

John S. Gero *Editor*

Design Computing and Cognition '18

 Springer

Design Computing and Cognition '18

John S. Gero
Editor

Design Computing and Cognition '18

 Springer

Editor

John S. Gero
Department of Computer Science
and School of Architecture
University of North Carolina at Charlotte
Charlotte, NC, USA

ISBN 978-3-030-05362-8 ISBN 978-3-030-05363-5 (eBook)
<https://doi.org/10.1007/978-3-030-05363-5>

Library of Congress Control Number: 2018963285

© Springer Nature Switzerland AG 2019

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The Second World War demonstrated that scientific research could be readily translated into technology. This opened up a much larger ambit of research than what had existed previously because of the change in its perceived value. The launch of the Sputnik satellite in 1957 by the Soviet Union prompted a reexamination of research in the West resulting in the establishment of the National Science Foundation and the Defense Advanced Research Agency in the US, which provided, and continue to provide, significant research funding to universities. Multiple disciplines that had not seen research as part of their activities began to view research, and in particular scientific research, as potentially beneficial. Design was one of these disciplines although the transfer of research to industry is not as well developed in design as in many other disciplines.

Design, when viewed from inside a discipline, often appears to be very narrowly concentrated within that discipline with little in common with design in other disciplines. This limited focus has made research into design less attractive as a field in its own right rather than being connected to any particular discipline. Further, the claim that the products of design are unique has been used to support claims that design cannot be studied scientifically because of its inherent lack of reproducibility. Science is built on the notion of regularities in phenomena with its implication of reproducibility. The regularities in designs and designing come from the structure of what has been designed and the processes used in their design, which make it open to scientific study. This division between designs and designing also matches computation well with its foundational ontology of representation and process.

The scientific study of designs and designing is often conflated with the notion that this makes designing scientific. Design science, a term coined by Buckminster Fuller in 1957, has come to denote three different streams of thought. Nigel Cross has called them scientific design, design science and a science of design. All three are represented by research in this volume, although the primary focus is on design science as the scientific study of design.

Design research treats designs and designing as a separate area from science and the humanities to produce a third domain of human endeavor: designing is designing. It borrows tools and techniques from other disciplines in the sciences and the humanities but remains distinct from them. The three waves of approaches to studying designing has been: formalization (through mathematics, logic and artificial intelligence), understanding the designer's mind while designing through cognitive science, and most recently understanding the designer's brain while designing through cognitive neuroscience.

The papers in this volume are from the *Eighth International Conference on Design Computing and Cognition (DCC'18)* held at the Politecnico di Milano, Lecco Campus, Italy. They represent the state of the art of research and development in design computing and design cognition including the nascent area of design cognitive neuroscience. They are of particular interest to design researchers, developers and users of advanced computation in design as well as to design educators. This volume contains knowledge about the cognitive behavior of designers, which is valuable for those who need to gain a better understanding of designing.

In these proceedings the papers are grouped under the following nine headings, describing both advances in theory and application and demonstrating the depth and breadth of design computing and design cognition:

- New Design Methods
- Design Cognition—Design Approaches
- Design Synthesis
- Design Theory
- Design Cognition—Design Behaviors
- Design Grammars
- Design Processes
- Design Modeling
- Design and Visualization

A total of 103 full papers were submitted to the conference, from which 40 were accepted and appear in these proceedings. Each paper was extensively reviewed by at least three reviewers drawn from the international panel of reviewers listed on the following pages. The reviewers' recommendations were then assessed before the final decision on each paper was taken. The authors improved their contributions based on the advice of this community of reviewers prior to submitting the final manuscript for publication. Thanks go to the reviewers, for the quality of these papers depends on their efforts. Special thanks to Sarah Abdellahi who helped put the volume together.

Charlotte, USA

John S. Gero

List of Reviewers

Henri Achten, Czech Technical University, Czech Republic
Janet Allen, University of Oklahoma, USA
Petra Badke-Schaub, TU Delft, Netherlands
Stefania Bandini, University of Milano-Bicocca, Italy
Daniela Barattin, University of Udine, Italy
Nicolo Beactini, Politecnico di Milano, Italy
Jose Beirao, Unibersity of Lisbon, Portugal
Eric Bianco, Grenoble INP Lucienne Blessing, SUTD, Singapore
Nathalie Bonnardel, Aix-Marseille Université, France
Yuri Borgianni, Free University of Bozen-Bolzano, Italy
Jean-Francois Boujut, Grenoble INP, France
Frances Brazier, TU Delft, Netherlands
Ross Brisco, University of Strathclyde, UK
David Brown, Worcester Polytechnic Institute, USA
Janet Burge, Wesleyan University, USA
Jonathan Cagan, Carnegie Mellon University, USA
Marco Cantamessa, Politecnico di Torino, Italy
Hernan Casakin, Ariel University Center of Samaria, Israel
Gaetano Cascini, Politecnico di Milano, Italy
Philip Cash, Technical University of Denmark, Denmark
Gabriela Celani, UNICAMP, Brazil
Amaresh Chakrabarti, Indian Institute of Science, India
John Clarkson, University of Cambridge, UK
Nathan Crilly, University of Cambridge, UK
Andy Dong, University of Sydney, Australia
Alex Duffy, Strathclyde University, UK
Chris Earl, Open University, UK
Claudia Eckert, Open University, UK
Athanasios Economou, Georgia Institute of Technology, USA
Tamer El-Khouly, The American University in Cairo, Egypt
Benoit Eynard, Université de Technologie de Compiègne, France

Stefano Filippi, University of Udine, Italy
Dan Frey, MIT, USA
Haruyuki Fujii, Tokyo Institute of Technology, Japan
Tsutomu Fujinami, JAIST, Japan
John S. Gero, University of North Carolina at Charlotte, USA
Ashok Goel, Georgia Institute of Technology, USA
Georgi Georgiev, University of Oulu, Finland
Gabriela Goldschmidt, Technion, Israel
Kosa Goucher-Lambert, CMU, USA
Ewa Grabska, Jagiellonian University, Poland
Kazjon Grace, University of Sydney, Australia
Andrés Gómez de Silva Garza, Instituto Tecnológico Autónomo de México, Mexico
Milene Guerreiro Goncalves, TU Delft, Netherlands
Tracey Hammond, Texas A&M, USA
Sean Hanna, University College London, UK
Armand Hatchuel, ParisTech, France
Laura Hay, University of Strathclyde, UK
Ann Heylighen, Katholieke Universiteit Leuven, Belgium
Ben Hicks, University of Bristol, UK
Ethan Hilton, Georgia Institute of Technology, USA
Hao Jiang, Zhejiang University, China
Yan Jin, University of Southern California, USA
Yehuda Kalay, University of California, Berkeley, USA
Udo Kannengiesser, neon IT-solutions GmbH, Austria
Sonal Keshwani, IIS, India
Mi Jeong Kim, Kyung Hee University, Korea
Maaikje Kleinsmann, TU Delft, Netherlands
Terry Knight, Massachusetts Institute of Technology, USA
Lauri Koskela, University of Huddersfield, UK
Spirios Kotsopoulos, MIT, USA
Ramesh Krishnamurti, CMU, USA
Djordje Krstic, Signalife, USA
Pascal Le Masson, Mines ParisTech, France
Julie Linsey, Georgia Institute of Technology, USA
Quentin Lohmeyer, ETH Zurich, Switzerland
Ade Mabogunje, Stanford University, USA
Mary Lou Maher, University of North Carolina at Charlotte, USA
Anja Maier, Technical University of Denmark, Denmark
Dan McAdams, Texas A&M, USA
Chris McComb, CMU, USA
Janet McDonnell, Central Saint Martins, University of the Arts London, UK
Chris McMahon, Technical University of Denmark, Denmark
Scarlett Miller, Pennsylvania State University, USA
Farrokh Mistree, University of Oklahoma, USA

Yukari Nagai, JAIST, Japan
Jeff Nickerson, Stevens Institute of Technology, USA
Jamie O'Hare, University of Bath, UK
Mine Ozgar, Istanbul Technical University, Turkey
Panos Papalambros, University of Michigan, USA
Rabee Reffat, Assuit University, Egypt
Monica Rossi, Politecnico di Milano, Italy
Federico Rotini, University of Firenze, Italy
Stephan Rudolph, University of Stuttgart, Germany
Somwrita Sarkar, The University of Sydney, Australia
Kristina Shea, ETH Zurich, Switzerland
Steven Smith, Texas A&M University, USA
Chris Snyder, University of Bristol, UK
Ricardo Sosa, Auckland University of Technology, New Zealand
Martin Stacey, De Montfort University, UK
George Stiny, Massachusetts Institute of Technology, USA
Mario Storga, University of Zagreb, Croatia
Rudi Stoufffs, National University of Singapore, Singapore
Joshua Summers, Clemson University, USA
Hsien-Hui Tang, UST, Taiwan
Toshiharu Taura, Kobe University, Japan
Megan Tomko, Georgia Tech, USA
Irem Turner, Oregon State University, USA
Barbara Tversky, Columbia and Stanford, USA
Robert Wendrich, Rawshaping, Netherlands
Ian Whitfield, University of Strathclyde, UK
Andre Wodehouse, University of Strathclyde, UK
Robert Woodbury, Simon Fraser University, Canada
Maria Yang, Massachusetts Institute of Technology, USA
Bernard Yannou, Ecole Centrale Paris, France
Seda Yilmaz, Iowa State University, USA
Robert Youmans, Google, USA

Contents

Part I New Design Methods

Toward the Rapid Design of Engineered Systems Through Deep Neural Networks	3
Christopher McComb	
Deep Component-Based Neural Network Energy Modelling for Early Design Stage Prediction	21
Sundaravelpandian Singaravel and Philipp Geyer	
Unsuccessful External Search: Using Neuroimaging to Understand Fruitless Periods of Design Ideation Involving Inspirational Stimuli	37
Kosa Goucher-Lambert, Jarrod Moss and Jonathan Cagan	
Designing with and for the Crowd: A Cognitive Study of Design Processes in NatureNet	55
Stephen MacNeil, Sarah Abdellahi, Mary Lou Maher, Jin Goog Kim, Mohammad Mahzoon and Kazjon Grace	
A Comparison of Tree Search Methods for Graph Topology Design Problems	75
Ada-Rhodes Short, Bryony L. DuPont and Matthew I. Campbell	

Part II Design Cognition—Design Approaches

Externalizing Co-design Cognition Through Immersive Retrospection	97
Tomás Dorta, Emmanuel Beaudry Marchand and Davide Pierini	
Demystifying the Creative Qualities of Evolving Actions in Design Reasoning Processes	115
Tamir El-Khouly	

The Effect of Tangible Interaction on Spatial Design Tasks	135
Jingoo Kim, Mary Lou Maher and Lina Lee	
Side-by-Side Human–Computer Design Using a Tangible User Interface	155
Matthew V. Law, Nikhil Dhawan, Hyunseung Bang, So-Yeon Yoon, Daniel Selva and Guy Hoffman	
Part III Design Synthesis	
Utility of Evolutionary Design in Architectural Form Finding: An Investigation into Constraint Handling Strategies	177
Likai Wang, Patrick Janssen and Guohua Ji	
Exploring the Feature Space to Aid Learning in Design Space Exploration	195
Hyunseung Bang, Yuan Ling Zi Shi, Guy Hoffman, So-Yeon Yoon and Daniel Selva	
Redefining Supports: Extending Mass Customization with Digital Tools for Collaborative Residential Design	213
Tian Tian Lo, Basem Mohamed and Marc Aurel Schnabel	
Voxel Synthesis for Generative Design	227
Matvey Khokhlov, Immanuel Koh and Jeffrey Huang	
Part IV Design Theory	
Model-Based Abduction in Design	247
Lauri Koskela and Ehud Kroll	
Ekphrasis as a Basis for a Framework for Creative Design Processes	265
Udo Kannengiesser and John S. Gero	
Notes for an Improvisational Specification of Design Spaces	285
Alexandros Charidis	
Design of Transfer Reinforcement Learning Mechanisms for Autonomous Collision Avoidance	303
Xiongqing Liu and Yan Jin	
Part V Design Cognition—Design Behaviors	
Building a Social-Cognitive Framework for Design: Personality and Design Self-efficacy Effects on Pro-design Behaviors	323
Hristina Milojevic and Yan Jin	

Cognitive Style and Field Knowledge in Complex Design Problem-Solving: A Comparative Case Study of Decision Support Systems 341
 Yuan Ling Zi Shi, Hyunseung Bang, Guy Hoffman, Daniel Selva and So-Yeon Yoon

What Do Experienced Practitioners Discuss When Designing Product/Service Systems? 361
 Abhijna Neramballi, Tomohiko Sakao and John S. Gero

Visual Behaviour During Perception of Architectural Drawings: Differences Between Architects and Non-architects 381
 Canan Albayrak Colaço and Cengiz Acartürk

Part VI Design Grammars

On John Portman’s Atria: Two Exercises in Hotel Composition 401
 Heather Ligler and Athanassios Economou

Monitoring China’s City Expansion in the Urban–Rural Fringe: A Grammar for Binjiang District in Hangzhou 421
 Ruichen Ni and José P. Duarte

Composite Shape Rules 439
 Rudi Stouffs and Dan Hou

Shape Grammars as a Probabilistic Model for Building Type Definition and Computation of Possible Instances: The Case Study of Ancient Greek and Roman Libraries 459
 Myrsini Mamoli

Grammars for Making Revisited 479
 Djordje Krstic

Part VII Design Processes

Rule-Based Systems in Adaptation Processes: A Methodological Framework for the Adaptation of Office Buildings into Housing 499
 Camilla Guerritore and José P. Duarte

Using Argumentative, Semantic Grammar for Capture of Design Rationale 519
 Raymond McCall

Identifying Design Rationale Using Ant Colony Optimization 537
 Miriam Lester and Janet E. Burge

Biased Decision-Making in Realistic Extra-Procedural Nuclear Control Room Scenarios 555
 Emil Andersen, Igor Kozine and Anja Maier

Part VIII Design Modelling

Modeling Collaboration in Parameter Design Using Multiagent Learning	577
Daniel Hulse, Kagan Tumer, Chris Hoyle and Irem Tumer	
Exploring the Effect of Experience on Team Behavior: A Computational Approach	595
Marija Majda Perišić, Mario Štorga and John S. Gero	
An Exploration of the Effects of Managerial Intervention on Engineering Design Team Performance	613
Joshua T. Gyory, Jonathan Cagan and Kenneth Kotovsky	
A Study in Function Modeling Preferences and its Variation with Designer Expertise and Product Types	631
Xiaoyang Mao, Chiradeep Sen and Cameron Turner	

Part IX Design and Visualization

Information Visualisation for Project Management: Case Study of Bath Formula Student Project	651
Nataliya Mogles, Lia Emanuel, Chris Snider, James Gopsill, Sian Joel-Edgar, Kevin Robinson, Ben Hicks, David Jones and Linda Newnes	
A Visualization Tool to Investigate the Interplay of External and Internal Processes	669
Mia A. Tedjosaputro and Yi-Teng Shih	
Visual Interactivity to Make Sense of Heterogeneous Streams of Design Activity Data	687
Yasuhiro Yamamoto and Kumiyo Nakakoji	
Style-Oriented Evolutionary Design of Architectural Forms Directed by Aesthetic Measure	705
Agnieszka Mars, Ewa Grabska, Grażyna Ślusarczyk and Barbara Strug	
Creative Sketching Apprentice: Supporting Conceptual Shifts in Sketch Ideation	721
Pegah Karimi, Kazjon Grace, Nicholas Davis and Mary Lou Maher	
Author Index	739

Part I
New Design Methods

Toward the Rapid Design of Engineered Systems Through Deep Neural Networks



Christopher McComb

The design of a system commits a significant portion of the final cost of that system. Many computational approaches have been developed to assist designers in the analysis (e.g., computational fluid dynamics) and synthesis (e.g., topology optimization) of engineered systems. However, many of these approaches are computationally intensive, taking significant time to complete an analysis and even longer to iteratively synthesize a solution. The current work proposes a methodology for rapidly evaluating and synthesizing engineered systems through the use of deep neural networks. The proposed methodology is applied to the analysis and synthesis of offshore structures such as oil platforms. These structures are constructed in a marine environment and are typically designed to achieve specific dynamics in response to a known spectrum of ocean waves. Results show that deep learning can be used to accurately and rapidly synthesize and analyze offshore structure.

Introduction

A significant amount of the final cost of a system is committed during design. According to the situated function–behavior–structure design framework, design consists of navigating from the requirements for a solution to the documentation of that solution [1, 2]. This process entails negotiating through several ontological categories, including function, expected behavior, derived behavior, and structure. The focus of this paper is on the tasks of analysis (deriving behavior from structure) and synthesis (generating a structure based on desired behavior). Many computational approaches have been developed to assist designers in the analysis and synthesis of engineered systems (e.g., computational fluid dynamics and topology optimization, respectively).

C. McComb (✉)

The Pennsylvania State University, University Park, PA, USA
e-mail: uum209@psu.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_1

However, these approaches are often computationally intensive, taking significant time to complete an analysis and even longer to iteratively synthesize a solution. The current work proposes a methodology for rapidly evaluating and synthesizing engineered systems through the use of deep neural networks.

The proposed methodology is applied to the analysis and synthesis of offshore structures. Examples of offshore structures include buoys, oil rigs, and cruise ships. The analysis of an offshore structure design often involves a simulation that combines multibody dynamics with computational fluid dynamics. This makes the analysis of solutions computationally intensive, precluding the use of design algorithms which are often stochastic in nature and require thousands of iterations [3–5]. The objective of the proposed work is to alleviate that problem by introducing a methodology for achieving two goals:

1. the rapid performance analysis of an engineered system, and
2. the rapid synthesis of an engineered system given desired performance characteristics.

The proposed approach makes use of deep neural networks to accomplish these objectives. Specifically, variational autoencoders are used to perform to reduce the dimensionality of the input and output data, making it possible to learn analysis and synthesis in a space of reduced complexity. The remainder of the paper is organized as follows. A background section reviews related work in machine learning and the design of offshore structure. The next section lays out the generalizable methodology for achieving both rapid analysis and synthesis of engineered systems. Results of applying this methodology to the design of offshore structure are presented and discussed. This paper concludes with a discussion of future directions for this work, highlighting the possible role of the engineering design community as a driving force in generative machine learning research.

Background

Neural Networks and Deep Learning

Artificial neural networks (referred to in the remainder of this paper simply as neural networks) are computational systems that are analogous to the biological neural networks that make up nervous tissue and animal brains. Neural networks can be trained to accomplish a variety of complex tasks, including regression, classification, and feature extraction. Jain et al. provide a more detailed introduction to neural networks [6]. Deep learning, which is the focus in this work, refers specifically to neural networks that have more than one hidden layer.

Neural networks have shown significant success in two-dimensional image recognition tasks. This success has led researchers to apply similar methodology to three-dimensional recognition tasks [7], facilitated by recent advances in computing that

enable such tasks to be performed at scale. Seminal dataset and classification efforts include ObjectNet3D [8], ShapeNet [9], VoxNet [10], and PointNet [11]. The automated synthesis of three-dimensional objects is still a nascent field in machine learning. Most approaches focus on creating objects with a given form and category (e.g., [12, 13]), rather than attempting to derive a deeper relationship between desired functionality and requisite form.

This work also makes use of autoencoders. These are specially designed neural networks that take an input, map it into a space with reduced dimensionality, and then output a reconstructed version of the input [14]. The two halves of the neural network (the encoder and the decoder, respectively) can then be used for specific and useful functions. The encoder can map an input into a reduced space, essentially performing data compression, while the decoder can take compressed values and reconstruct an output. This work uses *variational* autoencoders which map the input into a space of latent variables so that the training data are normally distributed [15]. This is accomplished by training the network with a loss function that measures reconstruction accuracy as well as how normally distributed the parameters in the maximally compressed layer are (typically Kullback–Leibler divergence). This ensures that the variables in the latent space are rich in information. Variational autoencoders have been used to compress a wide variety of different data, including human faces [16], handwritten numbers [17], and house numbers [18].

Neural networks of many varieties have been utilized in design and engineering to accomplish various tasks. For instance, Tseng, Cagan, and Kotovsky utilized a neural network to learn the preferences of a customer and then utilized that neural network as the objective function for a genetic algorithm [19]. Dering and Tucker utilized convolutional neural networks to predict the function of a product from its form alone [20]. The utilization of deep learning, and specifically autoencoders, also led to the creation of a computational framework that models the curiosity of a given user in order to provide surprising examples [21]. Neural networks have also been utilized to automatically predict quality defects in automotive parts [22] and to support design for additive manufacturing [23–25]. These examples, while not exhaustive, serve to highlight potential utility of neural networks for design and the need for a standardized approach to implementing them. The current research utilizes a generic, voxel-based approach for describing potential design solutions, thus ensuring significant representation flexibility.

Offshore Structures

Offshore structures are comprised of buoys, drilling platforms, and wave energy converters (WECs). WECs are an increasingly common type of offshore structure that are designed to extract energy from ocean waves. WECs may serve an important role in the future of humankind, since it is estimated that approximately 3.7 TW (3.7 trillion Watts) of power can be harvested from the world's oceans [26]. However,

in order to access that power, several challenges in the design of WECs must be overcome [27].

A substantial array of numerical methods have been developed for the simulation of offshore structures, including analytical methods, empirical methods, Navier–Stokes equation methods, and boundary-integral equation methods [28]. Analytical methods offer quick and rough estimates for devices with simple geometry, while most empirical methods attempt to maintain simplicity while making use of experimental values to increase accuracy. Navier–Stokes equation methods (NSEMs) can resolve highly nonlinear phenomena, but generally do not permit closed-form solutions, requiring the use of computational fluid dynamics.

Boundary-integral equation methods (BIEMs) are the focus of this work, as they are the industry standard for design and analysis of offshore structures. BIEMs produce a potential flow solution in the frequency domain [28]. This means that the outputs are given as spectra that indicate how much force, damping, or other quantities are applied to an offshore structure for incoming ocean waves with varying frequencies. Although they are far less computationally expensive than NSEMs, producing a full BIEM solution for a model with a high mesh resolution can still take hours. In addition, pure frequency-domain BIEMs are only weakly nonlinear [28] which makes it impossible to directly implement nonlinear control strategies within the simulation. One way to overcome this limitation is by numerically integrating over several frequencies of the BIEM solution to yield a time-varying series for different fluid phenomena [29]. These series can then be applied in an appropriate 6 degree-of-freedom (6DOF) solver to produce a time-domain simulation, from which important metrics such as average power production can be computed.

The application of the current work focuses on predicting frequency-varying wave force spectra as a function of structure geometry (and vice versa). It should be noted that a similar methodology could be applied directly to other frequency-varying fluid phenomena that are produced by a BIEM solution. The results of the current work could be integrated into software packages that utilize the BIEM+6DOF approach outlined above [30, 31]. This would enable rapid exploration of conceptual solutions, either by human designers and engineers or by agent-based design algorithms [4, 5, 32].

Methodology

The approach that is proposed in this work for rapidly evaluating and synthesizing engineered systems can be broken into three steps. This process is depicted graphically in Fig. 1. First, a diverse set of examples of a given system type must be generated, and the performance of each example must be analyzed using current methods (finite element analysis, computational fluid dynamics, experimental testing, etc.). The second step entails training two autoencoders: one for the engineered system and one for the performance assessment. The third and final step in the proposed methodology is the recombination of performance and system autoencoders

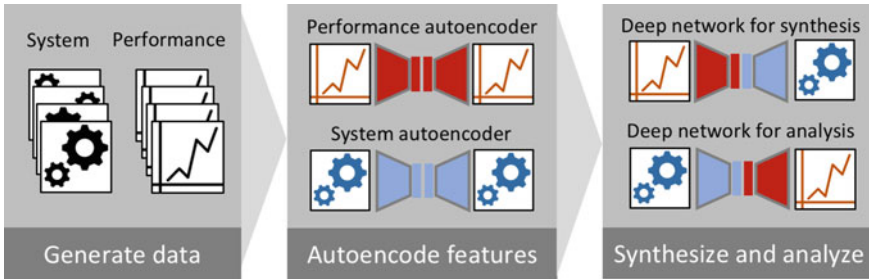


Fig. 1 Primary steps in the proposed methodology

into two new, deep networks that are possible of accomplishing rapid analysis of a system (encoding the system, decoding performance) as well as synthesis of a system according to desired performance (encoding performance, and decoding system). Both of these new networks should have one or more new layers that must be trained between the encoder and the decoder, enabling a mapping between the latent system space and the latent performance space (or vice versa). The use of autoencoders is critical as it permits the learning of synthesis and analysis in the latent space which has fewer dimensions (and thus less complexity) than the input or output.

The current work shares the application of the above methodology to analysis and synthesis of offshore structures. First, NEMOH, a BIEM solver, was used to simulate thousands of different floating body geometries, deriving frequency-varying response forces for each [35]. Next, the data generated in NEMOH was used to train two variational autoencoders, one of which modeled key features of frequency-varying response forces and the other modeled key geometric features of the input geometries. Finally, these autoencoders were used to instantiate two networks: one for predicting the force spectra of known geometries (analysis), and the other for generating geometries for a known force spectrum (synthesis). All neural networks were trained using the Keras neural network API [33] in conjunction with the Theano library [34]. A full implementation of this work, including training data, is available in the Python language under an MIT License.¹

Data Generation

The dataset used in this work was generated by instantiating 5000 different common shapes for offshore structures. These included wedges, hemispheres, cylinders, rectangular prisms, and cones. All shapes were generated to fit within a $10\text{ m} \times 10\text{ m} \times 10\text{ m}$ bounding box. These offshore structure shapes were then analyzed using the NEMOH BIEM solver [35], producing frequency-varying spectra describing the

¹<https://github.com/HSDL/WAnet/releases/tag/v1.0-beta>.

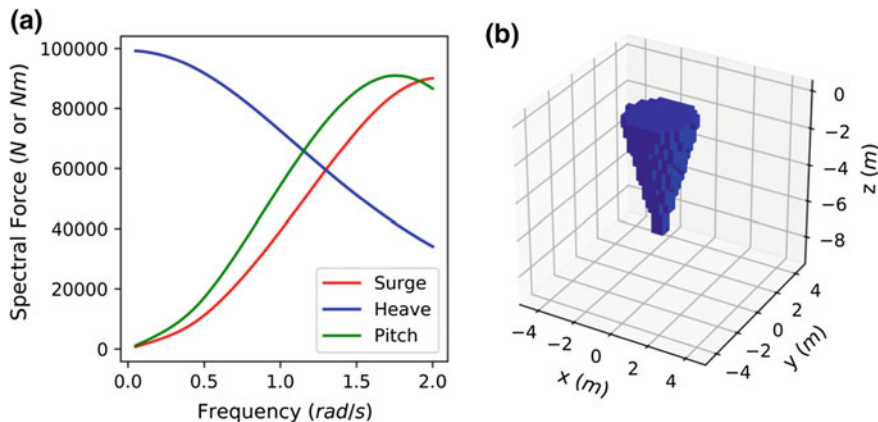


Fig. 2 Example of **a** force response spectra and **b** voxelized geometry

forces applied to the structure both parallel and perpendicular to the direction of the incoming ocean waves (commonly referred to as heave and surge, respectively), as well as a moment about the center of gravity of the body (referred to as pitch). An example of these spectra is provided in Fig. 2a.

The geometry for the offshore structures was originally provided to NEMOH as a mesh. The meshes were converted into a voxel-based format in order to make the geometry data more accessible to the proposed neural network approach. Voxels are a three-dimensional analogue of pixels. Specifically, the bounding box for the offshore structure was discretized into a $32 \times 32 \times 32$ grid, containing 32,768 voxels. The voxel values in this grid were defined as 1 (if the structure occupied part of the voxel) or 0 (if the structure did not occupy part of the voxel).

Thus, the final dataset consists of paired geometry-spectra observations. An example of a paired observation is provided in Fig. 2. Plots of force response spectra and voxelized geometries in the remainder of the paper will omit axis labels and scales in the interest of clarity and concision. This dataset was randomly separated into a training set (80% of the data, 4000 observations) and a testing set (20% of the data, 1000 observations). All accuracies reported in the remainder of this paper correspond to measurements on the testing set.

Training Variational Autoencoders

Two variational autoencoders were trained based on the data generated with NEMOH. The structure of these autoencoders is shown in Figs. 3 and 4. Both variational autoencoders are designed to compress the input data into an N -dimensional latent space, which describes the number of nodes in the smallest layer. The value of N is identified through a parametric search, detailed in the results section of this paper.

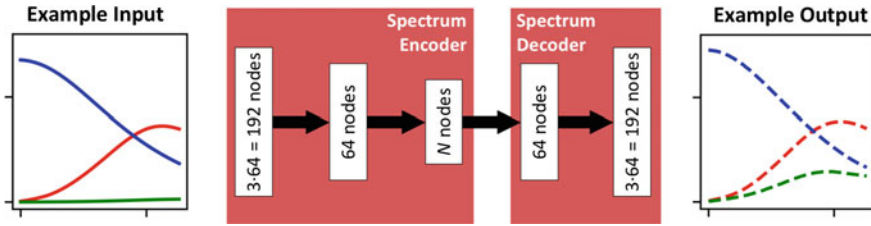


Fig. 3 Architecture for the force spectrum autoencoder

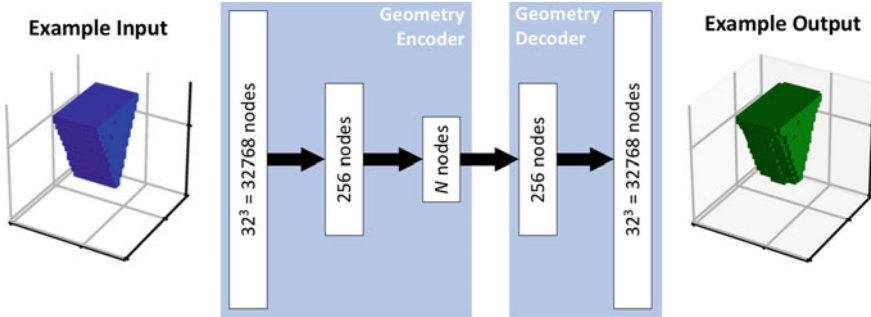


Fig. 4 Architecture for the voxel geometry autoencoder

Both autoencoders were trained using the root mean square propagation (RMSprop) algorithm [36]. The primary term in the loss function of the force spectra autoencoder was based on mean squared error while the primary term for the geometry autoencoder was based on binary cross-entropy. Both training algorithms also included a term for Kullback–Leibler divergence [37] of the values in the latent space (the innermost hidden layer) in the loss function. The computation of the loss functions in this way is standard for variational autoencoders.

It should be noted that the dimensionality of the latent space for both the spectrum autoencoder and the geometry autoencoder is described by a single variable, N , despite the fact that the force spectrum is much simpler than the structure geometry. This is an intentional decision, as equating the dimensionality of the spaces makes it more likely that a one-to-one mapping can be found between them.

Creating Neural Nets for Synthesis and Analysis

The variational autoencoders outlined in the previous section were recombined to instantiate two new deep neural networks, one for synthesizing geometries and the other for evaluating geometries. The structure for these neural networks is provided in Figs. 5 and 6.

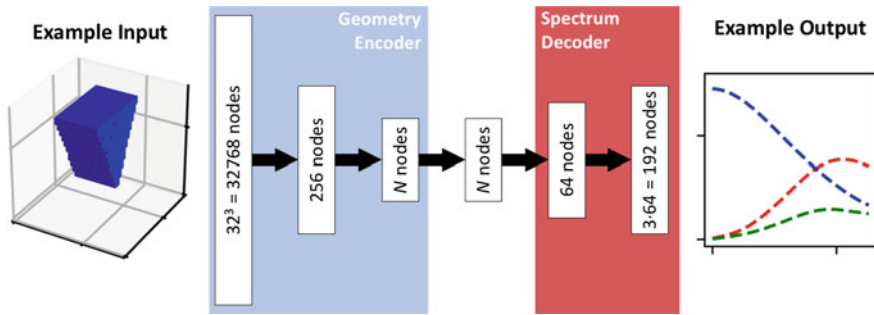


Fig. 5 Architecture for the analysis network

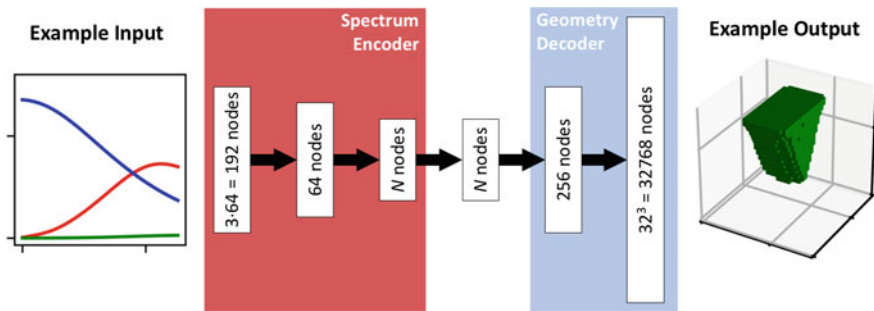


Fig. 6 Architecture for the synthesis network

The network designed for evaluating offshore structures utilizes the geometry encoder and the spectrum decoder (see Fig. 5). A layer of N nodes was included between these two elements, and this interior layer was the only layer that was trained. In essence, this network compresses the geometry of a structure into the N -dimensional geometry latent space (using the geometry encoder), maps the geometry latent space into the spectrum latent space (this is the trainable N -node layer), and then reconstructs the full force spectra, producing the desired output (using the spectra decoder).

The network designed for synthesis of offshore structure utilizes the spectrum encoder and the geometry decoder with a trainable N -node layer in between the two (see Fig. 6). This network compresses the spectra into the N -dimensional spectra latent space (using the spectra encoder), maps that into the N -dimensional geometry latent space (through trainable layer), and then reconstructs the full geometry (using the geometry decoder).

Both analysis and the synthesis networks were trained using the RMSprop algorithm [36]. The primary term in the loss function of the analysis network was based on mean squared error (since the output was a set of real-valued curves) while the primary term for the geometry autoencoder was based on binary cross-entropy (since the output was a set of voxel data with values between 0 and 1).

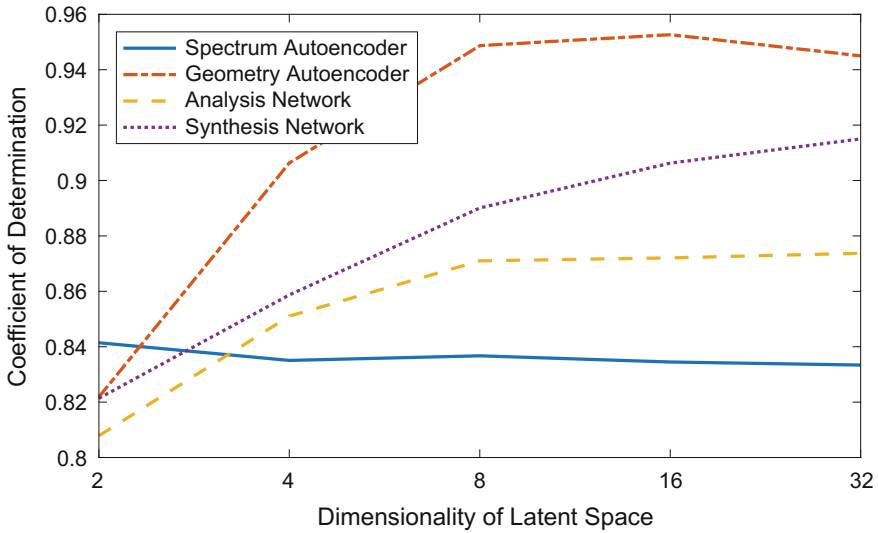


Fig. 7 Study to determine appropriate dimensionality of latent space

Results and Discussion

Results and discussion are provided in three subsections. The first subsection details a parametric study that was used to select the appropriate dimensionality for the latent space. The second reports the results and examples for the autoencoders (both geometry and spectrum). These autoencoders are critical as they permit the tasks of synthesis and analysis to be learned in a space of reduced complexity. The third subsection does the same for the recombined analysis and synthesis networks.

Determination of Latent Space Dimensionality

In order to determine the appropriate dimensionality for the latent space, a parametric study was conducted. All four networks used here (the spectrum autoencoder, the geometry autoencoder, the analysis autoencoder, and the synthesis autoencoder) were trained for increasing values of the dimensionality of the latent space, N . The results of this study are provided in Fig. 7. The horizontal axis shows dimensionality of the latent space and the vertical axis shows network validation accuracy (specifically, the percentage of variance explained by the trained network).

The data for the spectrum autoencoder is relatively flat, indicating that a small number of latent dimensions are sufficient to accurately reconstruct that data. The geometry autoencoder, in contrast, shows consistently increasing accuracy up to approximately 16 dimensions, at which point it begins to decrease. The analysis

and synthesis networks (which make use of portions of the autoencoders) continue to increase. However, training time increases substantially for larger latent spaces. Based on this study, a latent space dimensionality of 16 was selected as all networks are at or near maximum accuracy for this value.

Autoencoders

The force spectra autoencoder was trained for 100 epochs with a batch size of 100. The final mean squared error on the testing dataset was 9.90×10^9 . The total variance of the training data was 5.89×10^{10} yielding a coefficient of determination of 0.83. This indicates that this autoencoder can account for approximately 83% of the variance observed in the training data. Several randomly selected examples of original and reconstructed spectra are provided in Fig. 8. Although the curves are not reconstructed exactly, in all cases the reconstructed curves tend to share many similarities with the original curves. These similarities include slope, range, and the location of maxima and minima. However, some distinct differences become apparent for spectra that have low values. For instances, in Fig. 8a, b, the green spectrum (corresponding to pitch) is nearly flat in the original. The reconstructed version, however, shows significantly higher values for that spectrum. A similar overestimation is observed for the blue spectrum in Fig. 8d.

The geometry autoencoder was trained for 40 epochs with a batch size of 100. The final binary cross-entropy on the testing dataset was 0.17. By comparing the final binary cross-entropy of the model to the binary cross-entropy of a mean model (where the value of every voxel is the average over all voxels in the dataset), it is pos-

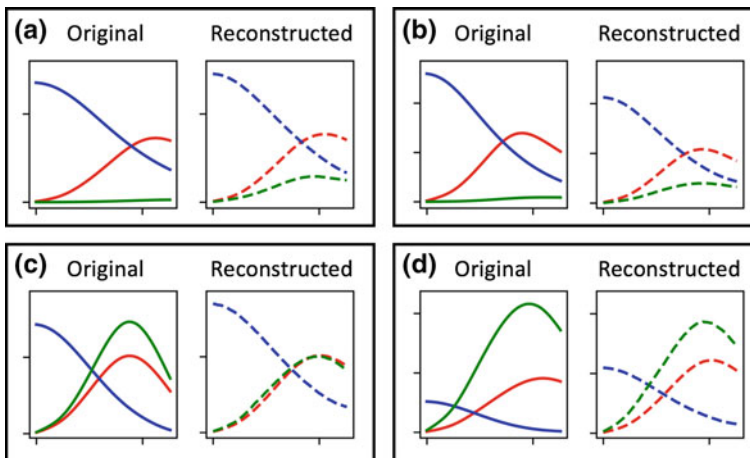


Fig. 8 Example results for force spectrum autoencoder

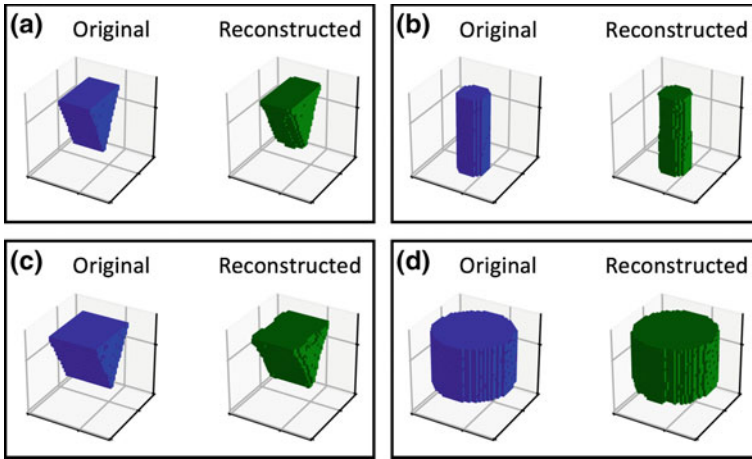


Fig. 9 Example results for offshore structure geometry autoencoder

sible to compute a coefficient of determination. The binary cross-entropy of the mean model is 3.42, yielding a coefficient of determination of 0.95. This value indicates that this autoencoder can reconstruct approximately 95% of the variance observed in the training data. Several randomly selected examples of original and reconstructed geometries are provided in Fig. 9. The original and reconstructed images are practically identical in many cases. The largest differences occur near sharp features, with the reconstructed showing a tendency to round corners and edges. In addition, flat faces in several of the geometries can be observed to bow outwards. It is possible that this could be corrected through the incorporation of convolutional layers in the autoencoder to better learn features that exist across size scales.

Synthesis and Analysis

This section reports the results of the neural networks designed to accomplish analysis and synthesis—these are the ultimate objects of the current work. These networks utilize portions of the autoencoders trained in the previous sections. Specifically, the network trained to perform analysis utilizes the encoder for voxel geometry and the decoder for force spectra. Conversely, the network trained for synthesis uses the encoder for force spectra and the decoder for voxel geometry.

The analysis network was trained for 25 epochs with a batch size of 100. The final mean squared error on the testing dataset was 7.49×10^9 , yielding a coefficient of determination of 0.87. Figure 10 shows several examples for the analysis network. From left to right, each example includes the geometry provided to the network as an input, the true spectra (the set of spectra produced by the geometry in NEMOH), and the predicted spectra (the output from the network). The characteristics of the

predicted spectra are similar in some ways to the reconstructed spectra in Fig. 8. The analysis network correctly predicts qualitative aspects of the curves, accurately producing curves with slopes, maxima/minima, and ranges that are similar to the true spectra. However, like the autoencoder, the analysis network tends to overestimate low values. This is evidenced in Fig. 10b, c. In addition, the true spectrum in Fig. 10c shows a very specific cusp feature which does not appear in the predicted spectrum. It is likely that cusp features of this type were rare in the training data, and thus are filtered out by the spectra decoder.

The analysis network was trained for 25 epochs with a batch size of 100. The final binary cross-entropy on the testing dataset was 0.45, yielding a coefficient of determination of 0.90. Figure 11 shows several examples of the synthesis network. From left to right, each example includes the set of spectra that was used as input, the true geometry (the geometry originally used to produce the input spectra in NEMOH), and the predicted geometry (the output from the network). In some of these examples, the synthesized geometry shows distinct departures from the true geometry. Sharp corners tend to be rounded off and flat faces bow outward slightly. This is expected, since similar behavior was observed in the geometry autoencoder.

In addition, it appears that in Fig. 11c a cone-type geometry was synthesized for what should have been a wedge. Similarly, in Fig. 11d, a square geometry was synthesized in place of what should have been a cylinder. At this point, the reason behind such idiosyncrasies is unclear. One possibility is that the departure from expected performance is due to simple errors in the synthesis. On the other hand, the synthesis network may have created a different geometry that provides force spectra that are very similar to what was desired. This will be a subject of future work.

The analysis and synthesis networks potentially provide very real utility for designers of offshore structure. The force spectra that are produced with the analysis network are important for simulation of offshore structures. However, the production of force spectra using BIEM methodology can take minutes for a simple mesh, which precludes the direct use of the approach in many optimization algorithms which might require tens of thousands of iterations. The use of the analysis network as an approximate BIEM makes the direct use of optimization algorithms more feasible.

Regarding the synthesis network, the ocean waves at a given location can be characterized with a power-frequency spectrum similar to the force spectra computed for the structure. If the peaks on the wave and device spectra align, then the device will absorb significant energy from the waves; if the peaks do not align, energy absorption is mitigated. Thus, many designers can estimate a desirable force spectrum for the structure based on known characteristics of the installation location, and use the synthesis network to directly generate a suitable geometry.

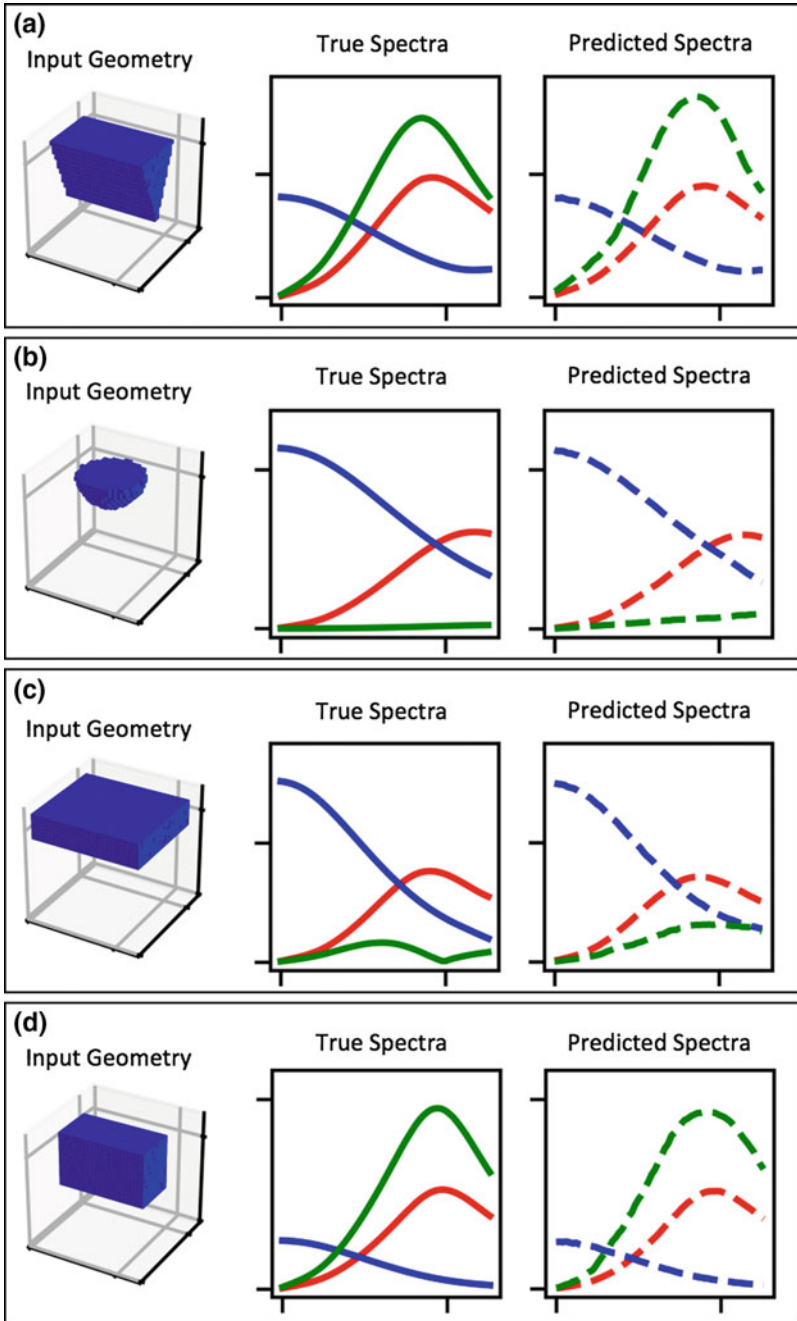


Fig. 10 Example results for analysis network

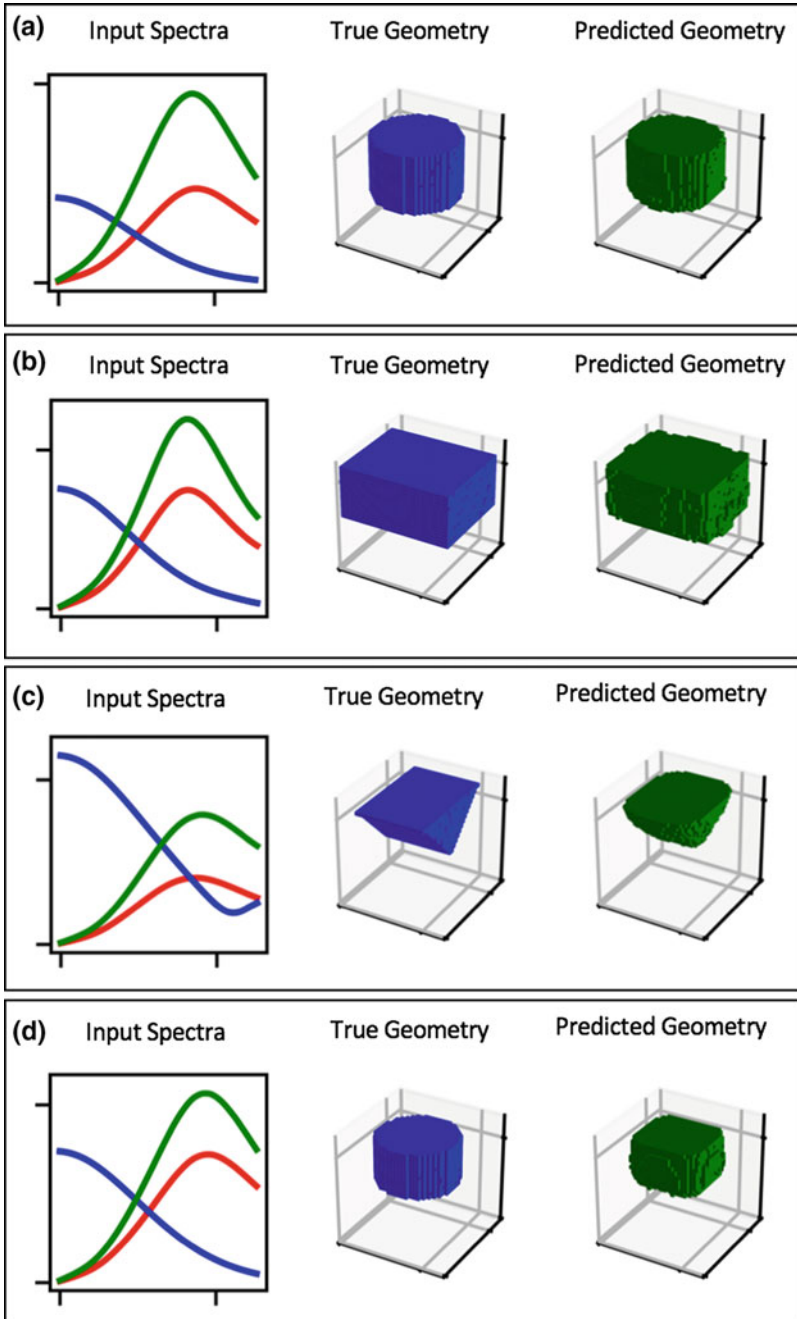


Fig. 11 Example results for synthesis network

Conclusions

The design of modern systems and products typically involves intensive computational analysis. For domains such as the design of offshore structures, these analyses can be particularly time-consuming. Standard methods for evaluating and synthesizing WECs and other offshore structure are too computationally expensive to efficiently implement within modern optimization and design algorithms. This work presented an autoencoder-based methodology for rapidly synthesizing and evaluating engineered systems in a space of reduced complexity, and applied that methodology to the synthesis and analysis of offshore structures.

The first step in the proposed methodology is the generation of data consisting of paired system design and performance information. In the offshore structure application of this paper, this consisted of voxel-based geometry paired with force spectra. The second step is the creation of two autoencoders that can compress and reconstruct both the system design and the performance information. The autoencoder for the force spectra achieved an overall reconstructive accuracy of 0.83, and provided strong qualitative reconstruction of the inputs (matching approximate range and location of maxima). The autoencoder for the voxelized geometry achieved an accuracy of 0.95 showing a strong ability to reconstruct common offshore structures, albeit with a propensity for rounding sharp corners. The third step of the methodology is the construction of networks for synthesis and analysis by reusing portions of the autoencoders. The analysis network (predicting force spectra based on geometry) achieved an accuracy of 0.87 and the synthesis network (predicting geometry based on design spectra) achieved an accuracy of 0.90. These results demonstrate that the proposed deep learning methodology is a promising means for accomplishing the rapid design of engineered systems.

Future work should investigate methods for increasing the accuracy of the autoencoders used here, as they are likely the limiting factor in the final accuracy of the analysis and synthesis networks. It may be possible to increase autoencoder accuracy through the use of convolutional layers [10] or the incorporation of generative adversarial network constructs [38]. In addition, the inclusion of eXplainable Artificial Intelligence (XAI) concepts [39–42] in conjunction with convolutional layers could provide designers with voxelized features that are aligned with high-performance solutions. Furthermore, although the geometries constructed by the synthesis network only differ slightly from the true geometries, the actual performance of the synthesized geometries is unknown. Future work should use NEMOH or another BIEM tool to directly evaluate the actual performance of synthesized geometries. In a similar vein, mapping differences between predicted and actual performance could indicate regions of the space that are particularly high performance.

Extensions of this work should also test the proposed methodology in other domains. As noted in the background section of this paper, machine learning for three-dimensional data is still nascent, particularly for synthesis tasks (typically referred to in machine learning as “generative” algorithms). Engineering design provides a large quantity of structured, three-dimensional data in the form of CAD files and pro-

totyped designs. Particularly, promising sources of training data include GradCAD and Thingiverse, online design communities in which users contribute 3D models. The existence of these, and other, sources of structured data positions the engineering design community as a future driving force in the evolution of generative machine learning methods.

Acknowledgements This material is based upon work supported by the United States Air Force Office of Scientific Research through grants FA9550-16-1-0049 and the Defense Advanced Research Projects Agency through cooperative agreement No. N66001-17-1-4064. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the sponsors. We also gratefully acknowledge the support of the NVIDIA Corporation for the donation of the Quadro P5000 GPU used in this work.

References

1. Gero JS (1990) Design prototypes: a knowledge representation schema for design. *AI Magaz* 11(4):26–36
2. Bhatta SR, Goel AK (1994) Discovery of physical principles from design experiences. *Int J AI EDAM (AI for Eng Des Anal Manufact)* (July 1992):1–22. <https://doi.org/10.1017/s0890060400000718>
3. Chakrabarti A, Shea K, Stone R, Cagan J, Campbell MI, Hernandez NV, Wood KL (2011) Computer-based design synthesis research: an overview. *J Comput Inf Sci Eng* 11(2):21003. <https://doi.org/10.1115/1.3593409>
4. Landry LH, Cagan J (2011) Protocol-based multi-agent systems: examining the effect of diversity, dynamism, and cooperation in heuristic optimization approaches. *J Mech Des* 133(2):21001. <https://doi.org/10.1115/1.4003290>
5. McComb C, Cagan J, Kotovsky K (2016) Drawing inspiration from human design teams for better search and optimization: the heterogeneous simulated annealing teams algorithm. *J Mech Des* 138(4):44501. <https://doi.org/10.1115/1.4032810>
6. Jain AK, Mao Jianchang, Mohiuddin KM (1996) Artificial neural networks: a tutorial. *Computer* 29(3):31–44. <https://doi.org/10.1109/2.485891>
7. Ioannidou A, Chatzilari E, Nikolopoulos S, Kompatsiaris I (2017) Deep learning advances in computer vision with 3D data. *ACM Comput Surv* 50(2):1–38. <https://doi.org/10.1145/3042064>
8. Xiang Y, Kim W, Chen W, Ji J, Choy C, Su H, ... Savarese S (2016) Objectnet3D: a large scale database for 3D object recognition. *Lecture Notes in computer science (including sub-series lecture notes in artificial intelligence and lecture notes in bioinformatics)*. 9912 LNCS: 160–176. https://doi.org/10.1007/978-3-319-46484-8_10
9. Chang AX, Funkhouser T, Guibas L, Hanrahan P, Huang Q, Li Z, ... Yu F (2015) ShapeNet: an information-rich 3D model repository. <https://doi.org/10.1145/3005274.3005291>
10. Maturana D, Scherer S (2015) VoxNet: a 3D convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS. IEEE), pp 922–928. <https://doi.org/10.1109/iros.2015.7353481>
11. Garcia-Garcia A, Gomez-Donoso F, Garcia-Rodriguez J, Orts-Escolano S, Cazorla M, Azorin-Lopez J (2016) PointNet: a 3D convolutional neural network for real-time object class recognition. In: 2016 International joint conference on neural networks (IJCNN). IEEE, pp 1578–1584. <https://doi.org/10.1109/ijcnn.2016.7727386>
12. Achlioptas P, Diamanti O, Mitliagkas I, Guibas L (2017) Representation learning and adversarial generation of 3D point clouds, pp 1–20. Retrieved from <http://arxiv.org/abs/1707.02392>

13. Wu J, Zhang C, Xue T, Freeman WT, Tenenbaum JB (2016) Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. (Nips). Retrieved from <http://arxiv.org/abs/1610.07584>
14. Hinton GE (2006) Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507. <https://doi.org/10.1126/science.1127647>
15. Kingma DP, Welling M (2013) Auto-encoding variational Bayes, (MI), pp 1–14. Retrieved from <http://arxiv.org/abs/1312.6114>
16. Rezende DJ, Mohamed S, Wierstra D (2014) Stochastic backpropagation and approximate inference in deep generative models. Retrieved from <http://arxiv.org/abs/1401.4082>
17. Salimans T, Kingma DP, Welling M (2014) Markov chain monte carlo and variational inference: bridging the gap. Retrieved from <http://arxiv.org/abs/1410.6460>
18. Kingma DP, Rezende DJ, Mohamed S, Welling M (2014) Semi-supervised learning with deep generative models. Retrieved from <http://arxiv.org/abs/1406.5298>
19. Tseng I, Cagan J, Kotovsky K (2012) Concurrent optimization of computationally learned stylistic form and functional goals. *J Mech Des* 134(11):111006-1–111006-11. <https://doi.org/10.1115/1.4007304>
20. Dering M, Tucker C (2017) A convolutional neural network model for predicting a product's function, given its form. *J Mech Des* 139(11):1–14. <https://doi.org/10.1115/1.4037309>
21. Grace K, Maher M Lou, Wilson D, Najjar N (2017) Personalised specific curiosity for computational design systems. In: *Design computing and cognition '16*. Springer International Publishing, Cham, pp 593–610. https://doi.org/10.1007/978-3-319-44989-0_32
22. Patel A, Andrews P, Summers JD, Harrison E, Schulte J, Laine Mears M (2017) Evaluating the use of artificial neural networks and graph complexity to predict automotive assembly quality defects. *J Comput Inf Sci Eng* 17(3):31017. <https://doi.org/10.1115/1.4037179>
23. Di Angelo L, Di Stefano P (2011) A neural network-based build time estimator for layer manufactured objects. *Int J Adv Manuf Technol* 57(1–4):215–224. <https://doi.org/10.1007/s00170-011-3284-8>
24. Xiong J, Zhang G, Hu J, Wu L (2014) Bead geometry prediction for robotic GMAW-based rapid manufacturing through a neural network and a second-order regression analysis. *J Intell Manuf* 25(1):157–163. <https://doi.org/10.1007/s10845-012-0682-1>
25. Chowdhury S, Anand S (2017) Artificial neural network based geometric compensation for thermal deformation in additive manufacturing processes, pp 1–10
26. Mørk G, Barstow S, Kabuth A, Pontes MT (2010) Assessing the global wave energy potential. In: *29th international conference on ocean, offshore and arctic engineering*, vol 3. ASME, pp 447–454. <https://doi.org/10.1115/omae2010-20473>
27. Czech B, Bauer P (2012) Wave energy converter concepts: design challenges and classification. *IEEE Ind Electron Mag* 6(2):4–16. <https://doi.org/10.1109/MIE.2012.2193290>
28. Li Y, Yu Y-H (2012) A synthesis of numerical methods for modeling wave energy converter-point absorbers. *Renew Sustain Energy Rev* 16(6):4352–4364. <https://doi.org/10.1016/j.rser.2011.11.008>
29. Babarit A, Hals J, Muliawan MJ, Kurniawan A, Moan T, Krokstad J (2012) Numerical benchmarking study of a selection of wave energy converters. *Renew Energy* 41:44–63. <https://doi.org/10.1016/j.renene.2011.10.002>
30. McComb C, Lawson M, Yu Y-H (2013) Combining multi-body dynamics and potential flow simulation methods to model a wave energy converter. In: *1st marine energy technology symposium*. <https://doi.org/10.13140/rg.2.1.3817.3285>
31. Ruehl K, Michelen C, Kanner S, Lawson M, Yu Y (2014) Preliminary verification and validation of WEC-Sim, an open-source wave energy converter design tool. In: *Volume 9B: ocean renewable energy*. V09BT09A040ASME. <https://doi.org/10.1115/omae2014-24312>
32. McComb C, Cagan J, Kotovsky K (2015) Lifting the Veil: drawing insights about design teams from a cognitively-inspired computational model. *Des Stud* 40:119–142. <https://doi.org/10.1016/j.destud.2015.06.005>
33. Chollet F (2015) Keras. GitHub

34. Al-Rfou R, Alain G, Almahairi A, Angermueller C, Bahdanau D, Ballas N, ... Zhang Y (2016) Theano: a python framework for fast computation of mathematical expressions. arXiv e-prints. abs/1605.0. Retrieved from <http://arxiv.org/abs/1605.02688>
35. Babarit A, Delhommeau G (2015) Theoretical and numerical aspects of the open source BEM solver NEMOH. In: 11th European wave and tidal energy conference (EWTEC2015), Nantes, France
36. Tieleman T, Hinton G (2012) Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude
37. Kullback S, Leibler RA (1951) On information and sufficiency. *Ann Math Stat* 22(1):79–86. <https://doi.org/10.1214/aoms/1177729694>
38. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, ... Bengio Y (2014) Generative adversarial networks. Retrieved from <http://arxiv.org/abs/1406.2661>
39. Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhutdinov R, ... Bengio Y (2015) Show, attend and tell: neural image caption generation with visual attention. <https://doi.org/10.1109/72.279181>
40. Chen K, Wang J, Chen L-C, Gao H, Xu W, Nevatia R (2015) ABC-CNN: an attention based convolutional neural network for visual question answering. Retrieved from <http://arxiv.org/abs/1511.05960>
41. Ba J, Mnih V, Kavukcuoglu K (2014) Multiple object recognition with visual attention. Retrieved from <http://arxiv.org/abs/1412.7755>
42. Mnih V, Heess N, Graves A, Kavukcuoglu K (2014) Recurrent models of visual attention. doi: 1406.6247

Deep Component-Based Neural Network Energy Modelling for Early Design Stage Prediction



Sundaravelpandian Singaravel and Philipp Geyer

Developing low-energy buildings calls for low-energy design and operations. Estimating operational energy of a building design supports major decisions taken at early design stages. To support early design decisions, accurate and quick predictions are required; a decision taken on predictions with poor quality can result in a wrong decision. The paper proposes a Deep Learning Model (DLM) structure that offers appropriate information about the dynamic thermal processes in a building to steer the design in the right direction. DLM prediction accuracy is similar to traditional building performance simulation (BPS) (achieves an R^2 higher than 0.97) while computation time is ~ 260 times faster than BPS. This accurate and fast feedback means decisions taken using DLM predictions will be valid as the design progresses.

Introduction

Our growing need to move towards a sustainable society calls for low-energy buildings. To design buildings meeting this requirement, building designs need to account for interactions that take place between environment, building form, materials and systems. Balancing these aspects aids in the development of a low-energy building design.

20% of the design decisions taken at early design stage affect 80% subsequent design decisions [1]. Hence, it is very important to have holistic design strategy at the start of design and to refine it as the design progresses. Today, low-energy design strategies are identified and validated through detailed Building Performance Simulation (BPS). BPS provides insights on how the building would operate for a

S. Singaravel (✉) · P. Geyer
KU Leuven, Leuven, Belgium
e-mail: sundar.singaravel@kuleuven.be

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_2

particular design under certain operational and environmental conditions. Energy demand estimates from simulations are the result of the dynamic interactions that occur between environment, heating/cooling system, occupant pattern and material's thermal conduction and storage properties. Insight on interactions and energy demand allows design teams to make the right design selections. During early design stages, BPS is used to identify building design options to meet the objectives of the low-energy design. Upon completion of the design process, BPS serves to validate if the design meets the targeted energy standard.

The main challenges in utilising BPS for developing holistic design strategy at the early design stage are computation time and information required to perform detailed energy analysis. Limited time for analysis of results in simulations limited to a few design options by far not exploring the available design possibilities. One workaround is to simplify the BPS model which increases the computation speed. Depending on the simplification performed by the energy modeller, a potential bias in the form of 'prediction gap' often occurs in the simulation outcomes [2]. Prediction gap is the difference between energy predictions obtained from detailed and simplified BPS. This gap could influence the reliability of a decision taken from the predictions. Hence, it is important to evaluate alternate methods that provide appropriate information for early design stage without compromising the reliability of a decision taken from an energy analysis.

Given the importance of early design decisions together with the limited amount of information and time for detailed BPS at early design stages, machine learning (ML) provides the ability to emulate BPS with simple input structure [3] and have high computational speed [4, 5]. Simple input structure allows us to develop models with abstract design element inputs like window-to-wall ratio rather than specific window dimensions. However, ML model is limited to the training data distribution. Hence, developing ML models with a wide set of design options is critical for its application in design.

Figure 1 shows the process for early design analysis using ML. The objective of early design analysis is to make the designers aware of the potential design options for low-energy design and how each design parameters impacts energy performance. This requires a repeated evaluation of design options, which is an iterative cycle of design exploration, as shown in the centre of Fig. 1; such an evaluation is supported by the developed ML approach. The ML model does not require all the detailed data as they might be developed in CAD or building information modelling (BIM); it requires only performance-relevant variables and parameters, which reduces modelling for performance prediction to the bare necessary information exchange. Identifying potential design options starts by understanding requirements and design variables (like window-to-wall ratio) constrained by building owner's expectations, architectural vision and energy efficiency regulations. One approach of examining constraints and variables in the design exploration process is generating a design sample matrix, for which design performance information is obtained through ML model. This approach is very efficiently supported by ML models. Potential design options and how design variables influence performance are identified and communicated to the design team. The design team works based on these insights.

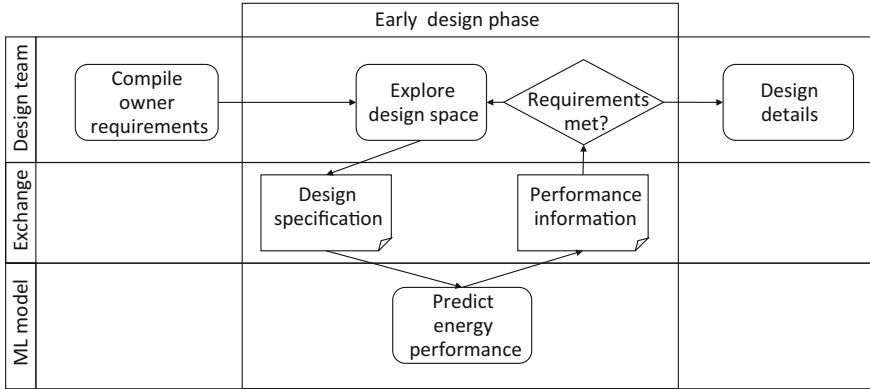


Fig. 1 Early design energy analysis through machine learning

A Component-Based Machine Learning Approach

To steer the design towards the right direction, designers not only rely on energy predictions but also on other factors influencing the energy predictions. These factors are heat flows via different elements of the building envelop, equipment gains and so on. The current approach is to apply monolithic ML models to represent building performance [6–10]. Limitations of a monolithic ML model are

- **Absence of typical design performance information.** Designers rely on different performance parameters such as heat gains via windows, internal gains, heat flows, etc., to make the right design changes. This information is not available in monolithic models. Thus, designers who use monolithic model’s for energy predictions would require domain experts to make the right design changes.
- **Complex BPS parametrisation and huge volume of data** are required to develop a model suitable for building design. No two building designs are alike. For monolithic models to cover modern design trends, such as zones with different heating parameters at one floor, would require data collection from all possible design options and training of a complex model structure. Thus, resulting ML model would be usable in limited situations.

To overcome these limitations, component-based ML (CBML) approach is proposed in [11, 12]. Component-based ML extends the reusability of ML models and provides the required information for a designer to support a design decision. Developing component-based ML models with appropriate information using data from BPS, ensures not only the required representation of dynamic effects to achieve reliable decision making but also provides the opportunity to evaluate more design options at the same time in the design exploration. Deep learning methods have shown better performance than traditional neural networks from modelling building energy performance through CBML approach [5, 13]. Furthermore, through

multi-task learning [14], a single model can represent multiple design performance indicator.

Aim and Scope of the Paper

This paper aims to show that through deep and multi-task learning, information required for energy efficient building design can be captured effectively. This includes ensuring

- Predictions from the ML model are similar to predictions from BPS. This similarity ensures that decisions taken based on ML model prediction are valid when evaluated with detailed performance analysis.
- Low computation time ensures evaluation of multiple design options at the early design stage.

As the main objective of the paper is to present deep learning model architecture developed through multi-task learning and its suitability for early design stage; design elements presented in this paper are limited to a simple but realistic architectural building design case.

Significance

Typical ML models developed in the domain of low-energy building design have a single-response variable like cooling energy demand prediction. Deep learning and multi-task learning have been successfully applied in domains like computer vision, natural language processing. Applications of such methods are limited in building design decision tools [15]. The concept presented in this paper can be extended to other design stages or evaluating different performance criteria (like thermal comfort, visible comfort and energy performance) for a building design. Models linking different design performance criteria enable holistic building design development. Through the presented approach multiple-response variables can be modelled at once; making this contribution significant.

Design performance indicators used in this paper are heat flows through envelop and energy demands. In the next section, the description of the proposed ML building energy model (BEM) is provided. This section is followed by showing the model's suitability for design applications, discussion and conclusions.

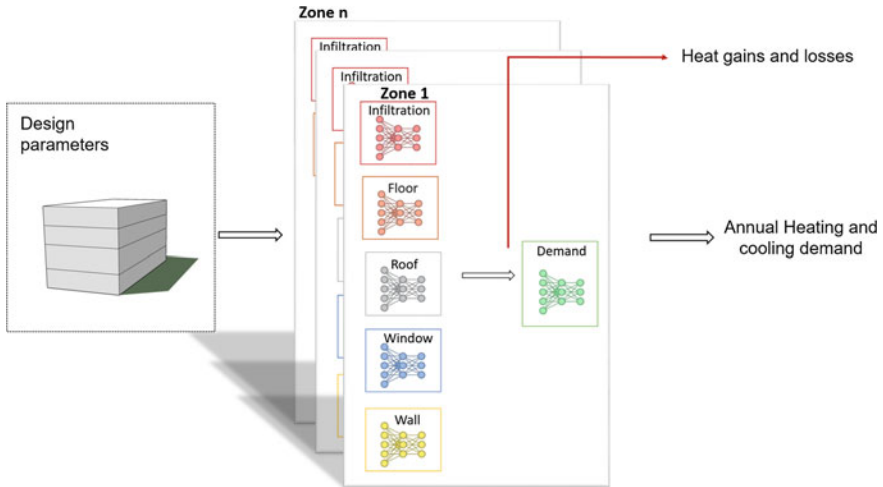


Fig. 2 Illustration of deep component-based neural network energy model

Deep Component-Based Neural Network Energy Model

A component is a representation of a building element, such as wall or HVAC systems. Deep learning model serves to represent the performance of a component or a system. The term deep component-based neural network energy model is coined as the proposed model represents BEM that predicts building performance and exploits both deep learning and component-based ML approaches. Figure 2 shows the deep component-based neural network energy models developed for early design stages. Deep learning model developed for this paper represents a thermal zone (e.g. Zone 1 in Fig. 2) with the outdoor-exposed wall, internal/ground floor, internal ceiling and roof. BEM is developed by aggregating predictions from different thermal zone models. This approach offers the following benefits:

- **It is easy to keep the models within training distribution while providing the possibility to evaluate complex designs.** Final energy prediction is the result of complex interactions that take place within a building. These interactions are linked to design, internal and external environment, confining the data collected to the specific design case. By capturing the information at a thermal zone level, the models can be utilized in new building design cases, while remaining under the training distribution of thermal zone.
- **It offers the granularity required for making design decisions.** By having models of thermal zones, it is easy to identify critical zones and develop design solutions suitable for this thermal zone.
- **Partial model update.** Certain design decisions can influence only a particular thermal zone in a building. Through the proposed approach, only appropriate

thermal zone model has to be updated. Eliminating the need to calculate the energy demand of the entire building again after small changes.

This paper limits design configurations to one thermal zone. Other types of thermal zone will be developed in upcoming research exploring further the approach’s potential to evaluate complex architectural design.

Deep Learning Model (DLM) Description

The developed DLM represents thermal zones in the ground, intermediate and top floors of a building. Each thermal zone model outputs (1) heat gains and loses via the thermal zone’s envelop and (2) heating and cooling demand of the thermal zone. The design determines the number of thermal zone inputs to be provided to the DLM. The outputs of the DLM are aggregated to obtain building energy predictions. In this section, the thermal zone deep learning model architecture and its performance are described.

Figure 3 shows the computational graph of the thermal zone DLM. The basic assumption behind this computation graph is that a ML model with appropriate inputs has high accuracy. By presenting the heat flow information together with design information to the final layers of the model, the robustness of the thermal zone’s energy predictions could be higher. Further research has to be done to evaluate its benefits. The model structure comprises four parts which are

- Shared hidden layer;

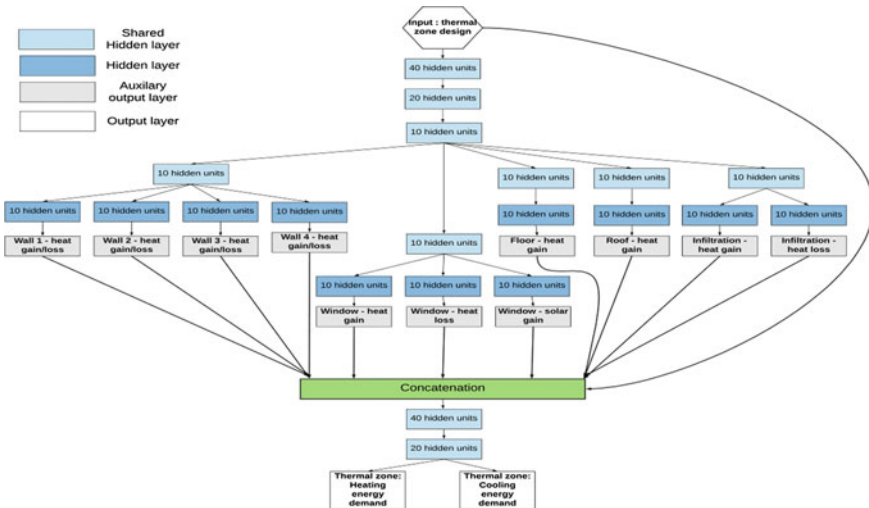


Fig. 3 Computation graph of thermal zone DLM

- Hidden layer;
- Auxiliary output layer;
- Output layer.

The shared hidden layers (light blue boxes in Fig. 3) are hidden layers that connect to hidden layers of specific tasks. These layers learn the features observed within the design input and feed their results into hidden layers. Hidden layers (dark blue boxes in Fig. 3) in-turn learn to predict from these general features. Auxiliary outputs, in this case, are heat gains and losses through thermal zone walls, windows, floor, roof, and infiltration. The output layer predicts zone's heating and cooling demand. Figure 3 shows the number of hidden units in each layer. Each hidden layer is activated through rectified linear units. Depending on the complexity of the data, the characteristics of the neural network layers can be modified.

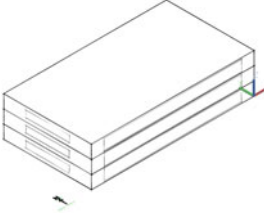
Training of the model is performed through multi-task learning method [14], where the training of all the layers is performed together. Since the accuracy of all output variables is important, the loss function has equal weightage for all output variables.

Description of Data

Training data is generated using parametric simulations in EnergyPlus. Table 1 shows the list of parameters and sampling range. Using Sobol sequence method, 1000 design combinations are generated. The generated samples represent different building design options. Annual energy rate and heat flow rate data per thermal zone are collected from the simulation models, resulting in a dataset of 3000 samples. Furthermore, the design parameters are uniformly changed in building design level and not at the thermal zone level. Major assumptions in the model are

- (1) Use of ideal HVAC systems (i.e. 100% efficiency);
- (2) The building is always occupied, and all equipment (i.e. lighting, equipment and HVAC) are used when the building is occupied;
- (3) Adiabatic internal floor, i.e. heat flows between zones are neglected. However, heat stored and released from floors are considered;
- (4) Fixed occupancy density of 10 m² per person.

Parametrizing these assumptions will increase the interactions within the simulation model. However, for training purpose, accumulated effects on an annual basis is used to steer the design. If these parametrized parameters are part of the input features, ML model should be able to learn these interactions. The effectiveness of the learning on more complex data is for future research.

Table 1 Design space covered by the training data


		Unit	Min.	Max.
Length (L)		m	20	80
Width (w)		m	20	80
Height (H)		m	3	6
Window to wall ratio (WWR)	South		0	0.95
	North		0	0.95
	East		0	0.95
	West		0	0.95
Orientation			-180	180
Wall U-value		W/m ² K	0.411	0.776
Window U-value		W/m ² K	0.5	2.0
Ground floor U-value		W/m ² K	0.411	0.864
Floor heat storage capacity		J/kg K	900	1200
Roof U-value		W/m ² K	0.191	0.434
Window g-value			0.1	0.9
Air change rate (ACH)		h ⁻¹	0.2	1
Lighting power		W/m ²	5	12
Equipment power		W/m ²	8	15

Computation Environment

The deep learning model is developed in python using Keras library with Tensorflow backend. Deep learning model computation time is estimated in a CPU with a clock speed of 2.7 GHz.

Performance of Thermal Zone Deep Learning Model

The generated data is split into the training, cross-validation and test dataset. Cross-validation data is used to tune the model parameters during training, while test data is used to evaluate generalisation of the trained model. The tuning parameters, in this case, are the parameter of the L2 regularisation for a fixed number of hidden units in each layer. The data is split in the ratio of 70/15/15. Test data performance is evaluated by coefficient of determination (R^2) and maximum absolute error.

Figure 4 shows the average mean squared error (MSE) for all response variable. As the MSE for both training and cross-validation (CV) error has not decreased after 300 epochs, the training of the model is stopped at epoch 300. Since both training and

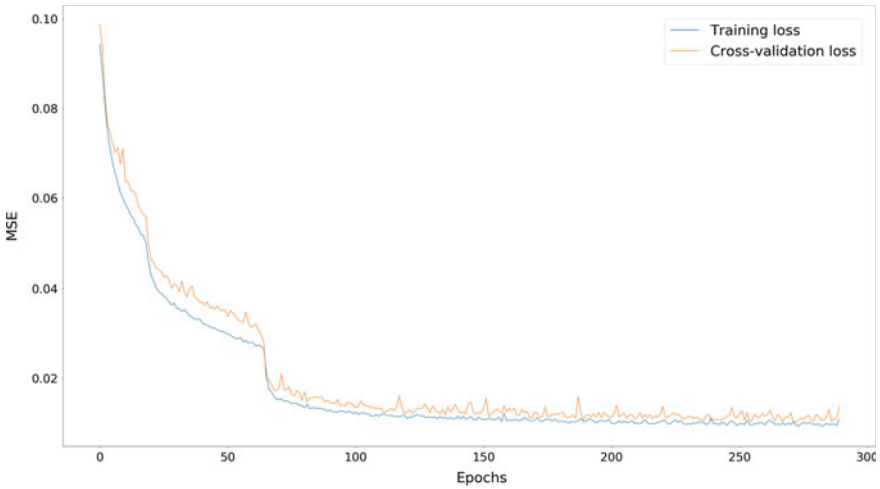


Fig. 4 History of training and cross-validation data loss

CV error have not diverged the model is not overfitting. However, the performance of the model determines the generalisation of the models.

Table 2 shows the performance of the model on training, CV and test data. Heating demand prediction has the maximum error of 21%. The error corresponds to 1.3 kWh/annum, while the maximum heating demand in test data is 6 kWh/annum. Figure 5 shows the predictions of heating and cooling demand from the ML model against simulation model. Reason for high error in heating predictions is most of the training data is concentrated in low heating demand area. However, the average test data maximum error and average test data R^2 for the DLM are 11.5% and 0.98 respectively. Hence, we can conclude that the model generalises within the training distribution.

Suitability of Deep Component-Based Neural Network Energy Model for Early-Stage Design Evaluation

The suitability of deep learning model compared with BPS for early-stage design is evaluated by the following criteria:

- **Reliability of a decision taken based on a prediction.** Evaluated by the similarities of predictions generated by deep learning model (DLM) and BPS. The aim is to ensure that decisions taken using DLM predictions will be valid when the design team switches to detailed BPS for validating building design compliance.
- **Computation time.** Having low computation time provides the possibility to explore more design options at early stages of design. High computation speed

Table 2 Performance of the deep learning model

Response	Training R ²	Cross-validation R ²	Test R ²	Test data maximum error (%)
Wall 1 heat flow	0.987	0.987	0.984	16.1
Wall 2 heat flow	0.990	0.988	0.988	7.7
Wall 3 heat flow	0.991	0.988	0.988	8.9
Wall 4 heat flow	0.987	0.983	0.982	17.8
Window heat gain	0.978	0.975	0.976	15.2
Window heat loss	0.989	0.974	0.983	10.6
Window solar heat gain	0.983	0.979	0.980	12.3
Floor heat gain	0.997	0.995	0.995	9.2
Roof heat gain	0.997	0.995	0.996	6.4
Infiltration heat gain	0.989	0.990	0.987	8.4
Infiltration heat loss	0.990	0.986	0.988	6.5
Heating demand	0.989	0.973	0.954	21.8
Cooling demand	0.992	0.990	0.989	8.9

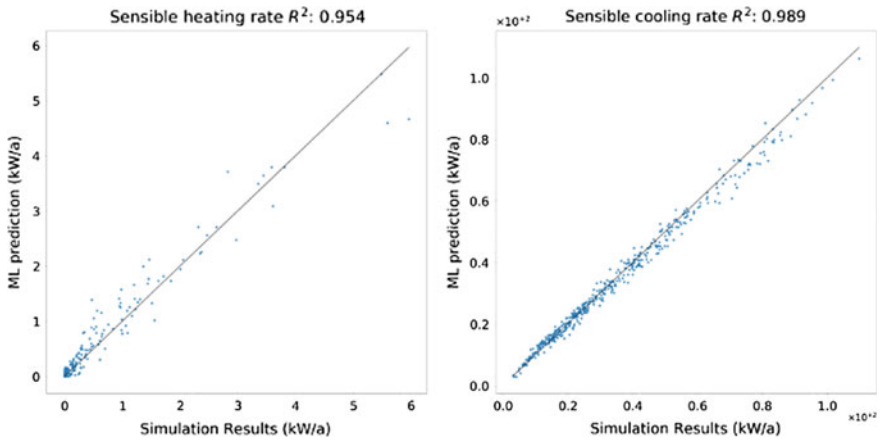


Fig. 5 Heating and cooling energy demand prediction accuracy

opens up the possibility of applying advanced optimisation and sensitivity analysis techniques for design space exploration.

For this evaluation, the developed method is applied to options of a building design test case. The design options evaluated resembles an actual building design shown in



Fig. 6 The design test case for evaluation through DML resembles EnergyVille I building in Genk (Photo By courtesy of EnergyVille/Nathalie Belmans.)

Fig. 6 illustrating that only through a simple thermal zone component, it is possible to evaluate real-world design cases. The design case is a new two-storey building with ground floor has a ceiling height of 5 m while other floors have a ceiling height of 3 m. Table 3 shows the parameter values of the design options evaluation. Option 1 and 2 have the same geometric characteristics, while the technical specifications are different. The transition from Option 1 to Option 2 represents a virtual decision process of improving a design option based on energy performance predictions as a basis for DLM evaluations.

Reliability of a Decision Taken with BPS and DLM

Figure 7 shows option 1 building-level predictions from BPS and DLM. This graph shows that predicted quantities based on the influential design parameters are similar for both prediction methods. No significant deviation occurs that could mislead decisions. Figure 8 shows option 2 building-level predictions from BPS and DLM and Table 4 shows the zone-level cooling energy predictions. It can be noted that with improvements in technical specifications in design, energy consumption rate has reduced in both BPS and DLM. The reductions observed by both the prediction methods are also similar. This shows that DLM can effectively capture complex interactions such as the effect of thermal mass on cooling demand. This also indicates that decisions taken on predictions from DLM are valid compared to traditional BPS predictions.

Table 3 Design options

		Option 1	Option 2
Length (<i>L</i>)		40	
Width (<i>w</i>)		40	
Height (<i>H</i>)		5—ground floor 3—middle and top floors	
Window to wall ratio (WWR)	South	60	
	North	90	
	East	50	
	West	80	
Orientation		0	
Wall U-value		0.58	0.39
Window U-value		0.90	0.50
Ground floor U-value		0.56	0.37
Floor heat storage capacity		1000	900
Roof U-value		0.28	0.19
Window g-value		0.5	0.1
Air change rate (ACH)		0.4	0.2
Lighting power		10	5
Equipment power		14	8

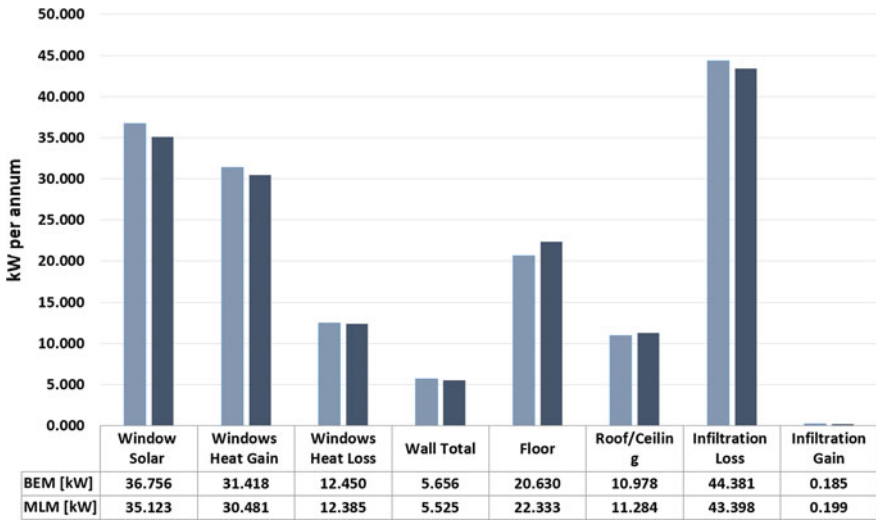


Fig. 7 Option 1: building-level predictions from (left bar) BPS and (right bar) DLM

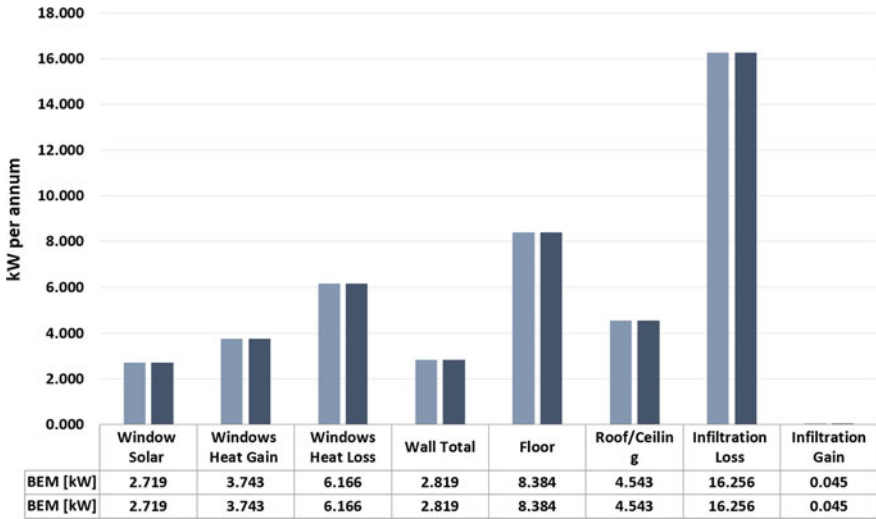


Fig. 8 Option 2: building-level predictions from (left bar) BPS and (right bar) DLM

Table 4 Zone-level cooling demand predictions

kW per annum	Option 1		Option 2	
	BPS	DLM	BPS	DLM
Zone 1	40.6	40.6	12.7	13.0
Zone 2	64.8	63.3	24.6	26.7
Zone 3	54.0	53.7	20.2	21.6

Table 5 Computation time of option 1 and 2

	BPS	DLM
Computation time (milliseconds)	9000	34–44

Computation Time

Table 5 shows the computation time to obtain results from BPS and DLM. BPS took 9000 ms for the simulating two design cases, while similar results were obtained from DLM in 34 to 44 ms. This time includes the time to load input data, scale input data, make predictions, rescale predictions, and aggregate predictions.

Discussion

The paper shows that through multi-task learning, a DLM with multiple performance indicators can be modelled through single training operation. The possibility to train a model with multiple performance indicators reduces the need to train separate components per indicator and thus reduces the overall model development time. DLM also offers the flexibility to scale easily to a larger dataset by simply increasing the size of the network. Size of the network can be increased by adding additional hidden units or hidden layer. Since, a larger model has more trainable parameters; giving it the ability to learn on a larger dataset. However, methods to work with large dataset need to be researched further. This flexibility to model multiple parameters on large dataset could result in general models that are suitable for design applications.

The paper proposes a component-based DLM structure that provides the flexibility to model building design options that are not explicitly part of the training data. For example, for a particular building design option within the training data, all zones have the same height. While the evaluated design cases have a larger zone height at the ground level than the remaining zones. However, this versatility of the model structure needs to be further evaluated when simplifying assumptions, such as adiabatic floors, are not present within the training data. For the adaptation to more complex models, potential DML extensions would be the incorporation of new features and addition of new components.

The evaluated design cases are simple. The current thermal zone model is limited to simple architectures and performance indicators suitable for architects. This will be extended further to cover different types of thermal zones and performance indicators. The hypothesis is that with more types of thermal zones more complex architectural designs can be evaluated. This will be further researched allowing to have general DLM for energy performance prediction in building design. Furthermore, depending on the type of thermal zone, certain components like floor or roof could be reused from the current thermal zone model. In such cases, dDeep learning methods like transfer learning and fine-tuning can be applied to make the model development process more efficient. An initial study has shown through transfer learning and fine-tuning, models with high accuracy can be developed in shorter training time [5]. In this study, heating DLM was developed from scratch with an R^2 of 0.99. Through transfer learning, another DLM that predicted cooling demand was developed with an R^2 of 0.99 [5]. The effectiveness of these methods for larger model needs to be further evaluated.

DLM is also developed on data generated from BPS with major assumptions in HVAC systems and occupancy. Data generated from detailed BPS will incorporate more dynamics. This will result in a model structure with more components and features. The component could be HVAC system and occupancy characterisation models. The required model structure and input structure for more detailed BPS require further research.

Conclusions

Results indicate that a DLM with multiple performance indicators for energy performance prediction is feasible. Predictions from DLM are similar to BPS, while the current DLM is about 260 times faster than BPS. This means that decisions taken in DML predictions will be similar to BPS predictions but much quickly available. Prediction accuracy combined with high computation speed makes DLM model suitable for early design decisions. However, the effectiveness of covering complex architectural design is still to be explored. Benefits of deep component-based neural network energy model based on this paper are

1. The versatility to apply in design situations.
2. Data has to be feed into the DLM only once to obtain different performance estimates for a thermal zone.
3. Prediction accuracy is similar to BPS and high computation speed compared to BPS.
4. Simplified modelling procedures are not requiring as much information as detailed BPS.
5. Model with multiple performance outputs can be developed at once. This makes the development process more efficient.

Due to these advantages, DML component method is well-suited for performance prediction replacing complex simulations in early design phases.

References

1. Attia S, Gratia E, De Herde A, Hensen JLM (2012) Simulation-based decision support tool for early stages of zero-energy building design. *Energy Build* 49:2–15
2. Singaravel S, Geyer P (2016) Simplifying building energy performance models to support an integrated design workflow. In: *International workshop of the european group for intelligent computing in engineering* edition, p 23
3. Horsey H, Fleming K, Ball B, Long N (2016) Achieving actionable results from available inputs: metamodels take building energy simulations one step further. In: *ACEEE summer study on energy efficiency in buildings*
4. Van Gelder L, Das P, Janssen H, Roels S (2014) Comparative study of metamodelling techniques in building energy simulation: guidelines for practitioners. *Simul Model Pract Theory* 49:245–257
5. Singaravel S, Geyer P, Suykens J (2017) Deep learning neural networks architectures and methods: building design energy prediction by component-based models. *Manuscr Submitt Publ*
6. Dong B, Cao C, Lee SE (2005) Applying support vector machines to predict building energy consumption in tropical region. *Energy Build* 37(5):545–553
7. Chari A, Christodoulou S (2017) Building energy performance prediction using neural networks. *Energy Effi* 1–13
8. Hou Z, Lian Z, Yao Y, Yuan X (2006) Cooling-load prediction by the combination of rough set theory and an artificial neural-network based on data-fusion technique. *Appl Energy* 83(9):1033–1046

9. Widén J, Wäckelgård E (2010) A high-resolution stochastic model of domestic activity patterns and electricity demand. *Appl Energy* 87(6):1880–1892
10. Yu Z, Haghghat F, Fung BCM, Yoshino H (2010) A decision tree method for building energy demand modeling. *Energy Build* 42(10):1637–1646
11. Singaravel S, Geyer P, Suykens J (2017) Component-based machine learning modelling approach for design stage building energy prediction: weather conditions and size. In: *Proceedings of the 15th IBPSA conference*, pp 2617–2626
12. Geyer P, Singaravel S Component-based performance prediction for sustainable building design based on systems engineering and machine learning: the energy perspective. *Manuscr Submitt Publ*
13. Singaravel S, Geyer P, Suykens J (2017) Deep neural network architectures for component-based machine learning model in building energy predictions. In: *EG-ICE*, pp 260–268
14. Ruder S (2017) An overview of multi-task learning in deep neural networks. *arXiv Prepr*
15. Amasyali K, Gohary N (2018) A review of data-driven building energy consumption prediction studies. *Renew Sustain Energy Rev* 81:1192–1205

Unsuccessful External Search: Using Neuroimaging to Understand Fruitless Periods of Design Ideation Involving Inspirational Stimuli



Kosa Goucher-Lambert, Jarrod Moss and Jonathan Cagan

This paper uses neuroimaging to provide insight into specific cognitive processes involved in design conceptualization with and without the support of inspirational stimuli. In particular, this work focuses on neural activity during unsuccessful search for a design solution. Twenty-one participants completed a brainstorming task while undergoing functional magnetic resonance imaging (fMRI). Participants were asked to think of concepts with and without the support of inspirational stimuli for 12 design problems. Behavioral results indicated that inspirational stimuli were most impactful after participants had time to begin developing solutions for a design problem. fMRI results during periods without inspirational stimuli indicated brain regions indicative of an impasse-based search strategy. This work elucidates cognitive mechanisms of continued search for insight into a design problem before a solution has been obtained. Furthermore, this work explores the meaning of distance for inspirational stimuli and what happens when stimuli are too far from the problem domain.

Introduction

Analogical reasoning and related processes have been formally investigated as a tool to support design ideation for over 30 years [1–9]. However, there is still a significant amount to learn regarding the cognitive processes that underpin design ideation involving inspirational stimuli (e.g., analogies). The overarching goal of this work is to understand unique brain networks that are activated during concept generation with and without the support of inspirational stimuli. Gathering insights into the neural

K. Goucher-Lambert (✉) · J. Cagan
Carnegie Mellon University, Pittsburgh, USA
e-mail: kgoucher@andrew.cmu.edu

J. Moss
Mississippi State University, Starkville, USA

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_3

activity during design ideation will allow for a more holistic understanding of how inspirational stimuli affect cognitive strategies undertaken by design practitioners. Uncovering this information will help design researchers to create more effective design theories, methods, and tools.

The research presented in this paper examines a piece of this overall goal. Broadly speaking, the analysis of neuroimaging data can be examined using either response models or block models. Response models focus on specific moments in time around a participant response (e.g., the few seconds surrounding when a design solution is indicated as being generated). Block models average neural activity over longer periods of time (e.g., overall activity during a multiple minute design conceptualization session). Examining the data in terms of each of these two mechanisms answers fundamentally different questions. The focus of this work is investigating brain activity using block models over longer periods of time. Again, the task involves generating design concepts with and without the support of inspirational stimuli. Averaging brain activity over the entire problem-solving block (in which multiple solution concepts may be generated) is truly capturing the “unsuccessful” search for design solutions. This is because there is proportionally more time spent being stuck searching for solutions than actually finding a solution for a given design problem. As discussed within the methods section of this paper, an approach was taken to filter out times in which ideas were successfully generated, therefore isolating the brain activity associated with unsuccessful search.

Background

This section provides a background of important prior research findings at the intersection of both the design research and neuroimaging literature, with a particular emphasis on analogical reasoning.

Design researchers have applied several neuroimaging techniques, including electroencephalography (EEG) [10, 11] and functional magnetic resonance imaging (fMRI) [12–14]. fMRI, the method of choice in this paper, is a brain imaging modality that measures changes in blood oxygen levels in short (~1 s) intervals of time. This change in blood flow gives an indication of brain activity and allows researchers to determine changes in activity due to specific experimental task conditions. Using fMRI, it is possible to gain an understanding of the cognitive processes involved during specific design related tasks beyond what could be determined from a traditional behavioral study. Neuroimaging techniques, such as fMRI, provide insight into what participants are truly thinking, feeling, and desiring at the time of mental judgments.

A few examples of prior investigations at the intersection of neuroimaging and design research include product preference judgments and design creativity. Sylcott et al. used fMRI to investigate tradeoffs between form and function preference decisions [14]. More recently, work by Goucher-Lambert et al. examined the neural signatures of product preference judgments involving sustainable design attributes [15]. Using fMRI, the presence of a network of brain regions associated with moral

reasoning and theory of mind (i.e., “what will other people think of my actions and behavior”) was present during sustainable product preference decisions. In a separate study from Alexiou and colleagues, the neural correlates of creativity in design during an apartment layout task were examined [12, 16]. This study indicated that the dorsolateral prefrontal cortex, a region of the brain critical to cognitive executive functions, including working memory and cognitive flexibility, was highly involved in design cognition during ill-structured design tasks. Finally, a study by Saggari et al. [18] used fMRI to study creativity during concept generation using a Pictionary-like game. Researchers found increased activation in several brain regions during concept generation compared to control, including the left parietal, right superior frontal, left prefrontal, and cingulate regions [17, 18]. Together, these prior works indicate that fMRI is most effective by providing links between specific features of design decisions to brain activation associated with separate cognitive tasks.

In this work, design ideation supported by “inspirational stimuli” is investigated, which is hypothesized to encourage cognitive processes closely related to analogical reasoning. Broadly speaking, the engineering design literature typically refers to analogical reasoning as a process by which information from a source domain or area is applied to a target through the connection of relationships and representations between the two [4, 19]. In this work, inspirational stimuli are provided to designers (participants) and then the relational mapping from the stimuli (source) to the problems (target) is left to the designer. Because of this key distinction, the stimuli provided in this work are not described as analogies themselves, and instead termed “inspirational stimuli.” However, if designers are able to construct the relational mapping from the stimuli to the problem, they are likely engaging in what is typically considered analogical reasoning.

Analogical reasoning has been intensely studied because analogies can serve as a powerful mechanism to assist designers in stimulating ideas more fluently, and/or ideas that embody positive characteristics (e.g., increased novelty, quality) [2–6, 8, 20–23]. In addition to the aforementioned areas of research regarding analogical reasoning, an additional body of prior work has centered on two fundamental features of utilizing analogies to inspire design activity. These are *when* an analogy should be presented and *what* analogy should be provided. In answering the *when* part of this question, research has identified that analogical stimuli are most impactful when presented after the development of an open goal (problem-solving has commenced, but the problem remain unsolved). Research from Tseng et al. found that when distant analogies were given after the development of an open goal, more novel and diverse solutions were produced [8].

Research answering what analogies are most effective has focused on understanding analogical distance. Primarily, research on analogical distance uses the terms “near” and “far” to discuss the distance of the analogy from the problem being examined [5, 24]. A near analogy implies that the analogy comes from the same (or closely related) domain, where a far analogy comes from a distant domain. Prior work has demonstrated differing results as to whether near or far analogies are more beneficial. For example, far analogies are usually considered to yield more novel solutions [25], yet they have also been found to cause to increase design fixation

[26]. In reality, it appears that the most impactful and effective analogies may reside between being too near and too far. Work by Fu et al. converged on this idea, by identifying the existence of a “sweet spot” of analogical distance; where analogies were most helpful to designers [5].

From a cognitive neuroscience perspective, analogical reasoning is a relevant and active area of research, as it is representative of some of the most unique characteristics of human logic, creativity, and thinking [27]. Neuroimaging studies in this area attempt to map the neural processes involved in analogical reasoning by breaking the process into separate component parts and studying them one at a time. Prior work has identified key component parts including encoding/retrieval (the source of the analog is identified and retrieved in memory), mapping (information from the source is matched or applied onto a target), and response [27]. Encoding and retrieval primarily depend on the type and complexity of the analogy being studied. The study presented here uses word-based inspirational stimuli. Prior work using word-based stimuli for analogical reasoning tasks of the form A:B::C:D has been shown to activate a temporal maintenance network associated with processing the words associated with the task [28]. Typically, the complexity of analogical stimuli has been controlled using text-based semantic approaches, such as latent semantic analysis [29]. The retrieval of the analogy from memory calls upon anterior parts of the prefrontal cortex (PFC) [29–32]. In addition, the rostralateral prefrontal cortex (RLPFC) has been identified as brain region, which supports higher level cognitive functions including analogical reasoning and episodic memory retrieval [33]. Wharton et al. implicated the left prefrontal and inferior parietal cortices as playing an important role in mediating analogical mapping [34]. These prior studies provide insights into the brain activation networks that may be observed in this work.

Methodology

An open-ended concept generation task using crowdsourced inspirational stimuli was used to investigate the cognitive processes involved in design ideation. While inside of an MRI machine, participants were asked to freely generate concepts for twelve different design problems from the literature. During problem-solving blocks, participants were presented with additional inspirational stimuli at varying distances (e.g., near vs. far). Using a combination of behavioral and neuroimaging data, an understanding of the ways in which inspirational stimuli impact design cognition was able to be determined. Of particular interest here was the overall neural activation patterns occurring over longer time periods, which was hypothesized to be consistent with the unsuccessful search for design solutions.

Participants

Twenty-one participants (13 male/8 female, mean = 27 yrs, SD = 5.4 yrs) were recruited to complete the fMRI study. Each of the participants provided informed consent in accordance with protocol approved by the Institutional Review Board at Carnegie Mellon University. In addition, all participants had design domain knowledge and experience as demonstrated by being an upper division student in Mechanical Engineering, or a Master's level student focusing on Design, Human-Computer Interaction, or Product Development. For their time, all participants were compensated monetarily and received a digital image of their brain.

Session Overview

The concept generation experiment that participants completed within the MRI machine was partitioned into three separate experimental conditions. Two of these conditions utilized inspirational stimuli at varying distances ("Near" or "Far"), and the third was a control condition in which words were reused from the problem statement. Each participant saw one condition per problem, and a total of twelve problems in the 1-hr session. The orders of these problem-condition pairs were presented in three separate counterbalanced groups.

The problems, as well as the inspirational stimuli used in this experiment, were identified in prior research from the authors, where a method was introduced to obtain useful inspirational stimuli with a crowdsourcing approach [35]. The motivation of this prior work was in part to address the difficulty in obtaining relevant and useful inspirational stimuli for wide varieties of design problems. The results of this work yielded an agnostic approach that utilized the naïve crowd to identify words, assessed analytically for their "distance", as inspirational stimuli for designers. Over 1300 crowd workers generated solutions. Near inspirational stimuli represented roughly the top 25% most used words, while the far stimuli sets were words that were only used once. The inspirational stimuli in this experiment were a subset of the extracted words from that prior experiment. The specific problems and words (inspirational stimuli) used for the fMRI experiment presented here are shown in Table 1.

The experiment consisted of a 1-hr brain scan, where participants generated ideas to various conceptual design problems. All experimental stimuli were presented in the MRI using the E-Prime Software package [44]. Subjects lay supine in the scanner, and viewed stimuli displayed using a monitor with a mirror fixed to the head mounted coil. Using a response glove strapped to their right hand, participants indicated a new response (including each time they had thought of a new idea) by pressing a button.

The timing of each trial is described in Fig. 1. For each problem, participants first read the problem statement for the given trial. Next, they began conceptualizing ideas for a 1-minute continuous block. During this period (WordSet1), participants were given a random subset of three words from the specific condition associated

Table 1 Problem statements and examples of inspirational stimuli from each experimental condition

Problem	Near words	Far words	Control words
1. A lightweight exercise device that can be used while traveling [36]	Pull, push, band, resist, bar	Roll, tie, sphere, exert, convert	Lightweight, exercise, device, while, traveling
2. A device that can collect energy from human motion [5]	Store, charge, shoe, pedal, step	Beam, shake, attach, electrons, compress	Device, collect, energy, human, motion
3. A new way to measure the passage of time [8]	Light, sand, count, fill, decay	Crystal, drip, pour, radioactive, gravity	New, way, measure, passage, time
4. A device that disperses a light coating of a powdered substance over a surface [6]	Spray, blow, fan, shake, squeeze	Rotor, wave, cone, pressure, atomizer	Light, coating, surface, powdered, substance
5. A device that allows people to get a book that is out of reach [37]	Extend, clamp, pole, hook, reel	Pulley, hover, sticky, voice, angle	Device, allows, people, book, reach
6. An innovative product to froth milk [38]	Spin, whisk, heat, shake, chemical	Surface, pulse, gas, gasket, churn	An, innovative, product, froth, milk
7. A way to minimize accidents from people walking and texting on a cell phone [39]	Alert, flash, camera, sensor, motion	Emit, react, engage, lens, reflection	Minimize, accidents, walking, texting, phone
8. A device to fold washcloths, hand towels, and small bath towels [40]	Robot, press, stack, table, rotate	Deposit, cycle, rod, funnel, drain	Fold, wash, cloths, hand, towels
9. A way to make drinking fountains accessible for all people [41]	Adjust, lift, hose, step, nozzle	Shrink, catch, attach hydraulic, telescopic	Way, drinking, fountains, accessible, people
10. A measuring cup for the blind [26, 42]	Braille, touch, beep, sound, sensor	Preprogram, recognize, pressure, holes, cover	Measuring, cup, for, the, blind
11. A device to immobilize a human joint [25]	Clamp, lock, cast, harden, apply	Shrink, inhale, fabric, condense, pressure	Device, to, immobilize, human, joint
12. A device to remove the shell from a peanut in areas with no electricity [43]	Crack, crank, blade, squeeze, conveyor	Melt, circular, wedge, chute, wrap	Device, remove, shell, peanut, areas



Fig. 1 Trial timing outline

with that problem (near, far, or control). A simple 1-back memory task was used to break up periods of concept generation, as prior research has determined that tasks lasting longer than approximately 1 min can have temporal frequencies that overlap with typical MRI signal drift [45]. After the 1-back task, participants continued generating design concepts for another 1-minute block (WordSet2). During this time, participants were shown the original set of three words (WordSet1), as well as an additional set of two new words (WordSet2). Following concept generation, participants provided ratings on a scale from 1 (low) to 5 (high) for four questions regarding the (1) usefulness and (2) relevancy of the inspirational stimuli, as well as (3) the novelty and (4) quality of their design solutions.

Data Analysis

fMRI Data Preprocessing

The raw neuroimaging data collected during the experiment were preprocessed and analyzed using the Analysis of Functional NeuroImages (AFNI) software package (March 1, 2017 version 17.0.11) [46]. A custom automated Nipype (Python language) preprocessing script was used to complete the preprocessing of the neuroimaging data into a form suitable for data analysis [47]. Preprocessing steps within this pipeline included slice scan-time correction, 3D rigid-body motion correction, high-pass temporal filtering, and spatial smoothing. Slice time correction aligned all slices within a brain volume to the first slice in that volume. Next, data from the functional image acquisitions were realigned to the first image of each run, and then again from this image, to the first run of each subject. The rigid-body rotation, translation, and three-dimensional motion correction algorithm examined the data to remove any time points where excessive motion occurred from the analysis. A high-pass Gaussian filter was used to remove low-frequency artifacts in the data. To reduce signal noise, the signal from each voxel was spatially smoothed using a Gaussian kernel (7 mm FWHM). Smoothing reduces the impact of high-frequency signal and enhances low-frequency signal. This causes more pronounced spatial correlation in the dataset. An anatomical image from each subject was co-registered to his or her corresponding functional images. The structural and functional images were transformed into Talairach space with 3 mm isometric voxels using AFNI’s *auto_tlrc* algorithm.

fMRI Data Analysis

In this work, the brain activity present during design ideation over long time scales is of interest. Here, the brain activity is averaged over the entire problem-solving period (WordSet1 and WordSet2). To do this, a mixed block model was utilized, which combined response regressors and the block regressors. The process of using response regressors (around the time of idea generation) in the block-level model has shown to be an effective way to measure sustained activity during task-level processing [48]. This allowed for an examination of widespread brain activity that is active across the whole concept generation period, while simultaneously filtering brain activity during idea generation.

At a block level, the resulting activity between contrasts is representative of the unsuccessful search for a design solution. By removing periods of productive idea generation captured in the response models, the block-level analysis captures brain activity representative of searching for a solution and not finding one. Neural activity at the block level was explored using the brain activation data from both WordSet1 and WordSet2 combined (2×60 s), as well as separately (60 s). The GLM block regressors were 1-parameter models with fixed shapes constructed using the AFNI BLOCK hemodynamic response type. The response regressors utilized the AFNI TENT (piecewise linear) and SPMG 2-parameter gamma variate models. The details of the implementation of how these models were executed can be found in companion work from the authors [49].

Behavioral Data Analysis

As mentioned previously, participants provided ratings across four metrics: the (1) usefulness and (2) relevancy of the inspirational stimuli being presented to them, as well as the (self-rated), (3) novelty and (4) quality of their design solutions. These scores were collected from participants after each problem during the fMRI experiment using the response glove provided to them. Repeated measures ANOVAs were used to determine whether there was a significant effect across the three experimental conditions (near or far inspirational stimuli, control) for any of these four rating areas.

Results and Discussion

This section introduces and discusses the behavioral and neuroimaging results from the fMRI design ideation experiment discussed previously. First, behavioral results are presented, as they help to better frame and interpret the results from the neuroimaging analyses. Results from a block-level fMRI analysis are presented, where

brain activation from within an entire problem-solving period (i.e., Near Condition, WordSet2) is averaged. Together, behavioral and neuroimaging analyses help to uncover the characteristics of design ideation involving inspirational stimuli, and in particular, periods of unsuccessful search during problem-solving.

Behavioral Results: Participants' Self-Rating Metrics

Mean values from participant self-reported ratings are shown in Fig. 2. Each value in the figure represents the mean rating (scale from 1–5) across participants for each of the three conditions. There were a total of 84 averaged responses for each metric. This amounted to the four problems of each condition that all of the 21 participants who completed the study saw (participants rated measures from each problem as a set). Using a repeated measures ANOVA, it was determined that there was no effect between conditions seen for how novel participants felt their solutions were ($F(2, 40)=0.43, p = 0.43$), or the overall self-rated quality of their solutions ($F(2, 40)=0.46, p = 0.63$). Prior work from the authors using these inspirational stimuli found that while participants did not self-report a difference in the novelty and quality of their design solutions, expert evaluators did perceive a statistically significant difference between the various conditions [35].

In addition to questions about their developed solutions, participants also provided ratings for the relevancy and usefulness of the inspirational stimuli. For each of these metrics, a highly significant effect was observed. For example, there was a strong correlation between participant ratings for the relatedness of the inspirational stimuli and its distance from the design problem. For this metric, participants rated all conditions to be significantly different from one another ($F(2, 40)=9.37, p \ll 0.01$). Near stimuli (mean = 3.7, SD = 0.97) were rated as being more relevant to the design problems than far stimuli (mean = 3.29, SD = 1.12) across all participants ($F(1, 20)=25.22, p \ll 0.01$). Similarly, there was a significant trend for participant judgments of the usefulness of the inspirational stimuli. The mean usefulness of the three conditions was different with a high degree of statistical significance ($F(2, 40)=76.73, p \ll 0.01$). Not surprisingly, participants rated the control stimuli (reused words from the design problem statement) as not useful (mean = 1.56, SD = 0.84). In addition, participants rated near stimuli (mean = 3.68, SD = 0.87) as being more useful than far stimuli (mean = 3.13, SD = 1.10). Finally, a separate contrast between the near and far conditions for the usefulness metric confirmed the significance of this difference ($F(1, 20)=11.12, p \ll 0.01$).

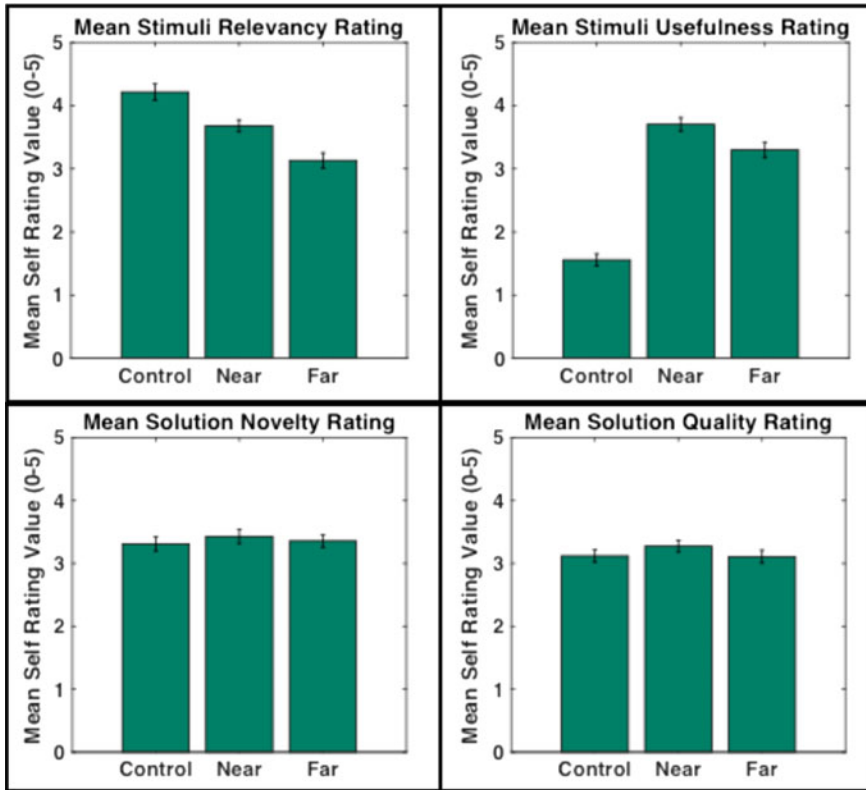


Fig. 2 Mean \pm 1 S.E participant self-ratings for relevance and usefulness of inspirational stimuli, and novelty and quality of design solutions ($N = 84$ per bar—21 participants * four samples of each condition)

Neuroimaging Results

Block Models: Brain Activation Patterns During the Unsuccessful Search for Design Solutions

Behavioral data provides insight into aspects of how design ideation is impacted by inspirational stimuli, but not *why*. This level of depth can be obtained using neuroimaging methods. As discussed in the methods section, a mixed event-related/block design was used to examine brain activity over the course of the entire problem-solving period. This gives a more holistic sense of brain activity while ideating about solutions, as the sharp areas of increased productivity during idea generation are masked by other forms of brain signal that are present throughout the duration of the block. In a sense, conducting a block-level analysis over the entirety of the 60-s block provides insight into brain activity when people are unsuccessful and are

struggling to develop a new solution. This is because the mixed model incorporates the response-level regressors. As a result, fine-grained activation patterns associated with successful ideation and mental search are modeled, and the resulting brain signal is consistent with the unsuccessful search for ideas.

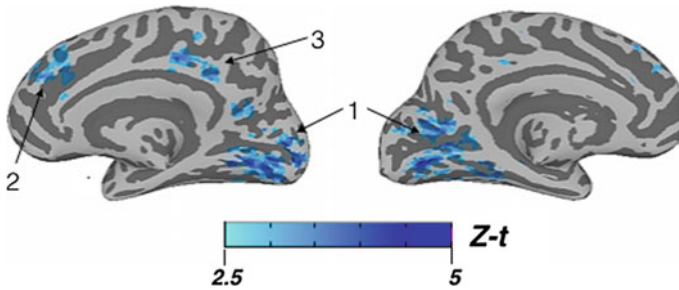
Contrasts were completed for all Condition (Near, Far, Control) and WordSet (WordSet1, WordSet2) combinations. From this analysis, only one contrast yielded significant group-level results: the Near–Control WordSet2 contrast. There is empirical evidence from this work that demonstrates that the impact of inspirational stimuli only truly takes effect in the second problem-solving block. This is consistent with prior research regarding open goals [50]. Research from Tseng et al. found that analogies were more helpful after an open goal already existed for the problem [8]. Therefore, one explanation for inspirational stimuli only having an impact during the second block of problem-solving is that the first block was needed to develop open goals for the problem, and having versus not having inspirational stimuli did not make a difference. This is a heuristic supported by the lack of significance in block-level contrasts involving WordSet1. For the Near–Control WordSet2 block-level contrast, the significant resulting brain activity clusters are all “negative”. This means that activity during the Control WordSet2 condition was greater than the Near WordSet2 condition. As mentioned previously, the significant areas of activation from this contrast are likely to represent areas associated with the unsuccessful search for a design solution during concept generation. From this analysis, it appears that this is occurring most when inspirational stimuli are not present and after an open goal has been established for a given problem (Control WordSet2 block).

All significant clusters of activation from this contrast are shown in Table 2 and Fig. 3. At the block level, increases in brain activity are seen in the primary visual cortex (V1), such as the bilateral lingual and calcarine gyri, as well as both posterior and anterior regions of the cingulate gyrus. This robust activation in the occipital gyrus (cluster 1) during the control condition points to increased time examining the problem statement when people are engaged in unsuccessful search. Prior research has linked increased visual activation to solving by analysis (as opposed to solving with insight), because participants have not yet found a source for insight [51]. In addition to visual processing-related brain regions, other areas of activation for this contrast were found in the posterior cingulate cortex (PCC). Research in cognitive neuroscience has still not reached a consensus regarding the exact role of the PCC. However, a comprehensive review of the role of the PCC in neuroimaging studies found that it may play a role switching between internal and external attention [52] (though to be fair, not much is generally known about switching between internal and external attention [53]). This type of activity makes sense, as switching between attention states would be necessary for participants as they continue to search for inspiration.

One explanation for the activation network established here (centered on the unsuccessful search for design solutions) is that participants are experiencing design fixation. Here, design fixation is defined as the impasse or mental block that occurs during the search for insight during a design problem, based upon the counterproductive impact of prior knowledge [54, 55]. Prior research demonstrates fixation is

Table 2 Near-control contrast for WordSet2 block. Individual voxels corrected to $p < 0.005$

	Region	B.A	x	y	z	k	Z-max	Alpha
1	R/L lingual gyrus, calcarine gyrus	18, 19	4.5	67.5	2.5	798	-4.58	<0.01
2	R/L superior medial frontal gyrus	8, 9, 32	4.5	-37.5	35.5	157	-3.74	<0.02
3	R/L posterior cingulate gyrus, paracentral lobule	31, 24	-1.5	22.5	31.5	72	-4.2	<0.08

**Fig. 3** Near-control contrast for WordSet2 block. Cluster numbering corresponds to Table 2

inversely related to the quantity of ideas being generated [54]. As the only significant brain activation occurred during WordSet2, it is possible that participants remain fixated on the initial ideas that they generated. Furthermore, not having additional inspirational stimuli in the control condition prevents participants from having a starting point to generate new insights into the problem space.

Further Identification of Brain Regions Indicative of Unsuccessful Search Using an Ancillary Block Modulation Analysis

The key result from the neuroimaging analyses was that there appeared to be a consistent network of brain regions (most notably areas in the occipital lobe including the lingual gyrus, cuneus, and calcarine gyrus) that seem to be linked to the unsuccessful search for a design solution. This was evident at the block level when contrasting Near WordSet2-Control WordSet2. To determine whether there was support for this

connection directly within the empirical data, an ancillary modulation analysis was completed.

The modulation analysis combined features of the block models and behavioral response data by modulating the amplitude of the block regressors based upon the number of responses participants made in a given block. Said otherwise, this analysis assumed that there was proportionality between the level of brain activity and the number of solutions the participant came up with during a given block. So, if a participant came up with fewer ideas during a block, then there would be a higher level of activity within regions associated with unsuccessful search.

This analysis indicated that unsuccessful search was present in all three of the experimental conditions. This in and of itself is not particularly surprising, due to the fact that unsuccessful periods are reasonably expected to occur when attempting to solve a difficult conceptual problem. Because the resulting values from this analysis are unweighted, it was not possible to directly compare the associated brain regions in one condition against another. To make this comparison, a region of interest (ROI) mask was created for the most statistically significant subset of these unsuccessful search regions. Following this, the mean brain activity for each condition during WordSet2 was sampled within each ROI to see whether there was a statistically significant difference between the conditions.¹

The extracted ROIs are listed in Table 3 and displayed in Fig. 4. The mean activity values from these ROIs were not statistically different, except for ROI 2. For this ROI, the mean activation was highest in the control condition ($F(2, 62) = 3.10$, $p = 0.052$). When comparing the mean activation for the near and control conditions within the extracted ROIs, the difference is highly significant ($F(1, 41) = 6.23$, $p = 0.017$). This shows that there was significantly more brain activity inside of the “unsuccessful search ROI” during the control condition. This ROI encompasses much of the same brain regions identified previously as being related to unsuccessful search these are occipital regions (for example lingual gyrus) and a portion of the posterior cingulate. The modulation and ROI results here, along with the distributive results from the block-level contrasts, lend strong support for the presence of an unsuccessful search region in these brain areas. This unsuccessful search region is most strongly correlated with the control condition, implying that solution search is more difficult in the absence of inspirational stimuli.

Put together with the previously presented results, in the absence of inspirational stimuli, participants engage in a unique search strategy. This strategy is represented by a specific brain activation network and is also present in the far condition (compared to the near condition). We call this network of brain regions and resulting solution search strategy *unsuccessful external search*. An increase in activity in primary visual processing-related brain regions, which make up the center of an identified unsuccessful search brain network, indicates that participants continue to explore

¹It should be noted that this method of ROI mask generation and sampling is similar to the analyses conducted in work by Goucher-Lambert et al. using the external neuroimaging database—Neurosynth. However, here the ROI mask was created based on a specialized analysis of the empirical data [56].

Table 3 Regions of interest for unsuccessful search modulation analysis

	Region	B.A	x	y	z	k
1	L middle/superior frontal gyrus	9, 32	31.5	-40.5	-3.5	881
2	R/L lingual gyrus, posterior cingulate	30, 19, 18	19.5	67.5	-18.5	508
3	R medial frontal gyrus, anterior cingulate	9, 32	-16.5	-46.5	8.5	267
4	R cerebellum	N/A	-43.5	52.5	-42.5	219
5	R postcentral gyrus, paracentral lobule	5, 6	-10.5	-46.5	50.5	154
6	L angular gyrus, middle occipital gyrus	39, 40	34.5	67.5	11.5	129

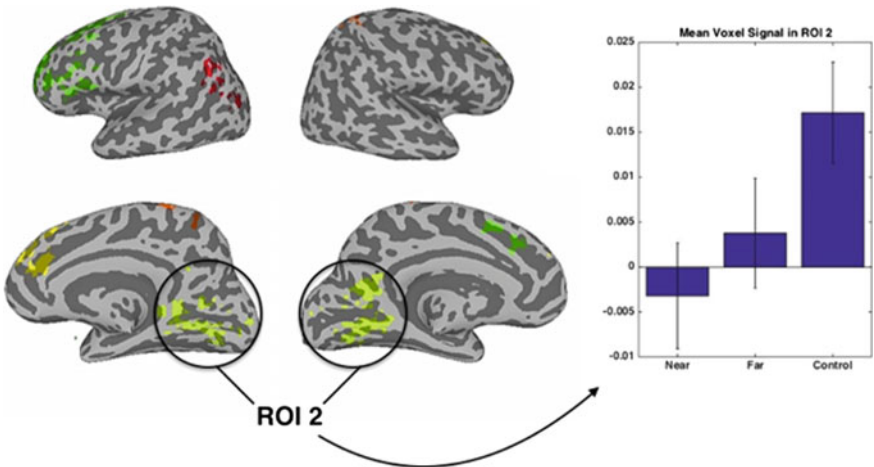


Fig. 4 Unsuccessful search ROI from modulation analysis—control condition shows highest level of activity

the design problem space for clues and insight. Prior research has also linked an increase in visual processing with participants being unable to solve problems with insight [51]. Furthermore, behavioral data from this experiment support the notion that individuals are less successful at generating ideas without inspirational stimuli.

Conclusion

The work presented in this paper used a neuroimaging experiment to investigate the neural correlates of the unsuccessful search for solutions during design problem-solving. Of particular interest were the impacts of design ideation with and without inspirational stimuli over longer time periods (minutes, compared to instances when participants generated new solution concepts). Investigating behavioral data and neural activity at this level provides insight into characteristics of unsuccessful search, which may be representative of design fixation. Inspirational stimuli at varying distances were compared against a control condition in which words were reused from the problem statement. Behavioral data gathered from participants self-reported ratings revealed that near-field inspirational stimuli are more useful and relevant compared to more distant stimuli. However, there was no significant difference in how participants rated the novelty and quality of their design solutions. Neuroimaging analyses provide insights into the mental processes during design ideation that participants are unable to verbalize. Mainly, fMRI data suggest that participants are more unsuccessful when not provided inspirational stimuli, or provided stimuli that are too distant. This leads to a specific brain activation network, which we term *unsuccessful external search*. Unsuccessful external search shows increased activation in brain regions associated with visual processing and directing attention outward. While the highest level of unsuccessful search was found in the absence of inspirational stimuli (control condition), distant stimuli show features of this search strategy. This suggests that when inspirational stimuli are too distant from the problem, participants continue to search through the external world (design problem and given words) in search of insight. Further work is needed to accurately characterize when a far inspirational stimuli (e.g., analogies) become too far and exhibit characteristics of unsuccessful external search. Taken together, this work demonstrates the effectiveness of inspirational stimuli on a neural level, opening the door for further advancements in the development of new design theory and methods.

Acknowledgements The authors would like to thank the staff at the Carnegie Mellon University Scientific Imaging and Brain Research Center for their assistance with fMRI data acquisition. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under grant DGE125252, the National Science Foundation under grant CMMI1233864, and the Air Force Office of Scientific Research under grant #FA9550-16-1-0049.

References

1. Findler NV (1981) Analogical reasoning in design processes. *Des Stud* 2(1):45–51. [https://doi.org/10.1016/0142-694X\(81\)90029-6](https://doi.org/10.1016/0142-694X(81)90029-6)
2. Chan J, Fu K, Schunn C, Cagan J, Wood K, Kotovsky K (2011) On the benefits and pitfalls of analogies for innovative design: ideation performance based on analogical distance, commonness, and modality of examples. *J Mech Des* 133(8):81004. <https://doi.org/10.1115/1.4004396>
3. Dorst K, Royakkers L (2006) The design analogy: a model for moral problem solving. *Des Stud* 27(6):633–656. <https://doi.org/10.1016/j.destud.2006.05.002>
4. Moreno DP, Hernández AA, Yang MC, Otto KN, Hölltä-Otto K, Linsey JS, ... Linden A (2014) Fundamental studies in design-by-analogy: a focus on domain-knowledge experts and applications to transactional design problems. *Des Stud* 35(3):232–272. <https://doi.org/10.1016/j.destud.2013.11.002>
5. Fu K, Chan J, Cagan J, Kotovsky K, Schunn C, Wood K (2013) The meaning of “near” and “far”: the impact of structuring design databases and the effect of distance of analogy on design output. *J Mech Des* 135(2):21007. <https://doi.org/10.1115/1.4023158>
6. Linsey JS, Wood KL, Markman AB (2008) Modality and representation in analogy. *AI EDAM* 22:85–100. <https://doi.org/10.1017/S0890060408000061>
7. Murphy J, Fu K, Otto K, Yang M, Jensen D, Wood K (2014) Function based design-by-analogy: a functional vector approach to analogical search. *J Mech Des* 136(10):1–16. <https://doi.org/10.1115/1.4028093>
8. Tseng I, Moss J, Cagan J, Kotovsky K (2008) The role of timing and analogical similarity in the stimulation of idea generation in design. *Des Stud* 29(3):203–221. <https://doi.org/10.1016/j.destud.2008.01.003>
9. Sternberg RJ (1977) Component processes in analogical reasoning. *Psychol Rev* 84(4):353–378. <https://doi.org/10.1037/0033-295X.84.4.353>
10. Nguyen TA, Zeng Y (2010) Analysis of design activities using EEG signals. In: Volume 5: 22nd international conference on design theory and methodology; special conference on mechanical vibration and noise, pp 277–286. <https://doi.org/10.1115/detc2010-28477>
11. Hu W-L, Booth JW, Reid T (2017) The relationship between design outcomes and mental states during ideation. *J Mech Des* 1–16. <https://doi.org/10.1115/1.4036131>
12. Alexiou K, Zamenopoulos T, Johnson JH, Gilbert SJ (2009) Exploring the neurological basis of design cognition using brain imaging: some preliminary results. *Des Stud* 30(6):623–647. <https://doi.org/10.1016/j.destud.2009.05.002>
13. Goucher-Lambert K, Moss J, Cagan J (2017) Inside the mind: using neuroimaging to understand moral product preference judgments involving sustainability (IDETC2016-59406). *ASME J Mech Des* 139(4):1–12. <https://doi.org/10.1115/1.4035859>
14. Sylcott B, Cagan J, Tabibnia G (2013) Understanding consumer tradeoffs between form and function through metaconjoint and cognitive neuroscience analyses. *J Mech Des* 135(10):101002. <https://doi.org/10.1115/1.4024975>
15. Goucher-Lambert K, Cagan J (2015) The impact of sustainability on consumer preference judgments of product attributes. *J Mech Des* 137(8):1–11. <https://doi.org/10.1115/1.4030271>
16. Gilbert SJ, Zamenopoulos T, Alexiou K, Johnson JH (2010) Involvement of right dorsolateral prefrontal cortex in ill-structured design cognition: an fMRI study. *Brain Res* 1312:79–88. <https://doi.org/10.1016/j.brainres.2009.11.045>
17. Saggari M, Quintin E-M, Bott NT, Kienitz E, Chien Y-H, Hong DW-C, ... Reiss AL (2016) Changes in brain activation associated with spontaneous improvisation and figural creativity after design-thinking-based training: a longitudinal fMRI Study cerebral cortex advance access. *Cerebral Cortex* 1–11. <https://doi.org/10.1093/cercor/bhw171>
18. Saggari M, Quintin E-M, Kienitz E, Bott NT, Sun Z, Hong W-C, ... Reiss AL (2015) Pictionary-based fMRI paradigm to study the neural correlates of spontaneous improvisation and figural creativity. *Sci Rep* 5(5):10894. <https://doi.org/10.1038/srep10894>

19. Gentner D (1983) Structure-mapping: a theoretical framework for analogy. *Cogn Sci* 7(2):155–170. [https://doi.org/10.1016/S0364-0213\(83\)80009-3](https://doi.org/10.1016/S0364-0213(83)80009-3)
20. Linsey JS, Markman AB, Wood KL (2008) WordTrees: a method for design-by-analogy. In: Proceedings of the 2008 ASEE Annual Conference
21. Stern PC (2000) Toward a coherent theory of environmentally significant behavior. *J Soc Issues* 56(3):407–424. <https://doi.org/10.1111/0022-4537.00175>
22. Damle A, Smith PJ (2009) Biasing cognitive processes during design: the effects of color. *Des Stud* 30(5):521–540. <https://doi.org/10.1016/j.destud.2009.01.001>
23. Cross N (2004) Expertise in design: an overview. *Des Stud* 25(5):427–441. <https://doi.org/10.1016/j.destud.2004.06.002>
24. Visser W (1996) Two functions of analogical reasoning in design: a cognitive-psychology approach. *Des Stud* 17(4 Special Issue):417–434. [https://doi.org/10.1016/s0142-694x\(96\)00020-8](https://doi.org/10.1016/s0142-694x(96)00020-8)
25. Wilson JO, Rosen D, Nelson BA, Yen J (2010) The effects of biological examples in idea generation. *Des Stud* 31(2):169–186. <https://doi.org/10.1016/j.destud.2009.10.003>
26. Jansson DG, Smith SM (1991) Design fixation. *Des Stud* 12(1):3–11. [https://doi.org/10.1016/0142-694X\(91\)90003-F](https://doi.org/10.1016/0142-694X(91)90003-F)
27. Krawczyk DC, McClelland MM, Donovan CM, Tillman GD, Maguire MJ (2010) An fMRI investigation of cognitive stages in reasoning by analogy. *Brain Res* 1342:63–73. <https://doi.org/10.1016/j.brainres.2010.04.039>
28. Cho S, Holyoak KJ, Cannon TD (2007) Analogical reasoning in working memory: resources shared among relational integration, interference resolution, and maintenance. *Memory Cogn* 35(6):1445–1455. <https://doi.org/10.3758/BF03193614>
29. Green AE, Cohen MS, Raab HA, Yedibalian CG, Gray JR (2015) Frontopolar activity and connectivity support dynamic conscious augmentation of creative state. *Hum Brain Mapp* 36(3):923–934. <https://doi.org/10.1002/hbm.22676>
30. Green AE, Fugelsang JA, Kraemer DJM, Shamosh NA, Dunbar KN (2006) Frontopolar cortex mediates abstract integration in analogy. *Brain Res* 1096(1):125–137. <https://doi.org/10.1016/j.brainres.2006.04.024>
31. Gonen-Yaacovi G, de Souza LC, Levy R, Urbanski M, Josse G, Volle E (2013) Rostral and caudal prefrontal contribution to creativity: a meta-analysis of functional imaging data. *Front Human Neurosci* 7(8):465. <https://doi.org/10.3389/fnhum.2013.00465>
32. Kowatari Y, Hee Lee S, Yamamura H, Nagamori Y, Levy P, Yamane S, Yamamoto M (2009) Neural networks involved in artistic creativity. *Hum Brain Mapp* 30(5):1678–1690. <https://doi.org/10.1002/hbm.20633>
33. Westphal AJ, Reggente N, Ito KL, Rissman J (2015) Shared and distinct contributions of rostrolateral prefrontal cortex to analogical reasoning and episodic memory retrieval. *Hum Brain Mapp* 912:896–912. <https://doi.org/10.1002/hbm.23074>
34. Wharton CM, Grafman J, Flitman SS, Hansen EK, Brauner J, Marks A, Honda M (2000) Toward neuroanatomical models of analogy: a positron emission tomography study of analogical mapping. *Cogn Psychol* 40(3):173–97. <https://doi.org/10.1006/cogp.1999.0726>
35. Goucher-Lambert K, Cagan J (2017) Using crowdsourcing to provide analogies for designer ideation in a cognitive study. In: International conference on engineering design. Vancouver, B.C., pp 1–11
36. Linsey JS, Viswanathan VK (2014) Overcoming cognitive challenges in bioinspired design and analogy. *Biologically Inspired Des* 221–244
37. Cardoso C, Badke-Schaub P (2011) The influence of different pictorial representations during idea generation. *J Creat Behav* 45(2):130–146. <https://doi.org/10.1002/j.2162-6057.2011.tb01092.x>
38. Toh CA, Miller SR (2014) The impact of example modality and physical interactions on design creativity. *J Mech Des (Trans ASME)* 136(9):[np]. <https://doi.org/10.1115/1.4027639>
39. Miller SR, Bailey BP, Kirlik A (2014) Exploring the utility of Bayesian truth serum for assessing design knowledge. *Human-Comput Interact* 29(5–6):487–515. <https://doi.org/10.1080/07370024.2013.870393>

40. Linsey JS, Markman AB, Wood KL (2012) Design by analogy: a study of the WordTree method for problem re-representation. *J Mech Des* 134(4):041009. <https://doi.org/10.1115/1.4006145>
41. Goldschmidt G, Smolkov M (2006) Variances in the impact of visual stimuli on design problem solving performance. *Des Stud* 27(5):549–569. <https://doi.org/10.1016/j.destud.2006.01.002>
42. Purcell AT, Williams P, Gero JS, Colbron B (1993) Fixation effects: do they exist in design problem solving? *Environ Plan* 20(3):333–345. <https://doi.org/10.1068/b200333>
43. Viswanathan VK, Linsey JS (2013) Design fixation and its mitigation: a study on the role of expertise. *J Mech Des* 135:51008. <https://doi.org/10.1115/1.4024123>
44. Schneider W, Eschman A, Zuccolotto A (2002) E-Prime reference guide. *Psychol Softw Tools* 3(1):1. <https://doi.org/10.1186/1756-0381-3-1>
45. Chein JM, Schneider W (2003) Designing effective fMRI experiments. *Handb Neuropsychol*, 2nd edn 9:299–325
46. Cox RW (1996) AFNI: software for analysis and visualization of functional magnetic resonance neuroimages. *Comput Biomed Res Int J* 29(3):162–173
47. Gorgolewski K, Burns CD, Madison C, Clark D, Halchenko YO, Waskom ML, Ghosh SS (2011) Nipype: a flexible, lightweight and extensible neuroimaging data processing framework in python. *Front Neuroinformatics* 5:13. <https://doi.org/10.3389/fninf.2011.00013>
48. Petersen SE, Dubis JW (2012) The mixed block/event-related design. *NeuroImage* 62(2):1177–1184. <https://doi.org/10.1016/j.neuroimage.2011.09.084>
49. Goucher-Lambert K, Moss J, Cagan J (2017) An fMRI investigation of near and far analogical reasoning during design ideation. *Des Stud* (submitted)
50. Moss J, Kotovsky K, Cagan J (2007) The influence of open goals on the acquisition of problem-relevant information. *J Exp Psychol Learn Mem Cogn* 33(5):876–891. <https://doi.org/10.1037/0278-7393.33.5.876>
51. Kounios J, Frymiare JL, Bowden EM, Fleck JI, Subramaniam K, Parrish TB, Jung-beeman M (2006) Subsequent solution by sudden insight. *Psychol Sci* 17(10):882–890
52. Leech R, Sharp DJ (2014) The role of the posterior cingulate cortex in cognition and disease. *Brain* 137(1):12–32. <https://doi.org/10.1093/brain/awt162>
53. Burgess PW, Dumontheil I, Gilbert SJ (2007) The gateway hypothesis of rostral prefrontal cortex (area 10) function. *Trends Cogn Sci* 11(7):290–298. <https://doi.org/10.1016/j.tics.2007.05.004>
54. Moss J, Kotovsky K, Cagan J (2011) The effect of incidental hints when problems are suspended before, during, or after an impasse. *J Exp Psychol Learn Mem Cogn* 37(1):140–148. <https://doi.org/10.1037/a0021206>
55. Vasconcelos LA, Crilly N (2016) Inspiration and fixation: questions, methods, findings, and challenges. *Des Stud* 42:1–32. <https://doi.org/10.1016/j.destud.2015.11.001>
56. Goucher-Lambert K, Moss J, Cagan J (2016) A meta-analytic approach for uncovering neural activation patterns of sustainable product preference decisions. In: *Design computing and cognition conference 2016*, pp 1–20

Designing with and for the Crowd: A Cognitive Study of Design Processes in NatureNet



Stephen MacNeil, Sarah Abdellahi, Mary Lou Maher,
Jin Goog Kim, Mohammad Mahzoon and Kazjon Grace

NatureNet is a citizen science project that, in addition to collecting biodiversity data, invites end-users to contribute design ideas to guide the its future design and development. This paper presents the NatureNet model of crowdsourcing design, then compares an analysis of the design process to published analyses of traditional face-to-face design processes. The protocol analysis approach is used to segment and code the design ideas submitted to NatureNet. We use the Function-Behavior-Structure ontology as a basis for comparison across the crowdsourced design data and design data collected in face–face sessions. The primary finding of this paper is that crowdsourced design results in a different distribution of cognitive effort when compared to traditional design processes.

Introduction

NatureNet is a citizen science project that encourages non-designers to contribute to its continual redesign. Society is now facing design challenges on a much larger scale as we become increasingly global and technological. Design solutions must not only respond to the needs and desires of their users, but must also be environmentally sustainable, attractive to multiple cultures, adaptable as technology changes, and intuitive to potential users. Tim Brown from IDEO proposes that designers cannot meet all of these challenges alone. The amount of problems to which design might contribute far exceeds the number of designers in the world, despite the continued

S. MacNeil (✉) · S. Abdellahi · M. L. Maher · J. G. Kim · M. Mahzoon
The University of North Carolina at Charlotte, Charlotte, USA
e-mail: smacnei2@uncc.edu

K. Grace
The University of Sydney, Sydney, Australia

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_4

best efforts of design schools. There is a need to rethink design, and one avenue is democratization: extending the capability and responsibility of design to all [1].

Crowdsourcing is a term used to describe situations where an open call is made to a large number of self-selected individuals to provide input to a process [1, 2]. Often associated with micro-tasks, such as labeling images or transcribing audio recordings, crowdsourcing may also help designers leverage greater cognitive diversity, among other potential benefits for the design process [3]. For instance, crowd-collaborative innovation platforms such as Quirky.com and OpenIDEO.com provide opportunities for expert and non-expert designers to contribute to design.

These crowd-collaborative innovation platforms and crowdsourced design in general have received more attention over the past years [4], but the majority of this work has focused on integrating the crowd into existing design processes or on modifying the design process and its organizational structures to improve the quality or heterogeneity of the design artifacts [5–9]. This has resulted in a variety of crowdsourced design models which often rely on either selecting the best design out of many competing designs or iterating between phases of design generation and design evaluation [10]. Consequently, few of the crowdsourcing design models afford opportunities for individual crowd members to be involved in all aspects of a single design.

This prior work provides a strong practical foundation for accomplishing crowdsourced design; however, they do not include cognitive studies of how the crowd designs. The design process that arises from the crowd is surely different from that of traditional design teams, but exactly how has yet to be explored. The question of how best to utilize and optimize crowdsourced design can benefit from an evidence-based understanding of it. In this paper, we describe the NatureNet model for crowdsourced design, present a protocol analysis of the design process followed by NatureNet's users, and compare that process to cognitive studies of small, co-located design teams.

NatureNet: Crowdsourcing Science and Design Contributions

NatureNet is a citizen science system designed for collecting biodiversity data from parks, creeks, backyards, and other natural settings (<https://www.nature-net.org>). Users are encouraged to participate in the design of the system in addition to collecting data about the environment at the park [11]. NatureNet is developed as a platform spanning both desktop and mobile devices, including a website, iOS app, and an Android app. Supporting these systems allows people to contribute regardless of the devices that they have available. The website and mobile apps provide four major functions (see Fig. 1).

Explore: map-based observations, this screen displays observations posted by users (e.g., a mushroom photo).

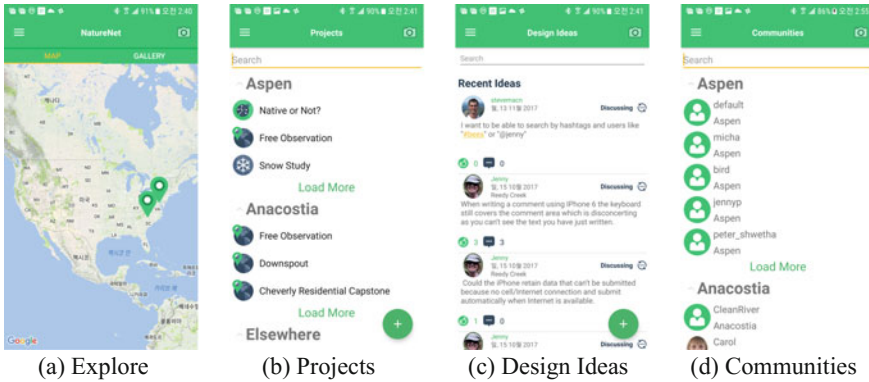


Fig. 1 The four major functions of NatureNet as shown on the android mobile app

Projects: project list, this screen provides a project list and each project displays the project description with observations (e.g., pond water project).

Design Ideas: design idea list, this screen display design ideas posted by users (e.g. an idea about adding hashtags).

Communities: user and group lists, this screen provides a list of users and groups with their contributions (e.g., “Reedy Creek”, a local nature center).

NatureNet as a Citizen Science Platform

As a citizen science platform, NatureNet allows users to contribute to ongoing environmental projects. To contribute to the projects, users take a mobile device into a park or other natural environment to gather photographs and notes. A submission or observation can include a photo, location, project, and comments with information such as water temperature, air temperature, and water pH. Alternatively, the user can submit a pdf with more detailed project information.

As an example of a NatureNet citizen science usage, a user may find the project “Planting for Pollinators” in the app, as shown in Fig. 2. The description of this project asks contributors to take a photo of a plant or pollinator. They then take a photo of a flower with a hummingbird pollinating it, add a description and then submit it. Their observation is shown on a map of submissions to the project, with the location detected automatically from the mobile device. Other users can engage with this new observation from their own apps or on a desktop by commenting on or liking it. Over time the project will elicit many contributions, allowing the scientists to get a better sense of the distribution of pollinators and plants.

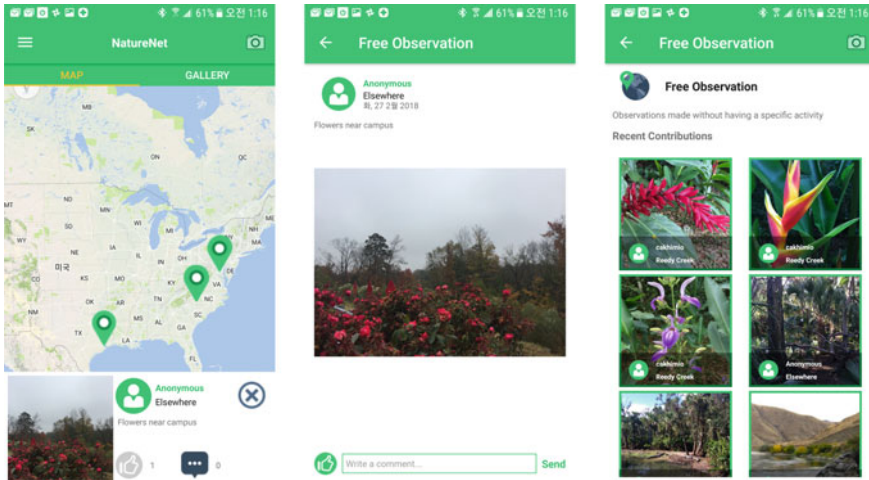


Fig. 2 Observations displayed on the android mobile app: explore (left), a single observation (middle), and a project (right)

NatureNet as a Crowdsourced Design Platform

As a crowdsourced innovation platform, NatureNet allows end-users to shape the design and development of NatureNet. End-users contribute through “design ideas”, which they can submit when they identify an issue with the existing design, have suggestions for a new feature, or want to do something new that is currently not supported. Users can also contribute by commenting and voting on existing design ideas. While these ideas can include new workflows and ways of using the platform, many contributions are based on the users’ direct experience with the visual design, information design, and/or interaction design of the platform.

To draw an example from our data, one user submitted the idea “I suggest allowing users to upload data from low-cost environmental sensors (Water quality, air quality).” This idea was submitted to the Design Ideas page as a “new feature.” This idea might require a significant change to the platform but also changes the types of projects that can be carried out. Other ideas, such as “Can I take a picture of a plant from multiple angles [as] part of one single observation”, might make the user experience more seamless but not fundamentally change what NatureNet does. Once these ideas are submitted, users can comment or vote on the idea. NatureNet team members can also comment on the idea. The interface for submitting and reviewing design ideas can be seen in Fig. 3.

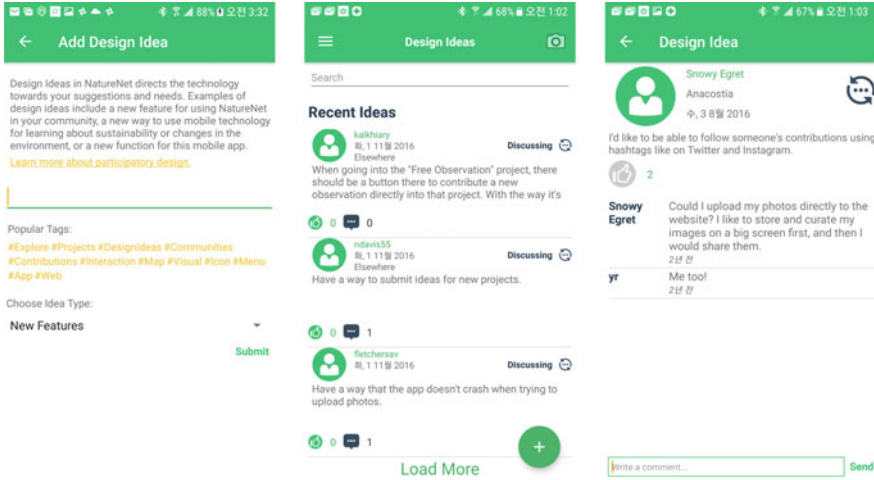


Fig. 3 Design ideas as shown on the NatureNet Android app: Adding a design idea (left), listing existing design ideas (middle), and viewing details of an existing design idea (right)

Crowdsourced Design in NatureNet

Crowdsourcing design is a way to quickly get a variety of design ideas. This process has been used with varying success. For instance, there are examples of the crowd outperforming experts [12], and examples of the crowd proposing obvious and redundant ideas [13]. Consequently, there are many techniques for boosting the creativity of the crowd such as combining crowd ideas [14]. Another technique for improving the quality of crowdsourced ideas is to guide design ideas by giving the crowd feedback about their contributions or explicit training [15]. Crowds generate many ideas, and given their potential for similarity [13], many crowdsourcing ideation platforms implement affordances for voting and for filtering ideas [16]. Through these many techniques, crowdsourced ideas can provide companies with many designs that have reasonable quality at relatively cheap prices. This balance between cost and quality is often made as an argument for and against crowdsourcing. In NatureNet, the crowd is not compensated monetarily but instead through the satisfaction of seeing their ideas implemented, through community reputation, and a voting system. This lessens the need for quality control mechanisms that are typically associated with crowd-work such as test questions.

In NatureNet, users can post ideas, comment on ideas to improve them or modify them, and vote. Throughout this process, the design and development team (referred from here on as the NatureNet team) tracks new ideas, discusses the ideas, provides feedback about feasibility. The NatureNet team has been made up of one designer, one design idea moderator, and one to three developers. As the NatureNet team integrates users' ideas they often combine multiple related ideas or create features that solve more than one problem, to address more design ideas, and save time.

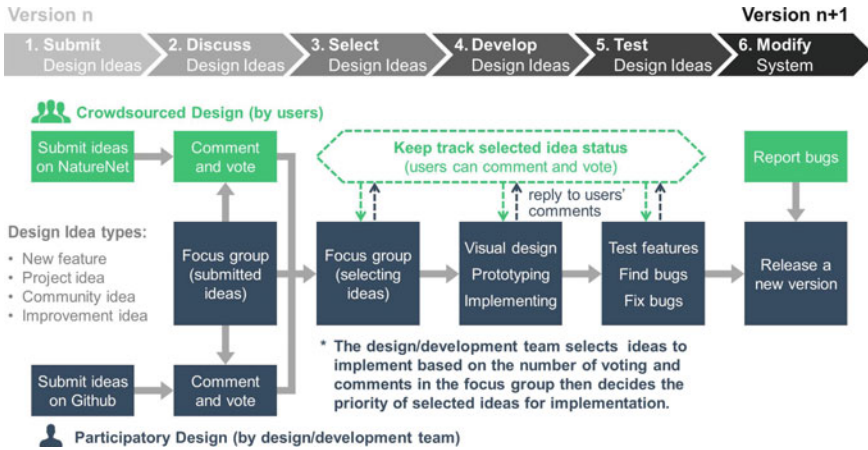


Fig. 4 Design process model shows how designs are contributed and implemented

This creative combination may lead to new and interesting solutions that are different but related to the individual ideas proposed by end-users.

The role of the NatureNet team is to manage the process, evaluate comments and contributions, and then implement the crowd’s designs. We conceptualize this as a cyclical process of ideation, discussion, and implementation, one cycle of which leads to a new version of the design. This process allows for the synthesis of crowdsourced perspectives and the selection of strong ideas by the NatureNet team for integration into the next version [17]. A design idea can have any of the following status labels, indicating its progress

- *Discussing*: submitted ideas (this is the initial status of each idea)
- *Developing*: selected ideas for the implementation
- *Testing*: implemented ideas before release
- *Done*: implemented and released ideas,

Figure 4 illustrates the transition from submitting a design idea to implementing a design idea in NatureNet. The crowd generates and contributes new design ideas in the current iteration of the design. The new design ideas can be tagged by the crowd or the NatureNet team for categorization using hashtags. In addition, the crowd can choose from a dropdown of the following idea categories when submitting their idea: *new feature*, *project idea*, *community idea*, and *improvement*. Design ideas are searchable by content and tag, which helps the NatureNet team track categories of design ideas over time.

The NatureNet team oversees the progression of these ideas through the process, communicating with the crowd throughout. To select an idea, the NatureNet team considers several aspects such as the idea type, number of likes, number of comments, difficulty of implementation, and priority. Once a design idea is selected for implementation, the NatureNet team develops prototypes, and evaluates them by

communicating with the crowd. Once a prototype is implemented, the new version gets feedback from the crowd and this process iterates to create subsequent versions.

This process leads to a model of design that is very different from traditional co-located small-team design. The crowdsourced design model adopted by NatureNet is different from traditional design methods in the following four ways:

Communication (direct, indirect): In a traditional design meeting, designers communicate with each other directly. In crowdsourced design, participants communicate indirectly, typically through a discussion forum.

Synchronicity (synchronous, asynchronous): The discussions in a traditional design meeting occur at the same time. Contributing design ideas in a crowdsourced design is asynchronous: they occur at any time.

Proficiency (qualified, unqualified): In a traditional design session, the participants are selected based on their qualifications for the design problem. In crowdsourced design, the participants need not be and typically are not qualified in the area of the design problem.

Optionality (required, volunteer): In a traditional design session, the participants are required to contribute to the design process. In crowdsourced design, the participants volunteer to participate.

A Protocol Study of Crowdsourced Design in NatureNet

In this section, we describe a protocol study of the NatureNet crowdsourced design process. This study investigates the cognitive processes adopted by the crowd designers, and explores how the four differences above impacted the distribution of cognitive activities in the design process. Protocols are records of designers' communications, usually verbal but increasingly text-based where online technology is involved. Protocols can be segmented to enable analysis, and this segmentation is based on a coding scheme. Function-Behavior-Structure (FBS) [19] and Level of Abstraction [18] are examples of coding schemes for design protocol analysis.

The FBS ontology provides a set of categories of cognitive issues that can be used to characterize what designers are thinking about during the design process [19]. In a typical design process, designing an artifact involves a series of elementary steps which transform, first, the desired "function" of the artifact into its "expected behavior"; then the expected behavior into a "structure" intended to enable the artifact to exhibit the expected behavior. After further steps of analyzing the structure for its "actual behavior" the structure is finally transformed into a design description from which an artifact may be produced. Reasoning about the function (F) refers to the manner in which the design object fulfills its purpose, i.e., the designer is working with the functional aspects of the problem domain. Reasoning about the behavior (B) concerns the description of the object's action or process in given circumstances and often deals with a response to some user action, i.e., the designer is concerned with the behavioral aspects of the problem domain. Behavior is either derived (Bs) or expected (Be) from the structure. Reasoning about structure (S) involves the consideration of

visual and conceptual elements, such as “Pond water”, search-bar, or mobile app. In addition to these main categories, requirements (R) represent intentions from the client that come from outside the designer. The FBS coding scheme categorizes the designer’s behavior based on his/her concentration of Function, Structure, or Behavior at each stage of the design [19].

The FBS coding scheme provides a standardized vocabulary which applies to design activity regardless of context. Gero states that “foundations of designing” are independent of the designer, their situation, and what is being design. Accordingly, all designs could be represented in a comparable way, as could all records of designing [20]. The FBS framework makes these uniform representations possible. As such, FBS allows us to compare synchronous and asynchronous protocols. It allows us to analyze crowdsourced and non-crowdsourced designs using the same coding scheme.

In traditional design, there are commonalities of process that can be observed regardless of the design context. Gero et al. [21] surveyed a set of thirteen design sessions and observed that the design issues that the designers are thinking about as they are designing, as coded by FBS had many similarities despite design contexts. They also present “empirical evidence of commonalities across designing independent of the designers’ geographical location, expertise, discipline, the specific design task, the size and composition of the design team, and the length of the design session” [22].

In the FBS coding scheme, issues can be categorized into problem-focused and solution-focused. Problem-focused issues are Functions (F), Requirements (R), and Expected Behaviors (Be). Solution-focused issues are coded as Structure (S) or Behavior derived from structure (Bs). P-S index is a concentration indicator defined by Gero et al. [23]. It is calculated as the ratio of number of issues in the problem space divided by the number of issues in the solution space.

$$\text{P-S index} = \frac{\sum (\text{F, R, Be})}{\sum (\text{S, Bs})}$$

A design session with a P-S index larger than 1 is a problem-focused designing style, and a session with a P-S index value less than or equal to 1 is a session with solution-focused style. We have used the P-S index to understand whether the crowd moves from a concentration on problem-focused design to solution-focused design as the usability of the NatureNet platform improves.

There have been several design studies looking at different aspects of design from creativity to technique and discipline based on FBS ontology [23–25]. Hence, basing our study on FBS in addition to providing the possibility of achieving comparable results regardless of the design topics makes the output cognitively comparable with other more common forms of design.

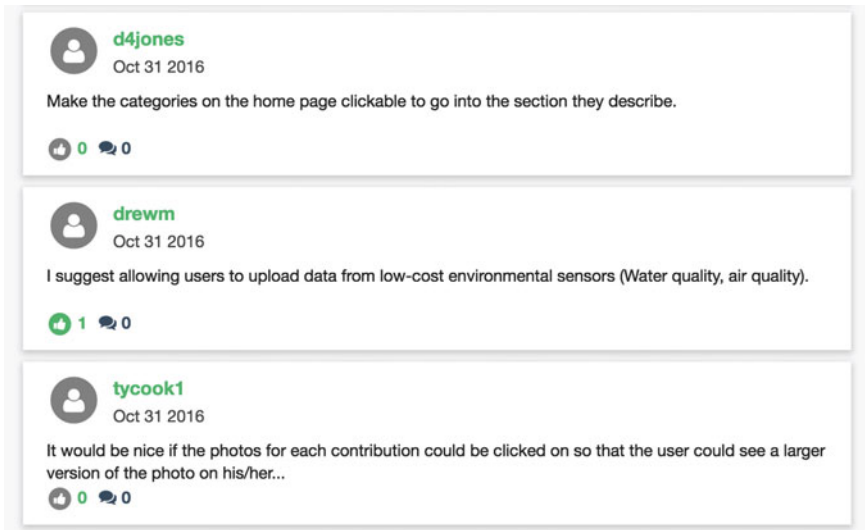


Fig. 5 A screenshot of some of the design ideas that were posted by NatureNet users (from www.nature-net.org)

Design Data Collected from NatureNet

We include 183 design ideas that were submitted to NatureNet by 74 different contributors in our analysis. These design ideas were collected from the NatureNet website, mobile apps, and the tabletop version of NatureNet. We did not include data about voting and comments in the corpus that we analyzed. These 183 ideas were collected between April 24, 2016 and October 5, 2017. All ideas are included in the analysis, regardless of whether they were chosen for implementation.

Users could see and interact with all the ideas that were previously contributed by other users. Figure 5 shows several design ideas captured from the web interface to demonstrate what end-users would see as they contribute new ideas. Ideas were often repeated because end-users did not search through previously submitted design ideas to see if their idea had already been proposed. This is consistent with previous work; BlueSky shows that crowdsourced ideas often repeat concepts and do not typically cover the design space [13].

Hypotheses

Considering different characteristics of NatureNet design model compared to traditional design processes, we hypothesize that the following patterns will be observed in the crowd data:

- Crowdsourced design ideas from end-users will have a stronger focus on Function than traditional design contexts because end-users have specific functional needs that are not provided in the current implementation.
- Crowdsourced design ideas will not focus as much on expected behavior because users are not experts in user experience design and lack an appreciation of expected behaviors.
- Similar to traditional design, we expect crowdsourcing design ideas to have a strong focus on structural aspects of the design. Previous studies of design activities show that design teams tend to have a larger percentage of time or issues related to the structure of the design when compared to function or behavior of the design [23, 26, 27].
- We expect NatureNet to oscillate between focusing on the problem and on the solution. Problems inspire solutions and new solutions may lead to new problems if the solution is incomplete or ineffective. In design, it is common for the problem and solution to co-evolve [28].
- The distribution of contributions among participants will be similar to behavior in online communities. In traditional design sessions, one person can “hold the floor”, thereby dominating the design session. We expect to see a small portion of the crowd exhibit similar behavior by submitting a substantial fraction of the ideas.

Analysis

To analyze the results, we used three coders in a group coding session. Before group coding, two coders coded independently to calibrate their codes. Their agreement as measured by Cohen’s Kappa was strong (0.62). An analysis of these codes show aspects of the crowd design data that was not present in traditional synchronous verbal protocols of designers and design teams. The crowd participants were often unaware of design ideas that preceded their own. This led to a lot of repetition and re-hashing of the most common design suggestions. We even observed users repeating their own ideas in cases where some time had passed. This informed the way that we coded segments in design ideas, leading us to treat each design idea as being independent of all other design ideas. This is different from traditional design contexts where each team-member can be considered to be aware of what other team members or doing and saying.

Results

To provide context for our results, we compare them with empirical results [22, 23] and a review of prior studies that have analyzed traditional design sessions [21]. To do this, we explored our results in three different ways: (1) the distribution of

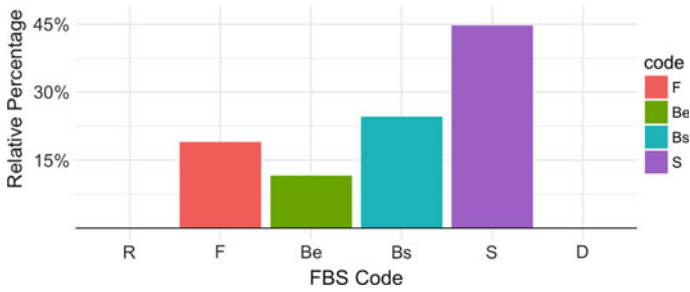


Fig. 6 The distribution of FBS codes across all of the design ideas received in the NatureNet platform. See Fig. 7 for a comparison with traditional design

FBS codes, (2) patterns in how users contributed ideas, and (3) temporal trends in the problem–solution index. We compared crowdsourced design to traditional design sessions in each of these aspects, providing insight into the differences and similarities between the two design contexts.

Distribution of FBS Codes in Design Ideas

The distribution of design issues is shown in Fig. 6. This distribution appears to largely replicate the results of [22], in which three design sessions each employing a different concept generation technique were coded, shown in Fig. 7. As shown in Fig. 7, Structure (S) is most common, followed by Analyzed Behavior (Bs), Expected Behavior (Be), and Function (F); in that order. Our data contains did not contain any requirements (R) or descriptions (D), so we omit those from our comparison.

The major difference between the crowd and the traditional design contexts is the frequency of Function (F), which was more common than Be and almost as common as Bs. We might expect that the lack of D in our dataset would cause all the other codes to increase proportionately, but the increase has gone almost entirely to the S and F categories. The higher occurrence of F is mostly likely either related to the way design ideas were obtained, or the fact that the data is from end-users rather than designers and therefore tend towards suggestions for new functions.

Obtaining design ideas asynchronously means that each idea occurs independently of the ideas that appear before it. As a result, there may be less shared context to reference when suggesting new ideas, and thus less “progress” from F to B to S. Instead we found that the crowd’s new ideas do not directly reference previously posted ideas. Since our crowd consists of primarily end-users, who may be motivated to submit design ideas based on immediate problems stemming from their use of the platform. Unlike traditional design processes where new ideas are considered within the broader scope of the design, end-users often suggest ideas independently. This

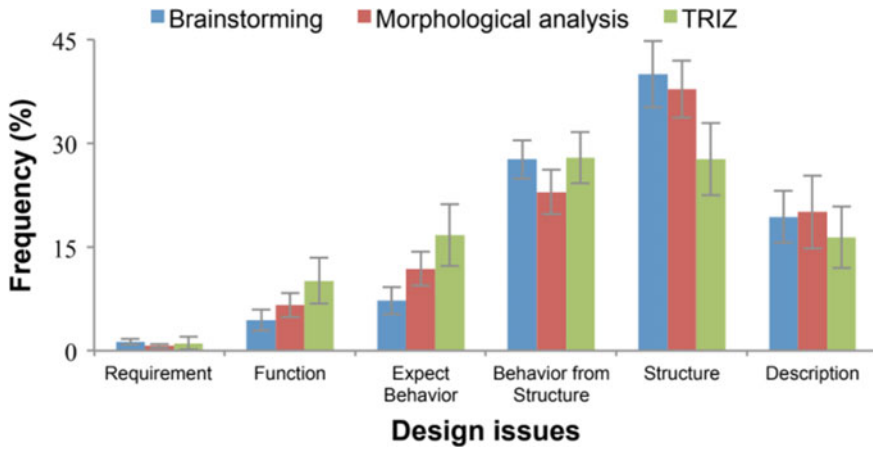


Fig. 7 The distribution of design issues (FBS codes) using three different concept generation techniques [22]. Error bars indicate the variation along each aspect between multiple design sessions in the review

might reflect a focus on immediate need with little additional context and without considering the larger scope of NatureNet.

We also explored potential relationships between the design issues that appear in each design idea. Correlations were computed using Pearson's Chi Squared Test. Significance was determined by applying a t-test to the individual correlations and Holm correction was used to correct for multiple comparisons. Three correlations were minor but significant: F-S (-0.21), F-Bs (-0.17), and S-Bs (0.31). F likely has a negative correlation with other design issues because it regularly appears alone in short design ideas, often those that describe a desired feature in a single sentence. Some of these ideas did not provide sufficient detail for the NatureNet team to understand the user's intent. This may stem from the fact that end-users are not designers. In contrast, ideas that discussed both Bs and S tended to be analyses of current features. For example, one idea described a new feature but did not indicate whether it corresponded to the mobile or web version of NatureNet. More scaffolding could help non-expert designers be aware of what context is relevant, but this presents a difficult interaction design problem: users often do not notice or ignore scaffolding. For example, most contributors did not use the provided dropdown list to indicate if the idea was a new feature, an improvement, a new community, or a new project.

Given both observations, we can see some initial evidence that F might take on a more significant role in crowdsourced design where end-users make up a significant portion of the crowd.

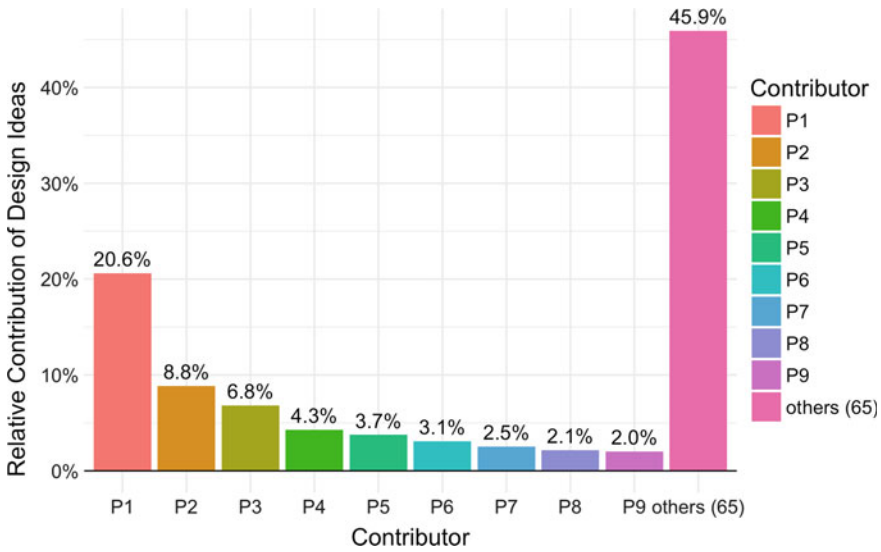
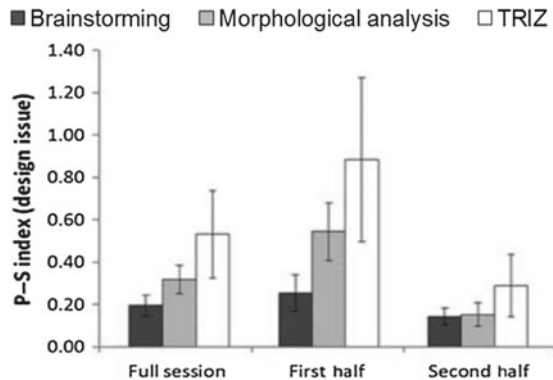


Fig. 8 Distribution of design ideas among all users. The number one contributor of design ideas accounted for 20% of the design ideas received. The top ten contributors, taken together, accounted for more than 50% of ideas

Fig. 9 An aggregated view of the design sessions from Gero et al.’s review of designing using three different concept generation techniques [23]



Relative User Contribution Frequencies

In traditional design settings, one designer can dominate a conversation by not providing opportunities for other designers to contribute. This happens when one designer talks longer and more frequently or when they interrupt others. This behavior may in turn demotivate and disincentivize participation by others. In asynchronous design contexts interruption is not possible and the conversation is parallel and distributed. Regardless, we still observed analogous conversational dominance, as some users contributed disproportionate numbers of ideas, shown in Fig. 8.

We analyzed the sequence of submitted design ideas for “streaks”: patterns of submissions by a single user with no interruptions by others. We also observed many cases where a large number of ideas were contributed by one user but with brief interjections from other users. In this analysis, we observed that there were 17 “2-streaks” (pairs of ideas submitted by a single user), 6 “3-streaks” (trios of consecutive submissions), one 4-streak, five 5-streaks, and an impressive 11-streak. Submitting as much as 6% of our data in a row with no interruptions can be considered equivalent to a designer “holding the floor”. Our interface privileges recently submitted design ideas, meaning that after a long streak all immediately visible design ideas will come from a single user.

Participation online is known to be uneven, with “superposters”, users who post much more than others, often making up a substantial portion of total posts in a discussion forum [29]. We speculate that, like in discussion forum contexts, superposting and sequential posting may be discouraging for new NatureNet users who have not yet contributed. Investigating superposter behavior and how it is interpreted by other contributors is an interesting area for future research, as is designing systems to encourage equitable participation in crowdsourced design.

P-S Index and Temporal Trends

We plotted the P-S Index for the first and second halves of our “design session”, shown in Fig. 10. The problem–solution index remained relatively consistent throughout the design session. This consistency is most similar to traditional brainstorming sessions which also have little variation over time, as shown in Fig. 9. In comparison, the analysis of the TRIZ and Morphological sessions shows that the first half of the session is more problem-focused while the second half is solution-focused. NatureNet more strongly resembles unstructured brainstorming, with new problems and solutions emerging continually. This does not imply that individual users of NatureNet are more balanced throughout their design sessions. We can see ideas as micro-design sessions and the unit of analysis here is the aggregate of those sessions. This analyzed macro-session was balanced over time. Ideas do not often build on previous ideas; instead, they are generated independently, and are often not immediately evaluated.

Discussion

In this paper, we compare crowdsourced design and traditional design by analyzing the design ideas that were contributed to the NatureNet citizen science platform. Crowdsourced design is different to traditional design sessions in synchronicity, temporality, expertise, and communication modality. These factors affect the way that end-users contribute to the design process. The analysis of the crowdsourced design ideas featured three main aspects: the FBS coding of ideas, user ideation

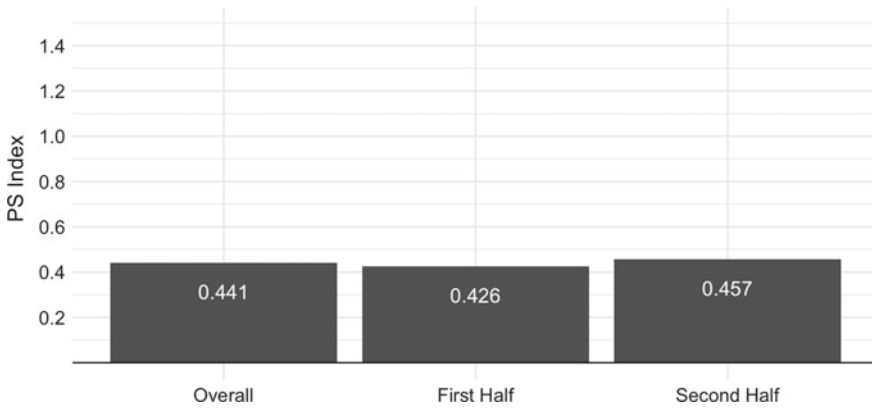


Fig. 10 First and second halves of the NatureNet “design session”. The ideas were ordered chronologically and then separated into halves to compare Fig. 9

behaviors, and temporal trends. These aspects were compared with more traditional design sessions (i.e., small, co-located design teams) based the review of design protocol studies in Gero et al. [21]. The differences between crowdsourced and traditional design include

- Participants in crowdsourced design post more ideas that contain references to functions than participants in traditional design.
- In crowdsourced design ideas containing function are often presented without much context, instead focusing on a single new feature.
- Crowdsourced design involved end-users in the design process and end-users do not usually know what information to include in design ideas for a design team to make sense of the idea, which suggests that scaffolding to obtain context from users must be carefully designed.
- In crowdsourced design attention does not oscillate between problems and solutions over time, instead problems and solutions are proposed continually throughout.
- In crowdsourced design a small portion of users post disproportionately and often in rapid succession.

One possible explanation for a significant number of ideas coded as Function is that many end-users have specific things that they want to be able to do with NatureNet. This would help explain why proposed new Functions frequently appeared by themselves as short idea such as “I want to be able to post audio clips.”

In another departure from traditional design, Expected Behavior (Be) appeared less frequently. We hypothesized that Be would be less frequent in crowdsourced design because new functions and structures could be explained in terms of existing behaviors, structures, and functions. On the few occasions that end-users described expected behaviors they did so from their own personal perspective, often without considering how it might affect others.

Due to these findings, future developments in crowdsourced design can include scaffolding to help users to think more broadly about their design ideas and to include more relevant information. In our current implementation, scaffolding has had mixed success. Suggesting hashtags does appear to increase their usage, but on the other hand our dropdown list of design categories was mostly ignored. It is difficult to conclude from this single study whether users' reticence to indicate the category of their submission reflects the behavior of the crowd or a usability issue.

Design ideas in NatureNet were more focused on solutions than problems. This focus did not vary much over time, which contrasts with the transition from problem-focused to solution-focused thinking observed in most design activities. An exception to this is brainstorming, which also tends to focus on solutions throughout the design session. Our crowdsourced design appears to be similar to brainstorming in this respect, although the balance between problems and solutions is different.

We observed that a small portion of users were responsible for a large proportion of design activity. While these behaviors were not meant to discourage others or dominate the design, their behaviors may have that effect on other users. If we refine the design of NatureNet based on the design ideas of the crowd, and 50% of the crowd's design activity is made up by just a few contributors, then we are designing NatureNet for those few people. The intent of crowdsourcing is to democratize participation, but this would more strongly resemble an oligarchy. Understanding how ideation patterns affect other contributors is an area for future research. End-users are a community and so the interactions between members and aspects such as equity and inclusion need be considered.

In summary, users' ideas are rooted in their experiences and they express solutions from their own perspective without considering the broader context in which the solution exists. This is to be expected since they are not designers. Furthermore, users did not appear to be aware of the ideas of others, often posting duplicate or highly similar solutions. Like brainstorming, the focus on problems and solutions did not change much over time, but with a slightly higher emphasis on solutions compared to brainstorming.

Limitations

We have presented the results from an extended crowdsourced design session; however, the results may not be representative of all crowdsourced design contexts. Crowdsourced design often integrates the crowd at discrete points in time such as during idea generation, evaluation, or modification. In those contexts, the crowd is may not be aware of the overarching design problem, goals, and solution. In our context, end-users have the option to be involved throughout most parts of the design process. They can suggest an idea, discuss the underlying problem, and actively develop solutions with other users and the design team. Although these things are possible, we observed that users did not always receive feedback, they often were not aware of other similar ideas, and they may not share a collective vision for the design

of NatureNet. Our notion of design is more similar to traditional design settings than discrete micro-task-based crowdsourced design. Our results compare the crowd to a design team, even though individual cognitive processes may differ. Finally, while design fixation is possible in any design setting, our crowd consists of end-users that are familiar with the existing system and its features. It is likely that this led to some amount of design fixation.

Conclusion

In this paper, we explore the design ideas that were submitted to the NatureNet apps by its end-users. These crowdsourced ideas are intended to improve the NatureNet platform, and can be considered a multitude of “micro design sessions”. We analyzed 183 design ideas and compared their content, distribution, and temporal trends with those of traditional design settings.

We have found that functions appear more frequently in this context, that problems and solutions appear consistently throughout the design lifecycle, and that a few users produce a large percentage of ideas. These findings have implications for crowdsourced ideation platforms and for systems that accept design ideas from their end-users. Further research includes: evaluate the quality, diversity, and creativity of the design ideas, and to develop affordances for a broader range of end-users to contribute and to consider the ideas of others on the platform.

References

1. Howe J (2006) The rise of crowdsourcing. *Wired Magaz* 14(6):1–4
2. Maher ML, Paulini M, Murty P (2011) Scaling up: from individual design to collaborative design to collective design. In: *Design computing and cognition '10*. Springer Netherlands, pp 581–599
3. Ponsonby AL, Mattingly K (2015) Evaluating new ways of working collectively in science with a focus on crowdsourcing. *EBioMedicine* 2(7):627–628
4. Kittur A (2010) Crowdsourcing, collaboration and creativity. *ACM Crossroads* 17(2):22–26
5. Chan J, Dow S, Schunn C (2014, Feb) Conceptual distance matters when building on others’ ideas in crowd-collaborative innovation platforms. In: *Proceedings of the companion publication of the 17th ACM conference on computer supported cooperative work & social computing*. ACM, pp 141–144
6. Chan J, Dang S, Dow SP (2016, Feb) Improving crowd innovation with expert facilitation. In: *Proceedings of the 19th ACM conference on computer-supported cooperative work & social computing*. ACM, pp 1223–1235
7. Xu A, Huang SW, Bailey B (2014, Feb) Voyant: generating structured feedback on visual designs using a crowd of non-experts. In: *Proceedings of the 17th ACM conference on computer supported cooperative work & social computing*. ACM, pp 1433–1444
8. Valentine MA, Retelny D, To A, Rahmati N, Doshi T, Bernstein MS (2017, May) Flash organizations: crowdsourcing complex work by structuring crowds as organizations. In: *Proceedings of the 2017 CHI conference on human factors in computing systems*. ACM, pp 3523–3537

9. Lasecki WS, Kim J, Rafter N, Sen O, Bigham JP, Bernstein MS (2015) Apparition: crowd-sourced user interfaces that come to life as you sketch them. In: Proceedings of the 33rd annual ACM conference on human factors in computing systems. ACM, pp 1925–1934
10. Wu H, Corney J, Grant M (2014) Crowdsourcing measures of design quality. In: ASME 2014 international design engineering technical conferences and computers and information in engineering. ASME
11. Maher ML, Preece J, Yeh T, Boston C, Grace K, Pasupuleti A, Stangle A (2014, Feb) NatureNet: a model for crowdsourcing the design of citizen science systems. In: Proceedings of the companion publication of the 17th ACM conference on computer supported cooperative work & social computing. ACM, pp 201–204
12. Poetz MK, Schreier M (2012) The value of crowdsourcing: can users really compete with professionals in generating new product ideas? *J Prod Innov Manag* 29(2):245–256
13. Huang G, Quinn AJ (2017, June) BlueSky: crowd-powered uniform sampling of idea spaces. In: Proceedings of the 2017 ACM SIGCHI conference on creativity and cognition. ACM, pp 119–130
14. Yu L, Nickerson JV (2011, May) Cooks or cobblers?: crowd creativity through combination. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, pp 1393–1402
15. Dontcheva M, Morris RR, Brandt JR, Gerber EM (2014, April) Combining crowdsourcing and learning to improve engagement and performance. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, pp 3379–3388
16. Ahmed F, Fuge M (2017) Capturing winning ideas in online design communities. In: Proceedings of the 17th ACM conference on computer supported cooperative work and social computing. ACM, pp 1675–1687
17. Grace K, Maher ML, Preece J, Yeh T, Stangle A, Boston C (2015) A process model for crowdsourcing design: a case study in citizen science. In: Design computing and cognition '14. Springer, pp 245–262
18. Purcell T, Gero J, Edwards H, McNeill T (1996) The data in design protocols: the issue of data coding, data analysis in the development of models of the design process. *Cross et al* 153:151–168
19. Gero JS (1990) Design prototypes: a knowledge representation schema for design. *AI Magaz* 11(4):26
20. Gero JS, Kannengiesser U (2014) The function-behavior-structure ontology of design. In: An anthology of theories and models of design. Springer, London, pp 263–283
21. Gero JS, Kannengiesser U, Williams C (2014) Does designing have a common cognitive behavior independent of domain and task: a meta-analysis of design protocols. In: International conference on human behavior in design
22. Gero JS, Kannengiesser U, Pourmohamadi M (2014) Commonalities across designing: empirical results. In: Design computing and cognition '12. Springer, Dordrecht, pp 265–281
23. Gero JS, Jiang H, Williams CB (2013) Design cognition differences when using unstructured, partially structured, and structured concept generation creativity techniques. *Int J Des Creat Innov* 1(4):196–214
24. Jiang H, Gero JS, Yen CC (2014) Exploring designing styles using a problem–solution division. In: Design computing and cognition '12. Springer, Dordrecht, pp 79–94
25. Kan, J. W., & Gero, J. S. (2009). Using the FBS ontology to capture semantic design information in design protocol studies. In *About: Designing. Analysing Design Meetings* (pp. 213–229). CRC Press
26. Yu R, Gero J, Gu N (2013, July) Impact of using rule algorithms on designers' behavior in a parametric design environment: preliminary result from a pilot study. In: International conference on computer-aided architectural design futures. Springer, Berlin, Heidelberg, pp 13–22
27. Williams CB, Lee Y, Gero J, Paretto MC (2013, Aug) Exploring the effects of the design prompt on students' design cognition. In: ASME 2013 International design engineering technical conferences and computers and information in engineering conference. ASME

28. Dorst K, Cross N (2001) Creativity in the design process: co-evolution of problem–solution. *Des Stud* 22(5):425–437
29. Huang J, Dasgupta A, Ghosh A, Manning J, Sanders M (2014, Mar) Superposter behavior in MOOC forums. In: Proceedings of the first ACM conference on Learning@ scale conference. ACM, pp 117–126

A Comparison of Tree Search Methods for Graph Topology Design Problems



Ada-Rhodes Short, Bryony L. DuPont and Matthew I. Campbell

In this paper, we discuss the relevance and effectiveness of two common methods for searching decision trees that represent design problems. When design problems are encoded in decision trees they are often multimodal, capture a range of complexity in valid solutions, and have distinguishable internal locations. We propose the use of a simple Color Graph problem to represent these characteristics. The two methods evaluated are a genetic algorithm and a Monte Carlo tree search. Using the Color Graph problem, it is demonstrated that a genetic algorithm can perform exceptionally well on such unbounded and opaque design decision trees and that Monte Carlo tree searches are ineffective. Insights from this experiment are used to draw conclusions about the nature of design problems stored in decision trees and the need for new methods to search such trees and lead us to believe that exploitative methods are more effective than rigorously explorative methods.

Introduction

AI tree searches are a common method for the creation of generative designs. This requires the problem to first be represented as a decision tree. Problems represented as decision trees have a benefit over conventional numerical optimization because the design space can have arbitrary complexity as opposed to being limited to a fixed vector of decision variables as in optimization.

Therefore, in this paper, we are exploring tree search methods for finding the best solution to a design problem. It is our conjecture that the design decision trees we define in this paper are different from typical tree search spaces explored by computer scientists. For example, the majority of tree search problems can be defined as path-

A.-R. Short · B. L. DuPont · M. I. Campbell (✉)
Oregon State University, Corvallis, USA
e-mail: matt.campbell@oregonstate.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_5

planning problems or game trees. Path-planning trees are distinguished by the fact that the tree terminates at easily discernible goal states and the tree often contains monotonicities in approaching that goal. Game trees also terminate in goal or end-game states despite the fact that two or more agents control the decision making and are typically operating under counterproductive utility functions.

Design decision trees are marked by four unique qualities. First, they are often *multimodal*. There is rarely a monotonicity in the metrics that can be used to guide us toward a solution. Second, the solutions are of *unbounded* complexity meaning that nonterminal states in the tree can be a valid solution even though additional decisions can be made on them to make more complex solutions. As result, there is no clear sense of a valid goal state, and in some cases, even the starting seed may be seen as a valid solution. Third, because design decision trees are often comprised of a structure (in this paper we use a graph structure), the design has *locations* within it that can be leveraged in the subsequent decision making. For example, if a generative graph grammar [1] is used to define transitions in the decision tree, then the mapping of the left-hand side of grammar rules through the recognition process may be reasoned about with some independence from the application or change produced by the rule (as indicated by the right-hand side of the rule). Finally, design problems often contain states that are not evaluable. This means that the effect of each decision cannot be readily determined if the resulting state does not update the defined metric of quality. As a result, sometimes multiple cascading decisions are required to arrive at an evaluable state. For example, the comfort, dynamic response, or aesthetics of a bicycle cannot be determined without completing decisions on all relevant parts such as the drive-train, frame, wheels, and suspension. Sometimes predictions can be made but within an automated process, the effort to make such predictions would be significant over merely invoking a few more decisions to arrive at an evaluable state. We refer to this final quality as *opaqueness*.

Typically, deterministic methods like best first search (e.g., A* [2]) are used in path-planning algorithms, but these are rarely useful for the generic class of “design” trees that are described here. G for searching trees—also known as Genet [3]—has stagnated in recent decades in favor of the more generic concept of Monte Carlo Tree Search (MCTS) methods [4–6]. However, the capabilities of these methods for problems portraying the four characteristics above (multimodality, unbounded, location, and opaqueness) is not well studied. This paper is a first attempt at exploring these methods to understand which are most applicable to design problems represented as decision trees.

Aims

This paper aims to establish an easily evaluable test problem to explore how different tree search methods perform as applied to graph topology design problems. In this paper, an emphasis was placed on the clear description of the implementation and

methods for the purpose of repeatability, and to lay a foundation for future work that uses other methods and approaches to study the Color Graph design problem.

Significance

This work has profound potential significance in the field of automated design for three reasons. First, it describes a class of problem that is rarely studied in the abstract. Second, it establishes a standard evaluation metric for unbounded opaque topological design problems. Finally, it presents findings that are widely generalizable to many real-world automated design problems.

Problem Definition

The unbounded opaque design decision tree problem has three unique qualities. They are (1) multimodal and non-monotonic, (2) unbounded, and (3) contain multiple internal locations.

Multimodality complicates the design and analysis of a system. This is because the design cannot be evaluated until it is complete, as early choices may be a good in the beginning but lead to complications later on. An example of this would be constructing a multistory building to minimize cost. If a choice is judged by its value before the building is completed, then it will likely not have a foundation can sufficiently support later levels. However, if the whole building is designed first and then evaluated, the designer will be able to determine if the foundation was sufficient.

The arbitrary complexity of the unbounded opaque design decision tree creates further complications. Traditional tree search methods are capable of finding an optimal solution in a bounded tree, but because this class of design decision tree can have a potentially infinite number of choices the problem become intractably large and no state can be described as globally optimal.

Multiple internal locations make the problem more complex. A new location is added to the system as it is constructed, resulting in a factorial branch factor. This means that the problem can become intractably large very quickly, and it becomes impossible to significantly sample the space.

Potential Applications and Generalization

While the exploration of unbounded opaque design decision trees is academically interesting for their own sake, this work is broadly generalizable to many applications. Including Automated Metallic–Organic Framework [7] design (directly inspired this work), architectural design [8], mechanical structures [9–12], piping systems like

HVAC [13], and truss design [14–16]. It should be noted that Color Graph is capable of being feasibly represented as a binary GA which may not be true for all design problems, however insights gained should still be generally applicable.

Method

In this paper, we present the results of two methods applied to the same design problem. A simple design problem was created that can be evaluated in a very short amount of time, roughly 0.001243 s on the machine that was used for this experiment.

The Color Graph Design Problem

A Color Graph is a directed graph composed of a seed node, n_0 , to which red, orange, yellow, green, blue, or violet colored nodes are added. In addition to the seed node, colored nodes can be added to other colored nodes already existing in the graph. Figure 1 shows three examples of Color Graph candidates found in the search tree.

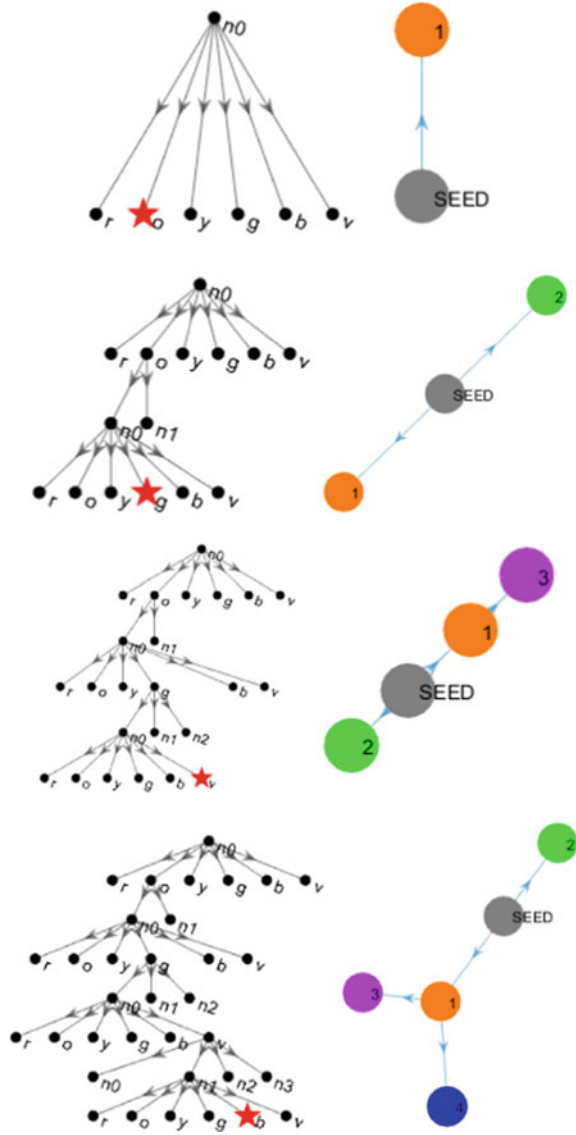
The design decision tree for a Color Graph alternates between location decisions and color decisions. There are six branches coming off the root representing the six potential colors for the added node. From each of those six color options, there are two branches representing potential locations in the Color Graph to add the next node. One for the Color Graph seed node, n_0 , and one for the first node added, n_1 . From each of those location options the design decision tree branches again with the six color options. The design decision tree continues to repeat this way between color and location, with the number of location options increasing every time a node is added to the Color Graph. Figure 2 shows the design decision tree and corresponding Color Graph for the first four nodes added.

The edges of a Color Graph design determine its quality. The edge's scores are determined by the source node and target node of each edge. For the case study presented in this paper, the edges have three arbitrary properties: α , β , and γ , with a



Fig. 1 Three examples of a color graph

Fig. 2 Design decision tree and corresponding color graph



range of -5 to 5 . These edge properties are independent and are randomly generated. When a completed Color Graph is evaluated, the scores for each edge are summed giving a three-dimensional score $[\Sigma\alpha, \Sigma\beta, \Sigma\gamma]$. Then, the design is rated on its proximity to a target score, which was $[0, 0, 0]$ for this paper. Table 1 shows example edge scores for a Color Graph, and an example Color Graph is scored based on its edges in Fig. 3.

Table 1 Edge properties

		Target node																							
		α						β						γ											
Source node		Red	Orange	Yellow	Green	Blue	Violet	Red	Orange	Yellow	Green	Blue	Violet	Red	Orange	Yellow	Green	Blue	Violet	Red	Orange	Yellow	Green	Blue	Violet
		Red	-3.15	3.83	-0.71	3.18	3.27	-3.49	0.38	1.06	-1.94	1.85	-2.20	-0.37	-0.06	-2.46	2.57	-3.85	-3.62	-2.53	-0.06	-2.46	2.57	-3.85	-3.62
Orange	3.56	3.45	-4.28	4.09	1.92	-1.10	1.55	-1.42	0.54	-0.96	4.06	-2.65	-2.82	2.11	0.74	2.83	4.89	1.81	-2.82	2.11	0.74	2.83	4.89	1.81	
Yellow	3.72	-4.54	2.93	-3.12	-4.44	0.85	0.18	3.95	-4.32	-2.88	-3.94	-3.00	1.88	1.33	-0.24	-4.92	0.98	-3.91	1.88	1.33	-0.24	-4.92	0.98	-3.91	
Green	1.14	0.76	1.13	-0.04	-3.27	1.43	-2.37	0.66	0.54	2.31	-1.89	4.99	0.74	2.62	1.65	2.80	1.22	1.29	0.74	2.62	1.65	2.80	1.22	1.29	
Blue	0.30	-4.27	-1.79	2.81	1.42	-4.48	4.56	-2.19	-4.55	1.45	2.52	-3.87	4.15	-3.46	1.03	4.29	3.66	4.23	4.15	-3.46	1.03	4.29	3.66	4.23	
Violet	0.93	-3.20	-4.42	3.99	4.58	4.47	-0.40	-4.58	-0.39	-0.71	-0.64	1.59	-3.41	0.96	4.89	-1.42	-4.40	-0.16	-3.41	0.96	4.89	-1.42	-4.40	-0.16	
Seed	-1.51	-0.22	-0.64	-3.32	-4.89	-2.38	1.34	-3.98	3.83	0.47	1.08	4.99	0.09	-4.52	3.58	0.86	3.58	-2.48	0.09	-4.52	3.58	0.86	3.58	-2.48	

Fig. 3 Color graph edge properties

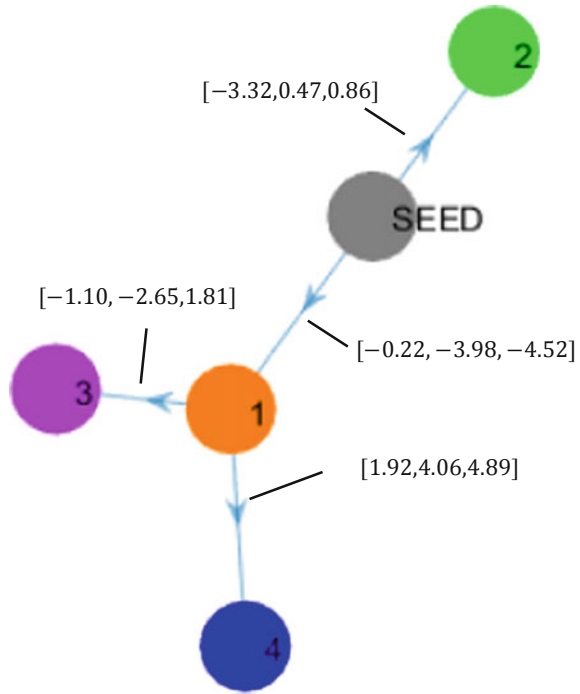


Figure 3 shows an example of how a Color Graph is scored using the edge properties from Table 1. The edge between n_0 (the seed node) and n_1 (the orange node) has an α value of -0.22 , a β value of -3.98 , and a γ value of -4.52 . We sum these with the edge scores from the remaining four edges and get totals of $\alpha = -2.72$, $\beta = -2.10$, and $\gamma = 3.04$ or as a three-dimensional coordinate in a design space $[-2.72, -2.10, 3.01]$. We compare this to our target design score of $[0, 0, 0]$ and determine the quality of the design by its proximity to the target. This can be found by calculating the distance between the two points using Euclidean distance, giving the design a final quality score of 4.59. A perfect score would be a 0 meaning that the design perfectly recreated the target design.

The Color Graph problem exhibits all of the properties of interest of the design decision tree. The design objective function for evaluating solution quality is multimodal and non-monotonically related to the number of nodes present in the Color Graph. For example, if the Color Graph in Fig. 3 was only the first four nodes of a larger design and a designer added a red node to n_1 , then the design quality would improve, but if a designer added a violet node to n_1 the quality of the design would decrease. This problem is compounded by the opaqueness of the design decision tree. The Color Graph design process is opaque as the final design performance cannot be determined from the performance of an incomplete Color Graph design. In many real-world cases, opaqueness exists in designs that are not immediately or intuitively obvious to a human, due to a large number of sensitive design variables and multi-

modality of the design. The Color Graph design decision tree is unbounded because it has no set limit on the number of nodes that can be added to the Color Graph. Last, it possesses an increasing number of internal locations, as the locations where nodes can be placed increases with the number of nodes already present in the Color Graph. This gives the design decision tree for the Color Graph a branching factor of $n \times 6$, making its growth both geometric and factorial. Figure 4 shows a subsection of the Color Graph design decision tree with nine levels (selecting node color five times and node location four times).

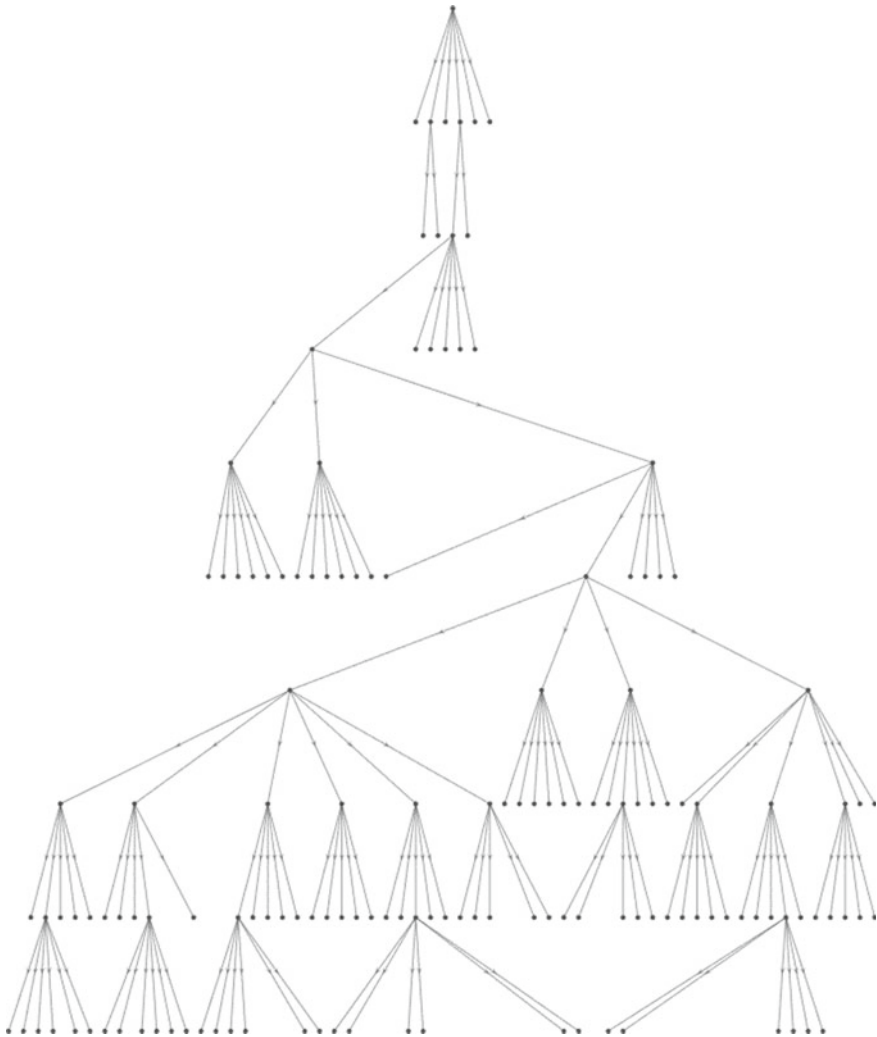


Fig. 4 A section of the design tree for a color graph

Algorithms Evaluated

Two algorithms were evaluated on their ability to design a Color Graph. The chosen algorithms were a genetic algorithm [17] and a Monte Carlo tree search [4]. We implemented both methods in the 2017 edition of MATLAB [18].

Genetic Algorithm

A Genetic Algorithm (GA) is a metaheuristic design algorithm inspired by natural selection. GA is one of the larger class of bioinspired algorithms called Evolutionary Algorithms (EA). For the Color Graph problem, the individual graph designs are represented by a set of chromosomes representing the location where a node is added, and the color of the added node. For example, a chromosome could be [red, seed; blue, node-1; green, seed; yellow, node-2; blue, node-3] as shown in Fig. 5, additionally, we can say the length of the set of chromosomes is itself defined by an additional chromosome that was not varied during these trials. The effect of modulating the length though can be explored in future work.

The algorithm starts by randomly generating 100 parent solutions. The algorithm then rates each solution with a fitness function, in this case, the Color Graph edge evaluation. The 10 top-performing individuals are copied into the next generation. The remaining 90 spots in the next generation are filled with the offspring of two parents randomly selected from the previous generation. Next, there is a small probability that a node color will randomly mutate. The GA is performed for 20 generations, and the final design is recorded.

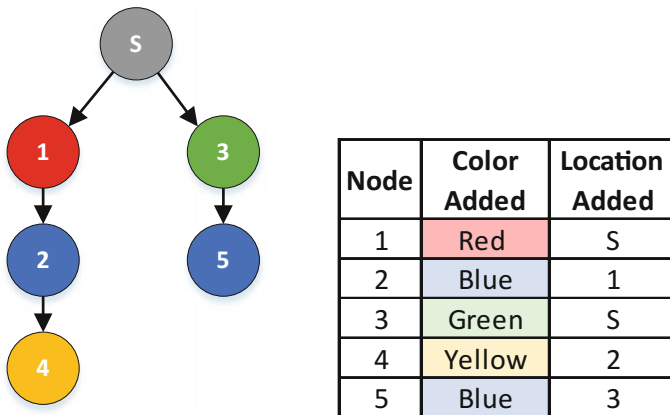


Fig. 5 Example color graph with chromosome [red, seed; blue, node-1; green, seed; yellow, node-2; blue, node-3]

Algorithm 1: GeneticAlgorithm

Input: Number of nodes N , and edge properties α , β , and γ , target properties t , generation size b , and number of generations G .

Output: A color graph design

1. Randomly generate b offspring
2. Compute offspring's α, β, γ properties
3. Compute offspring on proximity to t
 Loop Process
4. **for** $g = 2 : G$ **do**
5. Clone top 10 offspring to new generation
 Loop Process
6. **for** offspring $11 : b$
7. Select random parents biased to stop
 scoring of last generation
8. Select chromosomes randomly from parents
9. Randomly mutate chromosomes

Fig. 6 The genetic algorithm

The GA's greatest advantage is not its ability to search a tree in the traditional sense, but instead, they generate a constantly improving set of completed solutions. This has two advantages: (1) GAs find completed solutions, and (2) they do not have to follow a traditional search path and can make large moves to completely new branches. One potential weakness of the GA is that while it can quickly find a good solution, it is highly stochastic and there is no guarantee that the GA will converge on a globally optimal solution. Genetic algorithms have a tendency to become fixated on a region and never explore beyond it unless a serendipitous mutation should arise to introduce new regions of improvement. So, one could prescribe a level of quality that is considered good enough and running the GA until the level of quality is reached.

Each of the 100 solutions in all 20 generations was evaluated and had its design and the quality score recorded in a cell array. When a new generation was created, the top 10 offspring in the previous generation were copied into the new generation. Next, a normal distribution with $\sigma = 8$ and $\mu = 0$ was used in a Cumulative Density Function (CDF) to select two parents. Two random values between 0 and 1 were generated and used to look up the inverse value in the normal CDF. These numbers would correspond to the rank of the two parents. The chromosomes of the offspring were randomly selected from the two parents, with equal frequency. Next,

the random mutation would occur. For each added node, there was a probability of 0.05 that it would randomly mutate to a different color. This would preserve the overall structure of the Color Graph while switching the design decision tree to a different, but a similar branch. The best scoring design was recorded into the final results, along with the number of generations needed to reach the design. The GA algorithm is shown below in Fig. 6.

Monte Carlo Tree Search Algorithm

A Monte Carlo Tree Search (MCTS) is a type of heuristic search algorithm that is often used in the analysis of games. MCTS consists of four basic steps: (1) selection, (2) expansion, (3) simulation, and (4) back-propagation. During selection, MCTS uses a policy to select the best option currently available. A policy is a set of rules developed by MCTS that dictate what decisions should be made. Starting from the root node, the seed in a Color Graph, the search traverses until a leaf is reached in the decision tree. When a leaf is reached, MCTS begins expansion. During expansion, MCTS adds the next round of potential choices to the branch. MCTS then performs simulation, consisting of selecting a leaf node that was just added and randomly sampling the potential branches beneath it until completed designs have been generated. The completed designs are then scored on quality. Finally, quality scores are back-propagated through the branch and back to the root, expanding on the previous policy by adding one more level of informed decision. This is performed for all the adjacent leaves at this level.

One reason for interest in MCTS is that it has been successfully applied to similar problems trees in game theory before [19], however, when applied to problems that could be classified as unbounded opaque design decision trees, MCTS has been shown to have inferior performance. A motivation for studying and publishing this work is to better understand and describe why MCTS underperforms on unbounded opaque design decision trees, such as Color Graph.

The best implementation found to store the quality scores of nodes was inside a digraph object (a graph with directed edges) in MATLAB directly [20]. Starting at the root location (the Color Graph seed), the design decision tree was expanded to add the six possible color choices. A sample of 100 random designs was taken. The sample size of 100 was selected after performing a parameter sweep on sample sizes to determine what generated a policy the most effectively. To compare MCTS more directly to the GA, parallelization of the search was not used. This was to create a baseline for comparison showed the general feasibility of the method, not the computer's ability to brute force the problem. In order to keep MCTS from running for an infeasibly long time, a time was implemented that stops MCTS after 20 min has elapsed and records the current best quality score and policy. The MCTS algorithm is shown in Fig. 7.

Algorithm 2: Monte Carlo Tree Search

Input: Number of nodes N , and edge properties α , β , and γ , target properties t , sample size s .

Output: A color graph design

```

INITIALIZATION
1: Select the root node
SELECTION
  Loop Process
2: While design policy has not reached length  $n$ 
3:   While design policy exists
4:     Select branch with the best score
5:   End while
EXPAND
6:   If terminal leaf is a location in the color graph
7:     Add six new leaves representing color options
8:   Elseif terminal leaf is a color decision
9:     Add a number of leaves equal to the number of nodes in the
    color graph
10:  End if
SIMULATION
  Loop Process
11: While new unexplored leaf exists
12:   Generate  $s$  random completed design branches
13:   Calculate the average score for decision tree leaf
14: End while
BACK PROPAGATION
15: While current node  $\neq$  root
16:   Select node back one level in the branch
17:   Compute the average score for that node based on
SIMULATION

```

Fig. 7 Monte Carlo tree search algorithm

Experimental Setup

For both the GA and MCTS method, we conducted an experiment in which Color Graphs with 3, 5, and 10 added nodes are designed. Each algorithm explores graphs of variable sizes because the unbounded nature of the design decision tree means

that more choice could always be made, therefore it is important to study how each method behaves as the size of the decision tree grows.

For each method and size of Color Graph 10 trials were performed. Ten sets of random edge properties were generated prior to performance of the experiment, one for each trial. This was to ensure that the results would not be biased due to of a single set of properties, and enables comparison across methods and graph sizes.

The metrics of interest are the quality score of the designed graph, the length of time needed to reach the solution, and the number of graph designs analyzed during the process. Additionally, we recorded the number of generations before the solution was discovered (for the GA), the depth of the policy (MCTS). This allowed us to gain additional insight on the capability of the methods, even if they fail to generate a completed design within the allotted 20-minute limit. The best Color Graph from each trial is stored in a cell array for later review.

In addition to the GA- and MCTS-generated Color Graphs, purely random Color Graphs (consisting of randomly chosen colors placed in random locations) will be generated for each trial as a control, and analyzed as a baseline for comparison.

Results

The entire experiment took approximately 7 h and 45 min to run on an ordinary desktop computer with a 3.4 GHz Intel Xeon CPU [21] and 16 GB of RAM. A summary of the mean algorithm scores (with 0 being the best possible score) is shown in Fig. 8.

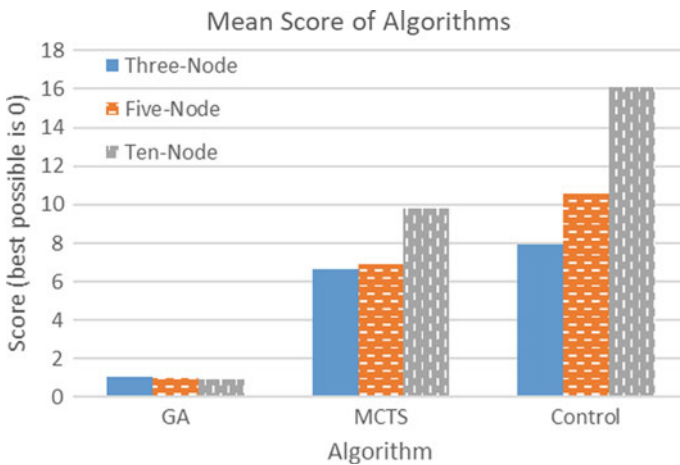


Fig. 8 Summary of mean scores

Three-Node Color Graph

In the first test, each method was used to generate a Color Graph with three added nodes, and we performed 10 trials for each of the methods. The methods were scored on their ability to improve Color Graph quality, with a best possible score of 0. Table 2 shows a summary of the results.

The best performing method was the GA with a mean score of 1.044, and a standard deviation of 0.457. MCTS had a mean score of 6.6296 and a standard deviation of 2.453. For this size, MCTS was only marginally better than the control and was 1.84 standard deviations away from a purely random design. The purely random control had a mean score of 7.945 with a standard deviation of 0.712.

Five-Node Color Graph

Again, the best performing method was the GA with a mean score of 0.9843, and a standard deviation of 0.413. MCTS had a mean score of 6.9008 and a standard deviation of 3.89. This significantly outperformed the purely random control, with a mean score of 10.59 and standard deviation of 0.929, but still performed much worse than the GA. Table 3 shows a summary of the results.

Another notable result is that for the five-node Color Graph, only three of the 10 MCTS trials successfully developed a full five-step design decision tree policy during the allotted 20 min. The other seven trials were only able to develop a four-step design decision tree policy, meaning that the final node's location and color are indeterminate and the final score was based on expected value if the branch were to be randomly completed. For comparison, the run that took the longest only lasted 11.6 s.

Ten-Node Color Graph

As in the first two tests, the best performing method was the GA with a mean score of 0.9290, and a standard deviation of 0.393. MCTS had a mean score of 9.774 and a standard deviation of 5.54, which is worse than the five-node test, still better than the control. The purely random control had a mean score of 16.07 with a standard deviation of 1.29. Table 4 shows a summary of the results.

During the ten-node test, MCTS completely failed to generate a completed policy within the allotted 20 min. In four of the trials, MCTS developed a five-node design decision tree policy, in five trials it developed a four-node design decision tree policy, and in one trial it only developed a three-node design decision tree policy. The longest the GA took to complete was 21.3 s.

Table 2 Results of the three-node test

		Three-node color graph						
		Genetic algorithm		Monte Carlos tree search		Control		
		Quality score	Run time (s)	Nodes evaluated	Quality score	Run time (s)	Nodes evaluated	Quality score
Mean		1.044	7.46	2000	6.630	309	6951	7.945
Std		0.457			2.45			0.711

Table 3 Results of the five-node test

		Five-node color graph				Monte Carlos tree search			Control
		Genetic algorithm		Nodes evaluated		Quality score	Run time (s)	Nodes evaluated	Quality score
Mean	0,9843	11.6	2000	6.901	1221	17,700	10.59		
Std	0,413			3.88			0.929		

Table 4 Results of the ten-node test

		Ten-node color graph						
		Genetic algorithm			Monte Carlos tree search			Control
		Quality score	Run time (s)	Nodes evaluated	Quality score	Run time (s)	Nodes evaluated	Quality score
Mean		0.9290	21.3	2000	9.774	1217	17,880	16.07
Std		0.393			5.54			1.29

Discussion of Results

Analysis of the results leads to several interesting observations.

The first and most obvious observation is that the GA significantly outperformed MCTS in both quality score and runtime with a p -value of 0.0001 or less in all three cases. This was likely related to the branching factor of the design decision tree being $n \times 6$ resulting in a tree that rapidly becomes too large to feasibly search. MCTS struggles with this because it must search through the actual tree in order to find good design solutions. The GA avoids this problem by not relying on the design decision tree structure, and instead, working directly on designs.

It is expected that this observation is broadly generalizable outside of these two methods. For example other Evolutionary Algorithms (EA) that work directly on the Color Graph without considering the design decision tree would likely be able to find a solution relatively quickly, on the other hand, algorithms that search through the design decision tree directly, such as an Ant Colony Optimization (ACO), are likely to fail to find particularly good solutions.

A second notable result is the usefulness of MCTS was very sensitive to the size of the Color Graph. For the three-node Color Graph, MCTS did not significantly perform better than random (p -value of 0.1205), because it was not able to find a particularly good node due to how large the design decision tree becomes. However, as for the five-node Color Graph test, MCTS performs significantly better than random (p -value of 0.0091). This appears to be because MCTS is capable of finding an okay solution, but as the problem grows larger pure random selection performs increasingly poorly. However, the capability of MCTS is limited as the problem size continues to grow because the runtime needed quickly becomes infeasible.

Conclusion

Based on the results of the experiment, it can be concluded that a Genetic Algorithm (GA) is much better suited than a Monte Carlo Tree Search (MCTS) to the problem of designing systems with design decision trees that are multimodal, unbounded, and contain multiple internal locations. In future work, we hope to explore this further and validate that it is a result of how the methods utilize the design decision tree different.

Additionally, it has been shown that the Color Graph design problem serves as a good benchmark for the study and comparison of various forms of automated design methods in real-world unbounded problems. The designs of Color Graphs can be evaluated at a rate of approximately 1000 graphs per second on commonly available desktop computers while preserving characteristic multimodality, unbounded complexity, and multiple internal locations.

Future Work

Future work will focus on further exploration of the Color Graph design problem using a wider variety of methods. Additionally, experimentation will be performed to explore questions of option design complexity, the complexity of design properties and behavior, and the development of new and novel methods for automated system design.

Acknowledgements This material is based upon work supported by the National Science Foundation under grant CMMI-1662731. Any opinions, findings, and conclusions or recommendations presented in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Rozenberg G, Ehrig H (1997) Handbook of graph grammars and computing by graph transformation
2. Kanal LN, Kumar V (1988) Search in artificial intelligence. Springer-Verlag
3. Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection. The MIT Press
4. Browne C, Powley E (2012) A survey of monte carlo tree search methods. *IEEE Trans Intell AI Games* 4(1):1–49
5. Perez D et al (2014) Solving the physical traveling salesman problem: tree search and macro actions. *IEEE Trans Comput Intell AI Games* 6(1):31–45
6. Perez D, Rohlfshagen P, Lucas SM (2012) Monte Carlo tree search: long-term versus short-term planning. In: 2012 IEEE conference on computational intelligence and games (CIG), pp 219–226
7. Manion CA, Arlitt R, Tumer IY, Campbell MI, Greaney PA (2015) Towards automated design of mechanically functional molecules. In: Volume 2A: 41st design automation conference, p V02AT03A004
8. Koning H, Eizenberg J (1981) The language of the prairie: Frank Lloyd Wright's prairie houses. *Environ Plan B Plan Des* 8(3):295–323
9. Patel J, Campbell MI (2008) An approach to automate concept generation of sheet metal parts based on manufacturing operations. In: Volume 1: 34th design automation conference, parts A and B, vol DETC2008-4, pp 133–142
10. Patel J, Campbell MI (2008) Topological and parametric tune and prune synthesis of sheet metal parts compared to genetic algorithm. In: AIAA/ISSMO multidisciplinary analysis and optimization conference
11. Swantner A, Campbell MI (2012) Topological and parametric optimization of gear trains. *Eng Optim* vol in review:1–18
12. Radhakrishnan P, Campbell MI (2010) A graph grammar based scheme for generating and evaluating planar mechanisms. In: Design computing and cognition '10, pp 663–679
13. Patterson WRJ, Campbell MI (2011) PipeSynth: an algorithm for automated topological and parametric design and optimization of pipe networks. *ASME Conf Proc* 2011(54822):13–23
14. Hooshmand A, Campbell MI (2016) Truss layout design and optimization using a generative synthesis approach. *Comput Struct* 163:1–28
15. Shea K, Fest E, Smith IFC (2002) Developing intelligent tensegrity structures with stochastic search. *Adv Eng Inform* 16(1):21–40

16. Shankar P, Ju J, Summers JD, Ziegert JC (2010) DETC2010—design of sinusoidal auxetic structures for high shear. *Eng Conf* 1–10
17. Whitley D (1994) A genetic algorithm tutorial. *Stat Comput* 4(2):65–85
18. MATLAB—The Language of Technical Computing. 09 Dec 2015. [Online]. Available: <http://www.mathworks.com/products/matlab/>. Accessed 09 Dec 2015
19. Browne CB et al (2012) A survey of monte carlo tree search methods. *IEEE Trans Comput Intell AI Games* 4(1):1–43
20. Graph with directed edges—MATLAB
21. Intel® Xeon® Processor E3-1240 v2 (8 M Cache, 3.40 GHz) Product Specifications. Intel® ARK (Product Specs). [Online]. Available: https://ark.intel.com/products/65730/Intel-Xeon-Processor-E3-1240-v2-8M-Cache-3_40-GHz. Accessed 16 Dec 2017

Part II
Design Cognition—Design Approaches

Externalizing Co-design Cognition Through Immersive Retrospection



Tomás Dorta, Emmanuel Beaudry Marchand and Davide Pierini

This paper presents an insightful explanation of designers' experience over time during immersive co-design sessions. The data was collected through immersive retrospection interviews here used to assess a co-design activity. Three teams of two proficient designers individually self-evaluated their perceived experience by observing an immersive video unfolding their respective co-design sessions inside a social virtual environment (Hyve-3D). Pinpointed shifts in these experiences guided subsequent individual immersive retrospection interviews, sparking the externalization of covert aspects of participant's co-design cognition. Analysis of these verbal accounts resulted in interesting insights about how designers' cognition proceeds during co-design activities, further pointing to the scaffolds of their evaluation of design ideas. Findings suggest that there frequently are parallel processes occurring in individuals' minds. We observed that internal ideation was mostly associated with optimal experience.

Introduction

With today's advent of virtual reality as consumer product, a lot of former usages of this technology are now being updated. Different uses have been developed, for example for training purposes (surgery, aviation, etc.) or for psychological interventions (phobias, PTSD, etc.). Of course, in the domain of computer-aided design, virtual reality represents a stepping-stone to help architects, designers and stakeholders to design and be inside their projects. The immersion places the users within their virtual realms. However, for researchers the 'cognitive realm' of what happens in the designers' mind remains concealed.

T. Dorta (✉) · E. Beaudry Marchand · D. Pierini
Hybridlab, University of Montreal, Montreal, Canada
e-mail: tomas.dorta@umontreal.ca

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_6

This paper presents the results of our endeavour to unveil designers' covert mental processes, using a social virtual reality (VR) design tool as a strategic device to access their cognition. This way, we wish to explore what were to happen if this technology was used not only for assisting the design process, but also *to place the users within their own inner realm along with the researchers*. Pairs of professional designers co-realized an ad hoc project in a virtual environment that was then instrumentalized for digging into co-ideation mechanisms through designers' experience. We collected data about affective states and shifts in affective experience in order to identify the moments during which something relevant occurred. We used these key moments to explore designers' cognitive activity through retrospectives interviews.

The results unveiled internal processes occurring parallel to overt information exchanges concerning ideation and self-evaluation activities; these aspects were often observed when designers were in an optimal experience. Furthermore, we explained some unspoken repercussions between participants' experience. The findings shed new light onto internal evaluation processes taking place simultaneously during idea propositions, even before the representation, going beyond the appraisal-after-exteriorization of Schön's model of *reflection-in-action*. The study generates new shades to be addressed in further research related to the source of the pleasant aspects of ideation, motivated either by internal process or sketching.

Previous Work

Accessing Designers' Cognition

In design, cognitive activity underlying idea generation (ideation) has mostly been studied through the analysis of verbal reports: interviews, dialogues analysis and protocol analysis, often used in design cognition research [1]. The latter is based on the *think aloud* technique, which requires designers to "speak their thoughts" during the activity of designing; meanwhile, their verbalizations are collected often along with a video recording showing their overt behaviour. Then, such verbalizations are segmented and analyzed as objective data from which cognitive activity is inferred [2]. Since verbalizations are collected during the activity (concurrent protocol analysis), designers have to tell what is *in their minds* (contents) as well as what they are mentally doing (process) in the very moment when this happens; therefore, memory biases and subjective interpretations are avoided. Yet, a part of designers' mental activity can remain covert: while thinking aloud designers seem to deeply cogitate during the silences in between the verbalizations [3]. In fact, during ideation sudden insights [4, 5], visual-spatial as well as perceptive processes [6–8] could be active and might not be compatible with (slow) verbalizations possibly requiring full attentional resources.

Another way to obtain verbal reports about design thinking is to ask designers how they built their design retrospectively, i.e. after an ideation session [9–11]. In order to

support memory recall, video recordings of the session can be presented. The major drawbacks of retrospection rely on that people could forget and not report something that was in their working memory and, because of designers' interpretation of the situation, they could report something that did not actually occur [2]. It seems that concurrent and retrospective protocols give similar results when analyzed using a process-based coding scheme [12]. However, in another study, concurrent protocols seemed to reveal mainly data related to the procedure used to carry on the activity at hand (i.e. what I am doing), while retrospective protocols, beside information about the procedure, allowed to access the reasons behind people's behaviour (i.e. why I did something) [13].

Co-design

During collaborative ideation, part of the thinking processes might be naturally verbalized because designers need to share knowledge about the project, explain their solutions, ask questions, and express opinions about the solution [14]. However, the need for communication with teammates is likely an additional task rather than an externalization of designers' thoughts. For example, explaining the idea I have *in my mind* could not reflect the cognitive activity that generated such specific idea, but rather the need of finding a way to transmit it to the teammates. Moreover, non-performance related inner speech (such as self-criticism, self-rewarding and social assessment) can still occur as a result of the appraisals processes without being externalized during conversations [15]. Therefore, a part of the cognitive activity can still remain hidden.

Designer's Experience

Comparably to visual-spatial and perceptive processes, appraisals of the on-going design situation and the related affective states specifically related to non-verbal communication and potentially driving designers' behaviour are barely represented in the language [16]. This is why we chose to directly use the assessment of the designer's subjective perception of the unfolding activity, namely their experience. Safin et al. [17] developed a high-granularity (i.e. high sampling rate) designers' experience evaluation coined Design Flow 2.0 based on retrospective self-observations in order to avoid interruptions of the design activity. This framework stands on the notion of optimal experience [18] measured using a two sliders interface enabling the continuous assessment (1-second resolution) of the perceived *challenges* and *skills* levels. The different balances between these levels result in a characterization of four general experience states, namely, *stress* (high challenges and low skills), *flow* (high challenges and high skills), *control* (low challenge high skills) and *disengagement* (low challenge and low skills) [17]. The state of *flow* is considered being an optimal

experience because it has been associated with better performance and with a positive affective state [18].

However, the limitation of this framework resides in the fact that participants are required to remember what happened, yet without providing an insightful explanation (rationale) of what was the cause of different experience states and their transitions. For example, if one participant rated a sequence of events as stressful, researchers have to interpret the cause of such state.

Immersive Retrospection to Access Experience Rationale

To overcome this shortcoming as well as to access covert cognitive activity, we used a twofold approach previously implemented [19]: (1) an immersive retrospective self-observation and (2) an immersive retrospective interview. In order to support the recall of previously lived events, we placed participants in an immersive video. This offers a viewpoint similar to that of the direct observation of the scenes occurring when the original activity was conducted [19]. By using virtual reality for immersion in past events, we intended to go further than traditional video self-observations (using conventional flat displays) by taking advantage of improved spatial awareness and embodiment in supporting the recall of the events. Such immersive videos were obtained using a 360° camera, pan-tracking the subject's position or their line of sight in post-production before the self-observations. It is also possible to use a recording of the immersive viewport itself, giving the same perspective(s)-taking during the sequence of events occurred in the virtual environment. Based on this twofold approach, it is possible to first immersively collect high-granularity data of the participant's experience and then to gather rich explanations of its underlying covert mental activity through a semi-structured immersive retrospective interview. In this study, we used a social virtual reality co-design system (Hyve-3D™) [20] not requiring VR glasses in which each user has a tablet for sketching and interacting (Fig. 1). During the interviews, we asked designers what they were thinking and what caused changes in their experience state.

Methodology

The study was conducted with six professional designers/architects paired in three teams of two (three females and three males). Each designer had worked with his/her teammate on occasions prior to the studied co-design sessions. Following a design brief of an ad hoc project (5 min.), namely proposing a new concept for a self-sufficient eating area of a campus building, each team was given a 20-minute period in the virtual reality collaborative sketching environment to proceed freely with the design activity. We here opted for a co-design setting to make straightforwardly observable exchanges (verbal and graphical not analyzed in this study), content that



Fig. 1 Immersive co-design sketching system (Hyve-3D) used during the co-design sessions (left) and the immersive self-observations experience assessment (right)

would ultimately serve as a reference to support the participants' recall of events during the retrospection process. Further analysis in future studies could put findings in perspective with considerations of overt behaviours.

Within the hour following each session, each participant underwent a 20-minute high-granularity experience assessment through immersive self-observation taking place in the same immersive co-design system. Using the Design Flow 2.0 method [17], designers were called to evaluate the evolving levels of their perceived *challenges* and *skills* in dealing with the design sessions. They faced an immersive recording of the team's co-design session's unfolding in the virtual environment along with a synchronized audio recording of their conversations.

After collecting this time-mapped experience data, preliminary visualizations of the fluctuations in designers' experience state (i.e. stress, flow, control, and disengagement) were produced and analyzed to extract *key moments* that would serve to frame the immersive interviews (Fig. 2). The selection of these key moments stood on the assumption that changes in experience state indicated subjectively important moments, and thus would provide relevant anchors that could act as the starting point for recollection on events of interest. This way, in the weeks following the original sessions, researchers conducted an *immersive retrospective interview* structured accordingly to these key moments, in order to obtain an insightful explanation of participants' experience during the co-design activity. Last, two additional questions were asked during brief exit interviews regarding the immersive retrospection technique concerning: (1) *how well they remembered the co-design activity* (extent and depth of memories) and (2) *the embodiment and the gestures looking at the immersive video* (vigour of experience). Qualitative content analysis of the interview reports in the form of in vivo, descriptive and simultaneous coding lead to the generation of a set of codes used to describe a variety of co-design-related elements (such as intents and events) namely here as *aspects*, explicitly evoked by the participants (Fig. 3). Those aspects were associated with them in term of co-occurrences (i.e. simulta-

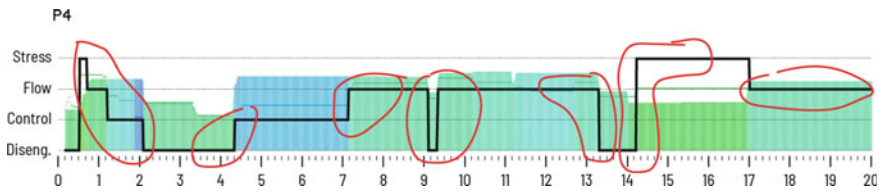


Fig. 2 Sample (participant P4) of experience states assessment and *key moments* selection (in red)



Fig. 3 Occurrences of aspects (design-related tasks) for each experience state classified according to three different activities (three colours)

neous appearance of two or more aspects during the interview). Time mapping the instances of such codes provided us with multi-dimensional data that was then used for various visualization crossing different factors (see figures in the results section).

Table 1 Distribution of experience states among participants

States	P1 (%)	P2 (%)	P3 (%)	P4 (%)	P5 (%)	P6 (%)	Overall
Stress	31.5	35.5	38.1	14.9	19.6	28.8	28.1
Flow	22.4	16.3	17.3	47.6	37.4	32.3	28.9
Control	32.6	37.4	28.3	18.6	16.2	21.7	25.8
Disengaged	13.5	10.8	16.3	18.8	26.8	17.2	17.2

Following research regulations of our institution, all the participants consented to be part of this study by signing a consent form accepted by the ethical committee of the University of Montreal: *Comité plurifacultaire d'éthique de la recherche* (CPÉR).

Results

Design Experience

Overall, except for *disengagement* (17.2%), the states of *flow* (28.9%), *stress* (28.1%) and *control* (25.8%) were almost equally rated by the participants (Table 1). However, mostly (three occurrences and up), participants' reports were associated to design ideation aspects while in the optimal experience (Fig. 3, *Flow* state). Also, those aspects were contextually associated to design activities (green).

Internal Cognition

Five from six participants directly reported (stating literally) to be *in their minds* while in *flow*, showing the highest rate among the psychological states (11 occurrences). This aspect was also frequently reported in other states, ranked second in *control* (four occurrences) and fourth in *stress* (six occurrences). This, otherwise, hidden aspect seems to indicate the internal cognition of the design activity.

In more detail, as shown in Fig. 4 (two co-occurrences and up), all participants associated to be *in their minds* only with aspects regarding an ideation process of proposing and evaluating design ideas. In particular, we highlight the fact that the self-critic process is intrinsic to this internal thinking, placing the *self-criticism* at the top of the co-occurrences (six times). On the other hand, there were three instances of quality judgements that ideas were good (*Good idea*) (Fig. 4).

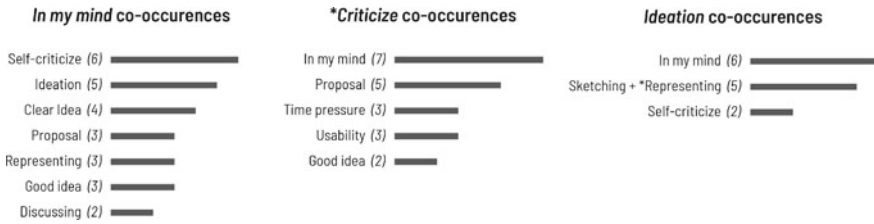


Fig. 4 Co-occurrences of aspects evoked at the same time of *in my mind* (left), variants of *criticize* (centre) and *ideation* (right). *Including criticize + self-critique or representing + self-representing

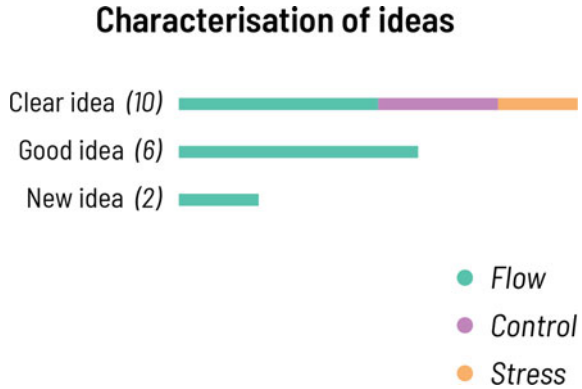
Design Evaluation

Conversely, participants during the retrospective interviews remarkably brought up evaluations mostly related to their own ideas (*proposal*) rather than others'. When evoking criticism-related aspects of the design activity, the claim of being *in their minds* was the most frequent co-occurrence (seven times) (Fig. 4, centre). Moreover, the criticism was perceived mostly as a stressful dimension including both *criticize* and *self-critique* (Fig. 3, *Stress* state). There were three occurrences of self-critic in the state of *disengagement*: two related to design activities (green) and one concerning the tool (red) when it was put aside for too long (Fig. 3, *Disengagement* state). Overall *self-criticism* was for the most contextually associated to design activities (Fig. 3, green).

Sketching and Ideation

The aspect reported as *ideation* by five of the six participants show a close relation to internal activity (*in my mind*), but also to the process of graphically externalizing design ideas (*Sketching*, *Representing* and *Other's representing*) (Fig. 4 right). Furthermore, this sketching activity was perceived by all the participants as pleasant (Fig. 3, *Flow* state, 10 occurrences). Also, four occurrences of sketching appeared as stressful for three participants: one relating it to a design activity trying to resolve a shape; two to the communication activities, to represent the idea enough (quantity) and 'appropriately' (quality); and one towards the end of the session for the speed of sketching using the tool (Fig. 3). The *ideation* aspect was most referenced in relation to the optimal experience, standing among *Good idea*, *Clear idea* and (*self*) *Proposal* (Fig. 3). The characterisation of ideas made by the participants themselves during the interview, as *clear* (ten times), *good* (six times), or *new* (two times) was for the most associated to the *flow* state (Fig. 5).

Fig. 5 Occurrences of aspects characterizing ideas related to the experience states



Using the Tool

Participants reported on the use of the immersive co-design tool during the activity, showing a positive assessment while in the *Control* or *Flow* state and negative (problematic) while in *Stress* or *Disengagement*. The use of the tool caused *stress* (Fig. 3, second rank of occurrences, red colour) for five of six participants, even if they had different levels of familiarity with the tool (two very familiar, three familiar not recent users, and one novice). Another five of six participants were at some point *disengaged* from the activity because of the tool (first rank) for various reasons, ranging from *how to use it* to *waiting* on the collaborator’s mastery during the activity. However, four of six participants reported to be in *control* of the situation related to the tool during the sessions (Fig. 3, *Control* state), with three occurrences related directly to others’ participation (*Other’s usability*, *other’s control* and *other’s representing*), placing it as the top referred aspect during the *control* state. Finally, only two participants reported about the tool while in *flow*, one person concerning the usability aspect, the other while using spatial references in the virtual environment.

Temporal Relationship of Participants’ Co-design Cognition

We illustrated the reported aspects from each co-participant of the same team, mapping them onto the different experience states through time as obtained during the experience assessment. Then, we associated those aspects with the ones they likely affected, as reported during the interviews (Figs. 6, 7 and 8 for all participants). The aspects that affected other participants’ experience were mostly associated with design (green) and communication (blue) activities (Figs. 6, 7 and 8). Table 2 synthesizes the repercussion of one’s psychological state over the other’s and shows only those that were identified as occurring in at least two teams.

Fig. 6 Participants 1–2
 Depiction of how one designer's aspects affected the other participant's experience over time as indicated by the arrows colour-coded according to the three different activities (green: design; blue: communication; red: tool)

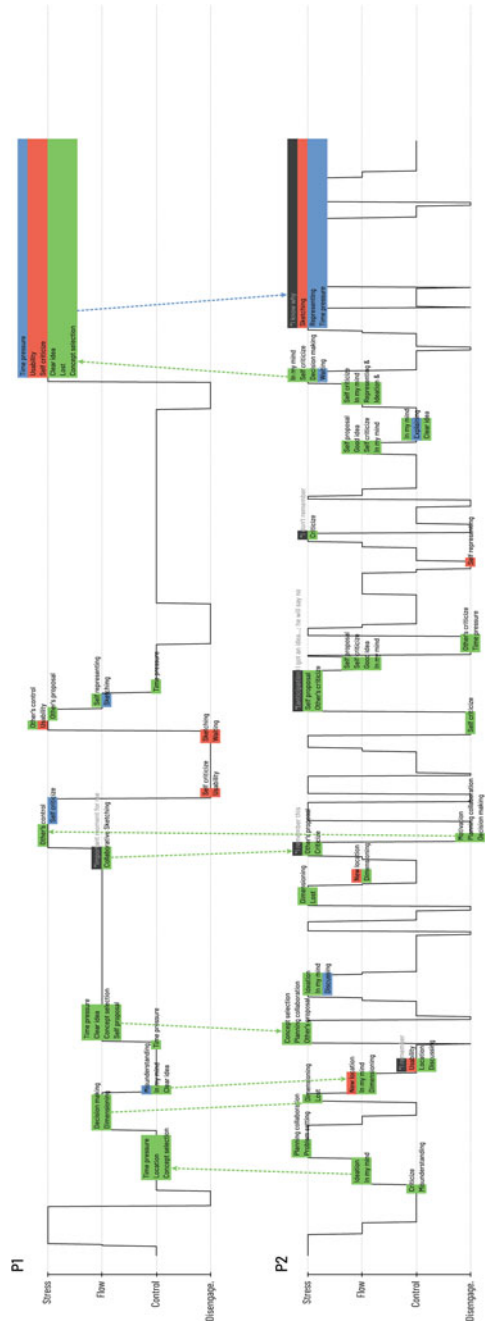


Fig. 7 Participants 3–4
 Depiction of how one designer’s aspects affected the other participant’s experience over time as indicated by the arrows colour-coded according to the three different activities (green: design; blue: communication; red: tool)

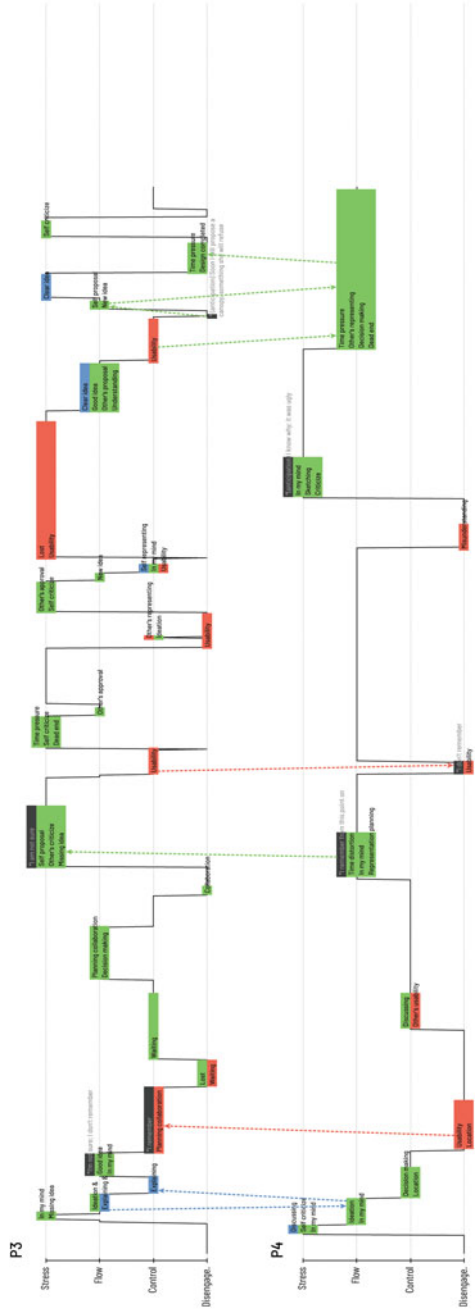


Fig. 8 Participants 5–6
Depiction of how one designer's aspects affected the other participant's experience over time as indicated by the arrows colour-coded according to the three different activities (green: design; blue: communication; red: tool)

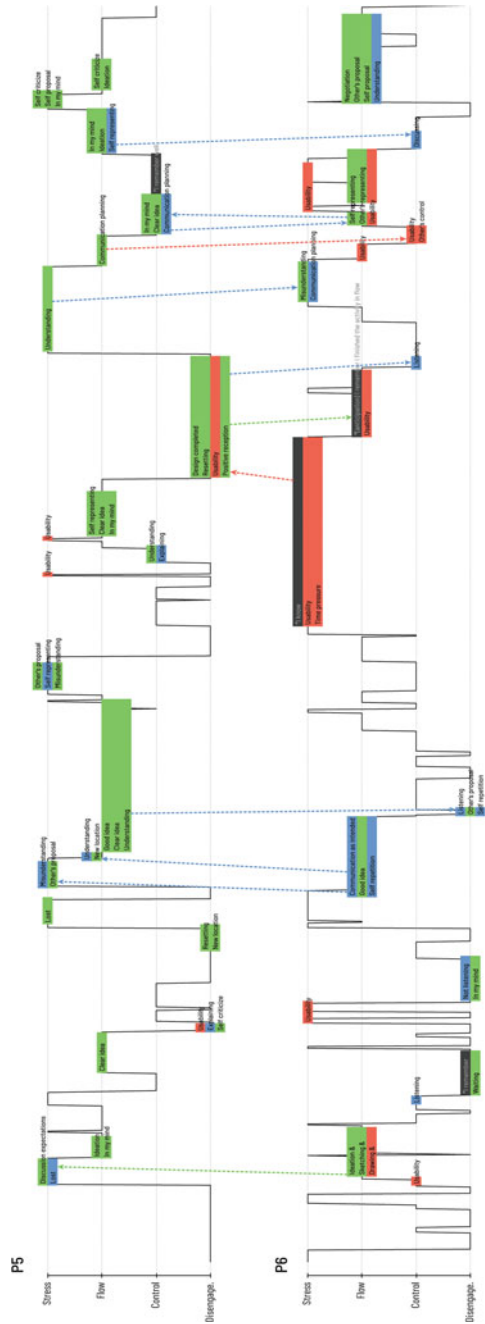


Table 2 Repercussions of psychological states for two or more teams

Team	1	2	3
Flow ↓ Stress	D: Dimensioning → <u>Lost</u> D: Self-proposal → <u>Other’s proposal</u> D: Sketching → <u>Other’s proposal</u>	D: Rep. planning → Other’s criticize	D: Ideation → <u>Lost</u> – Good idea → Misunderstanding
Flow ↓ Control	D: Ideation → Concept selection	C: Explaining → Decision making	C: Com. planning → Others’ control C: Representing → Discussing
Control ↓ Flow	D: Clear idea → Dimensioning	D: Usability → <u>Others’ representing</u>	C: Com. planning → <u>Others’ representing</u>
Flow ↓ Flow		D: Self-proposal → Others’ representing	C: Com. as intended → Understanding
Flow ↓ Disengagement		D: Decision-making → Design completed	C: Good idea → Self-repetition
Disengagement ↓ Control		T: Usability → Planning collaboration	C: Resetting → Listening
Stress ↓ Stress	D: Decision making → Lost C: Time pressure → Representing		C: Understanding → Misunderstanding

D—Design; C—Communication; T—Tool

Only the optimal experience of *flow* emerges as the origin of all four of its possible repercussions through two teams or more. More precisely, for the three teams we observed that the *flow*-induced *stress* and *control*. For only two teams *flow* provoked *flow* and *disengagement*. *Control* is the only other state having an effect in all teams, inducing a *flow* state (Table 2). Moreover, others’ state of *flow* induced the aspects of being *lost* and seeing *other’s proposal* as *stressful*, both of which were twice related to design activities. The state of *control* provoked twice an optimal experience while *others were representing*, for design and communication activities (Table 2, underlined aspects).

Multitasking

In addition, participants reported certain of the aspects as occurring in parallel with other tasks. Three participants mentioned such multitasking during the sessions, all

including *ideation* among parallel tasks such as *sketching* concerning the use of the tool, *representing* vis-à-vis the design activity and *explaining* regarding communication, all happening during a *flow* state.

Exit Interview

Concerning the question of *how they remembered the co-design activity*, participants generally reported short spans of immersive recordings (around 10 s.) as sufficient to gain recall and get re-contextualized, and for most of them, to “feel” again and rebuild the sequence of events and memories beyond the displayed content (pointing in real space). One participant was even able to pinpoint (feelings) when the experience changed.

On the other hand, to the question concerning *the embodiment and the gestures looking at the immersive video*, participants reported they remembered the gestures and their original positions regarding the screen. One participant referred to the missing information from the use of the tablets. Some others claimed the Hyve-3D’s 3D cursor (3D interactor’s avatar) and the appearing sketches allowed for a projection of themselves in the virtual space. One stated that the perspectives they took in the virtual space gave cues to remember.

Moreover, participants brought up other comments during the exit interviews in regard to the experience states. The *flow* was described while focusing on ideas that were in their minds and the collaboration as expected (the acceptance of their proposals), as one stated: “*I think internally, and I need to focus. So, when he talks and his ideas are different, I go away of that...*” and “*Flow was more collaborative-driven (team work) than ideation-driven*”.

Discussion

The short span (20 min.) of the ad hoc projects potentially made participants concentrate mostly towards design ideation related aspects, with limited wandering around other steps of extended design process (context analysis, resolving technicalities, etc.). This could explain the amount of contextually design-related reports (green). However, even when bearing in mind that the *flow* state was barely the most frequent state (*flow* 28.9% vs *stress* 28.1%), (Table 1), a large part of the aspects was related to its fluctuations (while in that state or shifting from or to that state), interestingly related to ideation (Fig. 3).

One important finding of this study is that when describing their experience state, participants often spontaneously recalled being *in their mind* (attentional focus oriented towards internal activities) during the design activity, even in such a collaborative setting. We see this claim as a gateway to their *co-design cognition*, externalizing their internal processes otherwise invisible during the activity. We chose in this study the co-design setting to make straightforwardly observable verbal and

graphic interactions calling them hereinafter first-order exchanges. Yet, the ideation and evaluation aspects happening in their minds had different concerns than *what was said and done during the activity*. Their occurrence indicates a gap with the first-order exchanges, explaining their (design) thinking, their reasoning, in short, the rationale of designing according to their felt experiences. Moreover, as shown in Fig. 4, the results suggest that there are parallel processes lying in proposing ideas (mostly their own), representing and evaluating them at the same time as participating in the first-order exchanges. More importantly, the evaluation in terms of self-criticism appears as intimately weaved with the development of ideas and their characterization (appraisal), in accordance with the *reflection-in-action* of Schön [6], nevertheless not necessarily triggered by the external representation. For example, in parallel to communications with the teammate, one designer reported having an evolving idea in his mind which was never exteriorized neither by talking nor sketching, finally dropped after judging it not relevant. Even if having *clear ideas* comes up not only in connection to a pleasant experience, graphically externalizing them by sketching was considered pleasant and accompanied to the aspect of *ideation*. One can ask why *ideation* was considered as an optimal experience: because of the *sketching*, the *collaboration-driven* activity or being *in their mind*?

As introduced earlier, traceable relationships between co-participants' cognition timelines (Figs. 6, 7 and 8) open the door to a second-order exchanges analysis. One's *flow* caused *stress* in their teammate under the aspects of *other's proposal* and being *lost*. This could refer to the fact that the perception of other's new ideas could engage stress because of possible confusion (*lost*). Moreover, one's *control* (of representing) induced a *flow* state in the corresponding teammate reporting the *other's representing*. This could mean that ideation by representing is appropriate in co-design and points towards a need to master the externalization medium to better support the collaborators' optimal experience. In this regard, during the study we noted different levels of familiarity with the immersive tool, potentially affecting the co-design activity at different levels, as shown above: sometimes explicitly *waiting* for the other's mastery or sometimes provoking *flow*. Furthermore, we eventually have to point out that the features of this co-design tool could orient the retrospection strongly towards sketching aspects, hindering the emergence of other facets which can occur using other representation tools (e.g. 3D modelling or parametric design tools).

In relation to the latter as reported in the exit interviews, during the retrospection participants evoked that the appearance of sketches and the viewpoints constituted cues helping them to reconstruct an extended sequence of events. Moreover, the immersive retrospective interview was considered as a successful approach to help them remember the causes of their experience during the activity. Some participants expressed *living* and *feeling* again what happened during the key moments, hinting at insights on the particular nature of *memorable events*. On this matter, some exclaimed interjections like "*I remember; I know; I remember well; Ah this is an important moment for me*" (black boxes in Figs. 6, 7 and 8). Furthermore, several participants anticipated specific moments which we note are mostly related to *stress* and *flow* (black boxes in Figs. 6, 7 and 8).

Conclusions

For the first time, we have access to the hidden reflections that explain the changes in designers' experience. We found that the processes of proposing, representing and evaluating design ideas internally can occur parallel to first-order (observable) exchanges. The immersive retrospective interviews gave us access to these parallel processes by further externalizing participants' co-design cognition through the glasses of their perceived experience. This technique was the trigger of the collected reports, using the immersive co-design tool (Hyve-3D) also as a research device supporting retrospective data collection. Despite the limited number of participants and their familiarity with the tool, the results of the study allow us to draft new explanations of the co-design dynamics hinting at *second-order exchanges*, between-designer links of experience states. Also, the internal self-evaluations seem to be related to the participants' own ideas before representing them, offering an update and refinement to Schön's model of *reflection-in-action*.

References

1. Gero JS, Mc Neill T (1998) An approach to the analysis of design protocols. *Des Stud* 19:21–61
2. van Someren MW, Barnard YF, Sandberg JAC (1994) *The think aloud method: a practical guide to modelling cognitive processes*. Academic Press, London
3. Lloyd P, Lawson B, Scott P (1995) Can concurrent verbalization reveal design cognition? *Des Stud* 16:237–259
4. Akin Ö, Akin C (1996) Frames of reference in architectural design: analysing the hyperacclimation (A-h-a-!). *Des Stud* 17:341–361
5. Cross N (2001) Design cognition: results from protocol and other empirical studies of design activity. In: Eastman C, Newstatter W, McCracken M (eds) *Design knowing and learning: cognition in design education*. Elsevier, Oxford, UK, pp 79–103
6. Schön DA (1983) *The reflective practitioner: how professionals think in action*. Basic Books, Avebury
7. Suwa M, Gero J, Purcell T (1999) Unexpected discoveries: how designers discover hidden features in sketches. In: *Visual and spatial reasoning in design (Vol. 99)*. Key centre of design computing and cognition. University of Sydney, Sydney
8. Tang HH (2002) Exploring the roles of sketches and knowledge in the design process. Department of Architectural and Design Science, Faculty of Architecture, University of Sydney
9. Bilda Z, Demirhan H (2003) An insight on designers' sketching activities in traditional versus digital media. *Des Stud* 24:27–50
10. Suwa M, Purcell T, Gero J (1998) Macroscopic analysis of design processes based on a scheme for coding designers' cognitive actions. *Des Stud* 19:455–483
11. Suwa M, Tversky B (1997) What do architects and students perceive in their design sketches? A protocol analysis. *Des Stud* 18:385–403
12. Gero JS, Tang HH (2001) The differences between retrospective and concurrent protocols in revealing the process-oriented aspects of the design process. *Des Stud* 22:283–295
13. Bowers VA, Snyder HL (1990) Concurrent versus retrospective verbal protocol for comparing window usability. *Proc Human Factors Soc Ann Meet* 34:1270–1274
14. Dorta T, Kalay Y, Lesage A, Pérez E (2011) Elements of design conversation in the interconnected HIS. *Int J Des Sci Technol* 18:65–80

15. Brinthaupt TM, Hein MB, Kramer TE (2009) The self-talk scale: development, factor analysis, and validation. *J Pers Assess* 91:82–92
16. Dong A, Kleinsmann M, Valkenburg R (2009) Affect-in-cognition through the language of appraisals. *Des Stud* 30:138–153
17. Safin S, Dorta T, Pierini D, Kinayoglu G, Lesage A (2016) Design Flow 2.0, assessing experience during ideation with increased granularity: a proposed method. *Des Stud* 47:23–46
18. Csikszentmihalyi M (1997) *Creativity: flow and the psychology of discovery and invention*. Harper Collins, New York
19. Marchand-Beaudry E, Han X, Dorta T (2017) Immersive retrospection by video-photogrammetry: UX assessment tool of interactions in museums, a case study. In: Fioravanti A, Cursi S, Elahmar S, Gargaro S, Loffreda G, November G, Trento A (eds) *Proceedings of the 35th eCAADe conference—volume 2, ShoCK!—Sharing Computational Knowledge!*, pp 729–738
20. Dorta T, Kinayoglu G, Hoffmann M (2016) Hyve-3D and the 3D cursor: architectural co-design with freedom in virtual reality. *Int J Archit Comput* 14:87–102

Demystifying the Creative Qualities of Evolving Actions in Design Reasoning Processes



Tamir El-Khouly

This paper detects the levels of contribution to design creativity of critical moves and sudden insights and identifies the syntheses the creative process may take. Using architecture case studies, designing situations are analysed as sketching episodes that reflect the structural units of reasoning to deduce common characteristics for the emergence of critical moves and sudden insights. Ethnographic observations are conducted on two architects where two factors are examined in an empirical study: creative actions emerge either through incremental improvement and emphasising the prevalent concept, or through non-incremental reasoning restructuring the design problem. Creative qualities are distinguished as actions that either accept or reject the prevailing paradigm or attempt to integrate multiple paradigms and form syntheses between different concepts. Sudden creative insights are likely to emerge in the latter case. A creative insight is particularly investigated when an unprecedented idea emerges to solve an intricate problem that is impossible to solve with the usual frame of reference. It comes with a proposition to ease the problem landscape to be solved. A creative solution is related to the quality in solving an intricate problem; the extent to which it meets the functional, structural and configurational requirements.

Introduction

Two main viewpoints describe design in the research arena: hierarchical or transformational. In a hierarchical process, ideas usually evolve ‘top-down’; the emergent products are outcomes of a structured thinking process and design decisions are prob-

T. El-Khouly (✉)
The American University in Cairo AUC, Cairo, Egypt
e-mail: t.el-khouly@ucl.ac.uk

T. El-Khouly
Ain Shams University, Cairo, Egypt

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_7

ably ‘process-oriented’ (the structuralists, e.g. [1]). The goals and problem definition are pre-determined in the preliminary stage before the design process proceeds, and execution and evaluation follow to assess the solution. This view holds that creative insights are created out through sequential cumulative syntheses of ideas (e.g. concepts that refer to the architect’s own experiences, or to the collective experiences of designers working in collaboration). Another view defines design as a transformational process, where ‘good’ design ideas emerge from reflection-in-action on the interim artefacts of mental representations; unexpected discovery evolves in practice and decisions are probably ‘action-centric’ (the constructivists, e.g. [2]). Here, the design problem, brief and solution co-evolve together, affecting each other iteratively [3]. The depth of an inter-relational design path can be tested in the context of examining the relations between ideas (including insights or intermittent actions) and the resulting products.

The dilemma lies in how to interpret the role of critical moves and sudden mental insights in the design creative process. Do sudden insights impose a particular structure of executional steps on the subsequent design actions and consequently drive the concept to certain levels of contribution to design creativity in an incremental reasoning process? Or do randomness and chance play a vital role in creativity, leading to novel variations in thinking and syntheses processes? It was argued that sudden insights impose a rigid structure of actions on the subsequent design moves to execute the occurring idea and conceptual form through procedural steps—the Simonian positivism view [4]. Others suggest that newly emerging products of sudden insights are subject to a series of developments and assessments when the designer looks at the whole picture and addresses contextual components of the design idea—the Schönian constructivism view [5]. Thus, ideas evolve and transform from one state to another while the designer considers pros and cons while the design is in progress.

References [6] and [7], however, introduced an interesting study on *structured imagination*, in which the argument on sudden mental insights was framed while posing the question: ‘are creative insights normally derived from existing cognitive structures and representations, or are they chanced upon arbitrarily?’ [7]: 208. While explaining the creative cognition approach, the point was clearly made that ‘creative discovery’ is not a type of ‘either/or’ question. Rather, an emphasis should be put on the methods that permit one to determine the relative roles that ‘randomness’ and ‘structure’ play in creative discovery. Hence, the contribution of this paper is a methodological development to detect the context behind the emergence of critical moves and evolving actions to identify the levels of contribution to design creativity. It describes an ethnographic observational study conducted on two architectural design cases to investigate how creative insights evolve.

Consequently, the research question is: *how the emergence of critical moves, and on the other hand, the levels of contribution to design creativity of the evolving actions and sudden insights can be distinguished?*

Such evaluative methods look at the progress of proposed design under test and provide the designer, and particularly the architect, with necessary information upon which the reframing of a solution or restructuring of a problem take place and may affect subsequent steps. The paper suggests a descriptive approach that adopts the

epistemology of practice to understand the evolution of ideas in the design reasoning process. In its contribution in the field of design research, the level of contribution to design creativity for each design move is detected and described, models of synthesis creativity and diversity and originality are identified in empirical architectural study, a method is proposed to segment the design process into structural units of reasoning via sketching episodes, code the dependency relationships among design moves to construct linkography patterns. The conclusions are twofold: first, creative qualities are distinguished as actions that either accept or reject the prevailing paradigm or attempt to integrate multiple paradigms and form syntheses between different concepts, and second, creative moves are most likely to emerge from unintended syntheses.

On the Nature of Design and Creative Insights

Two main opinions formed this debate on the nature of creative insights around which research efforts are clustered. According to one position, creative discovery is systematic and organised and is based on highly structured processes [8–10]. According to the other, *randomness* and *chance* play a vital role in creativity, leading to novel variations in thinking and syntheses processes [11–13]. The former opinion affirmed that insights are *retrieval* from memory, acting as ‘stimulus responses’ to a problem with an endorsed structured method of *trial-and-error* in order to develop a creative solution [10, 14]. The latter opinion stated that sudden insights result from rapid cognitive *restructuring* process of the design problem that distinguishes the problem-solving process in terms of a series of insightful processes; once an insight is perceived and realised, the problem solver can quickly implement its solution (the Gestalt view). A controversy has arisen between the two views, specifically enquiring: how would an insight help to solve the problem? In their explanation, Finke et al. [6] pointed at two positions: while the *memory* position states that an ‘incremental’ approach is the way to retrieve good ideas. Thus, insights are structured in the design process [15], the *restructuring* position states that an ‘unconscious’ way of thinking acts beyond our awareness where the design problem is to be restructured along the design process. Thus, sudden breakthroughs occur unconsciously and discontinuously [16, 17]. It was also debated that *unexpected discovery* and *discontinuity* of ideas are a driving force for *creative discovery* [18]. Sudden mental insights are hypothesised as occurring in the event of reformulating the design brief, and/or restructuring the entire design problem [19]. All these phenomena are of ‘creative cognition’ and are matters of investigation in this study.

A *design move* is an action of reasoning; a design ‘step’ transforms the design situation relative to the state it was in before that move, while a critical move is a key frame in the thinking process that are associated with the novelty of design [20]. A *design state* relies on what has been delivered by the series of sketching episodes; whether *incremental* (continuous evolution of the prevalent paradigm), *transformational* (bringing new elements to the concept under development) or *changeable*

(directing the design concept to totally new one). The transformation of ideas in the design process takes two types: *vertical* transformation develops the initial concept by adding more details to it; *lateral* transformation changes the existing concept to explore new ones, leading to a *divergent* style of thinking [21]. A *sketching episode* is defined as a transformation in perception from one state to another while marking out drawings before designing begins and as interim reflection when the sketch is still in progress [22, 23]. It is a depiction of an idea through sketching stated to deliver an eloquent role in the development of the concept. Any sign that the designer has decided to move from one frame of reference to another is considered an *insight* (as defined by [24]). A *creative insight* moves the perception to a completely different state that is independent of the current design situation. A *sudden mental insight* is a *stimulus response* that occurs in the mind suddenly when an unexpected idea is flashed [20]. It occurs to break out a *frame of reference* and shifts the design intention to a new one when a fixation effect is experienced causing blockage while solving the problem and generating the solution [24]. A *creative leap* was first proposed by [25] to indicate the effectiveness of creative insights on fostering the solution to overcome an experienced problem. The occurrence of sudden mental insights by which novel solutions become possible is considered a ‘situation-based’ event [24]. In our proposition, a *synthesis* process is the combination of more than one idea to arrive at an innovative solution to an intractable problem. Experimenting synthesis of various conceptual elements through sketching episodes may lead to the discovery of unpredicted creative solutions.

The Methodology Approach: Identification of Sketching Episodes

This study proposes an inductive approach based on wide data collection, moving from specific ethnographic observations to broader conclusions. From these observations, patterns can be identified and a hypothesis developed. The challenge is to understand the transformation of ideas from one state to another in the design process in order to judge the level of creative contribution. The key elements in the evolving actions where the design concept is shaped, reshaped or reconfigured are looked at across several design processes. By comparing interim consecutive artefacts (sketching episodes), two ways to describe the transformation of ideas from one state to another can be defined: ‘reframing the existing solution’ or ‘restructuring the pre-formulated problem’. This demonstrates whether there is a drastic change occurring in the flow or merely the introduction of new elements.

The aim is to show how stimuli responses help the designer to break away from one frame of reference to a new one. The pivotal moments where an unprecedented creative contribution occurs are highlighted and considered ‘aha’ events: creative hinges in concept reasoning development. False aha events might also occur, but later rejected.

This approach has two specific objectives: (1) to identify the beginning and ending of a design move; and (2) to disclose the mutual reflections with instantaneously externalised design artefacts between different means of representation (e.g. interim products, sketches, 3D models, etc.).

It is important to distinguish the gradual transformation of mental imagery from the view of design as perception-in-action (mental representation) through the sketching process. When mental imagery reflects ideas from the subconscious to the present situation, perception is instantaneous and deals with the synchronic designing situation in progress. Unintended discovery is more likely to happen in this case.

In this context, the ‘dialectics of sketching’ (as coined by [26]) can be identified, where two types of sketching episodes can be distinguished in reflection-in-action: in type 1 sketching from imagery is transformed into new combinations and is considered to be a rational mode of reasoning, whereas in type 2 sketching generates new imagery of forms in the mind and is a non-rational form of design thinking. The rational mode of type 1 is characterised by the systematic exchanges of conceptual or figural outcomes of sketching, while the non-rational mode of type 2 causes interactive manipulation with imagery. Examples of identifying the start and end points of the sketching episodes and the dependency relations among them to construct a linkograph are shown in detail in the empirical study.

The Empirical Study

Two experienced architects from two different countries were asked to design individually an ‘expo-pavilion’ for their own country. The brief was left open-ended, with no specific requirements or constraints. The architects could present their conceptual ideas through any means, without specific drawings or projections and without interference from the researcher. The design process lasted for one hour and all the design activities and verbalisations were video-recorded. The architects were asked to comment subsequently on the concept development and transformation of ideas for the serial order of sketches.

Architect A

Qualitative Description of the Design Process

Architect A chose to design an expo-pavilion to represent Greece. Her concept was based on the congregation of five pavilions. Design began bottom-up by setting up key conceptual elements of Greece, such as ‘sunlight’, ‘rocks’, ‘blue sky’, ‘water’, ‘Greek key patterns’, ‘olive and lemon trees’, ‘cubic forms’ and ‘shadows’. The architect proceeded to synthesise the concept of each pavilion around these elements. Some elements were repeated across pavilions but with different treatments, for example,

by using both natural skylights or light wells, and artificial lighting and interactive installations. Pavilion 1, for instance, took the concept of the synthesis of ‘sunlight’ and ‘central patio’. Pavilion 2 took the concept of the synthesis of ‘cubic stepped and linked forms’, with an ‘olive tree’ in the centre. The idea was transformed and developed by switching back-and-forth three times between two sketches with different projections: 2D plan and 3D perspective section. At pavilion 3, a lighting tunnel was introduced to the flow of reasoning. At pavilion 4 (episodes 71–78), an unprecedented novel idea interrupted the flow and restructured the whole design situation. At pavilion 5, the concept reinforced one of the predefined conceptual elements: the interlocking and stepped forms. Identifying the sketching episodes and dependency relations among them depends on examining the levels of contribution to creativity of critical moves and insights.

Creative Qualities for the Critical Moves and Sudden Insights

Concept initiation began with an *insightful* thinking process. The architect set up key elements that represented the nature of life in Greece. Some elements emerged by surprise, causing sudden changes in the prevailing concept of sketching. Representing the image of the Greek pavilion according to the architect’s subjective interpretation resembles a series of lateral transformations in concept initiation and development of which some ideas of sketching episodes suddenly evolved—taking the form of sudden flashes during transformation. Design episodes 2–8, 11 and 15 are conceptual ideas that appeared in one sketching medium independently of each other; see Fig. 1a.

The architect moved to another sketching medium and designed the first pavilion, the ‘Greek key forms’. Sketching episodes 17, 18 and 19 are incremental actions that retained the prevailing concept. Each episode represents a different architectural treatment: 3-D perspective (episode 17), 3D section (episode 18) and 2D plan (episode 19). However, the sketching process did not travel in only one direction across the three drawings. It switched back and forth to add masses and refine details. Episodes 17, 18 and 19 are creative moves that contribute to *reframing* the prevalent solution through advanced incremental actions; see Fig. 1b.

In the third sketch, referring to Fig. 1c, episodes 25, 27 and 31 are three drawings that make complete switches to another prevailing concept: pavilion 2, ‘the olive tree’. These actions are *vertical transformation* of the idea from one projection drawing to another and constitute incremental reasoning in the structure of design process. Episode 52 is a *back reflection* action in the mind, echoing one of the two sketching processes defined by [26] (the irrational mode that generates unprecedented ideas). The architect added a new conceptual element—a detail of ‘irrational openings and balconies’ typical of the architecture of Greece. However, episode 55 shows a sudden major shift from the preceding concept. The architect switched to sketching pavilion 3 to introduce a new concept different from the irrational openings. The lighting tunnel is an artificial installation centred on the concept of lighting and fading. Episode 55 is a sketching episode for a 3D perspective, while episode 60 is an entire switch: a vertical transformation to design a 2D longitudinal section. Episode 55 redefines the

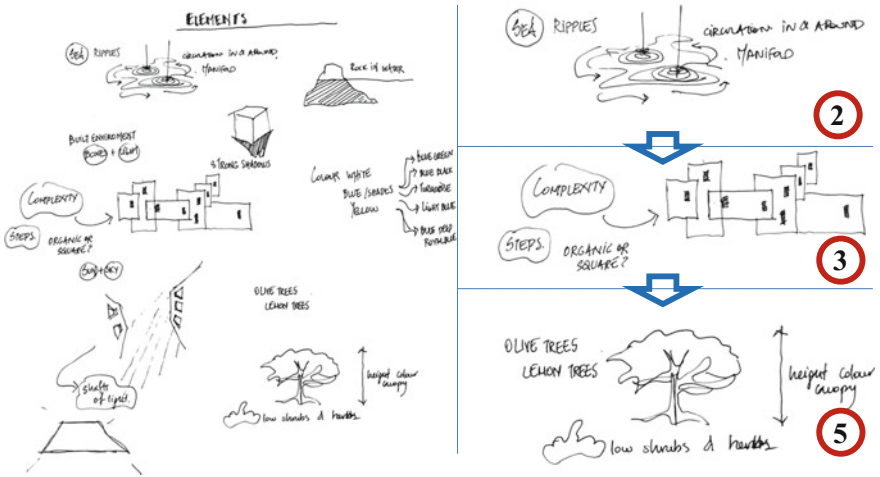


Fig. 1 a Identifying sudden changes in the flow of sketching and the creative qualities for the transformation from one sketching episode to another (architect A). **b** Identifying sudden changes in the flow of sketching and the creative qualities for the transformation from one sketching episode to another (architect A). **c** Identifying sudden changes in the flow of sketching episodes and the creative qualities for the transformation from one sketching episode to another (architect A)

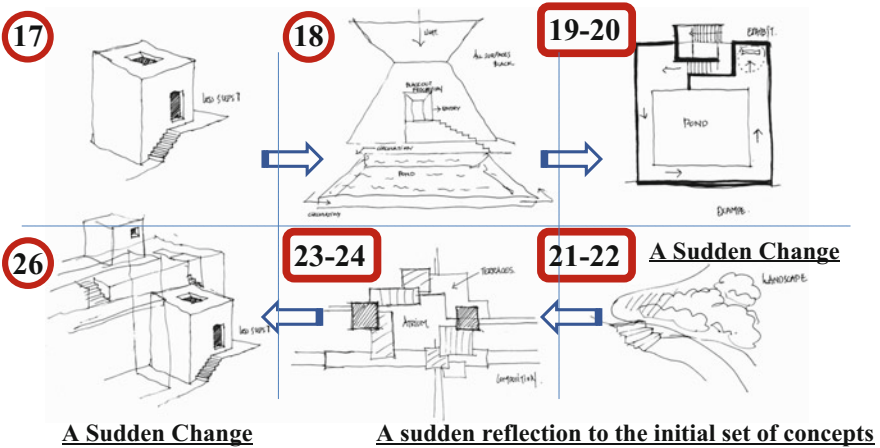


Fig. 1 (continued)

problem. Sketching episode 71 is a *sudden mental insight*. A paradigm shift occurred with the introduction of an unprecedented concept about ‘Greek emigration’ around the world. Pavilion 4 was an interactive installation for ‘cities and links’, which the architect said was inspired by the 2011 Serpentine Pavilion in London designed by Swiss architect Peter Zumthor, ‘which has everything to do with light’. The design then continued through another projection of 3D section at a complete shift at 72.

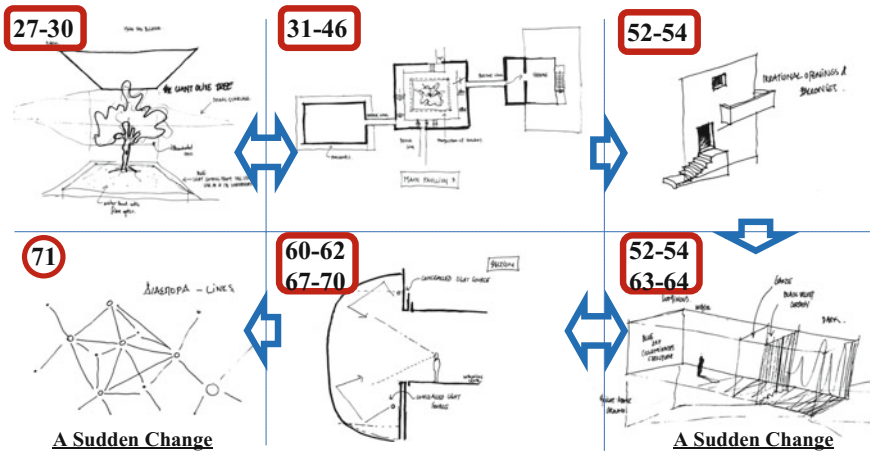


Fig. 1 (continued)

In summary, a sudden paradigm shift took place outlining unprecedented concepts: one on the *light-tunnel* experience and second on the *links* installation.

In summary, the transformations in concept development are taking shape from an episode to another; either reflecting the main set of conceptual elements, adding new ones and completing the design *parti* (conceptual artwork), reframing the solution and advance forward incrementation, or restructuring the design problem and reformulating the entire configuration. In the following, Fig. 2 introduces a method of coding the dependency relationships among the design sketching episodes to construct a linkograph, while Fig. 3 illustrates the linkograph of this design process.

Architect B

Qualitative Description of the Design Process

Architect B began by proposing conceptual ideas to express the diversity of society and life in the UK as ‘multi-ethnic, multicultural and multifaceted’ society such as a ‘user-generated sound disorder’ pavilion, reflecting the riots of 2011; ‘empire’ in history, science and industry; an amateur sports day event, e.g. ‘cheese-rolling’; science and discovery represented by a ‘chemical plant’; a ‘roulette wheel’ representing entertainment. The last concept was ‘empty box: the UK is what you make it’, allowing visitors to build their own image from different materials to express the UK’s eccentricity and continuing evolution and thus create their own understanding of the activities undertaken in the pavilion. The early stage of concept initiation reflected a divergent insightful thinking process, paving the way for a variety of syntheses and

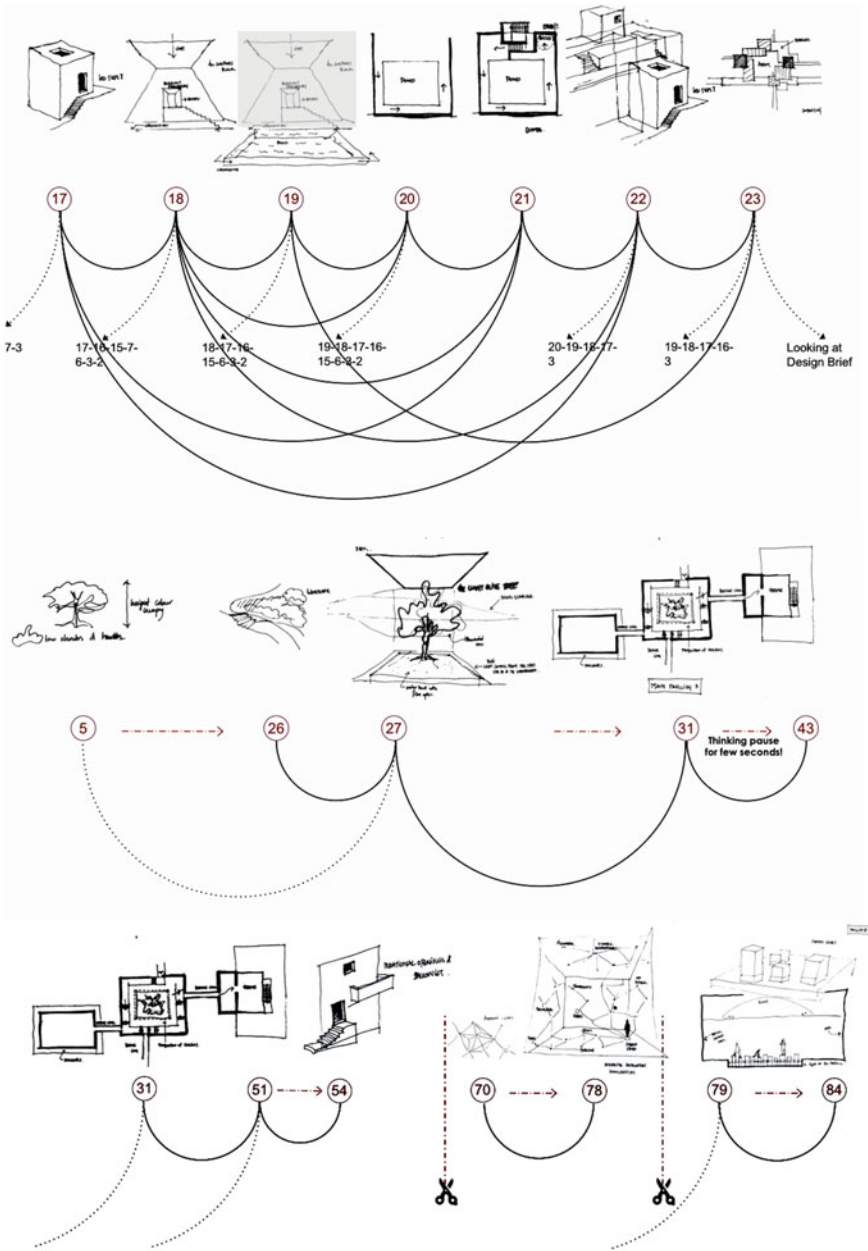


Fig. 2 Identifying the dependency relations between design segments (architect A)

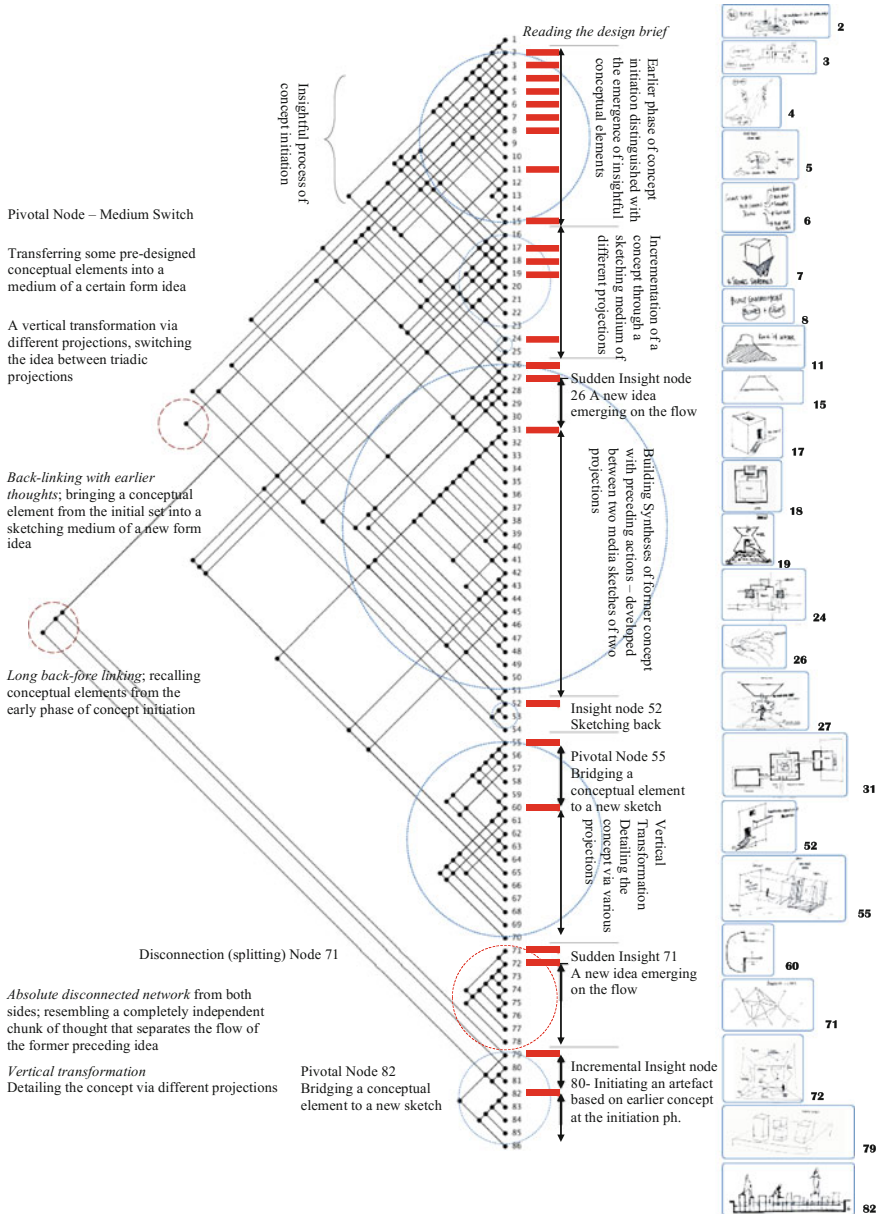


Fig. 3 The linkography protocol for the design process (Architect A)

hybridisation between the conceptual elements. This in turn increased the likelihood of designing several solutions throughout the process. A significant shift occurred in the transition from the middle to the final stage that led to exceptional ideas. The

decision was made to centre the ‘empty box’ as the prime element in the congregation. The remaining concepts were considered supplementary, but the architectural forms of the concepts remained similar to their definitions in the preceding phases. In the final stage, concepts were based on the earlier ideas and the end result did not produce any new unprecedented designs. Nevertheless, the variations in formulating the precedence of concept fluctuated throughout the process. The dependency relations between the design episodes were transcribed and coded to construct the linkograph. Reading the linkograph of this process is characterised by the long back- and fore-linking, divergence and convergence zones; see Fig. 5.

Creative Qualities for the Critical Moves and Sudden Insights

After reading the design brief at episode 1, a set of conceptual elements for pavilions emerged: ‘user-generated sound disorder’ (episode 2), ‘empty box’ (episode 6), ‘sports day—uphill cheese throwing’ (episode 9), ‘chemical plant’ (episode 11) and ‘roulette wheel’ (episode 14). Although those concepts express an image of the UK, they have no relation to any keywords in the design brief. However, the architect stated his aim at episode 3: ‘I am taking from the brief to make as many different ideas and sketches as possible’. The key concepts shifted the flow of sketching from one episode to another. Within the context of the prevailing concept, each action framed a new solution in relation to the whole configuration. All reflected the lateral transformation from one state to another to explore different ideas. At episode 16, the design converged the preceding ideas in one configuration. This action reflected the incremental mode of reasoning that directed the following designing actions constructively way until the designing episode (sketch 2-1) ends.

At episode 25, the final product was designed for sketch 3-1. A new 3D perspective depends on building synthesis with the preceding sketch 2-1 and with the conceptual elements. This action reflects the multiple exchanges of information and ideas with the preceding sketches creating back- and fore-linking. However, episode 16 remains the prime bridging point of convergence that directs the actions of design and synthesis in this medium. Episode 25 resembles advanced incremental action. Nevertheless, the decision to centre the ‘empty box’ pavilion in the congregation distinguishes this sketching episode (3-1) from what was designed earlier at episode 16 (sketch 2-1). Figure 4 illustrates the annotation of sketching episodes and contents. Figure 5 presents the linkograph of this design experiment.

Results

With *reframing*, we mean the events that preserve the prevailing design concept. The *reframing* events in the two cases are immense, acting to preserve the original concept and the flow of reasoning through the series of incrementation and reflections on the interim artefacts. The *restructuring* events, however, in cases A and B

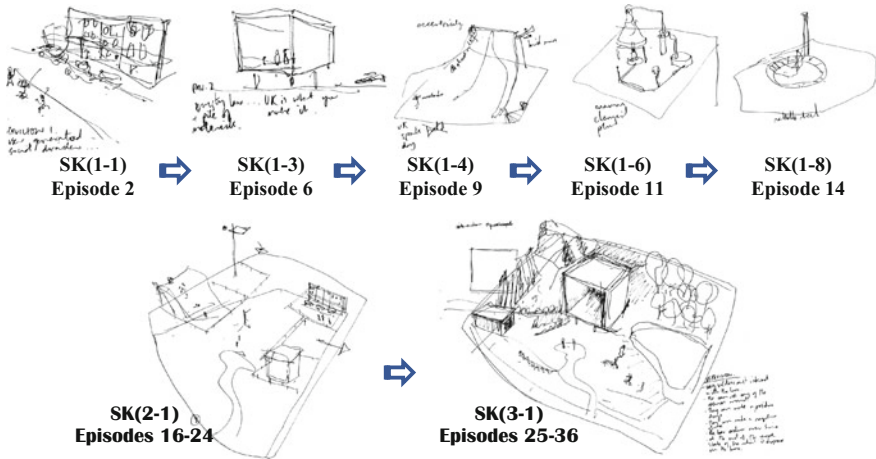


Fig. 4 Illustration of sketches—The interim and final products of the design process—The segmentation and chronological development of the conceptual and critical sketching episodes along the process (Architect B)

show discontinuity in the flow of design reasoning that shifts the process from a state to an entirely different one. A sudden insight is perceived structuring the subsequent design steps to achieve a goal that has not been defined before, redefining the design problem. The unintended combination of different ideas is important in the emergence of unprecedented high-quality solutions provided that responses to the functional, conceptual and configurational requirements. The next sections look at the aspects of synthesis and creativity that have formulated and structured the design concept in each case.

Models of Synthesis and Creativity: Diversity and Originality

El-Khouly [27] presented an empirical study on the role of the procedural and contextual components in creative discovery in architectural design processes. The following models are taken from [27: 269].

Architect A

- **Early phase of initiation:** a variety of conceptual elements are outlined.
- **Designing phases:** five different conceptual forms, where each form was developed through operational and finalisation stages.

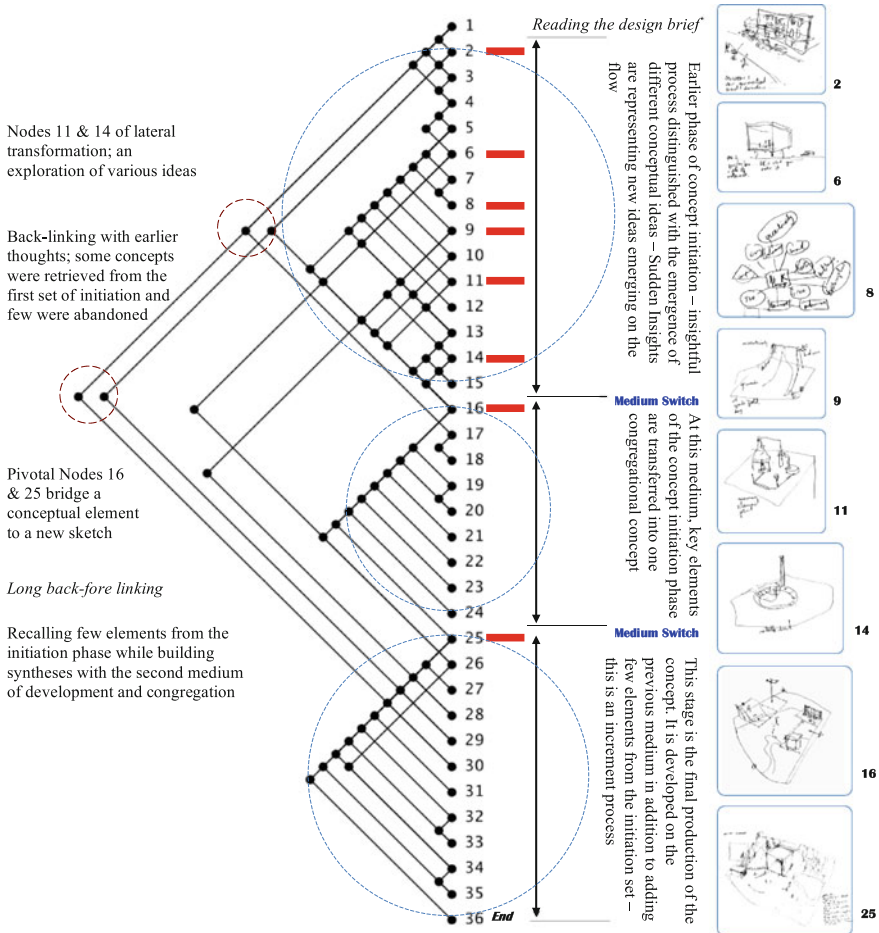


Fig. 5 The linkography protocol for the design process (Architect B)

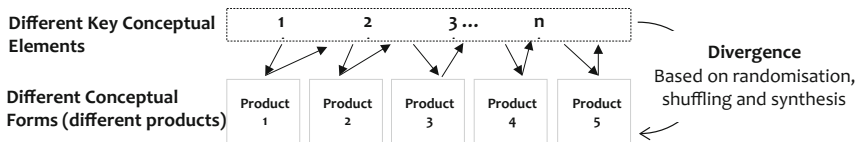


Fig. 6 An illustration of the divergence model

- **Model 1: Divergence model based on building synthesis** between different conceptual elements is achieved through; refer to Fig. 6:
- *Bottom-up process*: initiating the concept is based on independent units of conceptual elements and building synthesis.

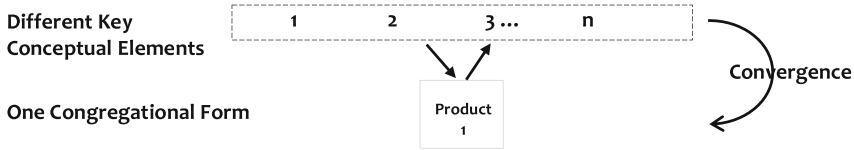


Fig. 7 An illustration of the convergence model

- *Randomisation and shuffling elements*: creating high variation and diversity between concepts of the final products.
- *Proposals are developed based on high uncertainty* (uncertainty is the motivation for exploration and creativity).

Architect B

- **Early phase of initiation**: different conceptual elements.
- **Designing phases**: one prime conceptual form, followed by operational and finalisation stages.
- **Model 2**: Convergence model is achieved through; refer to Fig. 7.
- *Design process*: initiating the concept is based on independent concepts and creating convergence into one conceptual form.
- *Convergence* into one framed design problem.
- *Proposal is developed with less uncertainty*.

The difference between both architects can be easily distinguished through their designing styles where architect A created back- and fore-linking between the variety of design episodes at earlier and later events, architect B adopted a convergent approach by integrating the conceptual episodes into one configuration at the end. Both styles are reflected on the linkography patterns, while the pattern looks structured by long back and fore linkages in the first case; see Fig. 3, it reflects two converging events in the second; see episodes 16 and 25 in Fig. 5 where most conceptual elements are brought into one composition.

From the linkographs that were constructed for the design experiments, two poles can be identified to distinguish linkography patterns: *diversification* and *integration*. While diversification reflects *divergence* and *distinctiveness* of design ideas, integration reflects *articulation* and *convergence*.

Identifying the Level of Contribution to Creativity of Design Episodes

The results of the observed study reflect three main categories for the levels of contribution to design creativity of critical moves and sudden insights:

- (1) Paradigm-preserving actions that reframe the solution, subcategorised as incremental advancement, replication or redefinition of the design situation.
- (2) Paradigm-rejecting actions that restructure the design problem. Actions move the field in a new direction from an existing or pre-existing starting point and are subcategorised as redirecting or reconstructing the situation. The actions might also move the field in a new direction from a totally new starting point—reinitiation.
- (3) Paradigm-integrating actions that attempt to ‘integrate’ multiple current paradigms. Creative design moves and sudden unprecedented solutions are most likely to emerge from the unexpected combination of synthesis. Actions are categorised as convergence or integration.

These results confirm those of Sternberg [28]; however, from the empirical study, some contributions may be added to his taxonomy. In some design cases, there are actions that move the situation from one idea to two different options: regenerating, reproducing various forms of solutions. Rather than initiating an action that moves the design towards a different direction from its current position (as in *redirection*), the designer might suggest moving to a different direction from a different point in the multidimensional space of design. Creative architects often question their original assumptions and begin again from a point that may stimulate different assumptions. This creative quality may lead to a paradigm shift in the design process.

Types of Evolving Actions and Creative Insights

Two prime types of design actions can be observed from the enclosed empirical study—as also confirmed in a previous work [29]:

1. *Bridging* actions: transfer information from one idea (design episode) to the next, which might cause collision between an old thought and the current situation leading to the emergence of unintended creative solution. The highlighted events in yellow in the linkography protocols Figs. 3 and 5 show traces of conceptual sketching elements that have been recalled to subsequent design situations to develop the concept in both design cases.
2. *Disconnecting* actions: the emergence of sudden insights could result in shifting the flow to a completely different state. In this situation, the new paradigm radically shifts the process, leading to disconnection of the pattern of design synthesis. Two or more separate chunks appear in the linkograph, embracing *disconnection* nodes within its pivotal structure; see Fig. 3.

The Architect's Idiosyncrasy and Identification of Design Actions

According to our ethnographic observations on both architects, a design episode can be inferred when the concept is crystallised through the presentation of a functional or conceptual element, the emphasis on a component on the design composition, or the projection to study a new dimension to the idea or the architectural configuration; whether functional or form composition, through the triadic drawings (plan, section, elevation), or the situations where the designer flips the tracing sheet over and rearranging the configurational relationships among the functional and composition elements. More on these idiosyncrasies will be presented in a separate study.

Length of Incubation Period in the Evolution of Ideas

To understand the context behind the emergence of creative insights, it is important to look at the length of *incubation* period (incubation of ideas) versus length of *externalisation* period (externalisation and drafting—execution). Incubation occurs when the design problem is set aside temporarily after an impasse is reached, and helps to achieve the sudden unconscious realisation of ideas. Sudden insights reflect a subconscious process, whereas incremental insights reflect conscious actions. Conscious actions are reflected by the interrelated chunk of patterns in the linkograph appearing as a direct dialogue with the sketch. Sudden insights emerge when the unconscious action collides with the conscious state and a longer period of incubation makes the collision more effective in disconnecting the linkograph.

From the case studies, it appears that incubation time produces better coherence to synthesise the conceptual elements in this phase. The relation between the length of incubation period and externalisation and operational phases of design, e.g. drafting, sketching or detailed drawing, can be outlined in three models:

1. Concept initiation has a short incubation time: the architect rapidly externalises the idea and design drawings.
2. Concept initiation has a long incubation time: the architect spends more time thinking of possible concepts before executing the idea.
3. The design process alternates between incubation and execution phases owing to emergent creative insights.

The design process becomes more structured as more insights emerge and stimulate the exploration of different options. Setting the goals in advance and descending through the process to achieve them makes the process hierarchical. Bottom-up design process makes it transformative. However, a cascade of unrelated, independent insights might cause hyper-stimulation, where the architect shifts from one state to another without developing the original concept.

Discussion

Two modes of reasoning can be identified from the case studies: rational top-down and non-rational bottom-up. In the *rational top-down* approach, the early phase of concept initiation often produces one original idea that is probably dependent on the instructions in the design brief. The attempt to solve the design problem relies on the functional programme. The designer who adopts a rational approach takes actions that break down the main problem into subsets and configures the design from the whole to the parts. Subsets of problems are related to the generic relation between form and function; once laid down, it is difficult to rearrange the matrix of relations for the overall configuration of form and function. In this way, the evolution of ideas probably results from the incremental reasoning of procedural components to execute predefined goals. In the *non-rational bottom-up* approach, the relationships between various conceptual 'seeds' are designed together to create the design configuration early in concept initiation. The unexpected discovery of ill-defined syntheses may result from trial-and-error attempts leading to critical moves. Unexpected creative insights may lead to redefining the goals, reformulating the whole configuration; this might even lead to restructuring the design problem and exploring the design space to generate the best solution. In one approach to find a 'good' concept, the designer may create various combinations between the sketching episodes of conceptual elements, achieving unexpected combinations of design configurations. Shuffling and reshuffling the matrix of relations between the functional elements and morphology of forms could lead to sudden creative insights.

The process of generating creative ideas is linked to understanding the context of the design situation and attempting to build on it by, for example, introducing modern vocabulary elements or innovative concepts for functional or morphological features, or synthesising prototypes of solutions for the type of building. The designer tries to generate ideas for creative solutions, which may lead to reliance on procedural or contextual components depending on the objectives of this stage.

Procedural components are the subsets of solving the design problem for synchronic concept development or implementation. *Contextual* components relate to the whole design problem for concept synthesis of back- and fore-linking between the diachronic stages of the design process. It may be concluded that sudden mental insights are likely to occur in a non-rational reasoning process of creative discovery based on unexpected syntheses. Variation in their sketching skills affected how the participating architects interpreted the resulting artefacts at each stage. The reflection-in-action, perceptions and gradual transformation differed for each architect: one might rely on a single sequence of sketching while another might move the flow from one idea to another to generate as many ideas as possible. Discontinuity while sketching is a drive for creative thinking: the unexpected discovery depends on the architect's rational or non-rational reasoning, reflected in how he or she sketches and transforms ideas from one stage to another.

Conclusion

The design process differs according to how it is affected by *procedural* or *contextual* components. Design is a *hierarchical* process when the problem-solving depends on significant involvement of procedural components to execute the concept and generate the solution through systemic actions. Each emergent act of reasoning is visited only once to execute a particular entity or form linear thinking and executing. *Procedural* components are stage-based and problem-oriented. Regardless of the design situation, the solution is generated according to certain actions in the designer's mind. *Procedural* components are based on the abstraction and analysis of the problem structure leading to the generation of various solutions from which a choice is made. Based primarily on the formulation of a solution-neutral problem statement, the reliance on procedural components suggests that the final design will be dependent more on logical deduction than on previous experience. In contrast, design is a *transformational* process when the generation of ideas is based on *contextual* components and the environment, and take into account reflection-in-action to transform mental imagery from one state to another. *Contextual* components are 'action-centric', design the situation from 'content-based' decisions, and affect how designers perceive and experience the problem. Addressing the designer's perception of the emerging problem (to identify the interim goal and generate a potential action for the next step) reveals the core nature of design activity, which exposes a shortfall in procedural components.

This paper has proposed an analytical method to identify the degree of influence of creative moves and *sudden mental insights* on the evolution and development of ideas in the design process by studying how two architects design for the same design brief. It was possible to identify the basic structural units in the reasoning process across the freehand sketching process for what was going on in each architect's mind in relation to practical ideas, and then capture the gradual transformation from each sketching episode to another and classify the design moves into two major categories: one in which *the prevailing idea is preserved*; and one that *resists the prevailing idea* by producing a new situation for an entirely new idea or by restructuring the design problem and generating an alternative solution in a different context.

The degree of influence and effect for the emergent actions in the reasoning process (creative quality) is determined by the value added to the evolution of the design concept to produce subsequent interim products. In the first category, which accepts the prevailing concept, one type of creative quality may be the displacement of the concept, known as 'forward incrementation'. Another, known as 'redefinition', may redefine the present situation but from another perspective, while still preserving the original prevailing concept; and a third type replicates the same idea: 'replication'.

Three types of creative qualities belong to the category of actions that reject the prevailing concept and attempt to replace it: one redirects the concept towards a different direction: 'redirection'; a second attempts to move the field of reasoning to where it once was—a reconstruction of the past—so that the flow may move onward from that point but in a different trajectory from the original one: 'reconstruction';

and a third type reinitiates the design concept by shifting the field of reasoning to a different starting point and then moves the design from that point: 'reinitiation'.

There is a third category of actions that attempt to integrate, merge two ideas and build synthesis between different seeds of conceptual elements: *integration*. While agreeing with [28] that human cognitive production cannot be limited to the reductionist view of being either 'creative' or 'non-creative', the research outlined in this paper contributes empirical proof for Sternberg's propulsion theory of creative contribution, which adopts a viewpoint that every mental product has a degree of creativity to some extent; a design move makes a creative contribution in the design reasoning process. The results outlined here are based on only a few cases, but further research has been extended to other design experiments, which could not be presented here for reasons of space and will be presented in subsequent studies.

Acknowledgements The study associated with this paper was conducted while pursuing my doctorate research fellowship at the Bartlett Faculty of the Built Environment, University College London (2007–2015) under the supervision of Profs. Alan Penn and Satu Teerikangas. I am thankful for their constructive advice and gratitude goes to the architects who took part in this study.

References

1. Piaget J (1971) Structuralism. Harper Torchbooks, New York
2. Goldschmidt G, Weil M (1998) Contents and structure in design reasoning. *Des Issues* 14:85–100
3. Penn A (2008) Architectural research. In: Knight A, Ruddock L (eds) *Advanced research methods in the built environment*. Wiley-Blackwell, Oxford, pp 14–27
4. Simon HA (1969) *The sciences of the artificial*. MIT Press, Cambridge, MA
5. Schön DA (1983) *The reflective practitioner*. Harper Collins, New York
6. Finke RA, Ward TB, Smith SM (1992) *Creative cognition: theory, research, and applications*. MIT Press, Cambridge, MA
7. Ward TB, Smith SM, Finke RA (1999) Creative cognition. In: Sternberg RJ (ed) *Handbook of creativity*. Cambridge University Press, Cambridge, UK, p 189–212
8. Perkins RE (1981) *The Nature and Origins of Boredom*. Thesis (Ph.D.), University of Keele
9. Ward TB (1994) Structured imagination: the role of category structure in exemplar generation. *Cogn Psychol* 27:1–40
10. Weisberg RW (1986) *Creativity, genius and other myths*. W.H. Freeman, New York
11. Bateson G (1979) *Mind and nature: a necessary unity*. Bantam Books, New York
12. Findlay CS, Lumsden CJ (1988) The creative mind: toward an evolutionary theory of discovery and innovation. *J Biol Soc Struct* 11:3–55
13. Johnson Laird PN (1988) *The computer and the mind*. Harvard University Press, Cambridge, MA
14. Schön DA (1963) *The invention and the evolution of Ideas*. Tavistock Publications, London
15. Weisberg RW, Alba JW (1981) An examination of the alleged role of "fixation" in the solution of several "insight" problems. *J Exp Psychol General* 110(2):169–192
16. Metcalfe J (1986) Feeling of knowing in memory and problem solving. *J Exp Psychol Learn Mem Cogn* 12:288–294
17. Metcalfe J, Weiße D (1987) Intuition in insight and non-insight problem solving. *Mem Cogn* 15:238–246
18. Weisberg RW (1993) *Creativity: beyond the myth of genius*. W.H. Freeman, New York

19. Chiang Y-C (2006) The role of design eureka in design thinking. Paper for WonderGround: design research society international conference 2006, Lisbon, 1–4 Nov 2006.
20. Goldschmidt G (1990) Linkography: assessing design productivity. In: Trappl R (ed) *Cybernetics and system '90*. World Scientific, Singapore, pp 291–298
21. Goel V (1995) *Sketches of thought*. MIT Press, Cambridge, MA
22. El-Khouly T, Penn A (2012a) Order, structure and disorder in space syntax and linkography: intelligibility, entropy, and complexity measures. In: *Proceedings of 8th space syntax symposium, Santiago De Chile, (SSS8 2012)*, pp 8242:1–22
23. El-Khouly T, Penn A (2012) On an integrated analytical approach to describe quality design process in light of deterministic information theory. In: Gero J (ed) *Design computing and cognition '12*. Springer, Dordrecht, pp 451–470
24. Akin Ö, Akin C (1996) Frames of reference in architectural design: analysing the hyper-acclamation (Aha!). *Des Stud* 17:341–361
25. Cross N (1997) Creativity in design: analyzing and modeling the creative leap. *Leonardo* 30(4):311–317
26. Goldschmidt G (1991) The dialectics of sketching. *Creativity Res* 4:123–143
27. El-Khouly TAI (2015) *Creative discovery in architectural design processes: an empirical study of procedural and contextual components* (Doctoral dissertation), University College London. <http://discovery.ucl.ac.uk/1467244>
28. Sternberg RJ (2003) *Wisdom, intelligence, and creativity synthesized*. Cambridge University Press, Cambridge, UK
29. El-Khouly T, Penn A (2013) Directed linkography and syntactic analysis: comparing synchronous and diachronic effects of sudden emergence of creative insights on the structure of the design process. In: Kim YO, Park HT, Seo KW (eds) *Proceedings of the ninth international space syntax symposium*. Paper 59. Sejong University, Seoul

The Effect of Tangible Interaction on Spatial Design Tasks



Jingoo Kim, Mary Lou Maher and Lina Lee

Tangible user interfaces (TUIs) enable physical affordances that encourage the spatial manipulation of multiple physical objects to interact with digital information. We claim that the affordances of tangible interaction can affect design cognition on spatial tasks. While many researchers have claimed that TUIs improve spatial cognition, there is a lack of agreement about what improve means and a lack of empirical evidence to support the general claim. While most cognitive studies of TUIs focus on a comparison of tangible and traditional GUI keyboard and mouse interaction, we focus on comparing the use of TUIs on spatial versus nonspatial design tasks to validate the claim that tangible interaction specifically affects spatial design tasks. The results show that TUIs encourage users to perform more epistemic actions, leads to unexpected discoveries, and off-loads spatial reasoning to the physical objects. We conclude that the positive impact of tangible interaction is more dominant in spatial design tasks than nonspatial design tasks.

Introduction

Tangible user interfaces (TUIs) have significant potential to support problem-solving during design tasks due to the physical and spatial characteristics of graspable interaction. In this paper, we focus on how the affordances of tangible interaction influence design cognition and support creativity when performing spatial design tasks, while combining spatial elements using tangible user interfaces. Spatial problem-solving involves acknowledging and accounting for the characteristics relating to the position, area, and size of things. These spatial characteristics in spatial problem-solving can be projected onto physical objects through the affordances of TUIs, thereby facil-

J. Kim (✉) · M. L. Maher · L. Lee
University of North Carolina at Charlotte, Charlotte, USA
e-mail: jkim168@uncc.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_8

itating the consideration of alternative spatial characteristics during design. There is evidence that gesturing with our hands aids thinking [1–3]. With this basic premise, this paper is concerned with understanding how tangible interaction effects users' creative cognition in different design contexts, what kinds of design problems are more affected by the use of tangible interaction, and how tangible interaction affects design cognition.

In this paper, we report on a study of how the affordances of tangible interaction have different cognitive effects in spatial tasks and nonspatial tasks. We claim that the use of tangible interaction in a spatial design task may be positively associated with creative design processes.

Tangible Interaction Studies

In TUIs, the user interacts with the system with multiple tangible objects, which are graspable by hands, and accordingly, the manipulation of these objects causes changes to the state of the digital system. There have been numerous studies of the differences in TUIs and non-tangible interaction. Fjeld and Barendregt [4] studied epistemic actions using three spatial planning tools with different degrees of physicality: no physical interaction, some physical interaction, only physical interaction as well as the potential relations between the three traditional measures of usability: efficiency, effectiveness, and satisfaction. This research shows that physical interaction offers a great deal of support for epistemic actions in the physical world [4]. Fleming and Maglio [5] compare tangible and non-tangible letter combinations and show that physical activity effectively complements internal, cognitive activity, providing a reliable way to simplify a search, explore the space of letter combinations, and identify potential words.

Tangible interaction provides physical affordances that are associated with spatial reasoning. Antle et al. [6] investigated the similarities and differences between how children solve an object manipulation task using mouse-based input versus tangible-based input. They provide evidence that direct physical manipulation of objects in a spatial problem-solving task supports children's ability to mentally solve the task through iterations of exploratory (largely epistemic) and direct placement actions. Maehigashi et al. [7] studied the influence of 3D images and 3D-printed objects on spatial reasoning. The study indicated that using a 3D-printed object produced more accurate task performance and faster mental model construction. Johannes et al. [8] show that tangible 3D molecular models help students develop a deeper understanding of core concepts in molecular biology. Smithwick and Kirsh [9] studied the cognitive role of tangible objects for architects engaged in design thinking. They show that the effects of tangible interaction influence the design space as it expands; leading architects to more divergent thinking and the physical interaction broadens the basis of creativity.

While many researchers have claimed that TUIs improve spatial cognition, there is a lack of research on comparing spatial and nonspatial tasks within a creative context. Although the precedents show potential impact of tangible interaction on

spatial tasks, questions arise concerning the differences when engaged in spatial and nonspatial design contexts. Therefore, this study investigates the impact of TUIs when the user is engaged in creative tasks, comparing spatial design tasks with nonspatial design tasks. Two previous studies [10, 11] influenced our experiment design are described in the following section.

Effect of TUIs and GUIs on a Spatial Design Task

Kim and Maher [10] studied how TUIs affect design cognition and found that tangible interaction affects the design process by increasing participants' "problem-finding" behaviors, a characteristic of creative design. Kim and Maher [10] compared participants engaged in a design task in two conditions: using TUIs on a tabletop system with 3D blocks to designers using GUIs on a desktop computer with a mouse and keyboard. These conditions are illustrated in Fig. 1. The participants were asked to perform a spatial configuration task by locating furniture and wall objects in a multi-purpose living and working space. They approached the study of spatial cognition in designing from three different perspectives: action, perception, and process. Based on these perspectives, they hypothesized about epistemic actions, gesture actions, spatial relationships, and design process as follows:

- H1: The use of TUIs can change designers' 3D modeling actions in designing—3D modeling actions may be dominated by epistemic actions.
- H2: The use of TUIs can change designers' gesture actions in designing—gesture actions may serve as complementary functions to 3D modeling actions in assisting in designers' cognition.
- H3: The use of TUIs can change certain types of designers' perceptual activities—designers may perceive more spatial relationships between elements, create more, and attend to new visuospatial features through the production of multiple representations.
- H4: The use of TUIs can change the design process—the changes in designers' spatial cognition may increase problem-finding behaviors and the process of re-representation, which are associated with creative designing.

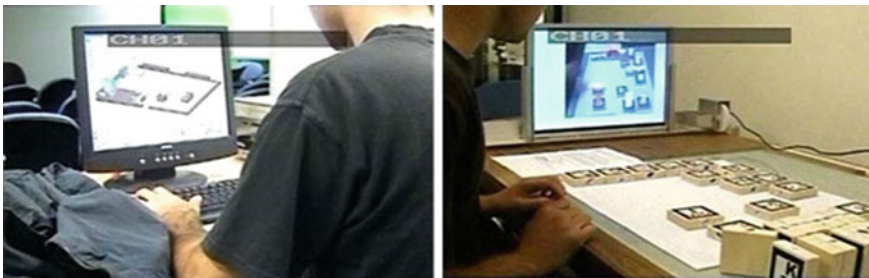


Fig. 1 Kim and Maher [10] experimental conditions: GUI with mouse and keyboard manipulation of objects (left) and TUI with 3D blocks manipulation of objects (right)

Through the validation of hypotheses using a protocol analysis, the study found that TUIs change designers' spatial cognition and these changes are associated with creative design processes. Specifically, the study shows the following:

1. The TUI condition produced more epistemic actions revealing information that is hard to compute mentally.
2. The TUI condition produced more immersive gestures using large hand movements assisting in designers' perception.
3. The TUI condition improved designers' perceptive ability for new visuospatial information on spatial relationships.
4. The participants using TUIs spent more time reformulating the design problem by introducing new functional issues.
5. During the TUI condition, the participants experienced "unexpected discoveries" and the process of re-representation.

In conclusion, the affordances of TUIs, such as direct manipulability and physical arrangements, when compared to the GUI condition, may reduce cognitive load associated with spatial reasoning, resulting in enhanced spatial cognition and creative cognition.

Effect of TUIs and Pointing on a Nonspatial Design Task

Maher et al. [11] and Clausner et al. [12] explored hypotheses on the effects of tangible interaction on creativity by comparing tangible interaction to non-tangible interaction while performing a word combination task. In their analysis, creativity is associated with ideas that are new, surprising, and valuable. Maher et al. [11] describe an experiment to measure differences between gesture and problem-solving actions in two conditions: a poster condition (no tangible interaction) and a cubes condition (tangible interaction) as illustrated in Fig. 2. In their experiment, participants are asked to make word combinations from a set of six nouns and give them meaning. They compared the participants' gestures and actions such as pointing in the two conditions. They explored the following hypotheses on the impact of tangible interaction on creative cognition.

- H1: Tangible interaction increases epistemic actions when compared to pointing interaction.
- H2: Tangible interaction encourages more fluid body movement than pointing interaction.
- H3: Tangible interaction encourages more collaborative actions than pointing interaction.
- H4: Tangible interaction elicits more functional and behavioral exploration than pointing interaction.
- H5: Tangible interaction induces use of both hands.



Fig. 2 Maher et al. [11] experimental conditions: non-tangible poster condition (left) and tangible cube condition (right)

Their analysis shows that tangible interaction encourages more epistemic actions, fluid body movement, and the use of both hands; meaning the cubes condition encouraged more gesture and action than the poster condition.

1. Tangible interaction encourages more epistemic actions.
2. Tangible interaction encourages more fluid body movement.
3. Tangible interaction encourages the use of both hands.

However, the two hypotheses that relate to collaboration (H3) and abstract thinking (H4) are not strongly associated with tangible interaction when compared to pointing interaction. The results of this study show that TUIs encourage more epistemic actions but not necessarily more creativity, when engaged in a word combination task.

The Effect of TUIs on Spatial Versus Nonspatial Design

In developing our experiment design, we extend the previous studies [10, 11] because the studies compared tangible and not tangible conditions within either a spatial or nonspatial design task. They do not provide insight into the comparative effect of TUIs on spatial versus nonspatial tasks. The most common issues that the previous studies discussed were the effect of tangible interaction on encouraging epistemic actions, gestures, and new functional issues that can influence the design process. This leads one to question whether the effects of tangible interaction on design cognition depend on the nature of the task: spatial or nonspatial task. Our goal in this paper is to identify the effect of tangible interaction in different creative tasks through the comparison of a spatial and nonspatial task. A comparison of the previous two experiment designs with the experiment design in this paper is shown in Table 1.

We assert that the affordances of TUIs encourage more cognitive actions associated with creativity in spatial design tasks than in nonspatial design tasks. Spatial thinking is dependent on the capacity to understand and remember the spatial relationships between objects [13]. A spatial design solution is synthesized by constructing

Table 1 Comparison of experiment designs

	Kim and Maher [10]	Maher et al. [11]	This paper
Comparison	Different condition and same task	Different condition and same task	Same condition and different task
Condition	TUI versus GUI (3D blocks versus mouse and keyboard)	Tangible versus pointing (cubes versus poster)	Tangible (graspable shapes and letters)
Task	Spatial design (spatial planning)	Nonspatial design (word combination)	Spatial versus nonspatial (shape combination versus letter combination)

and restructuring the spatial configuration of elements. TUIs enable the spatial thinking associated with synthesis to be transferred to physical objects in the environment. By off-loading to the environment, we claim that spatial interactions with tangible elements will affect the search process, encourage creativity, and facilitate spatial problem-solving. We have developed the following hypotheses about the effect of tangible interaction on creative spatial tasks.

- H1: Tangible objects encourage more epistemic actions in spatial design tasks than in nonspatial design tasks.

Epistemic actions are an exploratory motor activity to uncover information that is difficult to compute mentally. In contrast, pragmatic actions are a motor activity that directs the user closer to the final goal [14]. We are interested in the argument that epistemic actions can be used to reduce the memory, time, and probability of error of internal computation [14]. According to Kim and Maher [10], tangible objects facilitate a spatial design search process. A search process which finds a possible configuration of design elements is a major problem-solving activity relying on the spatial representation.

- H2: The affordances of TUIs facilitate more unexpected discoveries in spatial design tasks than in nonspatial design tasks.

Creativity in the design process is often characterized by an event occurring as a sudden insight which the designer immediately recognizes as significant [15]. TUIs provide more opportunities to externalize representations through ease of manipulation. This characteristic can affect the occurrence of sudden “insight” to find key concepts for a creative design. If the spatial design task involves more unexpected discoveries, we argue that tangible interaction affects the creative design process.

- H3: The affordances of TUIs facilitate more epistemic actions when speech production stops in spatial design tasks than in nonspatial design tasks.

People communicate and externalize their thoughts by speech production and action. If the spatial task is dominated by epistemic actions when speech production is stopping, we argue that TUIs encourage off-loading the spatial reasoning onto the

physical objects. There is evidence that gesturing with our hands promotes learning [16], and aids problem-solving [17]. When a problem occurs during the process of a design, the user will show more body movements in spatial tasks than in nonspatial tasks.

Experiment Design

Our experiment is a within-subjects design that compares pairs of participants engaged in two different tasks performed in a tangible interaction condition: a spatial design and a nonspatial design task. We collected video data during task performance and performed a concurrent protocol analysis: We include our observations of the participants in our analysis to augment the video data. The protocol including the informed consent document has been reviewed and approved by the UNCC institutional review board (IRB) and we obtained the informed consent from all participants to conduct the experiment.

The experimental task is a design task of combining prescribed graspable components. The spatial design task is to design a logo for a housing community mobile app by combining a given set of shapes. Participants are given seven geometric shapes. The nonspatial design task is to design a name for a new recipe app by combining a given set of letters. Participants are given five keywords which consist of individual letters printed on blocks. This task requires a selection and ordering of letters for making a meaningful name for an app.

For participants, we recruited architecture students because they are exposed to design tasks in their design studio courses and are familiar with the spatial reasoning skills required by the tasks. The experiment is a within-subject design with $n = 11$. We recruited 22 participants for 11 pairs.

The procedure consisted of a training session followed by two design task sessions. In the training session, the facilitator explains the spatial design and nonspatial design tasks to the pair of participants and shows a simple example for each. After the training session, the two design tasks are performed with a counterbalanced order. Participants were asked to think aloud and were naturally speaking during the task since the task is performed in pairs of participants. Participants were given 5 min to perform each design task. The facilitator does not interfere in the design process except to remind the participants to verbalize their thoughts if the participants do not think aloud for over 1 min (Figs. 3 and 4).

Segmentation and Coding

Segmentation and coding of protocol data are determined as a consequence of the hypotheses. Since our hypotheses are based on a claim about actions using tangible objects, a primary consideration is the percentage of time the participants were

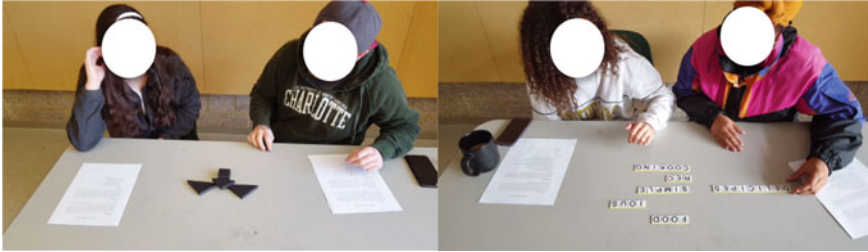


Fig. 3 Spatial design task (left) and nonspatial design task (right)

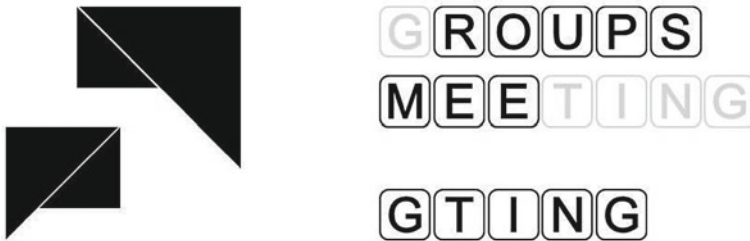


Fig. 4 Design tasks: spatial logo design (left) and nonspatial name design (right)

engaged in actions of interest, such as epistemic versus pragmatic actions. Accordingly, we segmented the video stream into equal time segments before coding the segments. After observing the typical duration of an epistemic action, we decided to segment the video data into equal 2 s segments. When an action is not maintained for the entire 2 s interval, the action that occurred for more than 1 s is coded. Our basic coding scheme is “action,” recorded as epistemic, pragmatic, or no action. After this code is applied to each segment, we coded the segments again to identify unexpected discoveries and whether the participants are talking versus not talking. Two of the authors performed the coding together for three of the sessions, to ensure agreement on the coding scheme. Then a single author coded all sessions twice, separated by a period of several days, followed by an arbitration process to identify and resolve differences in coding.

Kim and Maher [10] and Maher et al. [11] coded their data as epistemic versus pragmatic actions. These two studies describe epistemic and pragmatic actions as follows. “An example of epistemic action is the way novice chess players find it helpful to physically move a chess piece when thinking about possible consequences. Epistemic actions offload some internal cognitive resources into the external world by using physical actions. In contrast, pragmatic actions are motor activity directly aimed at a final goal” [11].

We distinguish epistemic actions and pragmatic actions in this study as follows. Epistemic actions (E) are counted as the total number of segments in which the participant is moving pieces to find a possible design. Pragmatic actions (P) are identified when the participants use tangible objects to show an intended design with

goal-directed actions. For example, if the pair of participants decides on “Foodie” as a name, then made the arrangement of letters without any change, the duration of action was coded as P. If the pair of participants only discussed without any action, we coded the duration as no actions (NA). We coded other gestures and actions not directly associated with making a design as NA. Examples include touching, grasping for, and pointing at pieces, because the intention of this coding scheme is to distinguish the pragmatic and epistemic actions.

We coded unexpectedness as a creative event. Creative events emerge as key concepts from a sudden insight in the design process [15]. The intention of this coding is to identify the creative events and actions leading to an unexpected discovery and sudden insight, which the participant immediately recognizes as significant. We associate unexpected discoveries with creativity in the design process.

Unexpected discoveries in this study are identified in two ways: emergent designs (ED) and unexpected spatial arrangements (US). Emergent designs (ED) are identified from the analysis of the verbal transcript and are associated with words like “Oh! Do you see that?” The duration of the design session in which the participants are seeing ED is the duration of speech production time that is derived from the total number of segments in which the participant is talking about an emerging concept. When a design idea emerges from an exploration of moving pieces, we coded the actions as ED. An example of an emergent design is when the participants discovered “mood” from arranging “doom” as a name. We considered both speech and actions as the basis for a duration of ED.

Unexpected spatial arrangements (US) are identified when the participants use the tangible objects in unexpected ways, such as stacking or making a shadow. US can occur by either a sudden insight during exploration of moving pieces, that is during epistemic actions, or from the participant’s thoughts, and the arrangement is made as a pragmatic action. We intentionally coded for ED and US, then coded the remaining segments as expected behaviors (EX).

We coded each segment as talking (T) which is when the participant talks alone or to the partner, and no talking (NT) which is when the participant does not talk more than 5 s. There are two types of external representations while the participants are performing the design tasks: speech productions and actions. A TUI allows participants to externalize a representation of the solution onto physical objects, which Alibali et al. [18] claims facilitates interaction and reduces cognitive load. The purpose of this coding scheme is to record the participants’ actions on the tangible objects when they are not producing speech. We then compare the actions during the no talking segments to analyze the differences in epistemic and pragmatic actions for spatial and nonspatial design tasks.

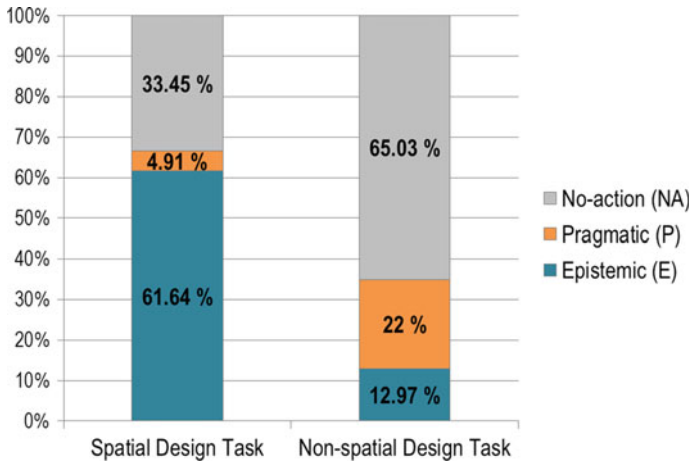


Fig. 5 The total percentage of segments; epistemic actions, pragmatic actions, and no actions

Analysis and Interpretation of Coded Data

In this section, we report the analysis of the coded data that are relevant to each hypothesis for the 11 pairs of participants. We interpret our analysis with respect to our hypotheses and provide examples of participant behaviors from our observations.

H1: Tangible objects encourage more epistemic actions in spatial design tasks than in nonspatial design tasks.

In Fig. 5, we show the average across all sessions for the epistemic, pragmatic, and no action coded segments in the two conditions: spatial design task and nonspatial design task. In the spatial design task, the largest amount of the total time (61.64%) is coded as “epistemic” action. In the nonspatial design task, the largest amount of the total time (65.03%) is coded as “no action.” In the spatial design task, epistemic actions are dominant, and the participants generate design alternatives while utilizing tangible objects: the participants engage in epistemic actions a larger amount of time in the spatial design task and they engage in pragmatic actions a larger amount of time in the nonspatial design task, as shown in Table 2.

The total percentage of time the participants are engaged in epistemic actions is 61.6% during the spatial design task. Participants try to find the solution for the given task by changing or realigning the positions of the tangible pieces. The participants were given a set of shapes, from which they were asked to design a logo. Various design alternatives emerged, in part, from the spatial characteristics and arrangements of the shapes. Among the 11 sessions in which participants were engaged in spatial design tasks, pragmatic actions did not occur for pairs *P6*, *P7*, *P8*, and *P9*, as shown in Table 2.

In contrast, the total percentage of time the participants are engaged in epistemic actions during the nonspatial design task is 13%. The nonspatial design task is to

Table 2 Epistemic, pragmatic, and no actions in both conditions (▲ indicates the higher value in the two conditions)

	Epistemic actions (E)		Pragmatic actions (P)		No actions (NA)	
	Spatial design task (%)	Nonspatial design task (%)	Spatial design task (%)	Nonspatial design task (%)	Spatial design task (%)	Nonspatial design task (%)
P1	52.0	0.0	7.3	23.3	40.7	76.7
P2	72.7	18.0	6.0	38.7	21.3	43.3
P3	70.0	0.0	3.3	21.3	26.7	78.7
P4	48.7	30.0	11.3	40.7	40.0	29.3
P5	60.0	13.3	5.3	8.7	34.7	78.0
P6	73.3	34.0	0.0	9.3	26.7	56.7
P7	60.0	6.7	0.0	17.3	40.0	76.0
P8	83.3	22.0	0.0	34.0	16.7	44.0
P9	69.3	9.3	0.0	5.3	30.7	85.3
P10	43.3	9.3	3.3	18.7	53.3	72.0
P11	45.3	0.0	17.3	24.7	37.3	75.3
Total	61.6	13.0	4.9	22.0	33.5	65.0
Total segment	2034 (1 h 7 m 48 s)	162 (5 m 24 s)	428 (14 m 16 s)	726 (24 m 12 s)	1104 (36 m 48 s)	2146 (1 h 11 m 32 s)

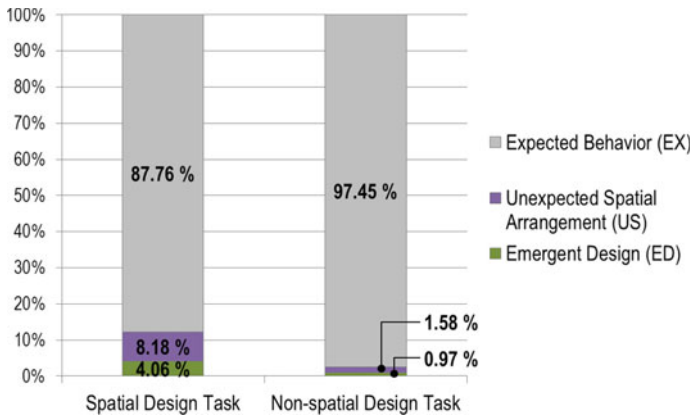


Fig. 6 The total percentage of segments; unexpected discovery, unexpected spatial arrangement, and expected spatial arrangement

design a new application name by creating a new word from the letters contained in existing words. Participants created possible word combinations in their heads, instead of moving their hands. From our observation, it was found that users distinctively think about a letter combination in advance, and merely express the outcome, instead of exploring possible alternatives of letter combination resulting from directly manipulating the given tangible letter pieces. Among the 11 sessions in which participants were engaged in nonspatial design tasks, epistemic actions did not appear at all for pairs *P1*, *P3*, and *P11*, as shown in Table 2. Goal-directed actions dominated in the nonspatial design tasks: participants combined words by moving each letter object in order to present the design alternative that the participant had identified by thinking about the task and looking at the letters.

A two-sample *t*-test was conducted to determine the significance of our results that tangible objects encourage more epistemic actions in spatial design tasks than in a nonspatial design task. Participants in spatial design task ($M = 0.61$, $SD = 0.13$) show more epistemic actions than nonspatial design task ($M = 0.12$, $SD = 0.11$), $t(20) = -9.11$, $p = 0.000000014$. Participants in spatial design task ($M = 0.22$, $SD = 0.11$) take less pragmatic actions than nonspatial design task ($M = 0.22$, $SD = 0.11$), $t(14) = -4.30$, two tail $p = 0.00072$. The percentage of time the participants were showing nonaction in the nonspatial design tasks is greater than in the spatial design tasks (spatial design task: $M = 0.33$, $SD = 0.10$; nonspatial design task: $M = 0.65$, $SD = 0.18$; $t(16) = 4.93$, two tail $p = 0.00015$).

H2: The affordances of TUIs facilitate more unexpected discoveries in spatial design tasks than in nonspatial design tasks.

As seen in Fig. 6, the total ratio of segments coded as emergent designs (spatial design task: 4.06%, nonspatial design task: 0.97%) and unexpected spatial arrangements (spatial design task: 8.18%, nonspatial design task: 1.58%) is higher in the spatial design tasks than the nonspatial design tasks.

An emergent design in the case of the spatial design task is shown in Fig. 7. The participant moved the tangible objects without any goal. On reflecting on the shape configuration, the participant combines shapes into the form of a box rolled into one by chance, and comes up with the “Home, Safe, Secure” concept from it. The participant said “Yeah could put a space between them. Then it becomes a community, and it can be all kinds of houses in the community.” By leaving a gap between figures, the participant develops the configuration into a community concept.

When examining the case of emergent design performed in a nonspatial design task, a participant associates “rice-cipe” with “king dicer.” By removing “R” from “Dicer” and replacing “Dice” which is the first letter with “R,” the participant comes up with a new word of “rice,” as shown in Fig. 8. As such, the case that a user came up with a new design idea based on the existing word without intention was discovered only twice (*P2*, *P8*) in 22 experiments in total, as shown in Table 3.

Figure 9 shows examples of unexpected spatial arrangements: stacking shapes (*P2*), placing shapes in 3D arrangements (*P5*), and the discovery of the shadow as an emergent property (*P2*).

Figure 10 shows examples of unexpected spatial arrangements in the nonspatial design task: the case of arranging letter pieces in the cross shape for *P4* and a new character, *T*, was created by *P1*. The participant came up with unexpected design solutions when there were not enough letter pieces or no alphabet letters.

Our results show that the affordances of TUIs facilitate more unexpected discoveries in spatial design tasks ($M = 0.12$, $SD = 0.12$) than in nonspatial design tasks ($M = 0.02$, $SD = 0.03$), $t(12) = -2.43$, two tail $p = 0.01$). However, due to the small number of unexpected discoveries in both conditions, further research is needed to validate H2.

H3: The affordances of TUIs facilitate more epistemic actions when speech production stops in spatial design tasks than in nonspatial design tasks.

As shown in Fig. 11, the amount of time the participants are talking and not talking are similar in both of the design tasks. When we examine the use of epistemic, pragmatic, and no action during the non-talking segments, there are significant differences in the two conditions. During spatial design tasks, the percentage of epistemic action was very high at 71.02%. During nonspatial design tasks, the percentage of time for no action is 80.8% as shown in Fig. 11. From our observations through the video data, a participant becomes quiet for these reasons: (1) the participant cannot think

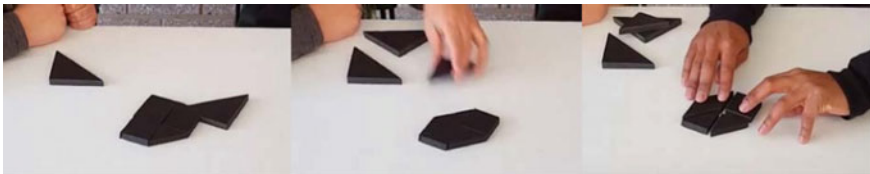


Fig. 7 Examples of emergent designs in spatial design task

Table 3 Emergent design, unexpected spatial arrangement, and expected spatial arrangement in both conditions (▲ indicates the higher value in the two conditions)

	Emergent design (ED)		Unexpected spatial arrangement (US)		Expected spatial arrangement (EX)	
	Spatial design task (%)	Nonspatial design task (%)	Spatial design task (%)	Nonspatial design task (%)	Spatial design task (%)	Nonspatial design task (%)
P1	11.3 ▲	0.0	7.3 ▲	6.7	81.3	93.3 ▲
P2	14.7 ▲	2.7	20.0 ▲	0.0	65.3	97.3 ▲
P3	0.0	0.0	31.3 ▲	0.0	68.7	100 ▲
P4	0.0	0.0	0.0	5.3 ▲	100	94.7
P5	4.0 ▲	0.0	18.7 ▲	0.0	77.3	100 ▲
P6	0.0	0.0	0.0	0.0	100	100
P7	0.0	0.0	2.7	2.7	97.3	97.3
P8	0.0	8.0 ▲	7.3 ▲	2.7	92.7 ▲	89.3
P9	0.0	0.0	0.0	0.0	100	100
P10	6.7 ▲	0.0	0.0	0.0	93.3	100 ▲
P11	8.0 ▲	0.0	2.7 ▲	0.0	89.3	100 ▲
Total	4.1 ▲	1.0	8.2 ▲	1.6	87.8	97.5
Total segment	134 (4 m 28 s)	32 (1 m 4 s)	270 (9 m 0 s)	52 (1 m 44 s)	2896 (1 h 36 m 32 s)	3216 (1 h 47 m 12 s)



Fig. 8 Examples of emergent design in nonspatial design task



Fig. 9 Examples of unexpected spatial arrangement in spatial design task

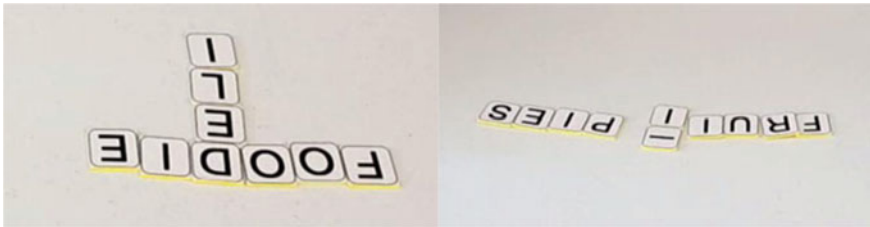


Fig. 10 Examples of unexpected spatial arrangement in nonspatial design task

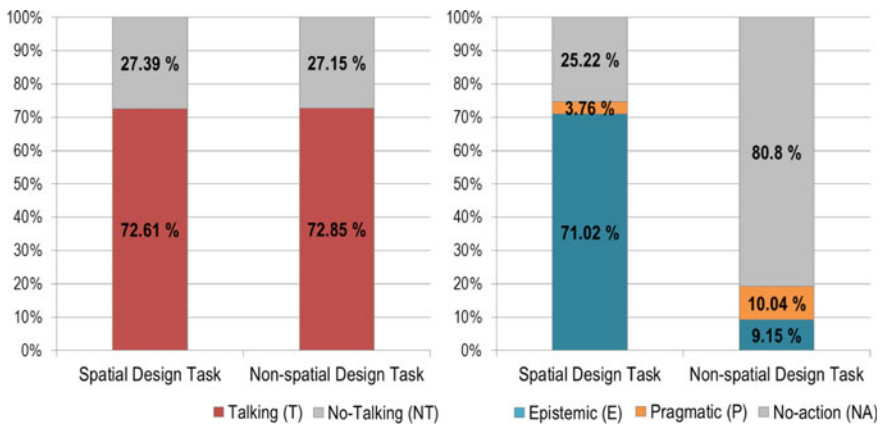


Fig. 11 The total percentage of segments; talking and no talking (left), the total percentage of segments in no talking; epistemic, pragmatic, and no actions (right)

of any new design alternatives or improvements on the current design alternative, (2) the participant is trying to understand the design of the other participant, or (3) the

participant is evaluating the quality of the current design alternative. Therefore, the non-talking section is closely related to the process of resolving the design.

With Hypothesis 1, we show that tangible objects facilitate the spatial design search process. Substituting this in H3, we investigated the frequency of the use of epistemic actions in the non-talking section, that is, in the process of resolving the design problem. Figure 12 shows the distribution of data values and central values in a total of 22 sessions. The box plot on the left shows the distribution of actions for all segments while the box plot on the right shows the distribution for only the non-talking segments. The overall flow and pattern of the two box plots are very similar. In other words, epistemic actions are prominent in spatial tasks and no actions are prominent in nonspatial tasks. Judging from much greater data variability, the lowered median value of epistemic actions and the higher median value of no actions in spatial tasks in the box plot on the left, it is noted that the user tries to find a design solution through various movements using tangible objects. The reason for examining the “No Talking” zone is significant because when design progress was being resolved, participants worked in silence and the tangible objects facilitated the participants’ spatial thinking.

In Table 4, we can see that in the case of P2, epistemic actions are the only action code for the non-talking segments regardless of engaging in a spatial or nonspatial design task. When seeing that epistemic action never appeared in the non-talking segments of five pairs (P1, P3, P7, P10, and P11) in nonspatial design tasks, it can be interpreted that tangible objects, in the case of letter pieces, encourage less spatial thinking during the search process. On the other hand, in the spatial design task, all the experiments show epistemic actions during the non-talking segments.

The affordances of TUIs facilitate more epistemic actions when speech production stops in spatial design tasks ($M = 0.75, SD = 0.19$) than in nonspatial design tasks ($M = 0.17, SD = 0.31$), $t(20) = -5.21$, two tail $p = 0.00004$. We conclude that TUIs

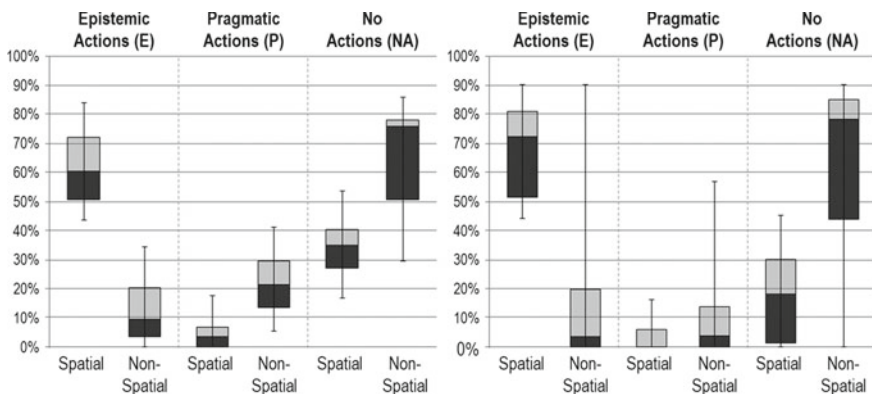


Fig. 12 Distribution of total segment by tasks type and actions; epistemic, pragmatic, and no actions (left), distribution of no talking segments by tasks type and actions; epistemic, pragmatic, and no actions (right)

Table 4 Epistemic, pragmatic, and no actions during no talking segments (▲ indicates the higher value in the two conditions)

	Epistemic actions (E)				Pragmatic actions (P)				No actions (NA)			
	Spatial design task		Nonspatial design task		Spatial design task		Nonspatial design task		Spatial design task		Nonspatial design task	
	(%)	▲	(%)	▲	(%)	▲	(%)	▲	(%)	▲	(%)	▲
P1	57.9	▲	0.0		15.8	▲	13.4		26.3		86.6	▲
P2	100		100		0.0		0.0		0.0		0.0	
P3	82.1	▲	0.0		1.8		4.2	▲	16.1		95.8	▲
P4	100	▲	40.9		0.0		27.3	▲	0.0		31.8	▲
P5	82.1	▲	7.5		17.9	▲	0.0		0.0		92.5	▲
P6	75.0	▲	36.4		0.0		0.0		25.0		63.6	▲
P7	56.1	▲	0.0		0.0		0.0		43.9		100	▲
P8	97.5	▲	3.7		0.0		63.0	▲	2.5		33.3	▲
P9	80.0	▲	4.9		0.0		4.9	▲	20.0		90.2	▲
P10	50.0	▲	0.0		0.0		17.1	▲	50.0		82.9	▲
P11	48.9	▲	0.0		11.1	▲	0.0		40.0		100	▲
Total	71.0	▲	9.2		3.8		10.0		25.2		80.8	▲
Total segment	642 (21 m 24 s)		82 (2 m 44 s)		34 (1 m 8 s)		90 (3 m 0 s)		228 (7 m 36 s)		724 (24 m 8 s)	

facilitate spatial problem-solving by off-loading cognition for spatial reasoning onto the physical objects.

Conclusion

This paper presents the results of an experiment to determine the effects of tangible interaction on creative spatial design tasks. The goal of the experiment is to compare the effect of tangible objects on problem-solving behavior in spatial and nonspatial design tasks. While other studies have shown that TUIs increase epistemic actions while performing tasks; in this paper, we focus on the relative impact of TUIs on spatial versus nonspatial design tasks. We explored three hypotheses relating graspable objects to the occurrence of epistemic actions and unexpected discoveries in spatial and nonspatial design tasks. We collected video data during task performance and coded the protocol data by the coding schemes based on actions and speech productions. Our results show that the participants in the spatial design task produced more epistemic actions (H1), more unexpected discoveries (H2), and more epistemic actions when speech production stops (H3). We conclude that TUIs are more effective in encouraging problem-solving behaviors associated with a creative design when the tasks require spatial reasoning.

References

1. Kessell A, Tversky B (2006) Using diagrams and gestures to think and talk about insight problems. In: *Proceedings of the meeting of the cognitive science society*, vol 28, p 28
2. Goldin-Meadow S, Beilock SL (2010) Action's influence on thought: the case of gesture. *Perspect Psychol Sci* 5(6):664–674
3. Graham TA (1999) The role of gesture in children's learning to count. *J Exp Child Psychol* 74(4):333–355
4. Fjeld M, Barendregt W (2009) Epistemic action: a measure for cognitive support in tangible user interfaces? *Behav Res Methods* 41(3):876–881
5. Fleming M, Maglio PP (2015) How physical interaction helps performance in a scrabble-like task. In: *CogSci*, pp 716–721
6. Antle AN, Droumeva M, Ha D (2009, June) Hands on what? comparing children's mouse-based and tangible-based interaction. In: *Proceedings of the 8th international conference on interaction design and children*, ACM, pp 80–88
7. Maehigashi A, Miwa K, Oda K, Nakamura Y, Mori K, Igami T (2016) Influence of 3D images and 3D-printed objects on spatial reasoning. In: *Proceedings of the 38th annual meeting of the cognitive science society*, pp 414–419
8. Johannes K, Powers J, Couper L, Silbergliitt M, Davenport J (2016) Tangible models and haptic representations aid learning of molecular biology concepts. *Grantee Submission*, 372–377
9. Smithwick D, Kirsh D (2015) Let's Get physical: thinking with things in architectural design. In: *CogSci*, pp 2236–2241
10. Kim MJ, Maher ML (2008) The impact of tangible user interfaces on designers' spatial cognition. *HCI* 23(2):101–137

11. Maher ML, Lee L, Gero JS, Yu R, Clausner T (2017) Characterizing tangible interaction during a creative combination task. *Design computing and cognition '16*. Springer, Cham, pp 39–58
12. Clausner T, Maher ML, Gonzales B (2015) Conceptual Combination modulated by action using tangible computers. In: Poster presented at the 37th annual meeting of the cognitive science society, Pasadena
13. Soleimani A, Green KE, Herro D, Walker ID (2016, June) A tangible, story-construction process employing spatial, computational-thinking. In *Proceedings of the 15th international conference on interaction design and children*, ACM, pp 157–166
14. Kirsh D, Maglio P (1994) On distinguishing epistemic from pragmatic action. *Cogn Sci* 18(4):513–549
15. Dorst K, Cross N (2001) Creativity in the design process: co-evolution of problem–solution. *Des Stud* 22(5):425–437
16. Cook SW, Mitchell Z, Goldin-Meadow S (2008) Gesturing makes learning last. *Cognition* 106(2):1047–1058
17. Trofatter C, Kontra C, Beilock S, Goldin-Meadow S (2015) Gesturing has a larger impact on problem-solving than action, even when action is accompanied by words. *Lang Cogn Neurosci* 30(3):251–260
18. Alibali MW, Kita S, Young A (2000) Gesture and the process of speech production: we think, therefore we gesture. *Lang Cogn Process* 15:593–613

Side-by-Side Human–Computer Design Using a Tangible User Interface



Matthew V. Law, Nikhil Dhawan, Hyunseung Bang,
So-Yeon Yoon, Daniel Selva and Guy Hoffman

We present a digital–physical system to support human–computer collaborative design. The system consists of a sensor-instrumented “sand table” functioning as a tangible space for exploring early-stage design decisions. Using our system, human designers generate physical representations of design solutions, while monitoring a visualization of the solutions’ objective space. Concurrently, an AI system uses the vicinity of the human’s exploration point to continuously seed its search and suggest design alternatives. We present an experimental study comparing this side-by-side design space exploration to human-only design exploration and to AI-only optimization. We find that side-by-side collaboration of a human and computer significantly improves design outcomes and offers benefits in terms of user experience. However, side-by-side human–computer design also leads to more narrow design space exploration and to less diverse solutions when compared to both human-only and computer-only searches. This has important implications for future human–computer collaborative design systems.

Introduction

A useful formulation of early-stage design is viewing it as an exploration of the space of possible designs [5]. This can be formalized as a *search* through the solution space, proposing and evaluating solutions in pursuit of some possible world [44]. This is a particularly tractable model of design in symbolically represented state spaces [16]. Given that search is also a core capacity of Artificial Intelligence (AI) [33] and [48], researchers were able to develop intelligent tools to aid in design problems through a variety of computational search methods [45] and [31]. In most cases of

M. V. Law (✉) · N. Dhawan · H. Bang · S.-Y. Yoon · D. Selva · G. Hoffman
Cornell University, Ithaca, NY, USA
e-mail: mvl24@cornell.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_9

design-as-search, both when a human designer and when a computer design tool are employed, the process is modeled as one of an individual designer [18]. Some researchers, however, have suggested that exploring a design space can be more powerful when designers work with others. Fischer calls design social by nature [15]. Indeed, collaborative design can transcend the capacity of the individual, leveraging specialized expertise across “symmetries of ignorance” to enable designs that address complex problems and spaces [2]. The usefulness of collaboration in design has engendered a strong interest in systems and tools that support collaborative design, precipitating the field of Computer-Supported Collaborative Design (CSCD) [42].

Beyond CSCD, the potential of design as a collaborative activity also suggests human–computer collaborative design, which is the focus of this paper. While many approaches to human–computer collaborative design either pose agents as support tools for humans [31, 37], and [14] or position humans as inputs to a computational process [7, 13, 26], and [4], research in human–computer teamwork suggests merit in a more balanced partnership between human and computer designers, modeling the interaction as a true collaboration [17].

In this paper, we present a system to support a side-by-side model of human–computer collaborative design using a digital-tangible “sand table” interface in combination with an AI search agent and a visualization of the design problem’s objective space (Fig. 1). In our model, the user searches the design space using a physical one-to-one mapping of the solution space, while the AI search algorithm uses the user’s designs as seeds to search the design space alongside the human designer, and subsequently presents the human with a visualization of the search process.

Our motivation to use a Tangible User Interface (TUI) stems from the fact that tangible and tabletop interfaces have been found to be particularly well suited for collaborative exploration of design spaces. On its own, a TUI affords designers the ability to employ senses and manipulations they are familiar with in the physical world to interact with virtual models [21]. TUIs have been found to promote learning [6] and [47], and interaction with physical media to drive innovative exploration in design spaces [28, 46], and [32]. Tangible interfaces can also impact the nature of collaborative design processes, and hence outcomes, e.g., the effect of a TUI on spatial cognition in groups can increase “problem-finding,” leading to higher creativity [28].

TUI’s have been extensively evaluated vis-a-vis graphical or screen-based interfaces [49, 52], and [35], including with respect to design tasks [27], so this is not the focus of this work. We instead set out to use the TUI as a collaborative platform for evaluating side-by-side exploration with an agent in a design space.

In this vein, we present an experimental study that compares side-by-side human—computer collaborative design with two baseline conditions: human-only design search and human observation of computer-only search. Dependent variables include the quality of the generated designs and user experience. The design problem we use to illustrate our approach is the *EOSS Sensor–Orbit Design Problem*, a real-world space mission design problem with multiple competing objectives.

The core contributions of this work are: (a) a digital–physical system that supports side-by-side human–computer collaborative exploration of a design space; (b) support for our hypothesis that this system results in better designs than either the

human or the computer working alone; (c) insights into the user-experience benefits of side-by-side human-computer collaborative design; and (d) limitations and design implications related to the effects of side-by-side exploration on the coverage and diversity of the design solutions explored.

The EOSS Sensor-Orbit Design Problem

Designing sensor configurations for Earth-Observing Satellite Systems (EOSS) is a real-world multi-objective design problem in Aerospace Engineering. The design of such systems has become increasingly difficult and important to space organizations planning satellite missions due to increasingly stringent mission requirements without the necessary budget increases to fully meet the increased demands [40].

Specifically, we engage the problem of deploying sensors on a climate-monitoring satellite constellation to optimally satisfy 371 measurement requirements (e.g., air temperature, cloud cover, and atmospheric chemistry) defined by the World Meteorological Organization (www.wmo-sat.info/oscar) at minimal cost [19]. A design in this space consists of assigning up to 12 different kinds of sensors to satellites in five different orbits around the Earth. Each sensor has different capabilities that address different measurement requirements to varying degrees, dependent on the orbit in which it is deployed. The cost of deploying various sensors is also highly

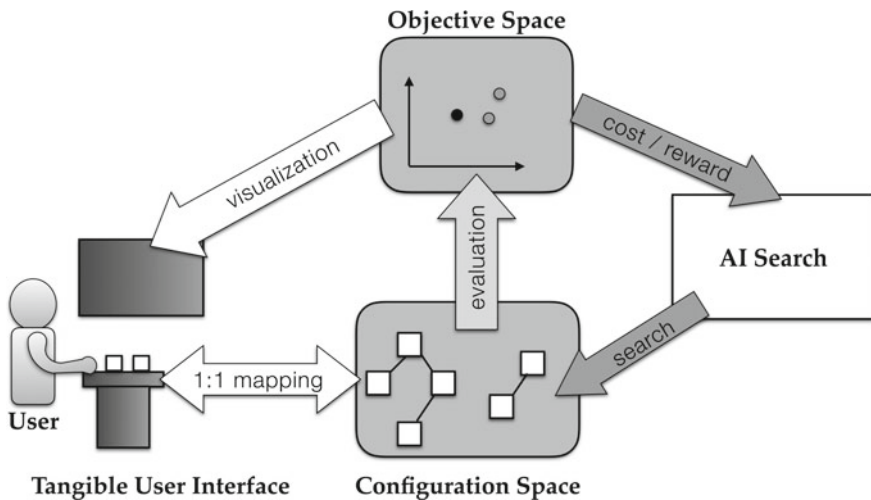


Fig. 1 Conceptual schematic of side-by-side human-computer collaborative design using a tangible interface. The human designer and AI search algorithm explore different designs simultaneously and affect each other’s position in the solution space. The human generates tangible physical representations of design solutions, while monitoring a visualization of the objective space. The AI search uses the human’s exploration to continuously seed its search and suggest design alternatives

orbit dependent, insofar as it affects the choice of launch vehicle and supporting subsystems, among other considerations. The cost and scientific benefits of a specific sensor configuration are further complicated by synergistic or deleterious effects that sensors deployed together can exert on each other.

Research Questions

The described system and study are elements of an ongoing project to both understand and realize novel forms of human–computer collaboration in physical design spaces. In this particular work, we explore the following research questions:

- **RQ1:** How do design solutions produced by a human and design agent working side-by-side compare to either human-only or algorithm-only generated solutions?
- **RQ2:** How does collaborating side by side with an intelligent agent affect user experience while exploring a design space?

The Collaborative Design Sand Table Tangible User Interface

Overview

Inspired by the affordances of TUIs for design and collaboration, we developed a tangible sand table interface to study collaborative design (Fig. 2).

Our mixed reality system consists of an interactive tabletop, a visualization, and a set of blocks. The blocks, which are mapped to sensors from the design problem, can be placed in regions designated as different orbits on the tabletop. The science benefit and cost associated with a particular block configuration are calculated using a custom simulation engine [41] and plotted on a visualization above the table. Points

Fig. 2 A user working collaboratively with an AI design agent using the presented digital-tangible sand table interface



on the visualization are color coded to indicate recency and whether they are user or agent generated.

The most recent point is plotted in red, the second most recent in pink, and all other points in various shades of purple such that a dark shade indicated a more recently generated point. All points on the plot are user selectable; the configuration used to generate any selected point is overlaid on the orbits in the tabletop workspace (Fig. 3).

Independent and Collaborative Design Agents

We developed two computational design agents to explore the sensor-orbit configuration design space, one that operates independently without user input, and one that explores the design space collaboratively with a human.

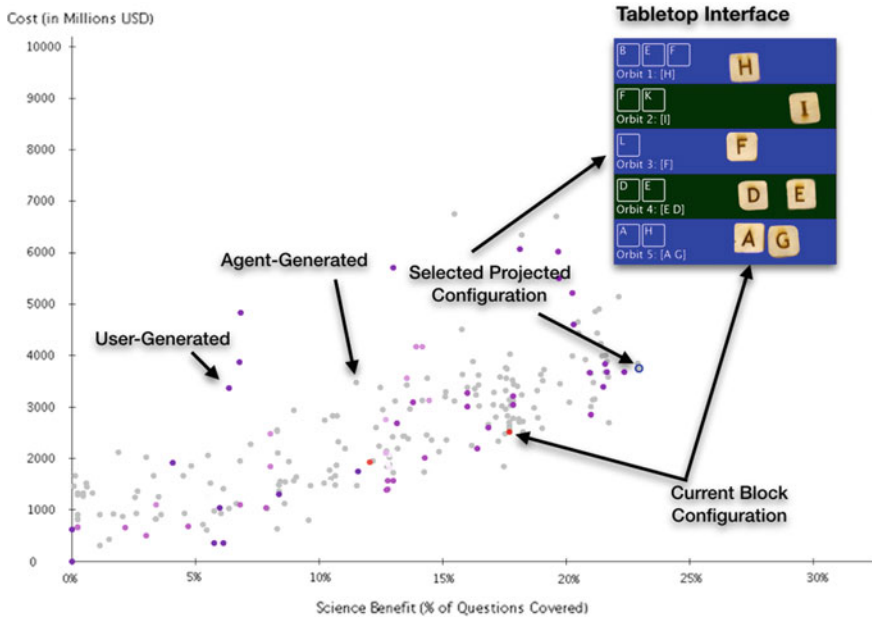


Fig. 3 This figure illustrates the tabletop (top right) and visualization interfaces. As users arrange sensor blocks into orbits, the system evaluates and plots the corresponding total cost and science benefit of the design on the scatterplot. The current design (in this case, instrument H in Orbit 1, I in Orbit 2, etc.) is plotted in red, the next most recent in pink. All other user-generated designs are plotted in a purple that fades over time. When a design agent generates a configuration, the system plots the corresponding output in gray. Finally, users can select an outcome to project its configuration on the table (in this case, instruments B, E, F in Orbit 1, etc.)

The “independent” design agent employs a Non-dominated Sorting Genetic Algorithm (NSGA-II) [9] to explore the design space. Evolutionary and genetic algorithms have long been associated with design exploration and NSGA-II is a conventional approach to exploring both design and multi-objective optimization spaces [8, 10, 22, 25, 38], and [30].

Inspired by recent work demonstrating the effect of simple local behavior on global outcomes in collaboration [43], the second, “collaborative,” design agent employs a simplistic version of local search modified to continuously orient its search space around the sensor-orbit configurations being explored by the human user. It does so by evaluating random one-block perturbations of the current table configuration. This allows the human and design agents to monitor one another while exploring the space in parallel, with the user choosing when to interact and cross search paths (Fig. 1).

Technical Specifications

Our tabletop TUI (Fig. 4) is designed in the tradition of the reactable [23]. An internally housed projector displays images on the 36'' × 30'' tabletop where an infrared camera detects objects placed on the surface. Blocks representing sensors are fitted with unique fiducial markers and tracked across the table surface using the camera and reacTIVision [24]. The NSGA-II agent was implemented via the jMetal optimization library [12].

Experimental Setup

We compare our side-by-side approach with two baseline methods, which are effectively “subsets” of the proposed approach. We ran a three-condition within-user study, which asked participants to explore the EOSS design problem on their own, by passively observing the NSGA-II agent, and side by side with the collaborative local-search design agent (Fig. 5).

Each study lasted roughly an hour and involved three treatment sessions. During each session, participants were asked to explore the design space through our interface, after which they were given up to 30 seconds to construct what they considered the “best” design based on what they had learned during exploration. They then completed a questionnaire assessing affect and user experience for that round. Following the study, users completed a post survey ranking the conditions and reflecting on their choices.

In the following, we describe the three conditions in detail:

1. *HUMAN-ONLY (HUM)*: Participants were instructed to explore the design space through the sand table interface on their own. They were given a set of blocks

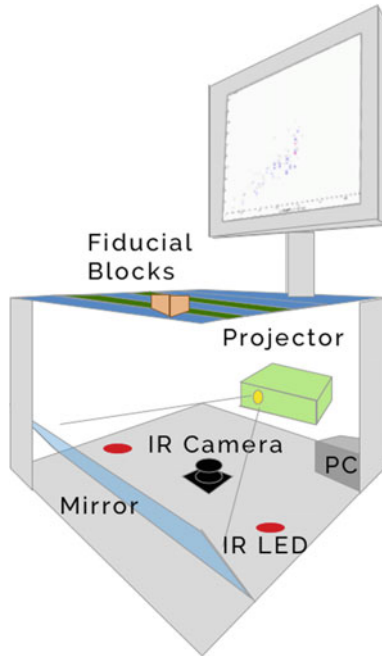


Fig. 4 The sand table projected a workspace onto a surface where a camera tracked blocks identified by fiducial markers. As the blocks move between regions on the surface, a simulation engine evaluates the associated configurations and plots them on a screen. All plotted points in the objective space can be selected and projected back onto the tabletop surface

for each instrument and explored using the tabletop and visualization without any assistance from a design agent. As described in the system description, participants could click on previously generated designs of their own to reflect on their design exploration at any time.

2. *OBSERVE-AGENT (OBS)*: Participants followed along as the NSGA-II design agent explored the space in real time, with all evaluated configurations plotted on the screen. Again, participants were able to select cost points as they were explored to see the corresponding configurations on the tabletop, and we allowed them to move around blocks on the table as well, although the system did not evaluate any block configurations.
3. *SIDE-BY-SIDE (SBS)*: Participants worked alongside the local-search design agent. As in the HUM condition, the system would evaluate and plot evaluations for the block configurations that users placed on the table. The local search agent would continuously explore minor variations of the current block configuration, which the system would evaluate and visualize for the user as well. For simplicity, we defined the local search neighborhood as any configuration at an edit distance of one from the current configuration (e.g., add or remove one instrument in any orbit). Users were free to monitor the agent's search path and adjust their own.

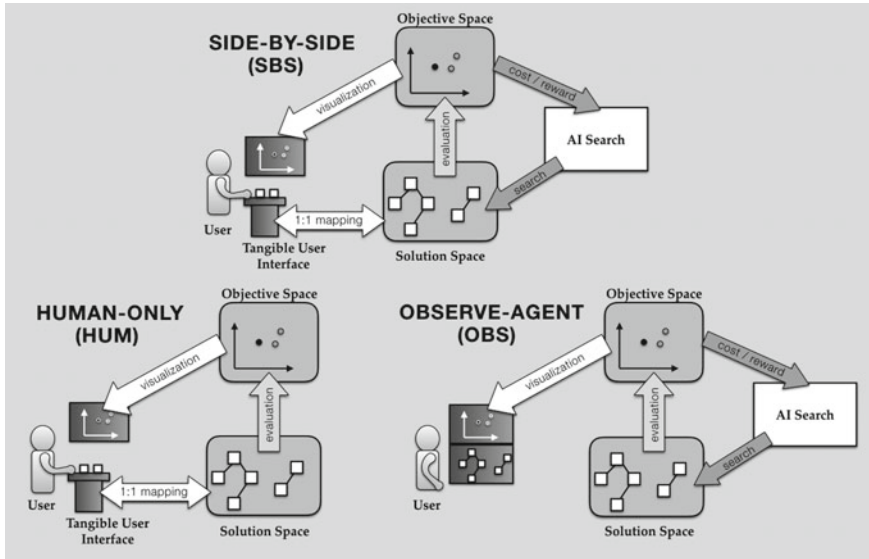


Fig. 5 The three design space search interactions studied: human-only search, human observation of agent search (NSGA-II), and side-by-side collaborative search

The instruments and orbits were randomly remapped between conditions with users informed in order to prevent knowledge carryover. The conditions were also randomly and uniformly counterbalanced against ordering effects due to fatigue or increased familiarity with the interface or task.

The study was exempted by the Cornell University Institutional Review Board (IRB) for Human Participants, based on the board’s criteria of data collection and risk. Any images in this work (e.g., Fig. 2) are not of participants, but are reenactments of the study by associates of the research team who consented to their image being used.

Hypotheses

Through our study, we examined the following hypotheses¹

- **H1: Design Quality:** The user–agent collaboration (SBS) will generate better designs than the user (HUM) or computer alone (OBS) will generate. While “better” is often difficult to quantify in a design problem; in this case, we will evaluate designs relative to a baseline Pareto front generated by a conventional genetic algorithm used in this domain—NSGA-II.

¹We initially intended to explore a third hypothesis addressing learning outcomes but were unable to do so due to an error in data collection.

- **H2: User Experience:** Users have a better experience when collaboratively exploring with an agent (SBS) compared to exploring on their own (HUM) or following the agent as it explores (OBS).

Results

Thirty-one subjects (13 females, ages 18–37) participated in our study. To attain a more diverse population sample, we recruited participants from a large city both through mailing lists and flyers at local universities and via ads on related social media groups and online bulletin boards. The resulting participant set came from a varied educational background: six had completed high school or a GED, 18 had a bachelor’s degree, and seven had a master’s degree, advanced graduate work, or a Ph.D. We describe our findings with regard to our hypotheses around design quality and user experience.

Design Quality

Given the multi-objective nature of the sensor–orbit problem, there is no clear single metric to objectively compare designs, a matter complicated by the unknown nature of the true Pareto frontier in this real-world problem.

For each participant and condition, we had a single design solution produced from a blank slate at the conclusion of the condition to compare within-user. Following [20, 50], and [36], we calculated the *generational distance* for each of the designs using their normalized Euclidean distance from a reference, empirically derived Pareto frontier. We constructed this reference Pareto frontier from the configurations generated by running NSGA-II over 80 iterations with a population size of 200 (Fig. 7). For reference, the NSGA-II agents in OBS evaluated an average of 267.6 unique designs in addition to the initial population of 200 over the course of the treatment. User designs were then compared relative to their distance from this reference frontier.²

One-tailed paired sample *t*-tests were conducted to evaluate the difference in quality of designs produced in the SBS condition, compared to each of the baseline conditions, HUM and OBS. The SBS condition produced significantly closer designs ($M=0.114$, $SD = 0.086$) in comparison to both HUM ($M = 0.167$, $SD = 0.138$, $t = -1.920$, $p = 0.032$) and OBS ($M = 0.155$, $SD = 0.124$, $t = -1.827$, $p = 0.039$), see Fig. 6a. These results suggest that participants tended to produce better designs after exploring the space with the collaborative agent, relative to the reference Pareto optimal front, supporting **H1**.

²In the case that user-generated designs dominated any configurations on the reference frontier, they were assigned the negation of this distance. Overall, we acknowledge that this choice of reference may limit the validity of our finding to a small time or a small number of function evaluations

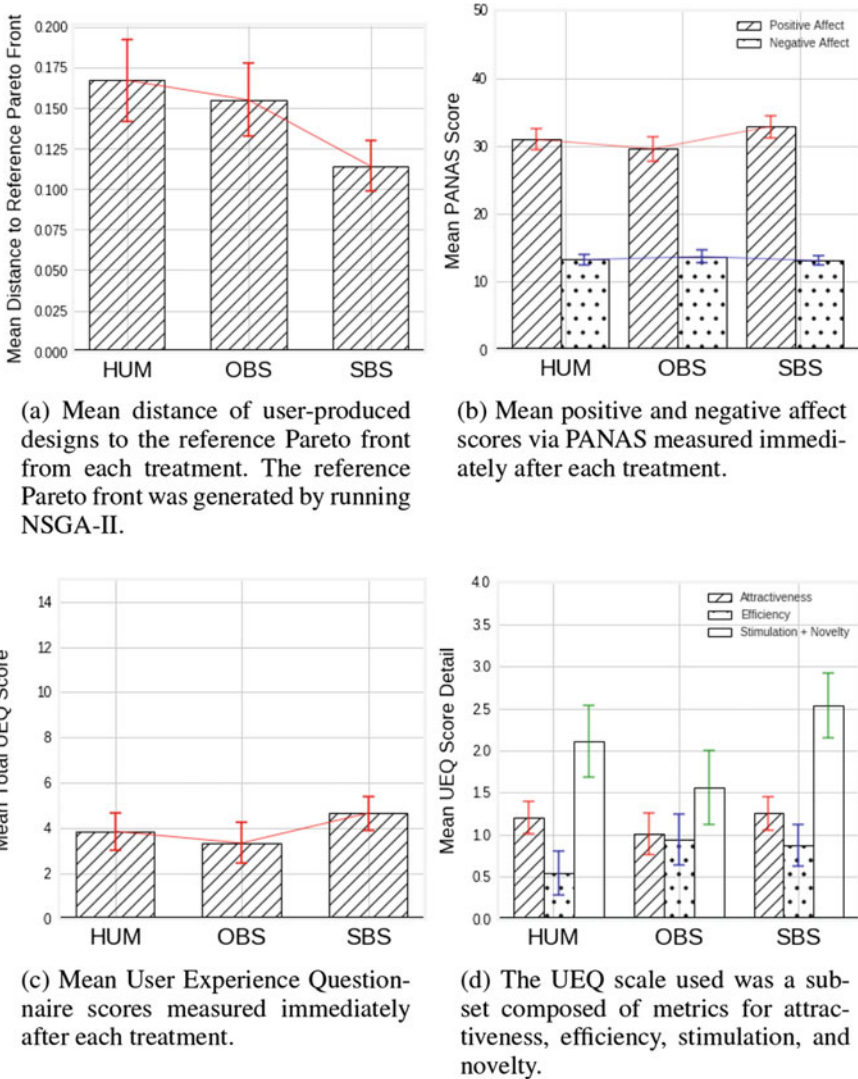


Fig. 6 Mean design quality and user experience scores across the three conditions

User Experience

Participants’ enjoyment was measured using the Positive and Negative Affect Schedule (PANAS) [51], and user experience via the User Experience Questionnaire (UEQ) [29]. Following the study, participants also ranked the treatments in order of helpfulness and enjoyment, and provided qualitative comparisons of the treatments in terms of helpfulness and enjoyment.

Participants displayed stronger positive effect in the SBS condition ($M=32.85$, $SD=8.964$) compared to HUM ($M=30.97$, $SD=8.677$, $t=1.455$, $p=0.078$), and compared to OBS ($M=29.56$, $SD=8.677$, $t=3.117$, $p=0.002$). One-tailed paired sample t -tests indicate that only the latter difference is significant, thus only partially supporting **H2**. No significant difference was found in participants' negative affect after the SBS condition ($M=13.13$, $SD=3.784$) compared to either HUM ($M=13.26$, $SD=3.838$, $t=-0.295$, $p=0.385$) or OBS ($M=13.71$, $SD=5.172$, $t=-0.668$, $p=0.255$) (Fig. 6b).

Participants scored the system more positively via aggregate UEQ scores after SBS design ($M=4.664$, $SD=4.117$) than either HUM ($M=3.858$, $SD=4.553$, $t=1.301$, $p=0.102$) or OBS ($M=3.339$, $SD=4.929$, $t=1.717$, $p=0.048$), although one-tailed paired sample t -tests indicate that only the second was barely significant and effect sizes were small (Fig. 6c). We employed a subset of the full UEQ scale, including the complete scales for attractiveness, efficiency, stimulation, and novelty. Interestingly, users rated OBS higher than HUM or SBS in terms of efficiency, but lower than the others in terms of attractiveness and the hedonistic scales of stimulation and novelty (Fig. 6d).

Finally, users overall ranked the treatments as (1. SBS, 2. OBS, and 3. HUM) in terms of helpfulness and (1. SBS, 2. HUM, and 3. OBS) in terms of enjoyment (Table 1). The rankings were aggregated using an extended Borda system [3], whereby each user's ranking was scored with three points for their first choice, two for their second, and one for their third choice.

Discussion

To summarize, we found that participants produced better designs after exploring the design space side by side with the collaborative design agent than after exploring on their own or observing and querying the NSGA-II algorithm visualization. Participants exhibited marginally higher effect and user experience when working side by side than either of the other modes. They also overwhelmingly rated this design method higher than the other two. In the following, we discuss implications of our findings, qualitative insights from user comments, and possible explanations that could lead to trade-offs when constructing collaborative design agents.

Qualitative Insights on Designing Side by side

Participants' post-study reflections provide some insight on why so many preferred exploring with the collaborative design agent (abbreviated as DA below) and how they perceived the DA. Several users pointed out complementary advantages they inferred in the DA, from speed to the ability to explore with more blocks at the same time. Others simply appreciated the experience of working together: "*Exploring with*

Table 1 Participants ranked and reflected on the three treatments at the conclusion of the study in terms of helpfulness and enjoyment

Treatment	Helpfulness rank (score) <i>n</i> = 31	Enjoyment rank (score) <i>n</i> = 27	Comments	
SBS	1 (81)	1 (70)	Positive	I liked the fact that I was being assisted along [...] It felt like as if two brains were working simultaneously
			Negative	It was distracting to see the agent coming up with points around me that weren't always improvements, and this made me feel less productive
OBS	2 (55)	3 (43)	Positive	It felt like watching the agent exploring by itself allowed me to see different trends without having to move the blocks myself [...] I was arriving at a better solution more quickly
			Negative	Observing the agent exploring was dreadful. Way too much information, and I couldn't control the variances in sequences to help myself understand the impacts of various instruments
HUM	3 (50)	2 (49)	Positive	Exploring alone makes it easier and enjoyable because it allows me to follow my own logic of exploration
			Negative	Exploring with blocks is too inefficient and make me feel frustrated. I felt lost without help from the computer

The rankings were aggregated using an extended Borda system (scores listed next to rank in parentheses). Four users did not respond for the enjoyment ranking

the DA felt more like a collaborative effort, rather than working alone or watching someone else work on something” or saw the back-and-forth with the design agent as a way to reduce the randomness of their search. Some participants derived confidence from working with the design agent: “It felt like as if two brains were working simultaneously and there was a hope to achieve optimal configuration”.

On the other hand, some expressed annoyance with the agent: “It would have been better if the computer gave better suggestions alongside working with me...”. At least one user saw the design agent as a playful antagonist: “I enjoyed exploring with the DA at the same time because I almost felt like I was competing against the DA.” Several developed ad hoc strategies for collaboration, e.g., splitting up the objectives: “After DA determined points from my selection, I rearranged the blocks to the DA point with the highest benefit. Then, I switched blocks to determine the lower cost”. The experiences described by participants in the side-by-side condition, whether positive or negative, suggest that users are capable of seeing such agents as collaborators and not just tools. In particular, the variety of implicit choices and ad hoc strategies users made in interacting with the design assistant while exploring the design space mirror observations prior work has made about human-to-human collaboration using TUIs, e.g., [52], including turn-taking, dominant–submissive pairs, and independent, parallel exploration. This supports the potential of intelligent agents acting as true collaborators in the design search process.

Does Working Side by Side Lead to Broader Search?

In order to gain intuition about why users generated better designs after SBS exploration, we examined the solutions they encountered during search under the different conditions (Fig. 7). Using one conventional way to compare sets of solutions, we found that the set of configurations considered by participants in the SBS condition tended to dominate more of the objective space, in terms of hypervolume [53] ($M = 0.626$, $SD = 0.134$), than those explored in HUM ($M = 0.561$, $SD = 0.145$), (Fig. 8a). This difference was significant via a one-tailed paired t -test ($t = 2.45$, $p = 0.010$). Designs explored by participants in the OBS condition also tended to dominate less hypervolume than in SBS ($M = 0.603$, $SD = 0.091$), although this difference was not significant ($t = 1.07$, $p = 0.146$).

To our surprise, however, users appeared to explore less broadly in the SBS condition than in either the HUM or OBS conditions. To quantify this, we define the *coverage* of the exploration as the number of possible sensor–orbit pairings that appeared in at least one evaluated configuration during the exploration. Similarly, to [34], we also use the normalized entropy of explored configurations as a measure of *diversity*. We calculate entropy as

$$H(X) = -\frac{1}{\log N} \sum_i^n p(x_i) \log p(x_i)$$

where X is the configurations explored, N is the number of configurations in X , x_i is a possible sensor-orbit pair, $p(x_i)$ is the probability of x_i appearing in a configuration in X , and n is the number of possible sensor-orbit pairs.

We find that participants tended to cover more of the orbit-sensor pairings when searching the solution space in the HUM condition ($M=44.93$, $SD=13.03$) and the OBS condition ($M=39.89$, $SD=12.79$) than in the SBS condition ($M=32.97$, $SD=10.53$). Both of these differences were significant via paired one-tail t -tests ($t=5.357$, $p<0.001$ and $t=2.428$, $p=0.011$ for HUM and OBS, respectively). We also find that the human’s search tended to be more disordered when either exploring alone ($M=1.482$, $SD=0.502$) or passively observing ($M=2.326$, $SD=0.672$), again

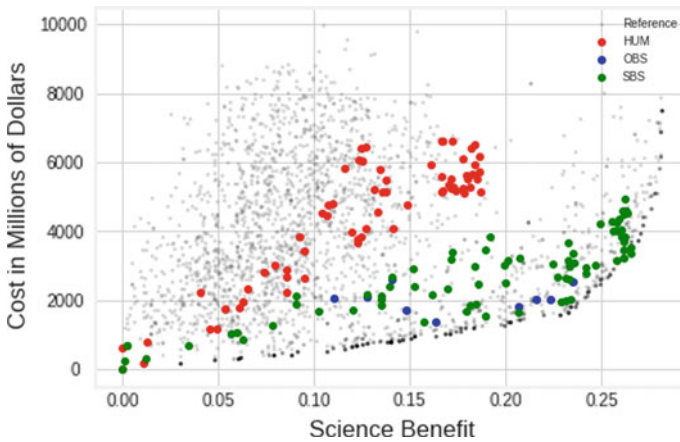


Fig. 7 This figure shows an example of all the evaluated configurations explored by a single user during the exploration phase in each study condition. The outputs used to generate the reference Pareto front are plotted in the background in gray

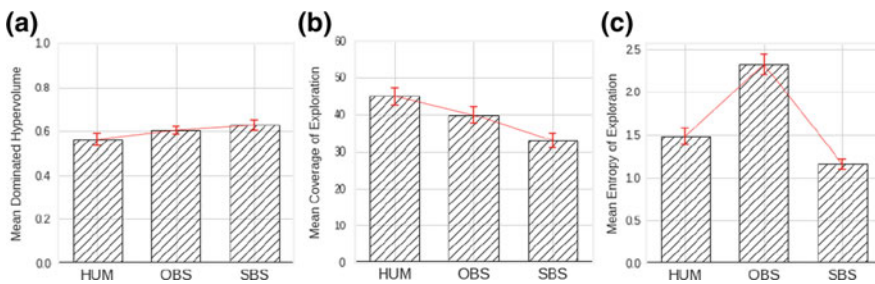


Fig. 8 In the SBS condition, participants tended to consider more Pareto optimal designs as measured by the overall hypervolume dominated by the non-dominated Pareto frontiers in each condition (a). Nonetheless, the search spaces explored by human participants when collaboratively exploring in the SBS condition tended to cover fewer possible sensor-orbit pairings (b), and exhibit lower information entropy (c) than in the other two conditions

both significant via paired one-tail t -tests ($t = 3.093$, $p = 0.002$ and $t = 8.414$, $p < 0.001$, respectively).

Participants' post-study reflections suggest that working with the design agent encouraged them to converge more confidently and quickly to a more focused region of the configuration space. For example, "*I could immediately see some sort of direction to move in instead of randomly guessing,*" and "*when we both (computer and I) are exploring together, less time is wasted, and productive results are easier to discover.*" Indeed, as one user put it, "*I felt lost without help from the computer.*"

However, as others observed, collaboration "*might have led to a bias in what order to use and I resulted in a lower science benefit than I had on my own,*" and "*exploring on my own gave me more freedom to try something completely different, and potentially get a more helpful combination.*" Participants appreciated this freedom, saying, "*it was really useful learning through trial and error,*" and "*exploring alone makes it easier and enjoyable because it allows me to follow my own logic of exploration.*"

This raises an important conundrum for the design of collaborative agents, insofar as the processes for achieving better designs through collaboration may not coincide with those that best encourage broader exploration of the design space or generate more creative designs. Some work with TUIs found similarly that rapid design exploration enabled by physical interfaces could actually reduce the degree to which users reflect in the design process [11]. This result also evokes prior work suggesting conversely that leveraging humans as a search heuristic can reduce the diversity of algorithmically generated solutions [39]. Insofar as a key benefit of collaborative design is its potential to foster broader exploration and emergence, future research should explore how interactions with collaborative design agents might expand, rather than contract, human designers' exploration.

Limitations and Future Work

Our findings are somewhat constrained by the complexity and domain-specific nature of the design problem we chose in contrast with the relevant sophistication and expertise of our users. The resultant abstractness of the problem made it very demanding for our users, and could have added to the variance in our results, although we attempted to account for this with a within-user design.

TUIs are especially useful for co-present collaboration in a shared physical workspace. Although our agent interacted with the user through the tabletop interface and display, it did not do so physically. This study is part of an ongoing project in which we plan to study collaborative exploration between a human and a physically embodied design agent in a shared workspace. Observing interactions between a virtual agent and a human through our TUI sand table is a first step toward this end.

This work also does not empirically compare the human–agent collaborative exploration to collaboration between humans. While some participants reported interacting with the agent in similar ways to what we see in the literature on co-present

human–computer collaborations, future work should directly examine these similarities in order to lay the groundwork for designing better collaborative agents in this vein.

Finally, while we adapted a design-as-search model, there are other potentially richer formulations (e.g., design-as-exploration) that may better model real-world design processes. Future work should consider other formulations of design which allow for important processes like problem reframing.

Conclusion

Humans and algorithms have different strengths and limitations in searching design spaces. Algorithms can quickly explore a large space and precisely compare solutions, while humans are adept at fast pattern recognition, generalization, and context integration. Egan and Cagan note the importance of both human intuition to handle difficult-to-translate qualitative processes and the objectivity and consistency of computation at scale [13]. This suggests benefits to be reaped by systems that model the human–machine interaction as a collaborative activity, building on the complementary skills of each agent, e.g., flexible and conversational mixed initiative collaborations or adjustable autonomy for different contexts [1].

In this paper, we described a new tabletop tangible sand box interface in order to study real-time collaboration between humans and design-search algorithms. Such side-by-side human–computer collaborative exploration of a design space via a physical one-to-one mapping of the solution space has not been studied before, despite the potential it offers designers to capitalize on benefits of both collaborative and AI-supported design.

In an experiment, we find that the proposed model of side-by-side design collaboration can lead a human designer to generate better designs than when working alone or observing an agent in terms of their selected final design's distance to a reference Pareto front, and, in the former case, to explore more hypervolume-dominant designs. We also find marginal benefits to user positive effect and user experience. In particular, side-by-side design positively overcomes some of the trade-off between efficiency and stimulation that exists when weighing human-only and computer-only designs.

However, we also find that this sort of collaboration might lead to lower solution space coverage and less diversity in the solutions explored. As we do not want human–machine collaborative design to reduce the creativity and open-ended exploration that early-stage design requires, these concerns should be considered in the development of such agents and future research.

This caveat notwithstanding, our work supports the feasibility of treating design agents not just as tools, but as peer collaborators in the exploration of possible solutions during early-stage design.

Acknowledgements This work was supported primarily by the Civil, Mechanical and Manufacturing Innovation Program of the National Science Foundation under NSF Award No. 1635253.

References

1. Allen JF, Guinn CI, Horvitz E (1999) Mixed-initiative interaction. *IEEE Intel Syst Appl* 14(5):14–23
2. Arias E, Eden H, Fischer G, Gorman A, Scharff E (2000) Transcending the individual human mind—creating shared understanding through collaborative design. *ACM Trans Computer-Human Int (TOCHI)* 7(1):84–113
3. Arrow KJ (2012) *Social choice and individual values*, vol 12. Yale University Press
4. Babbar-Sebens M, Minsker BS (2012) Interactive genetic algorithm with mixed initiative interaction for multi-criteria ground water monitoring design. *Appl Soft Comput J* 12(1):182–195
5. Balling R (1999) Design by shopping: a new paradigm? In: *Proceedings of the third world congress of structural and multidisciplinary optimization (WCSMO-3)*, vol 1, pp 295–297
6. Chen R, Wang X (2008) An empirical study on tangible augmented reality learning space for design skill transfer. *Tsinghua Science and Technology* 13 Supple (October):13–18
7. Cho SB (2002) Towards creative evolutionary systems with interactive genetic algorithm. *Appl Intel* 16(2):129–138
8. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Trans Evol Comput* 6(2):182–197
9. Deb K, Karthik S et al (2007) Dynamic multi-objective optimization and decision-making using modified nsga-ii: a case study on hydro-thermal power scheduling. In: *International conference on evolutionary multi-criterion optimization*. Springer, pp 803–817
10. Dhanalakshmi S, Kannan S, Mahadevan K, Baskar S (2011) Application of modified nsga-ii algorithm to combined economic and emission dispatch problem. *Int J Electr Power Energy Syst* 33(4):992–1002
11. Do-Lenh S, Jermann P, Cuendet S, Zufferey G, Dillenbourg P (2010) Task performance versus learning outcomes: a study of a tangible user interface in the classroom. In: *European conference on technology enhanced learning*. Springer, pp 78–92
12. Durillo JJ, Nebro AJ (2011) jmetal: A java framework for multi-objective optimization. *Adv Eng Softw* 42(10):760–771
13. Egan P, Cagan J (2016) Human and computational approaches for design problem-solving. In: *Experimental design research*. Springer, pp 187–205
14. Ferguson G, Allen JF et al (1998) Trips: an integrated intelligent problem-solving assistant. In: *AAAI/IAAI*, pp 567–572
15. Fischer G (2004) Social creativity: turning barriers into opportunities for collaborative design. In: *Proceedings of the eighth conference on participatory design: Artful integration: interweaving media, materials and practices-Volume 1*, ACM, pp 152–161
16. Gero JS (1998) Conceptual designing as a sequence of situated acts. In: *Artificial intelligence in structural engineering*. Springer, pp 165–177
17. Grosz BJ (1996) Collaborative systems (aaai-94 presidential address). *AI Mag* 17(2):67
18. Hay L, Duffy AHB, McTeague C, Pidgeon LM, Vuletic T, Grealy M (2017) A systematic review of protocol studies on conceptual design cognition: design as search and exploration. *Des Sci* 3:e10. arXiv:1011.1669v3
19. Hitomi N, Bang H, Selva D (2017) Extracting and applying knowledge with adaptive knowledge-driven optimization to architect an earth observing satellite system. *AIAA Information Systems-AIAA Infotech@ Aerospace*, p 0794
20. Ishibuchi H, Masuda H, Tanigaki Y, Nojima Y (2015) Modified distance calculation in generational distance and inverted generational distance. *EMO* 2:110–125

21. Ishii H, Ratti C, Piper B, Wang Y, Biderman A, Ben-Joseph E (2004) Bringing clay and sand into digital design—continuous tangible user interfaces. *BT Technol J* 22(4):287–299
22. Jeyadevi S, Baskar S, Babulal C, Iruthayarajan MW (2011) Solving multiobjective optimal reactive power dispatch using modified nsga-ii. *Int J Electr Power Energy Syst* 33(2):219–228
23. Jordà S, Geiger G, Alonso M, Kaltenbrunner M (2007) The reactable: exploring the synergy between live music performance and tabletop tangible interfaces. In: *Proceedings of the 1st international conference on Tangible and embedded interaction*, ACM, pp 139–146
24. Kaltenbrunner M (2009) Reactivision and tuio: a tangible tabletop toolkit. In: *Proceedings of the ACM international conference on interactive tabletops and surfaces*, ACM, pp 9–16
25. Kicinger R, Arciszewski T, De Jong K (2005) Evolutionary computation and structural design: A survey of the state-of-the-art. *Comput Struct* 83(23):1943–1978
26. Kim HS, Cho SB (2000) Application of interactive genetic algorithm to fashion design. *Eng Appl Artif Intell* 13(6):635–644
27. Kim M, Maher M (2005) Comparison of designers using a tangible user interface and graphical user interface and impact on spatial cognition. *Proc Human Behav Des* 5
28. Kim MJ, Maher ML (2008) The impact of tangible user interfaces on spatial cognition during collaborative design. *Des Stud* 29(3):222–253
29. Laugwitz B, Held T, Schrepp M (2008) Construction and evaluation of a user experience questionnaire. In: *Symposium of the Austrian HCI and usability engineering group*. Springer, pp 63–76
30. Laumanns M, Thiele L, Deb K, Zitzler E (2002) Combining convergence and diversity in evolutionary multiobjective optimization. *Evol Comput* 10(3):263–282
31. Liu H, Tang M (2006) Evolutionary design in a multi-agent design environment. *Appl Soft Comput J* 6(2):207–220
32. Maher ML, Lee L (2017) Designing for gesture and tangible interaction. *Synth Lect Human-Centered Interact* 10(2):i–111
33. McCarthy J (2007) What is artificial intelligence. URL: <http://www-formal.stanford.edu/jmc/whatisai.html>
34. Ozgur A, Johal W, Mondada F, Dillenbourg P (2017) Windfield: learning wind meteorology with handheld haptic robots. In: *HRI'17: ACM/IEEE international conference on human-robot interaction proceedings*, ACM, EPFL-CONF-224130, pp 156–165
35. Patten J, Ishii H (2000) A comparison of spatial organization strategies in graphical and tangible user interfaces. In: *Proceedings of DARE 2000 on designing augmented reality environments*, ACM, pp 41–50
36. Petersson K, Kyroudi A, Bourhis J, Ceberg C, Knöös T, Bochud F, Moeckli R (2017) A clinical distance measure for evaluating treatment plan quality difference with pareto fronts in radiotherapy. *Phys Imaging Radiat Oncol* 3:53–56
37. Ramchurn SD, Wu F, Jiang W, Fischer JE, Reece S, Roberts S, Rodden T, Greenhalgh C, Jennings NR (2016) Human-agent collaboration for disaster response. *Auton Agent Multi-Agent Syst* 30(1):82–111
38. Reed P, Minsker BS, Goldberg DE (2003) Simplifying multiobjective optimization: an automated design methodology for the nondominated sorted genetic algorithm-ii. *Water Resour Res* 39(7)
39. Selva D (2014a) Experiments in knowledge-intensive system architecting: interactive architecture optimization. In: *Aerospace conference, 2014 IEEE*, IEEE, pp 1–12
40. Selva D (2014b) Knowledge-intensive global optimization of earth observing system architectures: a climate-centric case study. In: *Sensors, systems, and next-generation satellites XVIII, international society for optics and photonics*, vol 9241, p 92411S
41. Selva D, Cameron BG, Crawley EF (2014) Rule-based system architecting of earth observing systems: earth science decadal survey. *J Spacecraft Rockets*
42. Shen W, Hao Q, Li W (2008) Computer supported collaborative design: retrospective and perspective. *Comput Ind* 59(9):855–862
43. Shirado H, Christakis NA (2017) Locally noisy autonomous agents improve global human coordination in network experiments. *Nature* 545(7654):370–374

44. Simon HA (1996) *The sciences of the artificial*. MIT press
45. Smithers T, Conkie A, Doheny J, Logan B, Millington K (1989) Design as intelligent behavior: an ai in design research program. In: Gero JS (ed) *Artificial intelligence in design*
46. Smithwick D, Kirsh D, Sass L (2017) Designerly pick and place: coding physical model making to inform material-based robotic interaction. In: *Design computing and cognition' 16*. Springer, pp 419–436
47. Starcic AI, Zajc M (2011) An interactive tangible user interface application for learning addition concepts_1217 131. 135. *Br J Edu Technol* 42(6):E131–E135
48. Thornton C, Du Boulay B (2012) *Artificial intelligence through search*. Springer Science and Business Media
49. Ullmer B, Ishii H (1997) The metadesk: models and prototypes for tangible user interfaces. In: *Proceedings of the 10th annual ACM symposium on user interface software and technology*, ACM, pp 223–232
50. Van Veldhuizen DA, Lamont GB (1998) Evolutionary computation and convergence to a pareto front. In: *Late breaking papers at the genetic programming 1998 conference*, pp 221–228
51. Watson D, Clark LA, Tellegen A (1988) Development and validation of brief measures of positive and negative affect: the panas scales. *J Pers Soc Psychol* 54(6):1063
52. Xie L, Antle AN, Motamedi N (2008) Are tangibles more fun? comparing children's enjoyment and engagement using physical, graphical and tangible user interfaces. In: *Proceedings of the 2nd international conference on tangible and embedded interaction*, ACM, pp 191–198
53. Zitzler E, Brockhoff D, Thiele L (2007) The hypervolume indicator revisited: on the design of pareto-compliant indicators via weighted integration. In: *Evolutionary multi-criterion optimization*. Springer, pp 862–876

Part III
Design Synthesis

Utility of Evolutionary Design in Architectural Form Finding: An Investigation into Constraint Handling Strategies



Likai Wang, Patrick Janssen and Guohua Ji

Evolutionary design allows complex design search spaces to be explored, potentially leading to the discovery of novel design alternatives. As generative models have become more complex, constraint handling has been found to be an effective approach to limit the size of the search space. However, constraint handling can significantly affect the overall utility of evolutionary design. This paper investigates the utility of evolutionary design under different constraint handling strategies. The utility is divided into three major factors: search efficiency, program complexity, and design novelty. To analyze these factors systematically, a series of generative models are constructed, and populations of designs are evolved. The utility factors are then analyzed and compared for each of the generative models.

Introduction

In the last decade, the use of evolutionary design (ED) has been gaining popularity in architecture as a strategy for architects to improve building designs. By defining generative models (GM) for the building design and evaluative models (EM) for the building performance, designers are able to use evolutionary algorithms to explore complex design search spaces and discover design alternatives for different objectives [1]. In some cases, novel design alternatives can be discovered that not only break conventional rules of thumb but also lead to innovative solutions that are able to resolve complex design challenges [2, 3].

L. Wang (✉) · G. Ji
Nanjing University, Nanjing, China
e-mail: dg1436002@smail.nju.edu.cn

P. Janssen
National University of Singapore, Singapore, Singapore

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_10

Among the three major components in ED (the evolutionary algorithm, the GM, and the EM), the GM has the most direct impact on outcomes of ED as well as the overall utility of the ED. This research focuses primarily on GMs for generating building geometries with a specific emphasis on constraint handling.

In architecture, GMs have become an important subdomain within ED research. Various innovative form-finding approaches have been explored by Frazer [4], Bentley and Kumar [5, 6], and others. Following these pioneers, other researchers have explored GMs with wide-ranging diversity [1]. Theoretically, the ED based on such GMs is useful for architects to explore design space and find solutions with excellent performance.

However, when such EDs are applied to real-world architectural designs it is often inapplicable to use since viable solutions are difficult to be found within reasonable time frames or deadlines set by practice. On the one hand, the process of ED is often prolonged by detailed performance simulations, the time cost by which can range from seconds to hours per design solution [7]. On the other hand, as GMs have become more complex to describe a detailed building design, the associated number of parameters and the resulting dimensions of the search space have also increased rapidly, which leads to an exponentially expanding size of design search spaces [8, 9]. Since there is usually a high proportion of invalid solutions in the search spaces [10], the expansion of the search spaces may also result in an increase in the number of invalid design solutions [11]. Therefore, the convergence of the evolutionary process is hindered due to the need to exclude large numbers of invalid design solutions.

Detailed simulations and large search spaces directly result in long running times of the evolutionary process which can severely weaken the utility of ED. Aside from the performance simulations which is out of designers' control, the search space is a critical factor in reducing the running times. Therefore, when constructing the GM for complex building designs, designers can incorporate constraint handling strategies in the GM in order to compress the search space [2, 12]. Such strategies can improve search efficiency by preventing computational resources from being spent on invalid design solutions.

In general, constraint handling can be categorized into two major classes: indirect and direct approaches. The main difference is that the indirect approach embeds constraints in the EM, while the direct approach embeds constraints in the GM [12]. These two approaches have different impacts on the evolutionary process. With indirect constraints handling, invalid solutions are identified and downgraded by the EM, which will lead to them becoming excluded during the evolutionary process. Direct constraint handling, in contrast, uses the GM to filter out invalid design solutions by including explicit or implicit rules [2, 13].

In general, the direct approach is preferred due to its ability to reduce the size of the search space, thereby improving the overall search process. However, with regard to overall ED utility, three main drawbacks have been identified, relating to *search efficiency*, *program complexity*, and *design novelty*.

First, search efficiency may be negatively impacted due to the introduction of disruptive nonlinearities into the genotype-to-phenotype mapping, which will result in a more irregular fitness landscape. The irregular change in fitness will make the

evolutionary search process more difficult to extract information to predict promising design subspaces [14].

The second drawback of embedding constraint handling into GMs is that it results in more complex control flows, which makes the program implementation and maintenance more difficult [5]. These characteristics are particularly demanding for architects who are mostly not good at programming.

The third drawback of embedding constraint handling into GMs is the fact that it may reduce design novelty. For architects, design novelty is a critically important factor. GMs must be able to generate designs that vary significantly in terms of their form and configuration.

Direct constraints handling, therefore, is a double-edged sword for ED. The resulting conflict between search efficiency, program complexity, and design novelty is a complex trade-off. However, current understanding of these factors and the trade-off between them in the field of ED is not well understood. Taking this as the point of departure, this study investigates the relationships between constraint handling and the three abovementioned factors. A series alternative GMs based on different constraints are constructed and populations of designs are evolved. The quality of these factors for each GM is then analyzed and compared.

Method: A Framework for Analyzing Utility

The aim of this study is to develop approaches that can help designers to evaluate which GM constraint handling strategies are suitable for design scenarios in terms of the three proposed utility factors. Even though absolute metrics are hard to come by, there are still various relative measures that can be used.

Search Efficiency

Search efficiency refers to the extent to which the GM enables the evolutionary search process to converge on viable design solutions. In practice, the search efficiency is typically one of the most pragmatic factors.

The search efficiency of alternative GMs can be objectively compared by analyzing the evolutionary search process. It is closely related to the size of the search space. Smaller search spaces will typically result in evolutionary processes that are able to find viable solutions in the short term and converge rapidly.

As additional constraints are embedded into the GM, the search space will continuously shrink, and search efficiency may be progressively improved. However, the negative impact of the more irregular fitness landscape also needs to be taken into account. Therefore, the overall effect of constraint handling on search efficiency remains an open research question.

Program Complexity

Program complexity refers to the complexity of the control flow of the GM code. In practice, the coding of complex constraint handling control flows can present significant technical difficulties for architects who are not good at programming.

The rising complexity of a program and the associated degradation of its maintainability cannot be measured by reference to the number of lines of program code. An alternative approach to measuring code complexity is *cyclomatic complexity* [15]. By counting the numbers of nodes and edges in the control flow graph of a program, the cyclomatic complexity measures the number of all linearly independent paths. This index has a close relationship with maintainability of programs. As the cyclomatic complexity increases, the control flow becomes more complex. This will typically result in extra coding effort and time that have to be spent on debugging and refactoring.

Lower program complexity, however, cannot ensure that the overall effort for ED implementation will be reduced. Since simple control flows are usually unable to avoid invalid solutions being generated, the more coding effort may have to be spent on implementing indirect constraint handling strategies, such as penalty functions, in the EM for downgrading undesired design solutions. As the result, coding effort saved by the simple GM control flow will, in many circumstances, be offset by additional coding effort in the EM.

Design Novelty

Design novelty refers to the ability of the GM to generate viable solutions that are unexpected. Discovering novel design alternatives is one of the main aims of ED. Therefore, the significance of design novelty may outweigh the search efficiency and the program complexity when it comes to the overall utility [16].

In most architectural design cases, the potential to discover novel design solutions has a close relationship with the formal variability of the phenotype space. If the phenotype space is overly restricted by constraints, it becomes more difficult for the evolutionary search process to find novel design solutions. Thus, although search efficiency can be improved by constraint handling, the overall utility may still be weakened, or even exhausted if fewer or no novel design solutions can be found.

Design novelty, however, is hard to evaluate objectively. Some measures have been developed [3, 17, 18], but these are themselves somewhat subjective and are hard to implement. In general, the degree of design novelty is highly subjective and largely determined by the needs of architects and projects. However, a visual appraisal of formal variability can provide a rudimentary way of differentiating the amount of design novelty from the architectural perspective. In reverse, GMs with low design novelty usually generate solutions that are visually very similar.

Case Study

In order to systematically investigate the evolutionary design utility factors for architectural designs, a case study high-rise office building design with an atrium and vertical gardens is introduced. The combination of atriums and vertical gardens is widely used as an effective strategy for improving environmental performance in many regions, from tropical to temperate climate zones. Examples include Commerzbank Tower in Frankfurt, Germany, and the Tongji University Multi-Functional Building in Shanghai, China [19].

In recent years, many GMs representing such building designs have been constructed for various design optimization problems. Based on these GMs, ED then can be used to explore possible configurations of these vertical gardens. The configuration of vertical gardens can be categorized as a subdomain of *facility layout problem*, which mainly addresses the various layout problems from the perspective of material handling costs, spatial efficiency, etc. [8, 20, 21]. However, in most cases, the adopted generative rules controlling combinations and allocations of vertical gardens are not properly constrained to avoid invalid solutions being generated. Thus, finding an appropriate trade-off between conflicting performance criteria requires atriums and vertical gardens to be carefully controlled and configured within the building volume [20].

Different constraint handling strategies can be incorporated in GMs to regulate the configurations of vertical gardens. In order to investigate the impact of constraint handling strategies on overall utility, four alternative GMs were implemented and tested. Each GM incorporated incrementally more constraint handling.

A GM with basic constraints was first constructed, referred to as the *naïve* GM (N-GM) and represents an elementary approach for generating the building. To compare the effects of constraint handling on evolutionary designs, three GMs with incrementally more constraints were constructed based on the same structure frame as in the N-GM. These three GMs, respectively, referred to as the *constrained* GM (C-GM), the *constrained-repaired* GM (CR-GM), and the *constrained-confined* GM (CC-GM), literally reflect their constraint handling strategies.

Figure 1 presents the random sampling generated by these four GMs. In general, the formal variability decreases with more constraints embedded. The distinct formal variability will result in significant effects on different utility factors of the ED.

For the case study, a fixed structural frame is used, consisting of a rectangular plan office floor with an open atrium in the center rising up through the whole building, flanked by two structural cores on both sides. The core-column structure is taken as the structural prototype as it has been widely applied in practice for its spatial and constructional efficiency. The size of each column grid is 8.4 by 8.4 m, which is proved can achieve a desirable balance between spatial and structural efficiency [22].

A *modular approach* is applied in the GMs, which partitions the floor plan into multiple fixed-size modules [20, 21]. Thus, each floor is divided into 11 cells based on this structural frame (Fig. 2). Except for the cells representing the structural cores

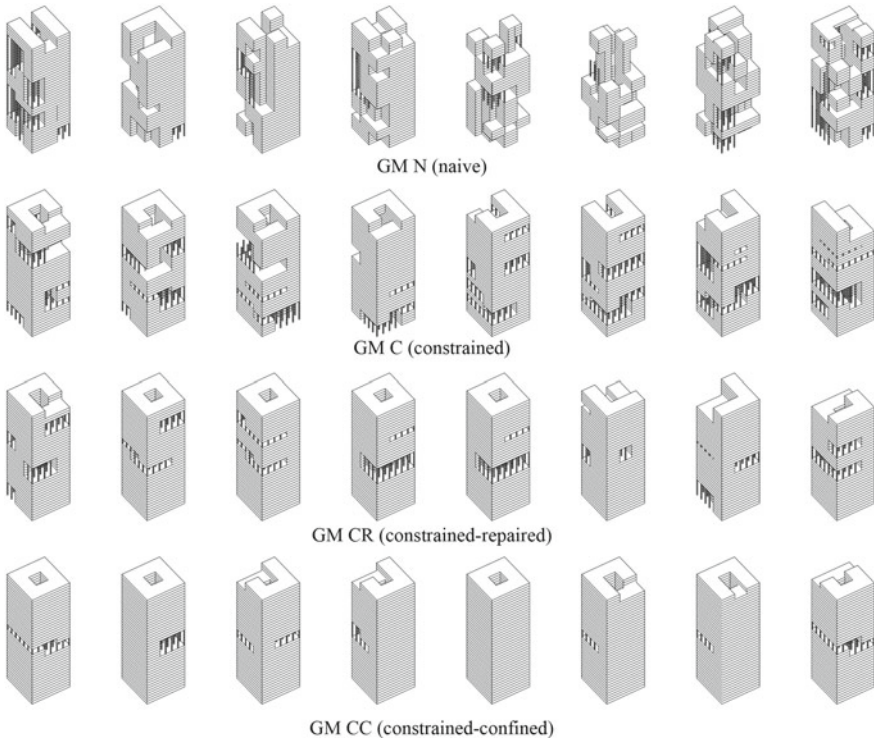


Fig. 1 Random sampling (not evolved) based on the presented GMs

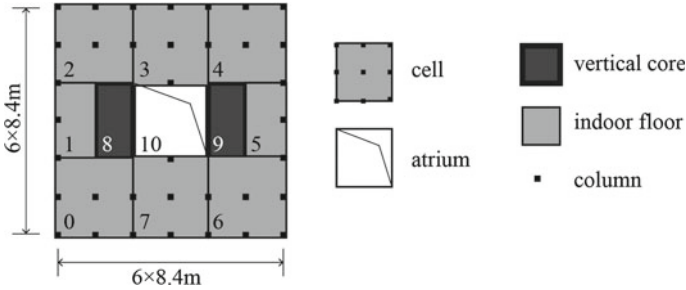


Fig. 2 The structural frame

(#8 and #9 in Fig. 2) which are fixed under all circumstances, all perimeter cells (#0–7 in Fig. 2) can be switched from solid to void, thereby creating complex patterns of interlocking indoor and outdoor spaces. In the presented case study, the tower is assumed to be 40 stories tall.

The evaluation model will first be briefly introduced, followed by the alternative generative models, each with varying levels of constraints.

Evaluation Model

A detailed simulation of environmental performance is beyond the scope of this study, and running relevant simulations would be also too time-consuming. In order to evaluate the generated solutions, a simplified EM based on an economic index is used. This index has the advantage that it is fast to calculate.

For each floor, the fitness function calculates the potential profit that can result from the rentable floor area and subtracts three construction cost factors: the core cost, the slab cost, and the facade cost.

- Potential profit: Rentable floor area multiplied by a factor that gives preference to south facing spaces and spaces on the upper or lower floors (due to the better view or accessibility).
- Core cost: The area of the structural cores in plan multiplied by a factor that increases with the rise of the floor level (due to the difficulty of construction on high).
- Slab cost: Slab area (excluding core but including outside spaces) multiplied by a factor that increases with the rise of the floor level (due to the same reason as the core cost).
- Facade cost: Facade area multiplied by a constant cost factor (due to façade cost mostly related to the material).

In reality, the gross area of buildings is regulated by urban planning codes. As the result, the EM also defines an upper limit of the gross area for the whole building. A solution whose gross indoor floor area surpasses a predefined limit (70,000 m² in this EM) will have its potential profit proportionally scaled back according to the excess area.

Based on the above EM, every design solution will have a distinct fitness. An analysis of randomly generated designs confirmed that the solutions that intuitively seem to be desirable also received high fitness values.

Unconstrained GM

As a comparative baseline for the four presented GMs, a GM called *unconstrained GM* (U-GM) is first introduced. This GM is not actually implemented and also not evolved for testing. No constraints are implemented in this GM, so the number of constraints is 0. The constraint-free mapping allows all possible solid-void combinations to be generated. The number of combinations (search space) is $8^{40} \approx 1.329e+36$. This search space is actually impossible to be searched through under current computational capacity. This GM represents the simplest way of generating the target design solutions; therefore, it can be used as the baseline to reveal the impact of different constraint handling strategies has on compressing the design space.

Naive GM

As to N-GM, floors are grouped into ranges with 2-to-5 consecutive floors, and each group has the same layout of the vertical garden (solid-void patterns). Applying floor groups is not only for reducing the number of parameters but also for the reason that single-floor vertical gardens are uneconomic and too dark.

The tower is divided into 10 groups. As the ten groups will each have variable floors, it may result in either too many or too few floors. Some simple rules are therefore applied in order to ensure that the correct number of floors is achieved. If the total floors are less than 40, then the topmost floor layout will be taken to fill the rest floors. If the total floors are greater than 40, then extra floors will be culled.

In this GM, the solid-void condition of every cell is defined by a binary switch. This results in a genotype–phenotype mapping that is straightforward (without conditional statements, iteration, or subroutines). This simple control flow is easy to implement and often applied to these types of optimization problems.

The genotype defines the layout for ten floor groups. For each group, the genotype contains two parameters. The first parameter is an integer between 2 and 5, defining the number of floors in that group ($p1$ in Fig. 3). The second parameter is a string containing 8 binary switches, defining the solid-void pattern for the eight perimeter cells in that floor group ($p2$ in Fig. 3). As a result, the design space of this GM is $(4 \times 2^8)^{10} \approx 1.268e+30$. The constraint handling strategies of floor groups and numbers of floors within one group impart N-GM with two constraints compared with the U-GM.

At the same time, the simple mapping process results in N-GM having the most regular fitness landscape compared with the other GMs (aside from the U-GM). The independent binary combinations allow a wide range of possible design solutions to be generated. However, this unrestricted diversity also allows many invalid design solutions to be generated as stochastic combinations of vertical gardens can result in unbuildable designs. For example, solutions may have very large or disproportionate voids or many separated small voids on the facades, or, in some other cases, suspended or large overhanging cells (see the first line in Fig. 1). Such problematic features will result in the expensive construction cost or make it hard to rent due to poor spatial accessibility or connectivity [20].

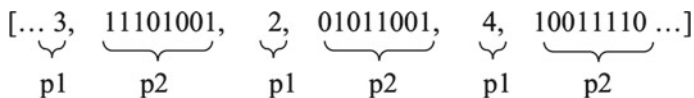


Fig. 3 Example of genotype data structure of N-GM

Constrained GM

The C-GM limits the number and size of vertical gardens. First, the number of vertical gardens is limited to one per floor, as multiple small vertical gardens result in a huge façade area which is costly in building materials. Second, the size of a vertical garden should be controlled and should not be significantly larger than that of the indoor space for the rental profitability. In addition, vertical gardens should be connected to the atrium to facilitate natural ventilation [19].

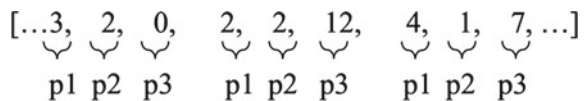
For C-GM, the genotype still defines the layout for ten floor groups. However, in order to constrain the GM to the above rules, certain modifications were introduced into the mapping process. For each floor group, the genotype now contains three parameters on different decision levels. The first parameter is the same as the N-GM and defined the number of floors in that group ($p1$ in Fig. 4).

The second and third parameters ($p2, p3$ in Fig. 4) replace the binary string ($p2$ in Fig. 3). These parameters are used to create voids through a mapping process with conditional statements. Since there are only two cells directly connecting the atrium, the vertical garden must include one of these two cells. Thus, the second parameter ($p2$ in Fig. 4) is either 0, 1, or 2. If the value is 0, then it indicates that there will be no void, in which case the third parameter can be ignored. If the value is 1 or 2, then it indicates which one of the two cells adjacent to the atrium will be included in the vertical garden. Finally, the third parameter ($p3$ in Fig. 4) is an integer that assigns a solid-void pattern from a predefined set for the vertical garden. To restrict the size of the vertical garden, the number of cells in each void pattern is limited to a maximum of 5, which results in a total of 13 unique patterns (Fig. 5). As the result, the search space of this GM is $(4 \times 3 \times 13)^{10} \approx 8.536e+21$. The two extra constraints on floor layouts make the C-GM with two more constraints compared with that of N-GM (totally four constraints).

By excluding most stochastic combinations of small voids in the building volume, the rationality of the generated design solution of C-GM is improved considerably (see the second line in Fig. 1). At the same time, there is a significant compression of the design search space compared with that of N-GM.

However, the explicit constraint rules also result in a more irregular fitness landscape, due to the introduction of conditional statements into the mapping. These statements result in discontinuities and *neutral mutations* in the genotype–phenotype–fitness mappings. Neutral mutation refers to genotypic mutations that have no effect on the phenotype and the fitness. (For example, in Fig. 5, $p2$ is a higher order decision level that will have a more significant impact on the design fitness. $p3$ becomes neutral when $p2$ defines that no vertical gardens are generated.) Such neutral mutations introduce many-to-one mappings in the GM, resulting in numerous

Fig. 4 Example of genotype data structure of C-GM



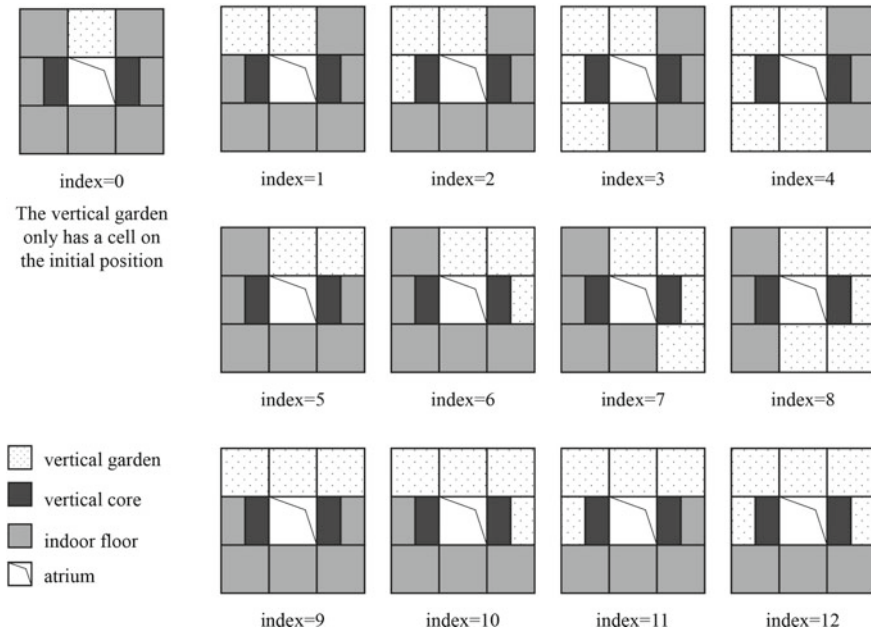


Fig. 5 Floor layout patterns

Table 1 The frequency of neutral mutation of the presented GMs

GM	N-GM	C-GM	CR-GM	CC-GM
Rental profit	7	30	35	20
Constructional cost	6	30	34	22
Gross profit	6	27	32	14

fitness plateaus. Such plateaus can trap the evolutionary process into subspaces with local optimals, thereby resulting in premature convergence [14, 23].

To analyze the frequency of neutral mutations in the presented GMs, 100 pairs of randomly sampled solutions from separated neighboring genotype subspaces were selected and evaluated, and pairs sharing the same fitness values were then counted (Table 1). As shown in Table 1, the additional constraints of the conditional statement result in a significant rise in the frequency of neutral mutations of C-GM compared with that of N-GM.

Although the configuration of the vertical gardens with C-GM has become more rational, the independence between floors layouts can still create certain types of voids that may be problematic. Two key types of problematic voids are identified: oversized voids in cases where two voids meet above one another and become merged, or cross-diagonal voids in cases where two voids meet at a point on the diagonal. Such voids are hard to avoid within the mapping process of C-GM.

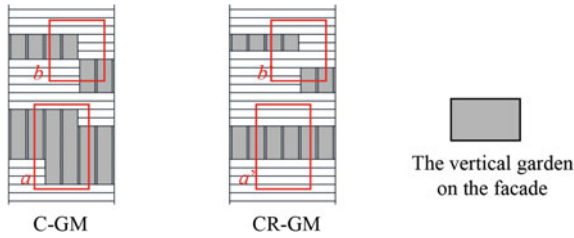


Fig. 6 Example of the repair operators

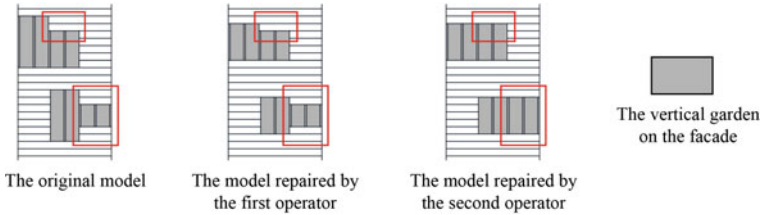


Fig. 7 Example of the first and the second repair operators

Constrained-Repaired GM

The CR-GM uses the same control flow as the C-GM. However, in order to correct the oversized voids and cross-diagonal voids generated by the C-GM, implicit repair operators are added. The repair operators will fix the oversized voids and cross-diagonal voids by switching cells on selected floors to become non-void.

In the case of the oversized void (larger than five floors), floors are iteratively removed from the top and the bottom of the void, until a suitable height is reached (*a-a'* in Fig. 6). In the case of the cross-diagonal void, all cells on the floor in the middle will be assigned non-void, so that the two voids become disconnected (*b-b'* in Fig. 6).

These repair operators may, however, result in additional problematic conditions being generated. In particular, inserting non-void floors in certain groups can result in many single-floor pendulous cells which are hard to rent or construct. Hence, an extra repair operator is defined in order to correct these conditions. This repair operator will identify isolated or pendulous cells and will switch them to the opposite solid-void condition (Fig. 7). Due to the fact that additional problematic conditions can continuously emerge after the execution of the first and the second repair operators, these operators are run in a loop until all infeasible conditions have been eliminated or the number of iterations reaches a predefined limit. The number of 30 is set as the limit in this GM.

These repair operators are able to filter out most invalid design solutions from the C-GM by further restricting the variability of vertical garden combinations (see the third line in Fig. 1). Including the first and the second repair operators, three more

constraints are implemented in the CR-GM compared with that of the C-GM (totally seven constraints). However, as the implicit rule has no effect on the data structure of the genotype, the genotype space remains intact.

With more constraints being embedded, the fitness landscape of CR-GM is further degraded, as the repair operators lead to additional neutral mutations in the phenotype space (Table 1), which makes the fitness landscape more irregular [24]. However, the number of possible combinations can be reduced remarkably by these neutral mutations. Lastly, a significant additional coding effort was required for implementing the more sophisticated mapping process.

Constrained-Confined GM

The CC-GM also uses the same control flow as the C-GM, but compared to the CR-GM, the search space is further compressed by only keeping parameters that have a mostly positive impact on overall fitness. (See the CC-GM solutions in the fourth line in Fig. 1.)

For CC-GM, vertical gardens are only allowed to be inserted in middle to upper floors, and there is a terrace on the roof. To ensure these features can be fully represented, CC-GM only has three floor groups (as opposed to the C-GM, which has ten floor groups).

The first two groups are assigned to floors ranging from 15 to 30 stories, and the third one defines the terrace on the roof. Therefore, the regularity of the fitness landscape of CC-GM is similar to that of C-GM, but the size of the genotype space is much smaller, which is $(4 \times 3 \times 13)^3 \approx 3.796e+6$. Compared with C-GM, seven more constraints are defined to disable the change of the remaining seven floor groups (totally 11 constraints).

Evolutionary Run

In order to further investigate the impacts of different constraint handling strategies on search efficiency, program complexity, design novelty, and overall utility, the evolutionary search processes based on the four presented GMs were run.

The evolutionary algorithm was executed using the Rhino–Grasshopper environment, and the standard genetic algorithm in the Galapagos was applied [25]. The population size was set to 100. Due to the large genotype space for some of the presented GMs, the population of the initial generation was raised to 1000. Meanwhile, to avoid the premature convergence, a higher mutation rate and a lower selection pressure were used. (In Galapagos, the settings are 25% for *maintain* and 25% for *inbreeding*.) At the same time, the number of 25 consecutive generations without new improvement solutions is set as the terminated threshold for the evolutionary

process. Last but not the least, in order to reduce the impact of stochastic variation, the evolutionary process based on each GM was repeated five times.

Results

For the presented case study, different constraint handling strategies impart each GM with distinct constraint numbers and genotype search space. Table 2 summarizes the number of constraints and the size of the design search space for all five GMs. In general, the size of search space decreases along with more constraints being embedded. Compared with the size of the search space of U-GM, the effect of the constraint handling strategies of C-GM on compressing the search space is more significant than that of any other GM.

Figure 8 shows the fitness progression trend lines of the evolutionary processes. For each GM, five trend lines are shown. The graphs show the best two solutions over time. The reason for recording the best two is that focusing only on the best solution can conceal the overall progress of the whole population. By recording the best two solutions, the improvement of the population can be revealed more subtly and precisely.

The tendency of the trend lines corresponds to the regularity of the fitness landscape. The smoother the landscape (such as N-GM), the more gently and smoothly the trend line grows, which visually reveals the correlation between the constraint handling and the utility of the evolutionary process. Except for N-GM, the fitness landscapes of the other three GMs are irregular to different extents. As a result, the trend lines also become correspondingly more irregular.

From the graph, it can be found that smoothness of the fitness progression trend lines has a strong correlation with the frequency of neutral mutations. The result in Table 1 shows that the frequency of the neutral mutations of both the C-GM and CR-GM are very high. Around or over 30% of the samples share the same fitness values, followed by CC-GM with about 20%, and N-GM with about 7%. As neutral mutations become more frequent, the trend lines grow more irregularly.

Aside from the neutral mutations, the repair operators also have significant impacts on the evolutionary process. Despite the fact that the search space of CR-GM is much bigger than that of CC-GM, the sophisticated repair operators of CR-GM not only facilitate the evolutionary process to converge faster but also result in the discovery

Table 2 The number of constraints and the size of design search space

GM	U-GM	N-GM	C-GM	CR-GM	CC-GM
The number of constraint	0	2	4	7	11
Design search space	1.33e+36	1.27e+30	2.06e+14	2.06e+14	3.796e+6

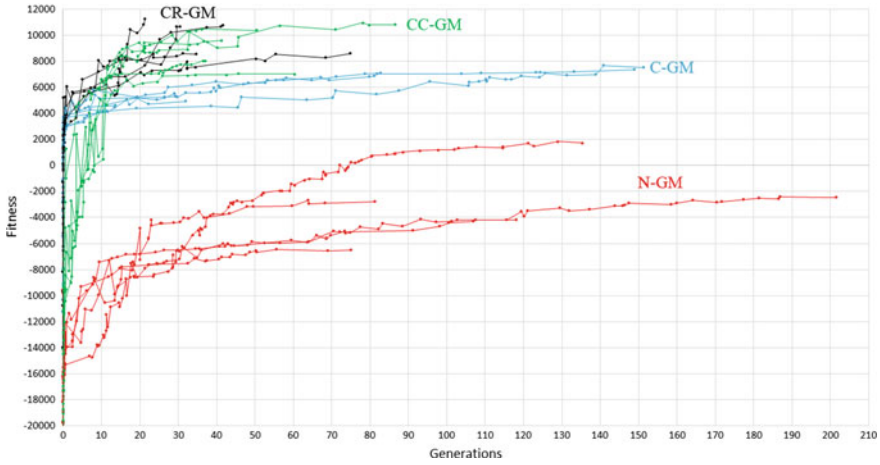


Fig. 8 Fitness progression trend lines for the alternative GMs. The number of generations is plotted along the x-axis, and fitness values on the y-axis

of better solutions. As the result, it can be assumed that the actual size of phenotype space s of CR-GM, which is compressed by the repair operator, is similar to that of CC-GM.

Search Efficiency

As demonstrated in Fig. 8, the search efficiency of the four presented GMs varies significantly. In general, adding more constraints both reduces the number of generations required to find viable solutions and improves the quality of the solutions that are found. The conclusion that can, therefore, be drawn is that, for this case study, the positive effects of a smaller search space outweigh the negative effects of an irregular fitness landscape.

Program Complexity

Constraint handling did not result in significant expansion in the programs' physical sizes of the GMs in this study (500–900 lines). However, the actual increase in the coding effort and time spent on the more complex control flows were considerable. By analyzing the cyclomatic complexity (M) based on the below formula [15], the latent effects brought by the complex control flows are revealed more precisely.

$$M = E - N + 2$$

Table 3 The cyclomatic complexity of the presented GMs

GM	N-GM	C and CC-GM	CR-GM
Nodes	2	6	11
Edges	2	8	16
Complexity	2	4	7

where E is the number of edges in the control flow, and N is the number of nodes.

As shown in Table 3, the complexity of the four GMs roughly increases exponentially, and the numerical differences of the values generally match the actual differences between the amount of coding effort and time spent. As the control flow becomes more complex, much more effort on debugging and refactoring has to be spent to maintain the program. However, the complex control flow and the associated effort can be offset or even outnumbered by the time saved in the evolutionary process.

Design Novelty

Figure 9 lists the results from the evolutionary processes. Similar to Fig. 1, the solutions become more rational as more constraints are embedded in the GM. However, improper use of constraint handling can make the evolutionary results suffer from poor design novelty. The design solutions generated by CR-GM and CC-GM are mostly predictable and lack *design surprise* which means that “the design is unexpected for the domain given previous experience [9, 10].”

In contrast, the solutions generated by N-GM have greater formal diversity but still cannot be regarded as having the desirable design novelty since they cannot be seen as being feasible architectural solutions. This is reflected in their low fitness values, which suggest that these solutions are not economical.

The results from C-GM suggest that there is a possible balance between the need for design novelty and design fitness. Although the designs are topologically similar to that form CC-GM and CR-GM (with similar locations and numbers of the vertical gardens), the less stringent constraint handling allows more unexpected solutions to be discovered. Furthermore, the regularity of the fitness landscape also facilitates the evolutionary process to search the design space more thoroughly, allowing a greater number of alternative design solutions to be evaluated. As the result, the C-GM solutions have more distinct formal features than the CR-GM and CC-GM solutions.

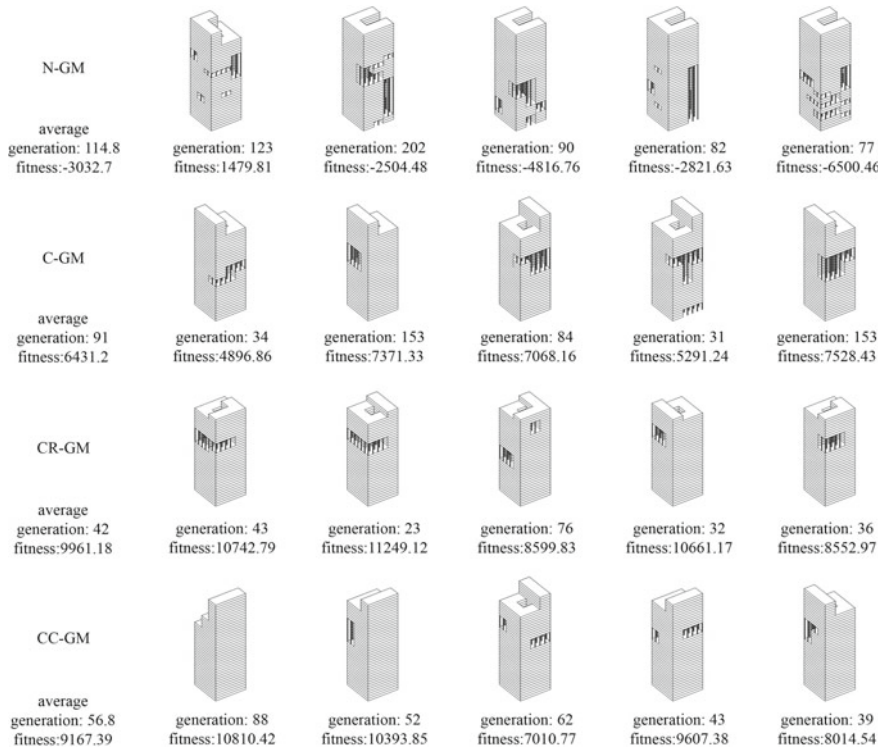


Fig. 9 Evolved design solutions based on the presented GMs

Table 4 Overall qualitative description of the presented GMs

GM	Constraint handling	Search efficiency	Design novelty	Program complexity	Utility
N-GM	Loose	Low	Very high	Low	Low
C-GM	Medium	Fair	High	Fair	High
CR-GM	Tight	Very high	Fair	High	High
CC-GM	Very tight	High	Low	Fair	Low

Utility

By summarizing the utility factors of the four presented GMs, a qualitative conclusion is drawn, as shown in Table 4. Due to the extremely poor search efficiency or low design novelty, it is fair to consider that N-GM and CC-GM are least useful for real-world scenarios.

The utility of the other two GMs, in contrast, is recognized as much better, but it is also affected by external conditions. For CR-GM, the ability to quickly discover viable solutions minimizes the number of evaluations that are required. This allows

it to be used in ED systems incorporating computational expensive simulations. However, the limited design variability may make this GM only effective for well-defined design problems.

On the contrary, if the simulation is relatively inexpensive or the design problem is ill-defined, C-GM is likely to be the better choice. The relatively regular fitness landscape of the C-GM facilitates the evolutionary process to search the design space more completely, and the potential to discover novel design alternatives is also higher.

Conclusion

In this study, the utility of constraint handling in GMs has been researched. For the case study investigated in this research, utility factors vary significantly when different constraint handling strategies are applied in the GM. On the one hand, constraint handling has a positive impact on both the search efficiency and design fitness. However, on the other hand, overly stringent constraint handlings can significantly weaken the other utility factors, especially design novelty.

Exclusively focusing on search efficiency by embedding evermore constraints in the GM is unlikely to be an effective strategy, as it will result in low design novelty and complex control flows which are hard to implement and maintain. As the result, a more balanced approach to constraint handling is critical to achieving effective and efficient evolutionary processes. For architects, in order to ensure that the resulting ED system is applicable for the purpose, the different utility factors should be carefully considered before constructing a GM.

Last but not the least, the impacts of constraint handling on the utility of ED may vary considerably across different GMs, and it is therefore not possible to draw generalized conclusions until more research under different design scenarios has been conducted. However, the importance of the overall utility is clearly revealed in this study, and further research will facilitate architects to carry out ED more efficiently and effectively in the future.

Acknowledgements This paper was supported by the National Natural Science Foundation of China (51378248) and the China Scholarship Council (201706190203).

References

1. Caldas L (2008) Generation of energy-efficient architecture solutions applying GENE_ARCH: an evolution-based generative design system. *Adv Eng Inform* 22:59–70
2. Eiben AE, Smith JE (2004) *Introduction to evolutionary computing*. New York
3. Gero JS (2006) Computational models of creative designing based on situated cognition. In: Hewett T, Kavanagh T (eds) *Creativity and cognition 2002*. ACM Press, New York, NY, pp 3–10
4. Frazer J (1995) *An evolutionary architecture*. Architectural Association, London

5. Bentley P, Kumar S (2003) Three ways to grow designs: a comparison of embryogenies for an evolutionary design problem. In: Proceedings of the 1st annual conference on genetic and evolutionary computation, vol 1, pp 35–43
6. Kumar S, Bentley P (2003) Computational embryology: past, present and future. *Adv Evol Comput*:1–16
7. Zhou L, Haghghat F (2009) Optimization of ventilation systems in office environment part II: results and discussions. *Build Environ* 44:657–665
8. Jo JH, Gero JS (1998) Space layout planning using an evolutionary approach. *Artif Intell Eng* 12:149–162
9. Chen S, Montgomery J, Bolufé-Röhler A (2015) Measuring the curse of dimensionality and its effects on particle swarm optimization and differential evolution. *Appl Intell* 42:514–526
10. Rasheed KM (1998) GADO: a genetic algorithm for continuous design optimization, Ph.D. dissertation, Rutgers University, New Jersey
11. Rasheed K, Ni X, Vattam S (2005) Comparison of methods for developing dynamic reduced models for design optimization. *Soft Comput* 9:29–37
12. Banzhaf W (1994) Genotype-phenotype-mapping and neutral variation—a case study in genetic programming. *Parallel Probl Solving Nat III* 866:322–332
13. Janssen P, Kaushik V (2014) Evolving Lego, rethinking comprehensive design: speculative counterculture. In: Proceedings of the 19th international conference on computer-aided architectural design research in Asia:523–532
14. Rothlauf F (2006) Representations for genetic and evolutionary algorithms. Springer, Berlin, Heidelberg
15. McCabe TJ (1976) A complexity measure. *IEEE Trans Softw Eng SE- 2*:308–320
16. Rebhuhn C, Gilchrist B, Oman S, Tumer I, Stone R, Tumer K (2015) A multiagent approach to identifying innovative component selection. In *Des Comput Cogn* 14:227–244
17. Brown DC (2015) Computational design creativity evaluation. *Des Comput Cogn* 14:207–224
18. Grace K, Maher ML, Fisher D, Brady K (2015) Modeling expectation for evaluating surprise in design creativity. *Des Comput Cogn* 14:189–206
19. Wood A, Salib R (2013) Guide to natural ventilation in high rise office buildings. Routledge
20. Liggett RS (2000) Automated facilities layout: past present and future. *Autom Constr* 9:197–215
21. Dino IG (2016) An evolutionary approach for 3D architectural space layout design exploration. *Autom Constr* 69:131–150
22. Aysin SEV, Özgen A (2009) Space efficiency in high-rise office buildings. *Metu Jfa*:2
23. Vanneschi L, Clergue M, Collard P, Tomassini M, Vérel S (2004) Fitness clouds and problem hardness in genetic programming. *Genet Evol Comput GECCO2004 Part II* 3103:690–701
24. Rothlauf F, Goldberg DE (2003) Redundant representations in evolutionary computation. *Evol Comput* 11:381–415
25. Rutten D (2013) Galapagos: on the logic and limitations of generic solvers. *Archit Des* 83:132–135

Exploring the Feature Space to Aid Learning in Design Space Exploration



Hyunseung Bang, Yuan Ling Zi Shi, Guy Hoffman,
So-Yeon Yoon and Daniel Selva

In this paper, we introduce the concept of exploring the feature space to aid learning in the context of design space exploration. The feature space is defined as a possible set of features mapped in a 2D plane with each axis representing different interestingness measures, such as precision or recall. Similar to how a designer explores the design space, one can explore the feature space by observing how different features vary in their ability to explain a set of design solutions. We hypothesize that such process helps designers gain a better understanding of the design space. To test this hypothesis, we conduct a controlled experiment with human subjects. The result suggests that exploring the feature space has the potential to enhance the user's ability to identify important features and predict the performance of a design. However, such observation is limited only to the participants with some previous experience with design space exploration.

Introduction

Over the last two decades, the “design by shopping” paradigm [1] has become a popular approach to tackle early-phase (conceptual design or system architecting) engineering design problems. An important step in this approach is called design space exploration (a.k.a. tradespace exploration), where the designer analyzes the structure of the design space and learns about the trade-offs in the system, sensitivities of design criteria to design decisions, couplings between design decisions, etc. For the remainder of this paper, “learning” in tradespace exploration refers to gaining knowledge about these parameters, and more generally about the mapping between

H. Bang (✉) · Y. L. Z. Shi · G. Hoffman · S.-Y. Yoon · D. Selva
Cornell University, Ithaca, NY, USA
e-mail: hb398@cornell.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_11

design decisions and design criteria. Through the process of design space exploration, designers can make a more informed decision for the selection of the final design.

However, design space exploration presents us with the challenge of information overload. The problem of information overload becomes more prominent in design tasks involving many design decisions, multiple objectives, and intricate couplings between them. It has been shown that as design problems get more complex, designers are overwhelmed by the size and the complexity of the data, thus leading to the degradation of their ability to understand the relationships between different variables [2–4].

To address this issue, various data visualization methods and tools have been developed for design space exploration [5–14]. Most of these tools focus on providing different views of designs defined in a multidimensional space, in some cases coupled with unsupervised machine learning methods such as clustering, feature selection, and manifold learning. However, due to the knowledge being implicit in visualization, these methods require an additional step for the humans to visually inspect the result and make interpretations. Therefore, the knowledge obtained through visualization can be ambiguous and subjective. Moreover, visually inspecting and finding patterns may be challenging without sophisticated rearranging strategies [15, 16].

Another complementary approach to learn about the design space is to extract knowledge using data mining algorithms that mine knowledge explicitly in the form of logical *if-then* rules [17–20]. These methods can be used to extract *driving features*, i.e., the common features (specific values of design decisions, attributes, or combinations thereof) that are shared by a group of designs that exhibit similar objective values [21]. For example, Watanabe et al. use association rule mining to analyze hybrid rocket engine designs, and find that 83% of all non-dominated (Pareto optimal) solutions had a similar initial port radius [22]. The major advantage of such knowledge is that it can be expressed relatively concisely and unambiguously through a formal representation [23].

While having been used successfully in the past to analyze design spaces, these methods are not without limitations. One of the limitations of the current methods is that they impose a rigid structure in the mined features (the conditional “if” parts of the rules); indeed, all features are represented as predicates (i.e., binary features) joined by logical conjunctions (i.e., “and” operator). From a mathematical point of view, this does not reduce expressivity, as any logical formula can be converted into a disjunctive normal form or DNF (i.e., a disjunction—OR—of conjunctive clauses) [24]. Therefore, any Boolean concept (a concept whose membership is determined by a combination of binary features [25]) can be represented using a set of rules (disjunction of rules). However, from a human learning point of view, the conversion to DNF often results in longer features, and thus harder to understand by humans.

Another limitation of the data mining methods is that they generate a large set of features without an easy way to identify the most useful and informative one [26]. Identifying a single feature that best explains the region of interest of the design space while staying compact could improve learning. One approach to select a single feature is to sort all features using one measure such as *confidence* or *lift* [27]. Intuitively, these interestingness measures provide a quantitative metric of the predictive power of the feature. However, selecting a single metric from a large list of alternatives

can be arbitrary, and may not necessarily be the right measure for the given design problem [28, 29].

In this paper, we present a new method, feature space exploration, to aid human learning in design space exploration, and a tool to use with the method. The aim of this method is to improve the designer's ability to identify important features and generate new insights. In order to foster learning, we enable designers to explore various forms of features and get immediate feedback on how well these features explain a certain region of the design space (e.g., a cluster, or the Pareto front). This is done by defining a space of all possible features (called the feature space), visualized on a 2D plane. Each axis in the plane represents one of the interestingness measures of features used in classification (e.g., *precision* and *recall* [29]) or association analysis (e.g., *confidence*, *lift*, and *Gini index* [28]). If one selects conflicting goodness measures such as *precision* and *recall* [30], the Pareto front of the feature space can also be defined. The designer can then use the visualization to observe how the goodness measures change in response to a change in the feature, and elicit his or her preferences among those two important measures. Due to its similarity to how a designer explores the design space, we refer to this process as "exploring the feature space". Exploring the feature space helps the designer identify the driving features that shape the structure of the design space. The process takes advantage of the intuitive and fast nature of exploring options through visualization, as well as the ease of learning through formal representations that are clear and concise.

To demonstrate the effectiveness of this new method to improve learning, we conduct a controlled experiment with human subjects. The experiment tests whether exploring the feature space improves the designer's learning, which is measured as his/her ability to predict whether a given design will exhibit desirable performance and cost. The result shows that exploring the feature space may indeed improve learning about the design space but only under certain conditions—for subjects who have received some formal training in design space exploration.

Example Design Problem: Architecting Earth Observing Satellite System

Before explaining how the proposed method works, we first introduce an example design problem to help explain the methodology in the remainder of the paper. It should be noted that the proposed method is not specific to a type of design problem. However, there are some implementation details that are tailored to the structure of this problem. This point will be elaborated on after the design problem is outlined.

The design problem is a real-world system architecting problem previously studied in [31]. The goal of the design task is to architect a constellation of satellites to provide operational observations of the earth's climate. There are two objectives: maximizing the scientific benefit and minimizing the lifecycle cost. The scientific benefit is a function of an architecture's satisfaction of 371 climate-related measurement objectives, generated based on the World Meteorological Organization's

OSCAR (Observing Systems Capability Analysis and Review Tool) database.¹ The level of satisfaction of each measurement objective is quantified based on the capabilities of each design and then aggregated to obtain a number that represents how much scientific benefit each design brings to the climate scientific community.

The design problem has been formulated as an assignment problem between a set of candidate measurement instruments (space-based sensors related to climate monitoring) and a set of candidate orbits (defined by orbital parameters such as altitude and inclination). Given a set P of candidate instruments and a set O of candidate orbits, the design space is defined as a set of all binary relations from P to O . Each instrument in P can be assigned to any subset of orbits in O , including the empty set. Therefore, the size of the design space is $2^{|P||O|}$, where $|P|$ is the number of candidate instruments and $|O|$ is the number of candidate orbits. In this work, we considered 12 candidate instruments and 5 candidate orbits, making a total of 2^{60} possible designs. Each design is represented by a Boolean matrix M of size 5×12 , where $M(o, p) = 1$ if instrument p is assigned to orbit o , and $M(o, p) = 0$ otherwise. Graphically, this can be displayed by a figure similar to Fig. 1. Here, each row represents a mission that will fly in each orbit, and the columns represent the assignment of different instruments. Note that in the examples that will follow throughout this paper, we replace the names of the actual orbits and instruments with numbers (e.g., 1000, 2000) and alphabetical letters (e.g., A, B, C) to simplify the presentation of the examples.

Once the design decisions and the corresponding objective values are provided in a structured format, the proposed method mostly considers the design problem as a black box. At the implementation level, however, there is one critical step necessary in order to run data mining, which is formulating the base features. The base features are predicates used to construct more sophisticated Boolean concepts related to the design space. In its simplest form, a base feature can be a single design decision

orbit	Inst 1	Inst 2	Inst 3	Inst 4
LEO-600-polar	AERO_POL	AERO_LID	CHEM_SWIRSPEC	
SSO-600-AM	CPR_RAD	VEG_LID	CHEM_UVSPEC	SAR_ALTIM
SSO-600-DD	VEG_INSAR	HIRES_SOUND		
SSO-800-DD	VEG_LID			
SSO-800-PM	CPR_RAD	CHEM_UVSPEC		

Fig. 1 An example architecture representation. Each row represents a single spacecraft flying in a certain orbit. For example, a spacecraft carrying the cloud and precipitation radar (CPR_RAD) and the UV/VIS limb spectrometer (CHEM_UVSPEC) will fly in a sun-synchronous orbit at an altitude of 800 km, and an afternoon local time of the ascending node

¹<http://www.wmo-sat.info/oscar/>.

Table 1 Base features

Name of the feature	Arguments	Description
Present	I_i	Instrument I_i is present in at least one of the orbits
Absent	I_i	Instrument I_i is absent in all the orbits
InOrbit	$O_i, I_j, (I_k, I_l)$	Instrument I_j (and I_k, I_l) is/are present in orbit O_i
NotInOrbit	$O_i, I_j, (I_k, I_l)$	Instrument I_j (and I_k, I_l) is/are not present in orbit O_i
Together	$I_i, I_j, (I_k)$	Instruments I_i, I_j (and I_k) are present together in any orbit
Separate	$I_i, I_j, (I_k)$	Instruments I_i, I_j (and I_k) are not present together in any orbits
emptyOrbit	O_i	No instrument is present in orbit O_i
numOrbits	n	The number of orbits that have at least one instrument assigned is n

set to 0 or 1. However, we introduce more complex base features to prespecify the structure of the patterns to be searched, thus biasing the search toward more promising regions in the search space. The base features used for the current system architecting problem are shown in Table 1. The formulation of such base features requires some domain-specific knowledge and insights obtained by observing the structure of the design problem. Based on those insights, we can speculate which form of features may drive the performance of a design.

For example, *Present* is a base feature that describes whether an instrument i is used in at least one of the orbits. This feature is equivalent to a disjunction of five base features (instrument i being assigned to each one of the orbits). *Present* may potentially speed up the search, since the decision whether to use an instrument or not has a bigger influence in the objective value compared to the decision of which orbit it should be assigned to. While such decision may or may not be useful in capturing the driving features, introducing the predicate *Present* helps searching that hypothesis space effectively. In the remaining sections of this paper, we will use this predefined set of base features to build more complex features.

Exploring the Feature Space

In this paper, we propose exploring the feature space as a learning aid for design space exploration. We define the feature space as a set of all possible features, visualized by mapping features in a coordinate system where each axis represents a different measure of the goodness of a feature (e.g., *precision*, *recall*, *F score*, *confidence*, *lift*, and *mutual information* [28, 29]). In the following sections, we introduce the

graphical user interface that enables visualizing and exploring the feature space and explain how a designer can use it for insight generation and learning.

iFEED

The capability to explore the feature space is built as an extension to the interactive knowledge discovery tool called iFEED [21]. Its goal is to help engineering designers learn interesting features that drive designs toward a particular region of the objective space as they interact with the tool. A user of iFEED can select a group of target designs, and run data mining algorithms to extract the common features that are shared by those designs. The main interface of iFEED is shown in Fig. 2. It consists of an interactive scatter plot, which shows the design space populated by thousands of alternative designs. When the user hovers his or her mouse over one of the points in the scatter plot, the information about that design is displayed below the scatter plot.

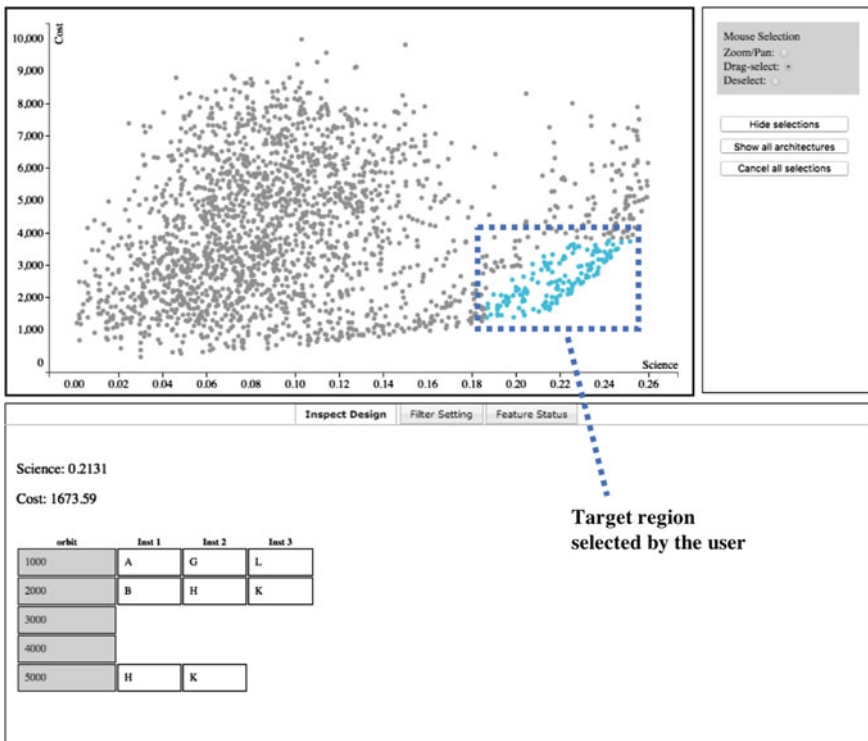


Fig. 2 The main graphical user interface of iFEED, which consists of a scatter plot showing the objective space and a display of the design that is currently viewed. Dots highlighted in cyan represent the target region selected by the user

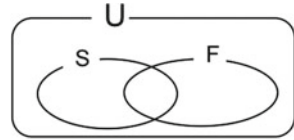
The displayed information includes values of the objectives and design decisions of a design.

The scatter plot can help the user select a region of interest in the objective space. When the user drags the mouse over the scatter plot, designs in the selected region are highlighted, and they are considered as target solutions when running the data mining process.

The data mining process is based on the Apriori algorithm, which is one of the earliest and most popular algorithms developed for association rule mining [32]. The algorithm has been extended to mine classification association rules, which follow the structure $X \rightarrow C$. Here, X is a feature that describes a design, and C is a class label that indicates whether a certain design belongs to the target region (cyan area in Fig. 2) or not. The data mining returns a list of features that are shared by the target designs. For more details on the data mining algorithm, readers are referred to [21].

Visualization of Feature Space

The features extracted by running the data mining algorithm have varying level of “goodness” in explaining the target designs. Such measures can be defined using various metrics used in binary classification and association rule mining [28, 29]. In this work, we use two measures of *confidence* defined as follows.



U: All possible designs
 S: Designs within target region
 F: Designs with the feature

$$conf(S \rightarrow F) = \frac{supp(S \cap F)}{supp(S)}$$

$$conf(F \rightarrow S) = \frac{supp(S \cap F)}{supp(F)}$$

Here, S is the set of all designs that are in the target region, and F is the set of all designs that have the particular feature that is being considered. *supp* stands for *support*, which is defined as

$$supp(X) = \frac{|X|}{|U|}$$

where U is the set of all designs in the database and $|\cdot|$ indicates the cardinality of the set. *Confidence* is often used in association rule mining to represent the strength of a rule [32]. $conf(S \rightarrow F)$ represents how complete the feature is in terms of the

fraction of the target region that exhibits the feature, while $conf(F \rightarrow S)$ represents how consistent or specific the feature is in explaining only the target region (fraction of designs with the feature that is in the target region). In fact, because we extract only binary classification rules, $conf(S \rightarrow F)$ and $conf(F \rightarrow S)$ are equivalent to *recall* and *precision*, respectively.

After we calculate both *confidence* measures for the extracted features, we can map them in a two-dimensional plane with each axis representing one of the *confidence* measures, as shown in Fig. 3. This visualizes the feature space as we defined at the beginning of this section. In the figure, each triangle is a feature obtained from the data mining algorithm. The general trend in the mined features shows that there is a trade-off between the two *confidences*, consistent with the relationship often seen between *recall* and *precision*.

The scatter plot displaying the feature space is also implemented as an interactive plot. When the user hovers the mouse over a feature in Fig. 3, the designs that have the feature are highlighted in the scatter plot as shown in Fig. 4. From these figures, the user can get a quick and intuitive sense of how the feature is distributed within the design space. For example, Fig. 4a shows a design space, and it highlights a feature whose $conf(S \rightarrow F)$ is high and $conf(F \rightarrow S)$ is low. This feature explains most of the target designs, but it is too general, such that it also covers many other designs that are not in the target region. In contrast, the feature highlighted in Fig. 4b has low $conf(S \rightarrow F)$ and high $conf(F \rightarrow S)$. The designs that have this feature fall mostly inside the target region, but only a small portion of the target region is explained by this feature.

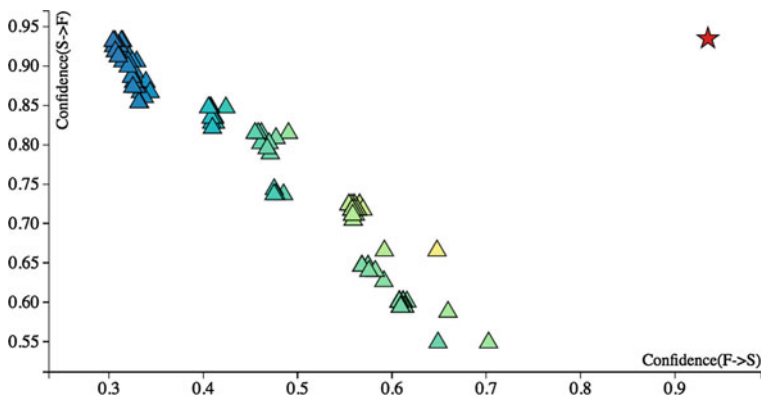


Fig. 3 Feature space plot, where each axis is one of the confidence measures. Each triangle represents one feature. The red star (upper-right corner) represents the utopia point of the feature space (Color figure online)

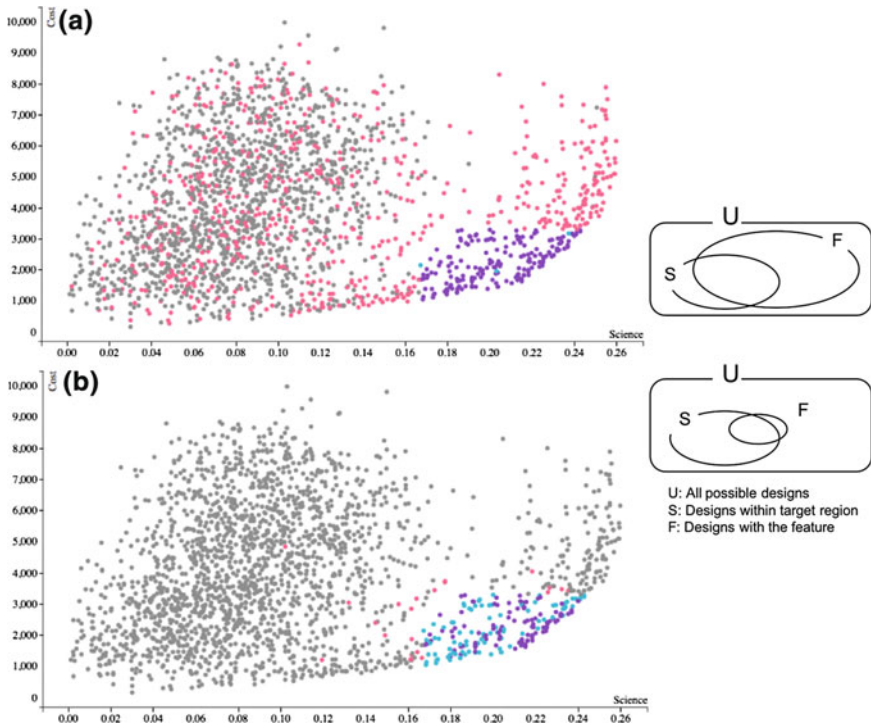


Fig. 4 Design space highlighting different features. **a** Designs that have the feature with high $\text{conf}(S \rightarrow F)$ and **b** another feature with high $\text{conf}(F \rightarrow S)$ are highlighted. The cyan dots are the target designs. The pink dots are the designs that have the feature. The purple dots are the overlap of those two sets of designs. The Venn diagram depicts the proportions of the highlighted designs

Representing and Modifying Features

When the user hovers a mouse over a feature in the feature space plot, a tooltip appears with the name of the feature. For example, the following text represents a feature that consists of two base features linked with a conjunction. “

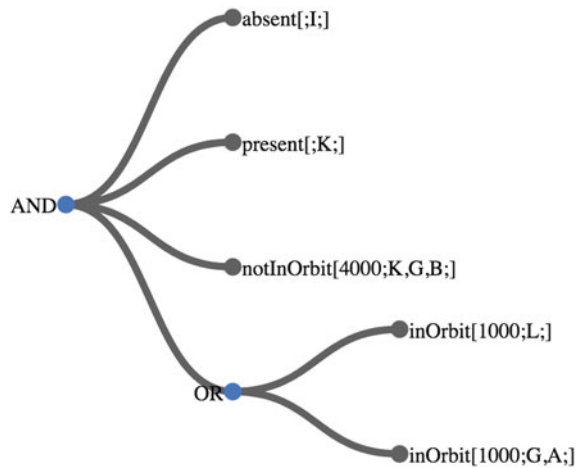
$$\text{absent}(I) \text{ AND } \text{present}(K)$$

”

In natural language, this can be interpreted as, “Instrument I is not used in any orbit, and instrument K is used in at least one of the orbits.” However, in our tool, such representation can only be used to view the extracted features and cannot be used to modify the given feature or input a new one.

In order to enable the user to modify and explore other features, we implemented a graphical representation of the feature as shown in Fig. 5. This representation uses a tree structure, consisting of two types of nodes. A leaf node represents a base feature,

Fig. 5 Representation of a feature using a graph. The displayed feature consists of five base features linked using both conjunctions and disjunctions. The feature can be interpreted in text as “absent(I) AND present(K) AND notInOrbit(4000, K, G, B) AND (inOrbit(1000, L) OR inOrbit(1000, G, A))”



and a logical connective node represents a logical connective (logical conjunction or disjunction) that links all its children nodes. Therefore, the feature shown in Fig. 5 can also be written in text as: “*absent(I) AND present(K) AND notInOrbit(4000, K, G, B) AND (inOrbit(1000, L) OR inOrbit(1000, G, A))*.” This graphical representation allows the user to easily see the hierarchical structure within a logical expression when both conjunctions and disjunctions are used. Moreover, the user can modify the structure of a feature by changing the location of nodes through a simple drag-and-drop. Being able to modify and test different features is important in order to quickly explore the feature space and gather information.

Search in Feature Space

While the user can explore the feature space by modifying and testing individual features, we also implement a local search method to speed up the exploration process. The local search extends a given feature by adding an additional base feature either using a conjunction (AND) or a disjunction (OR). The possible set of base features is set by the user during the problem formulation step, and its size is limited to a small number. Therefore, the system can test the addition of all possible base features, and return the new set of features that improve one of the goodness metrics.

To run the local search, the user has to select a feature from Fig. 3 by clicking on it. Then the user can choose to use either a conjunction or a disjunction in linking the new base feature to the selected feature. When a conjunction is used, the feature becomes more specific (the feature covers fewer designs), most likely leading to an increase in $conf(F \rightarrow S)$. On the other hand, if a disjunction is used instead, the feature becomes more general (the feature covers more designs), thus increasing $conf(S \rightarrow F)$. The newly generated features are compared with the existing set

of features and only the non-dominated ones are added to the visualization. This provides a quick and easy way for the user to explore the feature space effectively, advancing the Pareto front of the feature space.

Evaluation

To test the efficacy of exploring the feature space as a way to improve the user’s learning, we conduct a controlled experiment with human participants.

Hypothesis and Experiment Conditions

The aim of the experiment is to examine whether exploring the feature space improves learning, compared to when the user interacts only with the design space. Learning is defined here as learning the mapping between design decisions and objective values. Therefore, we set our hypothesis as the following:

- H1: Exploring the feature space improves a designer’s ability to predict the performance of a design.

To test this hypothesis, we use a within-subject experiment design and compare the learning in two different conditions: design space versus feature space exploration. The capabilities of the tool in these two conditions are summarized in Table 2.

In the first condition, called the design space exploration condition, we provide only the parts in the graphical user interface that are related to the design space. For example, the user can inspect each design shown in the design space (see Fig. 2) and observe the values of design decisions and objectives. The user can also modify each design by adding/deleting/moving instruments through drag-and-drop. After modifying the design, the new design can be evaluated to get the corresponding

Table 2 The capabilities provided in each condition

Capabilities	Design space exploration	Feature space exploration
Inspect designs	✓	✓
Modify and evaluate designs	✓	
Local search in the design space	✓	
Run data mining and inspect features		✓
Modify and evaluate features		✓
Local search in the feature space		✓

objective values. In addition, a local search in design space has been implemented to mimic the local search in feature space. The local search is done by randomly sampling four neighboring designs from the currently selected design, evaluating them, and displaying the newly added designs to the scatter plot. A neighboring design is defined as a design that can be reached by changing a single design decision from the currently selected design.

The second condition is called the feature space exploration condition. Here, the user is still able to inspect individual designs in the design space. However, other interactions in the design space (evaluate new designs, local search) are not allowed. Instead, the user can run data mining to obtain an initial set of features visualized in a similar manner to Fig. 3. Modifying, evaluating, and inspecting each feature is also enabled through the interface shown in Fig. 4. Moreover, the user can run a local search to quickly explore the feature space.

The conditions are designed to make the types of interactions as similar as possible in both conditions. The user can modify, evaluate, and inspect designs/features, and run local searches in the respective spaces.

Experiment Protocol

Participants are first provided with an interactive tutorial that explains the design problem as well as all the capabilities of the tool. The tutorial is designed to take around 20–30 min to finish. After the tutorial, each participant is given two different tasks within the same system architecting problem described above (architecting a constellation of climate-monitoring satellites). The two tasks differ in the set of capabilities provided (two experimental conditions). For each task, the participant is asked to find and take notes of the features that would be useful to identify whether an arbitrary design will be in the target region or not. The tasks are designed to be representative of a designer's effort to find patterns within a group of designs. Different target regions are specified and given to the user to investigate in each task. The two treatment conditions are presented in a random order, and a 10 min time limit is applied to each task to control how much time each participant spends in learning.

After each 10 min session, the participants are given a short quiz to measure how much they have learned during the interaction. For each question, a figure similar to Fig. 1 is given, and the user is asked to predict whether a given design will be located inside the target region or not. A total of 25 YES/NO questions are given.

Participants

We recruited 38 participants, all of whom are university students. The study was approved by the Cornell University Institutional Review Board. Written informed consent was obtained from all participants. The average age of the participants is 23.0,

Table 3 Descriptives: all subjects. The mean score shows the percentage of questions answered correctly out of 25 questions in each test

	<i>N</i>	Mean	SD	SE
Design space exploration	38	72.95	10.86	1.762
Feature space exploration	38	74.95	11.22	1.821

with a standard deviation of 4.05. There were 21 male participants and 17 female participants. 26 students identified themselves as majoring in the STEM field, and 12 students identified themselves as having majors other than STEM.

The recruitment was done through two different channels. First, we recruited from the general student population on campus and offered \$15 Amazon gift cards as compensation. 23 participants were recruited using this method.

Second, we recruited students who were taking a graduate-level course on Systems Architecture. These students were offered a small amount of extra credit for the class as compensation. The reason for recruiting from this second group of students was our previous experience running a pilot experiment and with other experiments with similar interfaces. We have observed in the past that participants who had not been exposed before to some basic concepts in design space exploration—such as design decisions, objectives, features, recall, and precision—often struggled to understand the task they were asked to perform and did not utilize all the capabilities of the tool that was provided. In addition to our main hypothesis, we also wanted to test if the participants' formal training in some of the important concepts has any interaction effect with their performance in each condition. 15 participants were recruited from the class.

Result

The test scores of all participants are summarized in Table 3. The average scores shown in the table represents the percentage of questions that were answered correctly out of 25 questions asked in each problem set. It shows that the average scores for both conditions are effectively the same. Running a paired samples one-tailed *t*-test gives a *p*-value of 0.209.

A more interesting result is observed when the participants are grouped based on whether they had the formal training (a first-year graduate course on system architecture) or not. We ran two-way repeated measures ANOVA to compare the difference in the mean scores of the two conditions while also considering the effect of the formal training of the subjects. Table 4 shows the within-subject effects, and Table 5 shows the between-subject effects. The result shows that there is no statistical significance when we only consider either the experiment condition or the formal

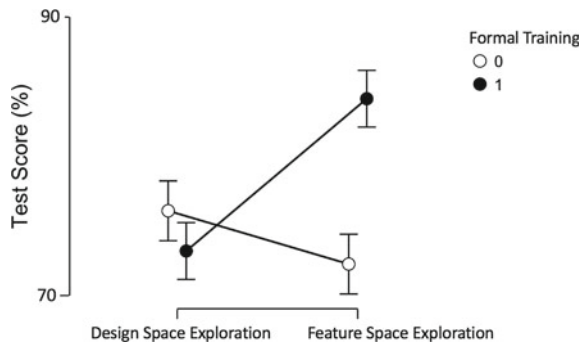
Table 4 Within-subject effects

	Sum of squares	df	Mean square	<i>F</i>	<i>P</i>
Exploration strategy	14.33	1	14.331	2.580	0.117
Exploration strategy * formal training	61.81	1	61.805	11.128	0.002
Residual	199.94	36	5.554		

Table 5 Between-subject effects

	Sum of squares	df	Mean square	<i>F</i>	<i>P</i>
Formal training	22.91	1	22.907	2.950	0.094
Residual	279.58	36	7.766		

Fig. 6 The test scores for each exploration strategy, factored by whether participants received formal training in system architecture design. The error bar shows the standard error



training (taking the System Architecture class or not) separately. However, there is a significant interaction effect between the two factors (the *p*-value is 0.002).

Figure 6 shows the average test scores after the participants have been divided into two groups (received formal training or not). These two groups of participants exhibit opposite trends in the scores. Those who have not received any formal training scored better in the quiz (one-tailed paired samples *t*-test: $t = 1.261, p = 0.890$) when they explored the design space ($M = 74.09, SD = 12.41$) than when they explored the feature space ($M = 70.26, SD = 11.11$). On the other hand, those who received formal training performed better (one-tailed paired samples *t*-test: $t = 3.759, p < 0.001$), when they explored the feature space ($M = 82.13, SD = 6.906$), compared to when they explored the design space ($M = 71.20, SD = 8.029$).

Discussion

From the experiment, we find that there is an interaction effect between the exploration strategy and the formal training. While the participants who had no formal training performed equivalently in both tasks, the participants who received formal training performed significantly better in the feature space exploration condition than in the design space exploration condition. This suggests that those who had previously been exposed to the basic concepts in engineering design and tradespace analysis found the feature space exploration more useful. A possible explanation for such observation is that the tool to explore the feature space is less intuitive and difficult to learn. Exploring the feature space requires reasoning at a higher level of abstraction, as it deals with what groups of designs have in common, rather than individual designs. Moreover, it requires understanding how features are represented (as shown in Fig. 5) as well as the basic concepts of interestingness measures in binary classification (e.g., *precision* and *recall*). While each subject is given a 30 min tutorial prior to the actual tasks, he or she may not be able to grasp all the concepts needed to make full use of all capabilities. This is also reflected in the qualitative feedback that we obtained from the participants after each session. Many participants reported that they had difficulty in understanding how to effectively use the capabilities to explore the feature space. Most of the participants thought that manually inspecting each design was more helpful.

While some of the participants who received formal training also made similar reports, others thought that exploring the feature space was more practical and useful in answering the questions in the quiz. It is possible that having received formal training helped them to better understand the tool. In the Systems Architecture class, the lectures cover a wide range of topics related to tradespace analysis including decision space, objective space, Pareto dominance, driving features, and sensitivity analysis among others. While this does not ensure a student's understanding of these subjects, we can assume that they have been exposed to, and thus familiar with, these topics.

The current experiment result supports our hypothesis that exploring the feature space improves learning, with a condition that the user has to be familiar with the key concepts of design space exploration and have been trained to reason in an abstract space. We believe that this is a promising result since the proposed method is mainly intended for professional engineering designers and systems engineers who are familiar with design space exploration.

Conclusion

This paper introduced a new concept in design space exploration, namely, that of exploring the feature space, where the feature space is defined as a set of possible features (combinations of values for various design decisions). The feature

space is visualized in a 2D plane, with each axis representing $\text{conf}(S \rightarrow F)$ and $\text{conf}(F \rightarrow S)$ —two measures that are equivalent to *recall* and *precision*, respectively. The designer can explore the feature space by modifying and testing different features and receiving immediate feedback on how the values of the goodness of features change in response. Such interaction provides a chance to learn and compare how well different features explain a selected region of the design space. This is in contrast to the conventional ways of presenting and selecting features in data mining, where the mined features are usually sorted by a single goodness metric and only a handful of them are inspected by the user. By inspecting the feature space, the designer can easily identify the major features that drive the performance of design and how well they explain the data.

The result from a controlled human subject experiment showed that the participants who received formal training in the key concepts of design space exploration performed better when they had a chance to explore the feature space, as opposed to when they explored only in the design space. This shows that feature space exploration has the potential to enhance designer's learning about the important features, which is reflected in their ability to predict the behavior of a design. For the purpose of this study, feature space exploration was tested separately from design space exploration. However, it is designed as a supplementary tool that helps the engineering designers to learn and gain new insights about what features constitute good designs. The designers can then leverage this knowledge to explore the design space more effectively.

A limitation in the result presented in this paper is that only the participants who received formal training performed better under the feature space exploration condition. While we obtained unstructured qualitative feedback after each experiment, how their previous exposure to design space exploration influenced the result is not clear. This will need to be investigated further in the future with a larger sample size and ways to measure how effectively each participant used feature space exploration.

There also exist other limitations with the current method to explore the feature space. The local search method to create and test new features is very simple and intuitive, but at the same time greedy and prone to overfitting. Using the local search, the users can easily generate features that have very high confidence metrics $\text{conf}(S \rightarrow F)$ and $\text{conf}(F \rightarrow S)$, but they are often too complex (large number of literals in a complex nested structure of disjunctions and conjunctions). When a feature becomes too complex, it becomes very difficult to comprehend and learn any insights from it. To resolve this issue, the authors are currently investigating new ways to populate the feature space.

References

1. Balling R (1999) Design by shopping: a new paradigm? In: Proceedings of the third world congress of structural and multidisciplinary optimization (WCSMO-3), vol. 1, pp. 295–297
2. Hirschi NW, Frey DD (2002) Cognition and complexity: an experiment on the effect of coupling in parameter design. *Res Eng Des* 13:123–131

3. Flager F, Gerber DJ, Kallman B (2014) Measuring the impact of scale and coupling on solution quality for building design problems. *Des Stud* 35(2):180–199
4. Grogan PT, de Weck OL (2016) Collaboration and complexity: an experiment on the effect of multi-actor coupled design. *Res Eng Des* 27(3):221–235
5. Eddy J, Lewis KE (2002) Visualization of multidimensional design and optimization using cloud visualization. In: *Proceedings of DETC'02*, pp 899–908
6. Stump GM, Yukish M, Simpson TW, Harris EN (2003) Design space visualization and its application to a design by shopping paradigm. In: *Proceedings of DETC'03*
7. Obayashi S, Sasaki D (2003) Visualization and data mining of Pareto solutions using self-organizing map. *Evol Multi-Criterion Optim LNCS* 2632:796–809
8. (Luke) Zhang X, Simpson T, Frecker M, Lesieutre G (2012) Supporting knowledge exploration and discovery in multi-dimensional data with interactive multiscale visualisation. *J Eng Des* 23(1):23–47
9. German BJ, Feigh KM, Daskilewicz M (2013) An experimental study of continuous and discrete visualization paradigms for interactive trade space exploration. *J Comput Inf Sci Eng* 13:1–12
10. Knerr N, Selva D (2016) Cityplot: visualization of high-dimensional design spaces with multiple criteria. *J Mech Des* 138(9):91403
11. Diaz MJ, Fullmer DD, Briceno SI, Mavris DN (2017) A collaborative approach to complex systems engineering using a web-based visual analytics framework. In: *55th AIAA aerospace sciences meeting*
12. Kudo F, Yoshikawa T (2012) Knowledge extraction in multi-objective optimization problem based on visualization of Pareto solutions. In: *2012 IEEE congress on evolutionary computation, CEC 2012*
13. Chen W, Chazan J, Fuge M (2016) How designs differ: non-linear embeddings illuminate intrinsic design complexity. In: *Proceedings of the ASME 2016 international design engineering technical conferences and computers and information in engineering conference IDETC/CIE 2016*, pp 1–12
14. Agrawal G, Bloebaum CL, Lewis KE (2005) Intuitive design selection using visualized n-dimensional pareto frontier. In: *46th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference, AIAA 2005–1813*, pp 1–14
15. Walker DJ, Everson RM, Fieldsend JE (2013) Visualizing mutually nondominating solution sets in many-objective optimization. *IEEE Trans Evol Comput* 17(2):165–184
16. Peng W, Ward MO, Rundensteiner EA (2004) Clutter reduction in multi-dimensional data visualization using dimension reordering. In: *Proceedings of IEEE symposium on information visualization, INFO VIS*, pp 89–96
17. Agrawal R, Imielinski T, Swami A (1993) Mining association rules between sets of items in large databases. *ACM SIGMOD Rec* 22(2):207–216
18. Quinlan JR (1993) *C4. 5: programs for machine learning*. Elsevier
19. Cervone G, Franzese P, Keese APK (2010) Algorithm quasi-optimal (AQ) learning. *Wiley Interdiscip Rev Comput Stat* 2(2):218–236
20. Santos JCA, Nassif H, Page D, Muggleton SH, Sternberg MJE (2012) Automated identification of protein-ligand interaction features using inductive logic programming: a hexose binding case study. *BMC Bioinform* 13(1):162
21. Bang H, Selva D (2016) iFEED: interactive feature extraction for engineering design. In: *Proceedings of the ASME 2016 international design engineering technical conferences and computers and information in engineering conference IDETC/CIE 2016*, pp 1–11
22. Watanabe S, Chiba Y, Kanazaki M (2014) A proposal on analysis support system based on association rule analysis for non-dominated solutions. In: *Proceedings of the 2014 IEEE congress on evolutionary computation, CEC 2014*, pp 880–887
23. Bandaru S, Ng AHC, Deb K (2017) Data mining methods for knowledge discovery in multi-objective optimization: part a—survey. *Expert Syst Appl* 70:139–159
24. Brachman R, Levesque H (2004) *Knowledge representation and reasoning*. Morgan Kaufmann

25. Feldman J (2010) The simplicity principle concept learning in human. *Psychol Sci* 12(6):227–232
26. Li WLW, Han JHJ, Pei JPJ (2001) CMAR: accurate and efficient classification based on multiple\nclass-association rules. In: Proceedings of the 2001 IEEE international conference data mining, pp 369–376
27. Liu B, Hsu W, Ma Y, Ma B (1998) Integrating classification and association rule mining. In: Knowledge discovery and data mining, pp 80–86
28. Tan P-N, Kumar V, Srivastava J (2004) Selecting the right objective measure for association analysis. *Inf Syst* 29(4):293–313
29. Sokolova M, Lapalme G (2009) A systematic analysis of performance measures for classification tasks. *Inf Process Manag* 45(4):427–437
30. Buckland M, Gey F (1994) The relationship between recall and precision. *J Am Soc Inf Sci* 45(1):12–19
31. Selva D (2014) Knowledge-intensive global optimization of Earth observing system architectures: a climate-centric case study. *SPIE Remote Sens* 9241:92411S–92411S
32. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules. In: Proceedings of the 20th VLDB '94 conference on very large data bases, vol 1215, pp 487–499

Redefining Supports: Extending Mass Customization with Digital Tools for Collaborative Residential Design



Tian Tian Lo, Basem Mohamed and Marc Aurel Schnabel

Fluctuating economies and changing family demographics have increasingly complicated the spatial requirements of contemporary housing. Advancements in digital design and communication technologies enable housing companies globally to move toward mass customization in response to market demands. This paper explores mass customization within the context of high-rise housing, where the shift lags behind new approaches in detached house design. We propose a comprehensive system based on the analysis of current housing trends and industry applications. A collaborative design model is then advanced for a collective design approach. Derived from game design concepts and implemented through a web-based application, our model enables the intuitive operation and visualization of design outcomes. Existing systems focus largely on individual houses; others fail to provide for collaborative design among users. This paper illustrates the importance of communication between users and architects within the system and demonstrates system integration by means of a computational method to enable efficient customization of high-rise housing.

Introduction

In urban centres, population growth has transformed housing into a high-rise mass housing typology in which individual families live in apartment units. To accommodate increasing demand, apartment blocks have been simplified to mass-produced

T. T. Lo (✉)
Harbin Institute of Technology (SZ), Shenzhen, China
e-mail: skyduo@gmail.com

B. Mohamed
Abu Dhabi University, Abu Dhabi, United Arab Emirates

M. A. Schnabel
Victoria University of Wellington, Wellington, New Zealand

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_12

'containers' for shelter and living. Such models, however, lack individuality; a crucial quality for successful housing developments. Individuality is commonly held to encompass varying factors of value and need; factors that differ widely among sociodemographic groups. These can have a remarkable impact on spatial requirements. According to recent reports by the Organization for Economic Co-operation and Development [1], household structure in modern society is changing and the typical 'two parents, two children' family becoming less common. This results in changing patterns of family living, education, work, entertainment and technology.

With globalization, variations in lifestyle and household structure are seen most clearly in developed cities. These variations result in mass housing designs ineffective for multi-faceted social needs; 'forcing' people to live in identical units designed and built on concepts of mass production for efficiency and affordability. Today, as a result of new socio-economic realities, homebuyers are becoming more selective in their demand for change. The 'one model fits all' approach seems to have run its course [2]. In response, a new strategy providing resilience for high-rise housing is necessary. While the viability of mass production techniques fluctuated in other economic sectors over time, in the 1970s a sharp increase in demand for personalized goods and products contributed to the call for a new production model. Termed 'mass customization' by Stanley Davis in his 1987 book 'Future Perfect', this process was formally systematized by Joseph Pine in 1993. Pine defined mass customization as the production of individual, customized goods and services. It relates to the ability to provide customized products or services through flexible processes, in high volumes and at reasonably low cost [3]. The process of mass customization is a multi-faceted one which encompasses various aspects, from managerial to technical. This production strategy aims to provide customers with individualized products, with near mass-production efficiency [4].

Similarly, a lack of variation and individual personalization in housing initiated a new participatory paradigm in design and production, aiming to allow homebuyers' input into the design of their homes. This has been realized in the work of many architects such as Le Corbusier, Walter Gropius, Frank Lloyd Wright, Buckminster Fuller and Jean Prouvé. In the 1960s, architect, theorist and educator John Habraken was working in the Stichting Architecten Research (SAR) (Foundation of Architects Research) group in the Netherlands. He developed the 'support' theory to involve communities in decision-making. The 'support' is a structure that is designed and built, similar to the 'housing system', but without non-load bearing elements. The community here comprises the homebuyers who will decide how elements are placed within the 'support'.

This paper represents an attempt to redefine 'supports' with the help of digital tools, towards adopting mass customization. We propose a computer-based participatory design model as a key solution for accommodating homebuyers' needs (Fig. 1).

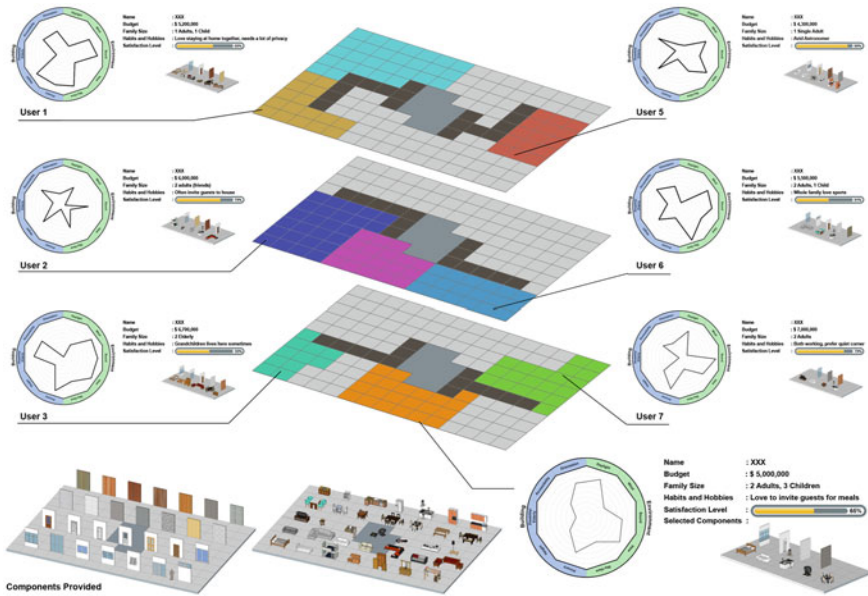


Fig. 1 Design collaboration between users with simple profile details and satisfaction score

Participatory Design in Housing

Enhancing user participation in design is not new and has historically produced limited success for industry. The notion of offering choices to homebuyers dates back to the 1960s when dwellings mass-produced during the post-World War II era incited architects to reflect on the traditional delivery methods of their designs. Seeking to include buyers in shaping their dwellings, architects experimented with participatory strategies and tools such as sketches, drawings and physical models. One such example is Frei Otto's 'Ökohaus'. Designed in 1978 for the International Building Exhibition (IBA) in Berlin, it represents a successful attempt to bring neighbours together to build their desired living space. As the medium of communication was mainly sketches and physical models, the design process itself took 2 years, despite being a small project for only a few households. For high-rise housing to adopt a higher level of participation with homebuyers, we need to reconsider the open building concept. Open building is an approach for building design promoted by Habraken and recognized internationally during the 1960s as constituting a new wave in the architectural field. The support system anatomy is based on dividing a building into three levels of decision-making: the tissue, the support, and the infill. They are separate, yet interdependent. The 'support' here is the physical, rigid part of the building, the structure and infrastructure users agree not to change. The 'infill' is the flexible part, adjustable on social, industrial, economic and organizational levels.

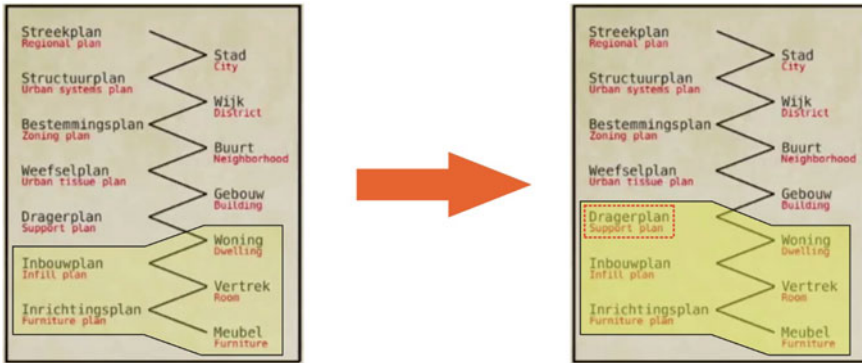


Fig. 2 Extending Habraken's [5] degree of user participation to the 'support' level

Habraken's level of user participation (Fig. 2) can be extended to evolve the 'support', so homebuyers can determine the location, the size, and the volume of their living space. This extended flexibility is necessary for mass customization to be efficient. The next stage is to provide the medium for increased participation.

Customization in Architecture: New Tools and Techniques

A unique architectural structure is often erected through the assembly of variously configured components. Most building projects fabricate elements on-site, directly processing materials such as concrete and masonry. More recently, the extensive application of CAD/CAM techniques within the building industry has resulted in comprehensive 3D building models, increasing efficiency in both design and construction processes. These same techniques further improve productivity by assisting with the design and production of off-site fabricated components which are later integrated with on-site activities.

Parallel to off-site fabrication, specialized manufacturers create customized products and components. Kieran and Timberlake [2] argue that mass customization has increasingly influenced construction processes and components over the past few decades. Most contemporary production approaches that employ specific digital design environments and manufacturing processes relate to the concept of mass customization, although this influence is at times subtle.

A major challenge of mass customization strategies in architecture lies in evaluating the efficacy of a product. It must be concurrently customizable, properly designed, in concordance with design codes and regulations, and accurately manufactured. Consumer products are usually modularized in a way that partially limits customization for technical pragmatism. However, architecture is unique in its interlinked structure of responsibilities. In the design, production and verification processes of

creating a building, there is usually no single party with the necessary specialization in all areas to manage the project. Accordingly, realizing a mass customization design and fabrication environment requires a high level of communication between users, designers and manufacturers. Fragmentation poses a major obstacle, as fabricators in the building industry generally consist of small to mid-size companies whose production volumes are insufficient for generating the economy-of-scale effects of modularized production in a typical mass customization model [2]. These aspects are derived from general theories and approaches to mass customization and situate the user, designer, and manufacturer in a complementary relationship via direct or indirect communication. Buildings, however, are products whose design involves typological, cultural and social aspects that are yet to be thoroughly considered in the customization process.

One example of an existing participatory design system is the ‘Barcode Housing System’ [6]. Developed a few years ago, the system allows prospective occupants to adjust their plan layout according to their needs. Architects collate the designs, stack them, respectively, and design a façade that consolidates the whole. However, the level of participation and options provided remain limited.

Digital technologies are abundant, yet their common application to customization within the housing industry is providing homebuyers only with a choice of unit layout, finishing and systems [7]. One of the primary challenges with a design system for participation in mass customization is the contrast between the simplified design parameters required for homebuyers with no architectural background, and the complexity essential for architects, who need a rich set of profession-specific details and data to ensure a buildable and successful project delivery.

The System

To enhance communication between architects and clients, (system X), a collaborative design platform, is developed. The technical aspects and functions of (system X) are explained in detail by the author et al. [8]. It is designed to enable the end users of a mass housing development to effectively communicate their needs and desires to one another, and to the architects, in the initial design phase. The process is supported by setting rules and parameters that are crucial to end users. This is inspired by simulation-game designs to provide an ‘easy-to-learn’ design process for a bottom-up collaborative approach. Using Java-Script-based code, WebGL, the proposed design tool can work to generate a wide variety of design options for individual occupants as well as to negotiate conflicting interests and outcomes. The system can be simplified into three main modules: the 3D design module, the data management module, and the real-time communication module. The 3D design module takes STL files as input then converts the data into a triangulated geometric model. The model is then manipulated behind the scenes by architects using a digital open structure that allows them to define design constraints and set the level of design freedom. Physical and environmental parameters are then mapped into the geometry accordingly and

act as a ‘scoring’ system for users during the design process. The data, including user profiles, are stored in a data management module that uses ‘redis’, an open source, in-memory data structure store. Node.js is another open source code, one capable of providing a scalable network for the communication module. This system uses Node.js to facilitate real-time design communication. For physical communication, simple text-chatting remains the best form so is implemented in the user interface. Using WebGL to incorporate the three modules is an ideal solution for generating 3D geometry in a web-based, collaborative interface. It helps to increase the speed of communication, essential for decision-making during the design process. It is a model that allows architects to cooperate closely with the potential inhabitants of a mass housing development, reacting to future inhabitants’ needs and desires. The setup, therefore, is quite different from a typical design process.

Figures 3 and 4 present the proposed collaborative design workflow that divides mass housing design into five major components: building form, structure frame, skin modules, inner partition system and utility cores. These subdivisions allow for design flexibility and for architects to have control of every design aspect. Using a Building Information Modelling (BIM) methodology, components are linked, rather than separated, to correlate and maintain information throughout the process. The five components offer selective control of information which can be published as open source for others to use or contribute to (system X) goes through four stages, each in collaboration with different stakeholders:

Stage One

The initial stage sees architects cooperate with developers to devise the building form, basing the targeted Gross Floor Area (GFA) and number of units on the demographics of the urban context. The difference here is that the architects only envision possible outcomes and prepare the necessary framework for the participatory design

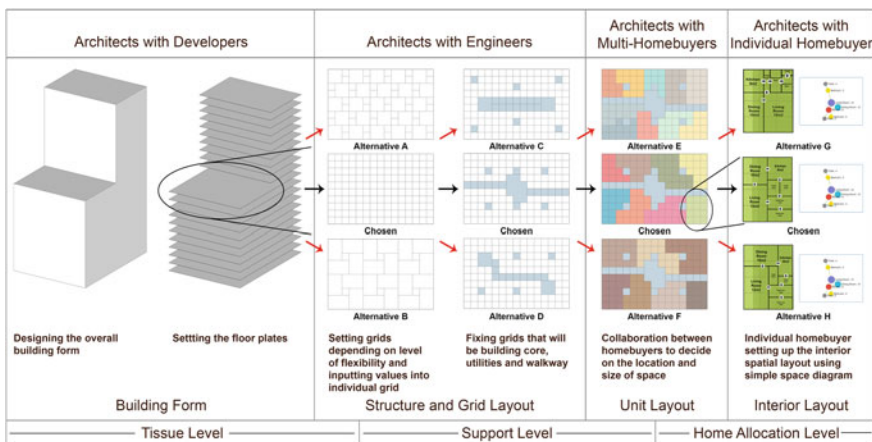


Fig. 3 Design process in the design system with flexible alternatives

model, where homebuyers become vital players in the process. Concurrently, environmental data such as building indoor daylight, average wind and surrounding noise are collected through digital simulation and site survey and serve as input for the next stage. In stage one architects also source potential suppliers who can provide mass-customized building components for homebuyers to choose from, and for construction.

Stage Two

Here, architects work closely with engineers to prepare the design framework of the apartment block. ‘Gridding’ the plans is a strategy employed to streamline the collaboration process. The architects can grid the layout per site geometry (it need not be a square grid if the architects are designing a unique housing plan) but the grid design must relate to the mass-customized components. The framework devised, the architects set the parameters of the grid, giving each a range of values useful to homebuyers throughout the customization process. For example, the most apparent parameter is the cost of each module within the macro grid. The architects can also set additional parameters (Fig. 4) such as daylight, sky-view, privacy, and views, based on the previously-collected data. In addition, some grid modules can be constrained to a fixed value, unavailable for selection by users. These are primarily communal blocks such as the building core, circulatory systems, utilities and public spaces where sole control belongs to the architects.

Stage Three

The framework prepared, the architects now release the model for homebuyers’ access. Homebuyers set up a personal profile outlining their needs and desires. They can then see the grid plans of the building within its surrounding site environment. This is a crucial point in the decision-making process, enabling homebuyers to better

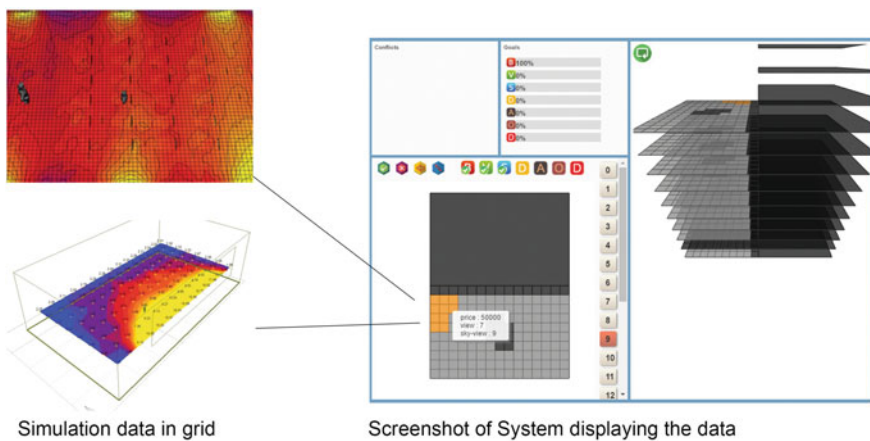


Fig. 4 Gridded simulation data acts as input for the system

understand the building quality. They then select the optimal space to suit their needs. The process is not totally open; there are system constraints which must be adhered to, for example, those spaces that must be adjacent to a fixed ‘utility’ grid, the ‘walkway’ grid or to exposed edges of the building. This ensures access to water and electricity, links to the walkway, and sufficient daylight. (system X) provides a ‘satisfaction’ indicator that shows homebuyers the extent to which their selection meets their needs. Given the anticipated conflicts within this collaboration process, (system X) alerts homebuyers to any clashes they may have with other homebuyers. Negotiations are then necessary. A resolution of the conflict can be negotiated by referring to the profile and satisfaction level of the homebuyers. If unresolved, the architect joins the process, acting as ‘judge’ to facilitate a successful solution. To encourage more collaboration between homebuyers, certain forms of remuneration such as ‘discount per square area’ or ‘greater advantage in being chosen as occupants’ form part of the overall system parameters. This process continues until all homebuyers are decided, and their satisfaction levels acceptable. The building need not be completely filled; some space can be reserved for future family expansion, or for public amenities.

Stage Four

With the space allocated, the next stage is for homebuyers to design their living space. They set up their spatial layout in a simple layout diagram and (system X) generates plan options that fit the diagram. Once a plan is selected, homebuyers can insert the customizable components prepared in stage one to ‘build up’ the whole space (Fig. 5). The components are sized so homebuyers can simply ‘drag and drop’ in (system X). The grids are designed to accommodate the components and (system X) accommodates their placement. For example, a 1.2 m door will not fit a 2 m grid, so (system X) fills the gap with ‘wall’ components that match the material and colour of the adjacent ‘wall’ component. While individual homebuyers are setting up the design, collaboration with adjacent homebuyers is necessary. This ensures sufficient privacy, the optimal use of space, and the harmony of adjacent building components.

Once every homebuyer is satisfied architects reclaim the lead role, working with other stakeholders towards finalizing design and preparing for construction. One option is transferring the building information to a BIM platform such as Revit to develop a comprehensive 3D model of the block, generating the details and quantities of components to be provided by various suppliers. Breaking down the building into



Fig. 5 Gridded simulation data act as input for the system

various levels and sub-levels provides a greater level of flexibility and integrating such techniques with a digital tool like (system X) enhances the level of participation. The next section demonstrates an example result and provides a deeper analysis of the decision support that this form of digital tool provides for mass customization.

The Level of Mass Customization

Introducing the choice of spaces and components to homebuyers at an early stage of the design process can greatly increase the level of customization. Within the consumer goods market, firms achieve the highest degree of customization, pure customization (Fig. 6), when allowing customers to have a direct impact early in the design process [9]. (system X), therefore, endeavours to engage homebuyers at an early stage of the design process and offers a collaborative environment with architects to enable spatial and component customization.

Customized housing can be structured on different levels, depending primarily on a series of factors: housing typology, enabling technologies, and interaction systems. Currently, common applications of customization within the housing industry are limited to layout selection and the internal and external appearance of a housing unit. In contrast, the proposed (system X) attempts to push boundaries further, enabling the homebuyer’s participation at the level of layout design. (system X) tackles the internal capabilities of the housing developer to implement a mass customization model through the application of computational design, combined advanced fabrication, and information management techniques.

Figure 7 illustrates a key difference between the customization of individual houses and that of high-rise buildings, namely, the external/internal finishes. Exterior façades are commonly designed wholly by architects to give a building a unique ‘design’ in itself. To push the limits of mass customization and provide a unique outcome for every homebuyer, we propose the exterior be included at the beginning design stage. By doing so, the cost of customization is reduced and a higher level of customization achieved. Since the whole design process is open and transparent, each homebuyer can view the design of another. If one homebuyer prefers the mate-

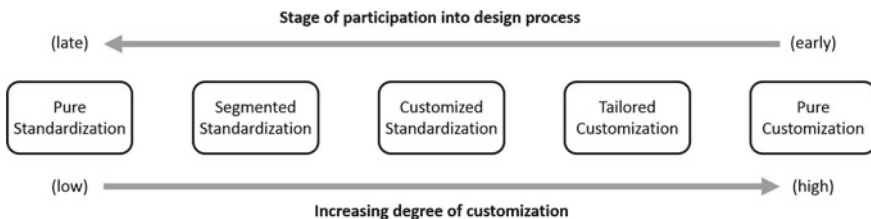


Fig. 6 Level of customization

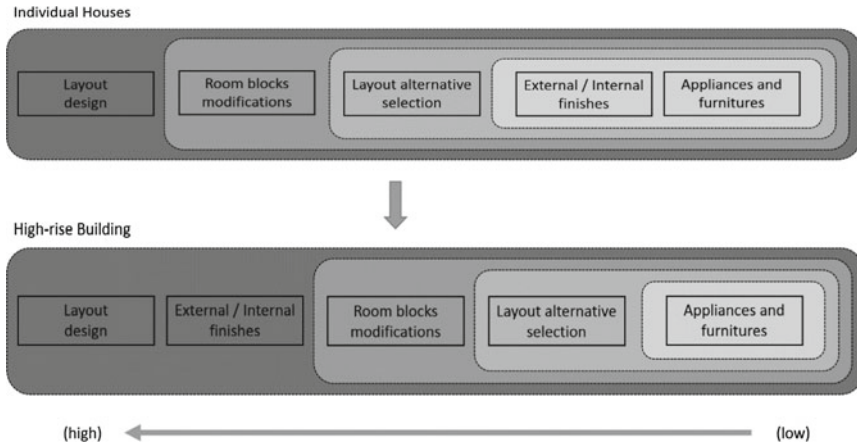


Fig. 7 Changing levels of customization in high-rise buildings

rials and finishes of another, they can agree on a design together and propose that to architects, who then help to negotiate and lower the cost of the customization.

Many homebuyers may find the customization process complex, so a decision support system also called an advisory system, operates in an interactive manner to guide homebuyers in their decision-making. Figure 8 shows the overall methodology of the design process adapted from Friedman's [10] flexible housing. The key participatory process is implemented digitally to handle the large amount of data and interaction of the high-rise context. With the ability to create individual profiles and an interface that simplifies the design process, layouts can be designed directly by homebuyers with the help of decision support imbedded in the system. The decision support is based on gamification to provide a user-friendly environment for homebuyers [8]. Such a process redefines the relationship between homebuyer, architect, and builder by repositioning the role of the architect within the customization process. Instead of designing apartment blocks collectively, the architect designs a system of coherent and partially interchangeable modules and component prototypes. Additionally, the architect is responsible for the configuration logic of the customization system. In this way, all modifications follow a theoretically pre-conceived scenario, overcoming potential design or technical challenges.

Simulation

We ran a simulation to validate the system and examine the intensity of mass customization in design. The results of the process are presented in (Fig. 9). The simulation is based on preparing a building framework for a site in Hong Kong, and a few initial housing types are decided based on its demographics. As the design process

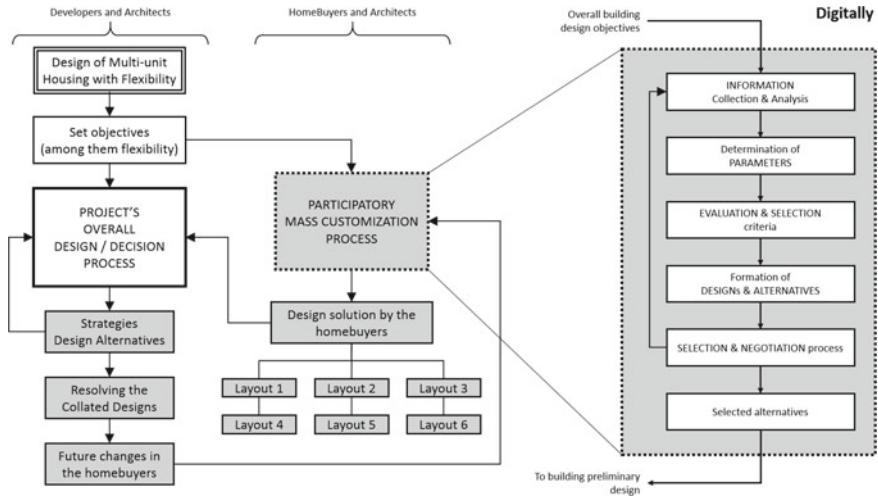


Fig. 8 The methodology of digital intervention in housing design process as decision support

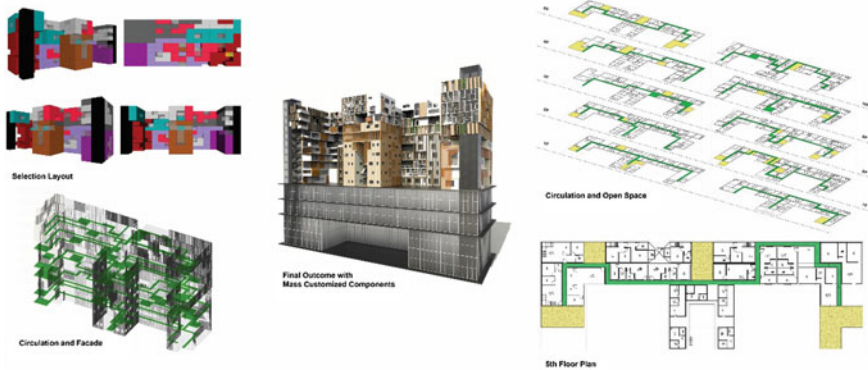


Fig. 9 A design outcome generated through a pilot study

was for study only, the ‘homebuyers’ are designers who are designing collaboratively based on family profiles collected from the area. The designers act as the ‘families’ and go through the design process to test it.

With the framework well prepared, the spatial layout was easily achieved. Variations appear as floor plans differ, yet structural integrity remains intact. There is a strong tendency to fill up every space, so constant directions are given to ensure ‘open’ spaces, avoiding an overly packed building and enabling shared activities. The interior design was also easily developed with only minor comments about the position of utility space grids not being ‘friendly’ and complicating the designs.

The most challenging part was the exterior finishes. To examine the engagement of the ‘homebuyers’ in mass customization, restraint was kept to a minimum. The work

to source materials and the design itself were done by homebuyers. They were given two criteria; the materials must be timber or wood, and the designs must be about horizontality or verticality. The 'homebuyers' could share resources and endeavour to minimize the budget as much as possible. Although the time taken to source materials and design the exterior was substantial, the process began when design participation started. Therefore, by the time participants have sourced materials; they have settled the design layout and reached the exterior design stage. Extra collaboration is required to consolidate the various designs.

One major advantage of such 'open' design is saving the architects from sourcing materials unsuited to the 'homebuyers'. 'Homebuyers' who were uninterested in sourcing materials settled for what others agreed upon. This reduced the number of variations considered at the design stage. At the same time, the cost of the construction is determined by the 'homebuyers' as they search for suppliers. Where design falls short of the architects' standards, they can inform the 'homebuyers' and propose appropriate alternatives to suit their intended outcome.

The outcome was successful, to a certain extent. The participatory design process was achieved with the 'homebuyers' engaging throughout the entire decision-making process. The breakdown of architectural elements and giving decision-making to the 'homebuyers' provided an opportunity to develop design variations to an extent very challenging for an assigned architect to achieve. However, although the outcome was a 'functional' building, many building standards were not considered. Moreover, the participants in this study were designers, and more investigation is required to know if this approach would work for real homebuyers.

Conclusion

This paper considers the integration of top-down mechanisms with a designer's stylistic inputs, and bottom-up ecosystems with homebuyers' customization in high-density housing design. As long as the architectural profession exists, designs will inevitably embody an architect's own stylistic preferences. And bottom-up customization in housing design cannot succeed without global design curation and management. Housing products are unlike fashion products, which can be customized as an independent entity. Housing is instead a collection of individual units based on negotiation between associative customizations. Our research finds that high-density housing design can be an on-going and transparent evolutionary process, rather than profit-driven and controlled by minorities. Based on parametric control and automation, open source design can be an iterative process with real-time feedback, generating more desirable housing products to suit individual needs, and resulting in open-ended design systems that move quickly from learned lessons to product solutions.

Future Potentials

At this time, our work posits potential for a higher level of participatory, bottom-up design processes by using the concept of ‘support’ in mass customization. Our system uses the digital tools of gamification models [11] to break down the residential high-rise into various components and sub-components. However, the process lacks the efficiency and fast response required to be relevant for industry. We believe that generative design systems can be implemented in the system to partially automate the design process and assist homebuyers in the decision-making process by matching their profiles to building parameters. This would save a significant amount of time and provide homebuyers with a starting point for engaging with design. Although the design process works well for new homebuyers, there are many complexities to consider when homebuyers decide to move away. Questions such as ‘what will happen to the space when the homebuyer moves?’ are still being investigated. However, methodologies such as the beginning of a housing typology will provide more opportunities for future changes and adaptations. Finally, Virtual Reality (VR) might be integrated for homebuyers to be fully immersed into the design process, further enhancing the participatory process. Better visualization can provide users with a better sense of space and volume.

References

1. OECD (2016) Family indicators (edition 2016), OECD social and welfare statistics (database). <http://dx.doi.org/10.1787/a6b32120-en>. Accessed on 20 Apr 2017
2. Kieran S, Timberlake J (2004) Refabricating architecture: how manufacturing methodologies are poised to transform building construction. McGraw Hill, NY
3. Pine B (1993) Mass customization: the new frontier in business competition. Harvard Business School Press, Boston
4. Blecker T, Friedrich G (2006) Mass customization: challenges and solutions. Springer, New York
5. Habraken J, Boekholt JT, Thyssen AP, Dinjens PJM (1976) Variations: the systematic design of supports. MIT Press, US
6. Madrazo L, Sicilia A, González MG, Cojo AM (2009) Barcode housing system: integrating floor plan layout generation processes within an open and collaborative system to design and build customized housing. In: CAADFutures, pp 656–670
7. Friedman A, Sprecher A, Mohamed BE (2013) A computer-based system for mass customization of prefabricated housing. *Open house Int* 36(1):20–30
8. Lo TT, Schnabel MA, Gao Y (2015) ModRule, a user-centric mass housing design platform. In: CAADFutures, pp 236–254
9. Silveira GD, Borenstein D, Fogliatto FS (2001) Mass customization: literature review and research directions. *Int J Prod Econ*:1–13
10. Friedman A (2011) Decision making for flexibility in housing. The Urban International Press, UK
11. Lo TT, Schnabel MA, Moleta T (2017) Gamification for user-oriented housing design: a theoretical. *Review CAADRIA*:63–73

Voxel Synthesis for Generative Design



Matvey Khokhlov, Immanuel Koh and Jeffrey Huang

In the ever-evolving world of computer rendering technologies, the demand for more original and diverse 3D models is ever-increasing, to the point where the man-hours required to create these models for video games, movies and architectural designs are now counted in hundreds, if not thousands. This paper proposes a new way to generate relatively large voxel models from smaller example voxel models given by the user. Based on the concept of Voxel Synthesis, this paper proposes two different models that use volumetric data of voxel models to extract implicit patterns and then recombine those patterns in new and unexpected ways. Without the need for any explicit rules or grammars, the results maintain both the formal and structural consistency of the input model. While the first results are small in scale, they show promise of real application with further optimizations.

Significance

As computer rendering technology becomes more powerful and affordable than it was 10 or even 5 years ago, the demand for three-dimensional models of relatively high fidelity also rises. The use of high fidelity models contributes to an increase in the perception of realism—a quality highly sought-after by both the movie industry and the surging virtual reality industry, particularly in the area of special effects and computer-generated imagery. The biggest challenge of using high fidelity models is the amount of time it takes to create them manually. Larger movies, video games and architecture projects would require extensive man-hours to create all the necessary three-dimensional models.

M. Khokhlov · I. Koh (✉) · J. Huang
EPFL, Lausanne, Switzerland
e-mail: immanuel.koh@epfl.ch

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_13

For the 2005 movie *King Kong*, Weta Digital was tasked with recreating a scene of New York in 1933. Instead of having artists manually model the entire scene (itself an impossible task given the time constraint), they developed a software package called ‘CityBot’¹ to procedurally generate approximately 90,000 unique and detailed models of buildings. The artists only had to recreate the important landmarks by hand, while the software would simply auto-generate the rest.

In the field of video games design, the idea of procedural generation is not new. *Beneath Apple Manor*² (released in 1978) first used procedural generation for automatic game-level design. However, it was *Rogue*³ (released in 1980) that later popularized this concept. Despite its usefulness, especially in the automatic creation of game-level layouts and terrain generation, it is less used for the generation of highly detailed urban environments. Today, the continual increase in hardware graphics power and the recent surge of games in the ‘open world’ genre have led game development studios to spend more time on asset generation. Resulting in a situation where large Western game developers have to outsource game asset creation to other studios in order to remain financially viable.

Within architectural design practice and research, the digital modelling of complex architectural details is equally time consuming. Although CAD/CAM-based parametric modelling tools have been increasingly adopted within the industry to counter this issue, setting up these parametric models with explicit geometrical rules and numerical parameters has remained, by and large, either too difficult for most architects to implement or not easily reusable for other projects. Architects learn the art and design of architecture from architectural precedents. It is thus not uncommon for them to re-appropriate or recombine exemplar conceptual, structural or stylistic motifs in their designs. This might seem particularly evident in some of today’s established architecture offices, such as Sou Fujimoto Architects and Kengo Kuma & Associates, as well as, an early master architect like Gerrit Rietveld. As seen in Fig. 1, recurring structural timber joint designs used by these 3 architects were digitally re-appropriated here as small user-input exemplar voxel models. Using our proposed voxel synthesis algorithm, we have demonstrated the possibility of generating new and larger recombinant architectural assemblies that significantly retain the original coherence of their respective input models.

Model Synthesis, proposed by Merrell [1], is an interesting approach to the procedural generation of three-dimensional models based on concepts originally found in the field of texture synthesis. The key concept is to have a relatively simple 3D model serve as an input to the algorithm, which generates a larger model of higher complexity while staying similar to the original input. This approach has the potential to ease and accelerate asset generation in video game development, whereby diverse models

¹<https://www.wetafx.co.nz/research-and-tech/technology/city-bot/>.

²<http://worth.bol.ucla.edu/>.

³https://archive.org/details/msdos_Rogue_1983.

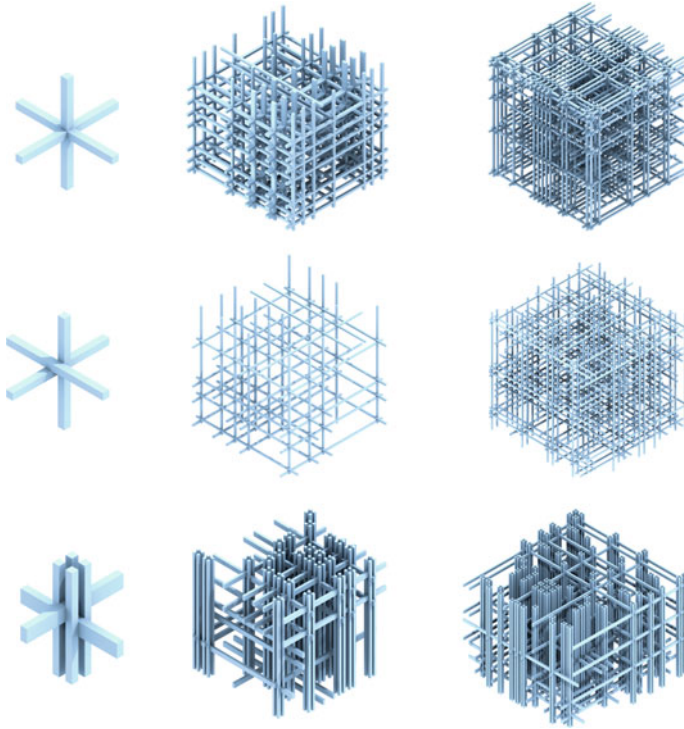


Fig. 1 Results of using small voxel models based on three different architectural construction details in generating coherent and larger output models with our proposed simple model. (Top row) Sou Fujimoto’s 2013 London’s *Serpentine Gallery Pavilion*. (Middle row) Gerrit Rietveld’s *De Stijl* period furniture. (Bottom row) Kengo Kuma’s 2006 *Yusuhara Town Hall*

could be generated from simpler input models hand-crafted by a human designer. In fact, the *Wave Function Collapse* algorithm created by *ExUtumno* [2]⁴ builds on this very concept of *Model Synthesis* to synthesize 2D textures and has the potential of being extended to three dimensions.

Aim

We want to present a proof-of-concept—a completely automated system based on the model synthesis of three-dimensional inputs (specifically voxel model inputs), which only requires a single-input model from the user, and nothing else.

⁴<https://github.com/mxgmn>.

Related Work and Background

Procedural Generation and Modelling

Procedural modelling is an active and ongoing research area. The earliest techniques involved using fractal geometry to create natural landscapes [3]. L-systems, among others, is a popular technique for creating realistic looking plants and foliage [4, 5].

Many procedural techniques have been developed for the modelling of urban environments. One of the most famous techniques for procedural architectural design is *shape grammars*, first conceived by Stiny [6]. Wonka et al. subsequently introduced the concept of split grammars that split larger shapes into smaller components and can thus generate more detailed architecture models [7].

Funkhauser and others have developed a system in which a user can choose and cut parts of an already existing 3D model, then query a database to find matching pieces of other already existing 3D models, to finally compose new models [8]. While it fixes and improves the creation process by having a machine suggest which object could be mixed and matched, it does not really create completely new shapes, which is one of the goals of this paper.

A newer approach consists of leveraging volumetric convolutional networks and generative adversarial nets to generate three-dimensional objects from a probabilistic space [9]. It also goes beyond simply rearranging different parts from the already existing model and instead taps into the probabilistic latent space of a volumetric object to generate new objects without supervision. In a similar way, our paper uses volumetric data in the form of voxels—a data format unlike typical polygonal meshes used in 3D modelling environments.

Texture Synthesis

Since the algorithm discussed in this paper shares similar features with texture synthesis algorithms, it is worthwhile to briefly discuss some of the ideas found in texture synthesis. Texture synthesis techniques have been used for decades to generate large digital images from small image sample by reconstructing its structural content. The seminal paper by Efros and Leung [10] laid the foundation for most of the texture synthesis techniques used today. Since then, a few techniques have been developed to accelerate and improve the efficiency of texture synthesis algorithms, including one which splits the image into optimal patches for further reassembling [11].

Although texture synthesis has already been extended to 3D for procedural modelling [12], it works mostly for the regular patterns and is better suited to decorate already existing models. A detailed explanation of the difference between texture and model synthesis is provided in [1, Sect. 2.3].

Model Synthesis

Example-based model synthesis introduced in [13] follows the same concepts as texture synthesis algorithms, but applies them to three-dimensional space. By giving a small example model as an input, the model synthesis algorithm generates a large model that resembles the input model. By giving an appropriate model as input, model synthesis can be a powerful tool capable of generating large, consistent models. One of the main drawbacks of the discrete model synthesis is that the user needs to manually divide his model into usable model pieces, label them correctly and define the appropriate neighbour pieces in the six possible directions (up, down, left, right, forwards and backwards) for each model piece. One of the primary motivations of this paper is to see if it is possible to circumvent the need for manually created labels and discover if we could potentially infer the neighbour placement rules from the structure of the input model.

Continuous model synthesis algorithm [14] uses the information about the connectivity between the adjacent boundary features of the input model and creates a larger output model based on that information. This removes the need for manual separation and assignment of labels and neighbour rules to the model pieces. The drawback of this algorithm is that the continuous version of model synthesis is best suited for polyhedral structures, but not curved or highly tessellated models. However, this version of the algorithm also removes the need for aligning the input model on a grid, thus removing the constraints on what type of model we can serve as input.

Wave Function Collapse

The Wave Function Collapse (WFC) [2] is a new development in the field of texture synthesis. It builds on the approach of model synthesis by dividing the model into smaller pieces and propagating the constraints to construct a coherent model. However, inspired by the uncertainty principle found in quantum physics, it is able to produce larger texture from a smaller input while respecting *local similarity* and consistency. The idea is to divide the input texture into $N \times N$ unique patterns, where N is the number of pixels per dimension. Each unique pattern is assigned a unique label. The user provides the output texture size and the algorithm starts with all output pixels in their *unobserved states*—that is each pixel value is a superposition of all colours from the input texture and has the potentiality of collapsing into any of the assigned unique patterns. Then the algorithm enters the observation–propagation cycle:

1. During the observation step, an unobserved $N \times N$ pattern with the lowest entropy is *collapsed* into a definite state (i.e. one label from the list of available labels is assigned to this pattern).



Fig. 2 Overlapping model. On the left, we can see a small input bitmap which is used to generate a larger output bitmap on the right

2. During the propagation step, the label constraints that the observation step has imposed on the neighbouring unobserved patterns are propagated throughout the entire output.

The algorithm is completed when the output is in a completely observed state (all the pixels in the output have exactly one label assigned to each of them).

By *local similarity*, the author implies that:

1. Each $N \times N$ pattern that appears in the output should occur at least once in the input.
2. Over a sufficiently large number of outputs, the distribution of the $N \times N$ patterns in the output should approximate the distribution of the patterns in the input texture. That is the probability density of each $N \times N$ pattern in the output should approximate the probability density of each pattern in the input.

Similar to discrete model synthesis, the algorithm may end up in a contradictory state if a pixel cannot be assigned a label after the constraint propagation. Likewise, finding out if a certain texture allows a non-failing solution is NP -hard. It is thus impossible to have a solution that never reaches contradiction, unless $P = NP$. The proof can be found in [1].

The author of WFC proposes two different models for his algorithm.

Overlapping Model

The overlapping model is a model in which the user gives a small example bitmap as input and divides it into $N \times N$ overlapping patterns. It then constructs an adjacency matrix by trying to overlap each possible pair of unique patterns and deciding which pairs can form compatible neighbours. Since the adjacency rules are inferred from the structure of the texture, the user does not need to provide any input aside from the example texture. The creation of the new texture is thus completely automated at the cost of a loss of precise control over the pattern placement by the user. See Fig. 2 for an example of the result.

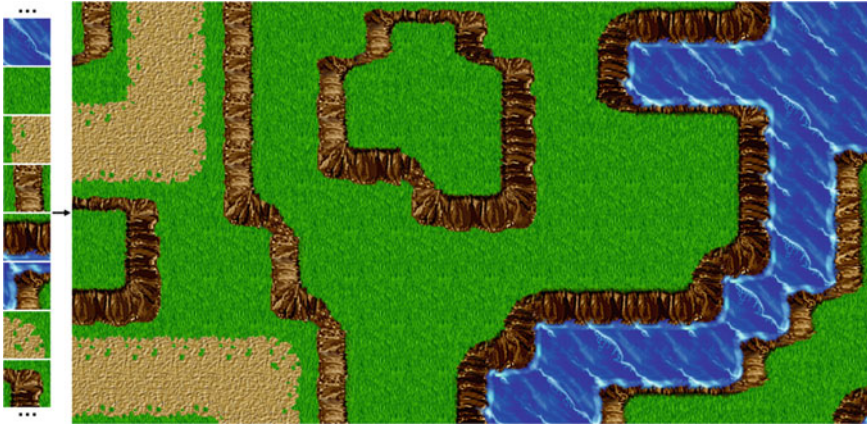


Fig. 3 Simple tiled model. The input tiles are displayed on the left, which combined with the user-defined rules to produce a coherent and larger tilemap

Simple Tiled Model

The Simple Tiled model takes in an array of tiles of any size and a set of adjacency rules that define the set of allowed neighbours in the four possible orthogonal directions on a 2D plane. The only constraint is that all tiles in the array must be of the same size. This approach enables the user to create large consistent tilemap textures from smaller tiles and can be especially useful for video game development, such as the popular *Pokemon* game which uses tilemaps for its game environments). The paper has implemented a user interface incorporating a special symmetry system to significantly reduce the number of manual rules assignments needed for each tile. See Fig. 3 for an example of the result.

Approach

The main advantage of using voxel models as inputs is that, due to the nature of voxels, the input model can provide us with information about its volume and internal structure, and not just the surfaces (as is the case with a polygonal mesh). This allows us to extract information that would not be possible with a polygonal mesh. It presents us with the possibility of inferring adjacency rules directly from the model's structure, instead of having to manually define them by the user, thus further automating the synthesis process.

Two different approaches were implemented and tested. The first approach is similar to the discrete model synthesis algorithm, with the main difference being that no user-defined rules are manually created, as the model would infer the rules

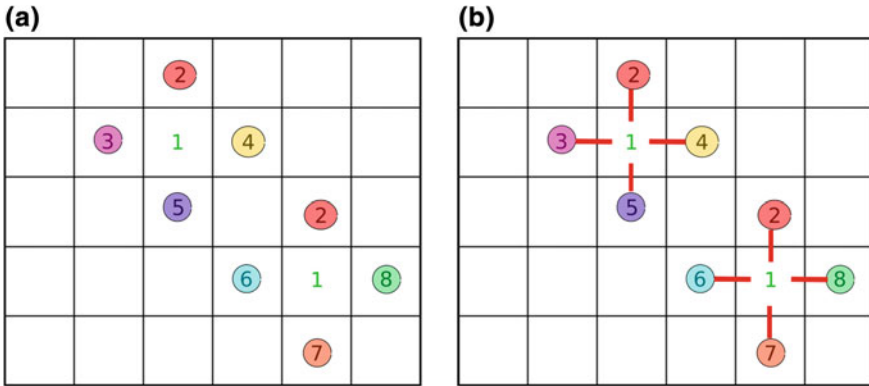


Fig. 4 Diagram showing how the direct adjacency rules are created for the pattern with the label 1 in 2D. This same principle applies to 3D space. After the initialization has finished, the adjacency map for the pattern with label 1 will be: up \rightarrow {2}, down \rightarrow {5, 7}, left \rightarrow {3, 6}, right \rightarrow {4, 8}. **a** Suppose we have several instances of the pattern with the label 1 in the input. **b** During the initialization, all its direct neighbours will be added to its adjacency map

from the input voxel model’s structure itself. The second model takes inspiration from the overlapping model of the WFC algorithm, but extends it to 3D by using three-dimensional convolutions to infer adjacency rules from the structure of the input example model. As we will see, both approaches have their advantages and disadvantages.

Simple Model

In many ways, the simple model resembles the discrete model synthesis algorithm and the simple tiled model from the WFC algorithm.

1. Initialization

The initialization procedure is as follows:

1. Divide the input model into equally sized $N \times N \times N$ patterns, where N is the number of voxels per dimension.
2. Assign a unique label to each unique tile. More specifically—if the same tile appears several times in the input, all the instances of this tiles will have the same label.
3. In order to construct the adjacency rules for each label, initialise an empty list for each of the 6 possible directions (left, right, up, down, forward, backward).
4. For each instance of the pattern in the input, add the neighbouring pattern labels to the list according to their respective directions, as shown in Fig. 4.

5. (Optional) To increase the number of possible neighbours in each direction, check for each possible pair of patterns if they ‘fit’ together in each of the six directions.
6. The output, whose size is given by the user and measured according to the number of patterns, is initialised as a three-dimensional array. Each array element is initialised with a list containing all the unique pattern labels present in the input. Each element in the array starts the algorithm in an *undefined state*, that is at first any element in the output can be any pattern found in the input.

2. Observation/Propagation cycle

The algorithm then enters the *observation and propagation* cycle, one which is similar to that of WFC algorithm. The *observation* begins by randomly choosing an *uncollapsed* element in the output (i.e. an element whose list of possible patterns consists of more than one element). Based on the probability distribution of the patterns in the input, exactly one label from the list of possible labels is chosen for that element and is followed by the removal of the remaining labels. We thus end up with an element in the output that is *collapsed*, which means that it has exactly one label attached to it. The *Propagate* procedure takes the coordinates of this newly collapsed element as the input and proceeds as follows:

1. Initialize the queue that will contain the 3D coordinates as *nodesToVisit* and initialize the output matrix as a three-dimensional array with a list of all the possible labels contained at each coordinate.
2. Enqueue the 3D coordinates that were given as parameters to the *nodesToVisit* queue.
3. While the *nodesToVisit* queue is not empty:
 - (a) Take the first element in the queue.
 - (b) For each of the six possible directions:
 - i. Add the direction vector to the current coordinates and label the new coordinates as *nodeToBeChanged*.
 - ii. Get all the allowed neighbour labels for each of the possible labels at the current coordinates and in the current direction, call them *allowedNeighbours*.
 - iii. Remove all the labels that are not present in the *allowedNeighbours* from the list of the available neighbour labels in the output matrix at the *nodeToBeChanged* coordinates.
 - iv. If at least one label has been removed from the list contained at the *nodeToBeChanged* coordinate and *nodeToBeChanged* is not already queued up in the *nodesToVisit* queue then enqueue it to the *nodesToVisit* queue, otherwise go to step 3.

The propagation of the adjacency constraints ensures that the global state of the output is consistent and that all the elements are placed next to their allowed neighbours.

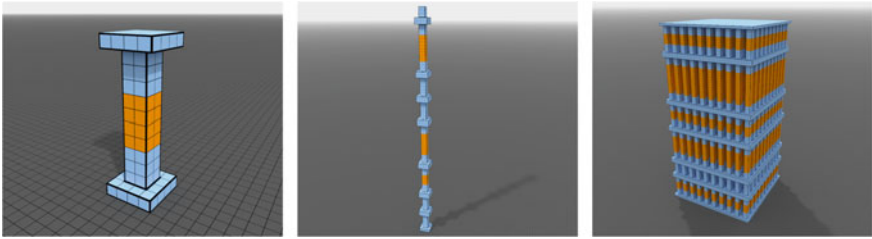


Fig. 5 Outputs produced by the simple model for a small column-like input model. The left-most image depicts the input model, the middle image show a $1 \times 20 \times 1$ output with $N = 4$ and the right-most image depicts a $10 \times 20 \times 10$ output with $N = 4$

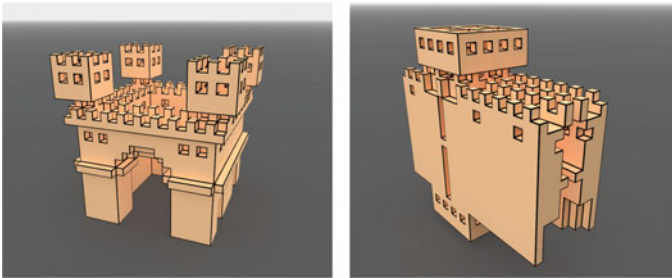


Fig. 6 Result of using a more complex castle-like input model (measuring $21 \times 21 \times 21$) to produce an output with $N = 5$

The biggest difference when compared to the discrete model synthesis and the simple tiled model in WFC is that the initialization procedure tries to infer the adjacency rules by simply ‘cutting up’ the input model into equally sized patterns and looking at which pattern is next to which. The problem with this approach is that a lot of the patterns that appear just once in the input end up with just one possible neighbour per direction. This can pose a higher probability of arriving at a contradiction during the algorithm’s propagation cycle, which is why the optional step 5 in the initialization procedure has been introduced. In case there are too many elements with just one possible neighbour, this optional step can help to increase the number of possible neighbours per direction if the two patterns can ‘fit’ together. This could serve as a simple way to increase the number of possible neighbours at a cost of $O(K^2)$ operations, where K is the number of unique patterns in the input model.

Results

From the resulting outputs shown in Figs. 5, 6 and 7, the simple model algorithm seems to work well on simple example inputs, such as the column and the bridge input.

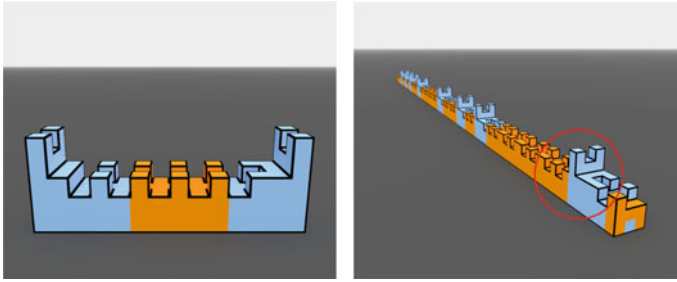


Fig. 7 A bridge-like input model (on the left) produces a relatively simple output that chains the $5 \times 5 \times 5$ input patterns. An inconsistency produced by the optional pattern fitting step is highlighted in red

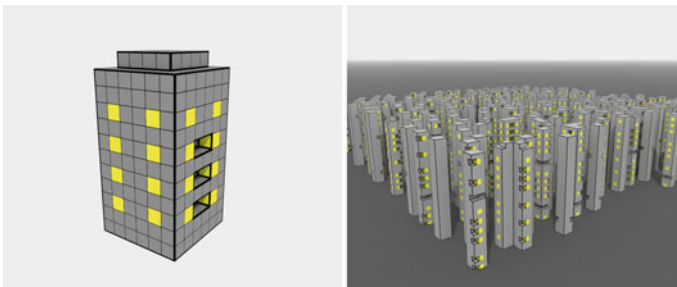


Fig. 8 A simple building input model can be used to generate an entire city-like model without any further user input

Figure 8 further suggests that the simple model is capable of producing more interesting results, given further adjustment by the user, that could be used as a backdrop to a virtual urban landscape where little detail is needed.

It should be noted that the choice of the N value plays a very significant role in the output produced. N should be dictated by the size of the input and the desired granularity of the output. If N is too small ($N = 1$), the algorithm will produce noise as output, since it will take every single voxel as a pattern. If N is too big, the entirety of the input model will be interpreted as a single pattern, resulting in exact copying in the output.

The optional step 5 in the initialization procedure helps to prevent contradictions in cases where there are too many unique patterns resulting in insufficient neighbour choices during the propagation step. The drawback, however, is the introduction of adjacencies that are not necessarily correct from the creator’s perspective. For example, from the larger bridge output in Fig. 7, one could identify few connections (highlighted in red) that do not really make sense to the creator. For some larger outputs, this discrepancy might be less noticeable. Nevertheless, proper verification would have to be made by testing and generating much larger outputs.

Limitations

During the execution of this model, we can arrive at a contradiction (i.e. when an element in the output has zero possible elements that can be assigned to it). The larger the size of the output model, the more probable it is for the algorithm to reach a contradicting state—as is the case with the discrete model synthesis algorithm as well. This makes it quite difficult to produce larger outputs in its current form. One possible solution to this issue could be via an optimisation that produces a larger input by dividing it into smaller blocks. This optimisation is further discussed in the ‘Future Work’ section.

Convolutional Model

The Convolutional model tries to fully exploit the fact that our model is made of voxels with the goal of not only mixing and matching the patterns that are present in the input model, but creating brand new structures while remaining stylistically true to the example input. This model takes inspiration from the overlapping model presented in the WFC and is an attempt to apply the same concept to three-dimensional space. Since voxels are basically pixels in 3D space, we can draw parallels between the usage of pixels in the overlapping model of the WFC algorithm and the usage of voxels in the convolutional model.

The input convolutional model, as well as, the desired output model can be defined as periodic (i.e. it can repeat itself over and over again) by the user.

1. Initialisation

During the initialization step, we again divide the input into $N \times N \times N$, but this time instead of taking solid patterns, we overlap all of them in order to have at most $K_x \times K_y \times K_z$ patterns for the periodic model and at most $(K_x - (N - 1)) \times (K_y - (N - 1)) \times (K_z - (N - 1))$ patterns for the non-periodic model, where K_x, K_y, K_z are the sizes of dimensions of the input model. The periodicity of the model is decided by the user and can be set as a flag before the execution.

Now instead of trying to find all the possible neighbour patterns in just six directions, we try and find all the patterns that can overlap with one another in a coherent way. Figure 9 shows several ways in which two $3 \times 3 \times 3$ patterns can overlap with one another. In order to find all the possible overlaps between two $N \times N \times N$ patterns h and g , we use three-dimensional convolution as defined by

$$y(i, j, k) = \sum_m \sum_n \sum_p f(i + m, j + n, k + p) \cdot h(i, j, k)$$

where

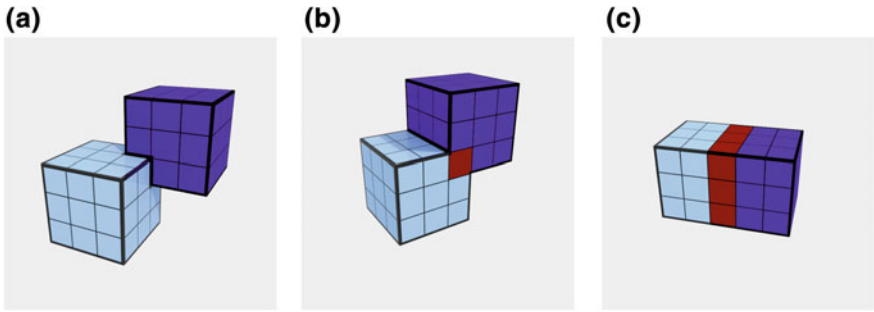


Fig. 9 Examples of different ways in which two $3 \times 3 \times 3$ cubes can overlap. **a** Just the corner, **b** whole edge and **c** or even the whole side

$$0 \leq mn, \quad p < N + (N - 1)$$

$$0 \leq i, j, k < N$$

and f is the kernel made of patterns h and g .

And we modify it by substituting the multiplication by a Boolean operation, which helps us check if the two patterns can fit together at point (i, j, k) , as described in the following procedure:

The *FitPatterns* procedure takes two patterns as input and consists of the following steps:

1. We construct a kernel from the first and the second patterns and label it i . We take the second pattern and label it h .
2. If for every possible value of the (x, y, z) coordinates between $(0, 0, 0)$ and $(N - 1, N - 1, N - 1)$ the value $i(x, y, z)$ is equal to $h(x, y, z)$, then return true, otherwise the procedure returns false.

Using this information, we can now construct a neighbourhood matrix for each unique pattern by comparing it to each pattern. Each value in the matrix is a list of possible neighbours in that direction. For example, $2 \times 2 \times 2$ pattern will have a neighbourhood matrix consisting of $3 \times 3 \times 3 = 27$ elements, where each element is a list of possible neighbours at that position.

The initialization procedure for the convolutional model is thus:

1. Divide the model into equally sized $N \times N \times N$ patterns that overlap.
2. Assign a unique label to each unique pattern.
3. Initialize the neighbourhood matrix for every unique pattern. (i.e. create an empty list of possible neighbours for each value).
4. For each possible pair of unique patterns check if they can be overlapped at each neighbour coordinate using the procedure described in the *FitPatterns* procedure. If they can be overlapped, add the pattern to the adjacency matrix.
5. Finally the output (the size of which is still given by the user, but is now measured in voxels instead of patterns) is initialized as a three-dimensional array. Each array

element is initialized as a list containing all the unique pattern labels present in the input.

2. Observation/Propagation cycle

The algorithm then starts the propagation cycle, much like in the simple model, except that this time we do not check the neighbours in just six directions, we check them for all the values of the adjacency matrix. The *Propagate* procedure takes the origin coordinate as input and is as follows:

1. Initialise *nodesToVisit* as an empty queue which will contain the 3D coordinates. Initialise the *outputMatrix* as a three-dimensional matrix with a list of all the possible labels contained at each coordinate.
2. Enqueue the 3D coordinates that were given as parameters to the *nodesToVisit* queue.
3. While the *nodesToVisit* queue is not empty:
 - (a) Take the first element in the queue and define its coordinates as the current coordinates.
 - (b) For each of the possible directions as defined by the adjacency matrix of the current label:
 - i. Add the direction vector to the current coordinates and label the new coordinates as *nodeToBeChanged*.
 - ii. Get all the allowed neighbour labels for each of the possible labels at the current coordinates and in the current direction, name this list *allowedNeighbours*.
 - iii. Remove all the labels that are not present in the *allowedNeighbours* from the list of the available neighbour labels in output matrix at the *nodeToBeChanged* coordinates.
 - iv. If at least one label has been removed from the list contained at the *nodeToBeChanged* coordinate and *nodeToBeChanged* is not already queued up in the *nodesToVisit* queue then queue it up in the *nodesToVisit* queue, otherwise go to step 3.

Results

The results produced by the convolutional model are shown in Figs. 10, 11 and 12. When using smaller and more abstract inputs, this model can create brand new variations from the original input patterns, giving rise to visually interesting new combinations. Unfortunately, with more structured and less abstract inputs, the coherence of the results seems to also reduce. This may be attributed to the fact that there are no explicit user-defined rules and that the adjacency rules are only inferred implicitly from the convolutions of the patterns. As a result, the overall coherence of the output model suffers directly, especially when N is small. A solution might be to apply

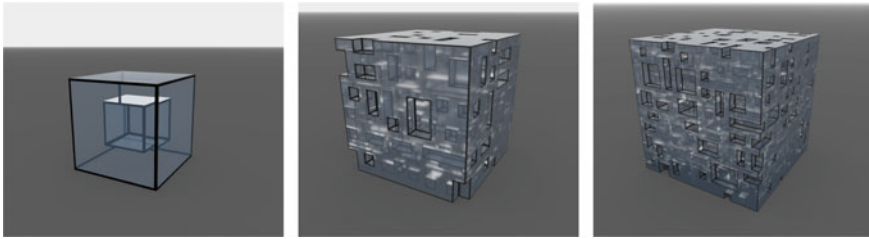


Fig. 10 Outputs produced by the convolutional model for the hollow cube input measuring $4 \times 4 \times 4$ voxels, throughout the outputs we can see the variations of the rectangular hole in the input

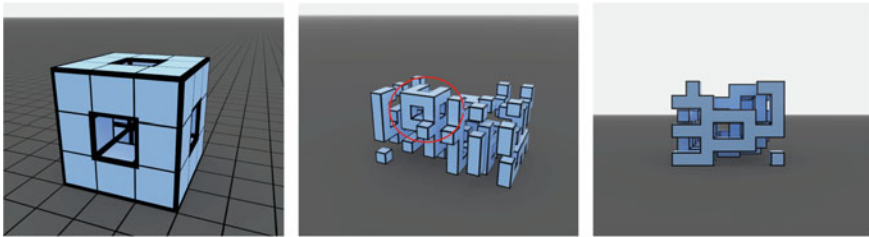


Fig. 11 Output produced by the convolutional model for the cube that is hollow in six directions. The model produces an output of size $12 \times 8 \times 6$ that contains patterns that we have not seen before. Highlighted in red is the original hollow cube modified in a new way

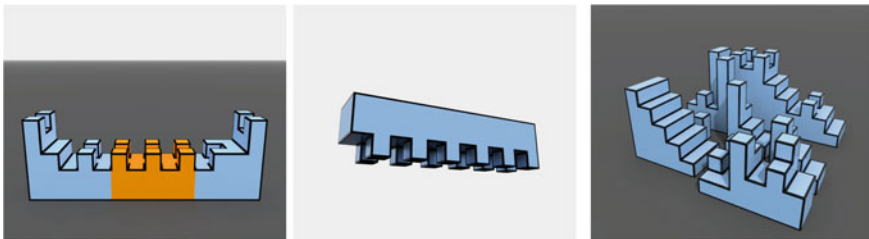


Fig. 12 A bridge-like input model (left image), with N set to 2, produces a simple and expected output (middle image). However, the convolutional model can also produce results (right image) that do not necessarily make much sense when the value of N is too low

additional heuristics and constraints. Again, unfortunately, with bigger values of N comes a dramatic increase in model generation time as well.

Similar to the simple model, the choice of the values for N is extremely important. Intuitively, this makes sense for the convolutional model. Since an $N \times N \times N$ captures volumetric information from the input model, the bigger the N the more information it will capture. However, the bigger the pattern, the more constraints will be applied on the adjacency of that pattern. Thus if N is too large, the pieces will be big, but not much variation would be possible, and consequently the possibility of creating new patterns is much reduced.

Limitations

One obvious limitation of the convolutional model is the slowness of the execution, which limits its practical use when applied to larger input/output models.

The complexity of the initialisation phase is

$$O(K^2(N + (N - 1))^3)$$

where K is the number of unique patterns and N is the number of voxels per dimension of a pattern.

The worst-case complexity of a single *observation and propagation* step can be approximated to

$$O(L(K, M_x, M_y, M_z) \cdot M_x \cdot M_y \cdot M_z \cdot (N + (N - 1))^3)$$

where

$$L(K, M_x, M_y, M_z) = K \cdot M_x \cdot M_y \cdot M_z$$

It would thus prove to be difficult to use the convolutional model in its current state on large example models. Further optimizations are definitely needed in order for this model to be practical for larger inputs.

Another drawback to this approach is that it still suffers from cases of contradiction. The same optimization proposed for the simple model could also be applied to the convolutional model in order to reduce the number of contradictions. This is discussed in the ‘Future Work’ section.

Conclusion

The main goals of this project were to apply the model synthesis concept to three-dimensional voxel models, without simply copying patterns found in the original input, but to generate new ones. Another goal was to automatize the process, whereby the user would only need to provide minimal initial input. Two different models were implemented in our attempt to accomplish these goals.

Our simple model resembles the discrete model synthesis [13] and the simple tiled model of the WFC algorithm [2], but further utilizes the inherent structural and volumetric information of voxel input model to automatically generate pattern adjacency rules. While it yields some interesting results with smaller example inputs, the overall quality of the generated 3D models outputs seems to suffer when compared to the results achieved with the discrete model synthesis algorithm [13]. The presented algorithm is particularly successful when using architectural components or joints as inputs, as seen from the generated outputs in Fig. 1. The works of Sou Fujimoto,

Kengo Kuma and Gerrit Rietveld are used as our exemplar voxel inputs. These $15 \times 15 \times 15$ input voxel models describe the fundamental structural motif used in their respective building designs, namely Fujimoto's 2013 London's *Serpentine Gallery Pavilion*, Kuma's 2006 *Yusuhara Town Hall* and Rietveld's famous *Cartesian node* also used in many of his *De Stijl* period furniture. The novel combinations seen in our results suggest that these simple rigid architectural modules can be generatively reconfigured using the simple model proposed here.

Our convolutional model further exploits the volumetric approach by using three-dimensional convolutions to define adjacency rules—a unique approach that would not be possible if typical polygonal meshes are used instead. The small-scale results are very promising and demonstrate its capability to create brand new pattern variations, without simply repeating patterns found in the original example input. Unfortunately, this method proves to be very computationally expensive in its current form, thus further optimizations would be needed in unlocking its full potential.

Future Work

One possible way of producing bigger and more complex outputs with the current algorithm is via the optimization described in [1, Sect. 3.3.6]. It consists of dividing the larger output into overlapping smaller sub-outputs. Each sub-output is to be computed individually using either the simple or the convolutional model. Since each sub-output overlaps with their respective neighbouring sub-outputs, the constraints could be propagated accordingly, thus preserving the cohesiveness of the final output.

As mentioned previously, additional heuristics can be introduced to gain further control over the convolutional model. Among these are density and height heuristics. For example, a simple heuristic of constraining all ground-touching input voxel patterns to only appear at the ground level in the output is highly plausible. In this way, a higher degree of constraint may give rise to a more structured output.

References

1. Merrell PC (2009) Model synthesis. Ph.D. thesis, University of North Carolina at Chapel Hill
2. ExUtumno (2016) Wavefunctioncollapse. <https://github.com/mxgmn/WaveFunctionCollapse>
3. Fournier A, Fussell D, Carpenter L (1982) Computer rendering of stochastic models. *Commun ACM* 25(6):371–384
4. Lindenmayer A (1968) Mathematical models for cellular interactions in development I. filaments with one-sided inputs. *J Theor Biol* 18(3):280–299
5. Prusinkiewicz P, Lindenmayer A, Hanan J (1988) Development models of herbaceous plants for computer imagery purposes. In: *ACM SIGGRAPH computer graphics*, vol 22. ACM, pp 141–150
6. Stiny G, Gips J (1971) Shape grammars and the generative specification of painting and sculpture. In: *IFIP Congress* (2), vol 2
7. Wonka P, Wimmer M, Sillion F, Ribarsky W (2003) Instant architecture, vol 22. ACM

8. Funkhouser T, Kazhdan M, Shilane P, Min P, Kiefer W, Tal A, Rusinkiewicz S, Dobkin D (2004) Modeling by example. In: *ACM transactions on graphics (TOG)*, vol 23. ACM, pp 652–663
9. Wu J, Zhang C, Xue T, Freeman B, Tenenbaum J (2016) Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: *Advances in neural information processing systems*, pp 82–90
10. Efros AA, Leung TK (1999) Texture synthesis by non-parametric sampling. In: *The proceedings of the seventh IEEE international conference on computer vision, 1999*, vol 2. IEEE, pp 1033–1038
11. Efros AA, Freeman WT (2001) Image quilting for texture synthesis and transfer. In: *Proceedings of the 28th annual conference on computer graphics and interactive techniques*. ACM, pp 341–346
12. Lagae A, Dumont O, Dutre P (2005) Geometry synthesis by example. In: *Shape modeling and applications, 2005 international conference*. IEEE, pp 174–183
13. Merrell P (2007) Example-based model synthesis. In: *Proceedings of the 2007 symposium on interactive 3D graphics and games*. ACM, pp 105–112
14. Merrell P, Manocha D (2008) Continuous model synthesis. In: *ACM transactions on graphics (TOG)*, vol 27. ACM, p 158

Part IV
Design Theory

Model-Based Abduction in Design



Lauri Koskela and Ehud Kroll

In prior literature, design abduction has been conceived in sentential (propositional) terms. The aim in this presentation is to explore the significance of internal mental models and images, and their external projections, in design abduction. Seminal and current literature on model-based reasoning in cognitive psychology and philosophy of science are reviewed. A retrospective case study on the invention of the airplane by the Wright brothers reveals that most occurrences of design abduction were model based. Conclusions and reflections flowing from the findings are presented.

Introduction

In prior research, the authors have endeavored to re-propose the conception of abduction in design [1, 2]. The background to this is that most research on abduction has been carried out in the framework of science. However, given the differences in context, abduction in design is argued to show characteristics not yet found or identified in science. For example, abduction can arguably occur in connection to practically all reasoning types in design, whereas in science abduction has been connected to regressive reasoning from effects to causes.

In this presentation, the new understanding of abduction in design is deepened through illumination provided by recent trends in cognitive science. Since the 1980s, the understanding of human reasoning as operating by means of mental models, through which the world is simulated, rather than only through formal rules of logical inference, has gained foothold in psychology [3]. As in discussions on reasoning,

L. Koskela (✉)
University of Huddersfield, Huddersfield, UK
e-mail: L.Koskela@hud.ac.uk

E. Kroll
ORT Braude College, Karmiel, Israel

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_14

in general, research on abduction has initially focused on logical inferences [4]. Peirce discussed abduction through syllogisms, logical sentences, and the subsequent literature has largely taken the same approach, called “sentential” by Magnani [5]. Magnani has seminally extended the discussion on abduction to models, especially in science, and hence the terms model-based abduction and model-based reasoning, which refer especially to construction and manipulation of visual representations, thought experiments, and analogical reasoning. Although there has been recent growth of work related to this topic [6], model-based reasoning in design has received little attention. Thus, the aim in this presentation is to explore the significance of mental models, and their external projections, in design abduction.

This article is structured as follows. First, the re-proposed understanding on design abduction is briefly recapitulated. Then, the new trend of seeing reasoning as model based in cognition research is presented. Given that contemporary authors have found Peirce as the seminal scholar for model-based cognition, it is appropriate to discuss his related thinking. Next, ideas on model-based abduction emerging in philosophy of science are examined, followed by a synthesis of these theoretical advances. How the current scholarly work on design treats models is then analyzed. Further, for exploring how model-based reasoning, especially abduction, occurs in design, a case study is reported. The paper is completed by conclusions.

Re-proposed Concept of Abduction in Design

The re-proposed concept of abduction in design [1] has been motivated by the observation that abduction in philosophy of science carries implicit contextual assumptions, which are not compatible with the context of design. According to the seminal views of Peirce, an abduction leads to a new idea, still hypothetical, by means of often subconscious, uncontrolled mental processes. Peirce examined abduction in the context of scientific discoveries, where it is triggered by an anomaly, such as a surprising observation. However, in design, abduction is triggered by a problem that the designer is not capable of solving through habitual or known solutions. The context of design is plainly different to that of science.

It is argued in [1] that given such differences of context, abduction in design has characteristics not found or at least discussed in science: Design abduction may occur in any part of the design process—not just in the beginning as in typical accounts on abduction in science. Abduction can occur in connection to practically all inference types in design—rather than just through regressive inferences as commonly assumed in science. Design abduction usually leads to an idea new in the context—rather than to entirely new ideas as in science. The primary criterion of an abducted insight in design is its utility—rather than its truth as in science.

Based on contextual differences between science and design as well as on empirical knowledge of different phenomena comprising design, the following types of abductive inferences in design were identified and discussed [1]: regressive abductive inference, abductive composition, manipulative abduction, abductive transformation,

abductive decomposition, abductive analogical reasoning, abductive invention of requirements, abduction for integration of theories, and theoretical model abduction.

There are interesting implications from this outcome for design theory and philosophy of science. The mental moves, which lead to new ideas in design, have for the first time been determined (although this list cannot be considered to be exhaustive). As abduction as a mental move is ubiquitous and generic, the hypothesis arises that the conception of abduction in science (covering only regressive inferences) has been too restrictive.

However, the re-proposed understanding of abduction in design also both inevitably encounters extant gaps in knowledge and reveals new disparities. The classical theory of abduction has been unable to explain from where the new idea emerges, at least if the somewhat apologetic argument that its origin is in intuition or subconsciousness is not accepted as satisfactory. Furthermore, the classical type of abduction, a regressive inference, has been seminally represented through a syllogism. This type of representation does not seem compatible with many newly defined types of design abduction. These two topics will be examined below.

Mental Models as Part of Reasoning in Cognitive Psychology

The understanding of human reasoning as based on mental models, through which the world is simulated, has recently acquired a strong position in psychology [3]. Up to the 1980s, the mainstream theory held that in reasoning, language-like representations of propositions are manipulated based on formal rules. Such rules are contained in formal logic, decision-making theory, or probability calculus [7]. The newer theory holds that based on linguistic representations of the meaning of propositions, mental models of the considered situation are constructed. Then, reasoning is based on these mental models. Since the 1980s, a number of variants of the model theory have been developed [7]. Here, the approach of the seminal advocate, Johnson-Laird, is adopted.

The theory of mental models is based on three assumptions [3]. First, a mental model is characterized as an internal model of a possibility [7]; it represents what is common to a distinct set of possibilities. Second, a mental model is iconic; it is structurally similar to what it represents: “A natural model of discourse has a structure that corresponds directly to the structure of the state of the affairs that the discourse describes” [3]. Third, mental models represent what is true; we usually construct models of what is possible and true, as opposed to what is not possible.

Do visual (internal) images fall into this kind of mental models as defined? Visual images are iconic [3], and they can underlie reasoning. However, Johnson-Laird contends that images may impede reasoning, and visual imagery is not necessary for reasoning. Visual imagery is thus not the same as building a mental model but there may be a close relation, as behind an internal image may lie a mental model [8].

According to Johnson-Laird [3], no clear distinction is drawn in reasoning among deduction, induction, and abduction—reasoning based on mental models “is more a simulation of the world fleshed out with all our relevant knowledge.” Further,

he forwards the ability to refute an inference through counterexamples as the heart of human rationality. External diagrams (or graphs) are closely related to mental models; they are often used to help reasoning. It is noteworthy that here, a diagram represents a model in the mind [9], rather than an external entity. Tversky [10–12] has interestingly focused on such projection of thought into the world. She argues [12] that when thought overwhelms the mind, the mind puts it into the world in diagrams or gestures. Thus, human actions organize space to convey abstractions; she calls this *spraction*. Accordingly [10]: “The designed world is a diagram.”

After the 1980s, the idea of model-based reasoning has received support from neighboring disciplines, often also in the form of questioning the novelty of it. Thus, Hintikka [13] has proposed that modern logic anyway has operated based on the idea of a model. Importantly, it has been pinpointed that already Peirce had discussed diagrammatic reasoning and the basic reasoning moves. Indeed, Johnson-Laird himself says [9]: “Mental models are similar to Peirce’s graphs.”

Logical and Cognitive Studies by Peirce

The American pragmatist philosopher Peirce developed highly original views on logic and cognition, which have recently found resonance in the circles of philosophers, logicians, and cognition researchers. Further, Peirce developed a notation called existential graphs [14], which, as mentioned above, is near the later idea of mental models.

He held that thinking is diagrammatic [15]: “I do not think I ever reflect in words: I employ visual diagrams...”. Further [16]:

We form in the imagination some sort of diagrammatic, that is, iconic, representation of the facts, as skeletonized as possible. The impression of the present writer is that with ordinary persons this is always a visual image, or mixed visual and muscular; but this is an opinion not founded on any systematic examination.

Peirce viewed reasoning to be iconic [9]. This refers to his division of signs into icons, indices, and symbols. An icon, such as a diagram, represents its object by likeness. Images and metaphors are other types of icon. An index represents its object by drawing the attention to the particular object meant without describing it. A symbol indicates its object by means of an association of ideas or habitual connections between the symbol and the object it stands for.

Peirce connected deduction to (intramental) diagrammatic reasoning [17]. Abduction, according to Peirce, is an inference through an icon. The following account of a compositional design problem clarifies the phrase “through an icon” (he focused on causal abduction in science and did not use the term abduction in connection to this typewriter example; however, all the hallmarks of design abduction are present) [18]:

Suppose I have long been puzzling over some problem, — say how to construct a really good typewriter. Now there are several ideas dimly in my mind from time, none of which

taken by itself has any particular analogy with my grand problem. But someday these ideas, all present in consciousness together but yet all very dim deep in the depths of subconscious thought, chance to get joined together in a particular way such that the combination does present a close analogy to my difficulty. That combination almost instantly flashes out into vividness. Now it cannot be contiguity; for the combination is altogether a new idea. It never occurred to me before, and consequently cannot be subject to any acquired habit. It must be, as it appears to be, its analogy, or resemblance in form, to the nodus of my problem which brings it into vividness. Now, what can that be but pure fundamental association by resemblance?

Thus, it seems he thinks that there is an icon, a mental model of the problem, and the subconscious mind retrieves a resembling icon, a model of a solution structurally corresponding to the problem model. He also comments on how this retrieval occurs: by resemblance (or similarity) rather than contiguity. By the term association by contiguity, he means that similar ideas are conjoined in experience until they become associated. By the term association by resemblance, he refers to the situation that “an idea calls up the idea of the set in which the mind’s occult virtue places it, and that conception perhaps gives, owing to some other circumstance, another of the particular ideas of the same set.” Peirce explains [19]:

Association by similarity is related to association by contiguity somewhat as our inward consciousness is related to outward experience; the one association is due to a connection in outward experience, the other to a connection in our feelings.

The importance of the distinction between association by contiguity and by similarity is that they, respectively, realize induction and abduction [20]:

The mode of suggestion by which, in abduction, the facts suggest the hypothesis is by resemblance, – the resemblance of the facts to the consequences of the hypothesis. The mode of suggestion by which in induction the hypothesis suggests the facts is by contiguity, – familiar knowledge that the conditions of the hypothesis can be realized in certain experimental ways.

Remarkably, what Peirce suggests here is an explanation on from where novel creative ideas emerge. These Peircean concepts have affinity to Gärdenfors’ [21] proposal on conceptual spaces as geometrical regions in the mind; indeed, Bruza et al. [22] have used this idea of conceptual spaces for modeling abduction.

Model-Based Reasoning and Abduction in Philosophy of Science

Magnani has seminally discussed model-based abduction, especially in the context of science and mathematics. He defines [5] model-based reasoning as the construction and manipulation of various kinds of representations, not necessarily sentential and/or formal. With the term model-based abduction, he refers to visual abduction but also abductions involving analogies, diagrams, thought experiments, and visual

imagery. In turn, according to Magnani, manipulative abduction is a kind of abduction, usually model based, that exploits external models; the strategy that organizes the manipulations is unknown a priori, and the results achieved are new and add properties to the concept. Thus, insights gained through geometrical constructions or sketching may be manipulative abductions.

He recognizes three types or roles of external representations (models), which help to provide abductive outcomes toward explanation or creation of novel concepts (in the latter case, the results are nonexplanatory as there is no preexisting concept or phenomenon to explain), namely [23],

- Mirror role (to externalize mental models),
- Unveiling role (to reveal imaginary entities), and
- Optical role (to see what otherwise would not be visible, due to smallness, largeness, or other obstacles).

Discussion on Findings from the Literature

Difficulties, Obstacles, and Pitfalls

The topic of model-based abduction provides for a multitude of difficulties, obstacles, and pitfalls, some of which deserve to be briefly discussed. The first difficulty is that the term abduction continues to be understood in different ways. An excellent example of this confusion is provided by the entry by Douven on abduction in the Stanford Encyclopedia of Philosophy [24]. Douven states the term “abduction” is used in two related but different senses. According to him, in the historically first sense, it refers to reasoning in generating hypotheses, while in the sense in which it is used most frequently in the modern literature, it refers to reasoning in justifying hypotheses. The entry is about abduction in the latter sense, also often called “Inference to the Best Explanation” (IBE). Unfortunately, Douven fails to note that the first sense is by no means historical; it suffices to pinpoint to the continuous stream of papers [25–28] trying to clarify the misunderstanding around abduction and to return to the Peircean understanding. Briefly, the argument is that the term abduction, originated by Peirce, has been appropriated, without good justification, from 1960s onward to the meaning of IBE.

The difference between these two understandings is, first, that the Peircean abduction is an account of generating explanatory hypotheses, while IBE represents an account of both generating and evaluating scientific hypotheses [27]. Second, a Peircean abduction is expected to produce a new idea. An abduction understood as IBE may produce both novel and already known hypotheses. In this presentation, abduction is understood in its original Peircean meaning.

Abduction is at the crossroads of logic, cognitive psychology, and many other disciplines, which do not necessarily communicate with each other. Here, the relation between abduction and creativity provides an example. Peirce defines abduction

as the only type of inference that is capable of creating a new idea [29]. The other characteristic of abduction, according to him, is *uberty*, referring to value in productiveness. Abundance and fruitfulness are dictionary meanings of the word. Now, the standard definition of creativity [30] is: *Creativity requires both originality and effectiveness*. Novelty and value are presented as possible synonyms for originality and effectiveness by these authors. Thus, abduction and creativity are closely related as they more or less share the same characteristics. However, they are not quite the same, as abduction produces a hypothesis, and regarding creativity, such a constraint is usually not required. As creativity is seen as the central characteristics of design, this connection deserves to be carefully explored.¹

Converging Outcomes

In spite of the difficulties mentioned, the considered streams of research show many converging, interesting results:

1. The distinction between sentential versus model-based reasoning is significant. The sources argue in a persuasive way that much of human reasoning takes place with support of internal models; the role of sentential reasoning remains somewhat unclear.
2. These internal models come in two main types, namely, as mental models, allowing for diagrammatic representations, or as visual images.
3. The human mind tends to project internal models externally, into sketches, diagrams, gestures, and physical models; such external models are then representations of the internal models.
4. Deduction may occur through model-based reasoning, either through a counterexample or by observing diagrams or other external models.
5. Abduction often seems to happen through model-based reasoning, for example, in the case of manipulative abduction or abduction based on analogy.
6. The underlying ideas on model-based abductions seem to be able to provide understanding on how novelty emerges as well as from where the new ideas come.

¹The concept of generativity, argued to be different from creativity, has recently been defined as the capacity to generate new propositions that are made of known building blocks but are still different from all previously known combinations of these building blocks [31]. Obviously, this refers to novelty. These authors add criteria, such as “impact of a new entity on the others,” which seems to be related to productiveness or effectiveness. Thus, the difference between generativity and creativity is not entirely clear. This issue is of significance as Hatchuel et al. [31] claim that generativity is an essential ontological property of design that provides it with a unique scientific identity.

Internal and External Models in Prior Design Theory

The role of models in design has generally been discussed in recent design theory [32, 33]. This is of course expected as the outcome of design is a representation of the targeted artifact. Design has also been characterized as progression through increasingly concrete models [33]. In such reviews, a number of instances of model-based reasoning have been mentioned; however, such discussions are scattered.

The duality of design reasoning is well acknowledged, for example, in the understanding of the differences between novice and expert designers [34], where the former tend to make a sequence of (sentential) inferences and the latter retrieve a model from their experience to start with. The most obvious example of model-based abduction may be drawing and sketching [35], which has attracted scholarly attention. However, otherwise little is known about model-based abduction in design.

One of the focal areas of design research is made up by comprehensive theories or frameworks, both descriptive and prescriptive, such as German systematic design [36], function–behavior–structure [37], and the C-K theory [38]. Their focus is on the total process of design, and understandably they have the tendency of largely abstracting away the cognitive aspects of design reasoning.² However, there are two proposed approaches to design that would seem to support or be compatible with model-based reasoning: parameter analysis and the proto-theory of design.

Parameter Analysis (PA)

PA [40] has been empirically developed by observing experienced designers. It defines two spaces, concept space and configuration space, between which there is an iteration consisting of three steps: parameter identification (PI), creative synthesis (CS), and evaluation (E). Additionally, such principles as steepest-first development, minimalistic configurations, and constant evaluation are followed. PA has been taught for over 25 years to mechanical engineering students, and it has recently been the subject of conceptual and theoretical research [41, 42].

In contrast to most other theories or models of designing, PA seems to focus on the cognitive aspects of design. The triplet PI–CS–E arguably describes how a designer selects a subproblem, forms intramental and extramental models of a solution, evaluates it and depending on the outcome, moves to the next subproblem or to another attempt at solution. Among the other principles, especially minimalistic configurations and constant evaluation are also closely related to model-based reasoning. It

²The C-K theory may provide an interesting exception. Ullah et al. [39] argue that the process of generating innovative concepts, as described by the C-K theory, cannot be explained by classical abduction. They suggest that it is a motivation-driven process, and that motivation comprises two facets, compelling reason and epistemic challenge, which help conceive a new concept when faced with lack of knowledge.

can be said that PA is centered around model-based reasoning by a designer. This focus probably originates from the empirical source of the approach.

Proto-theory of Design

Aristotle considered designing (and planning) to happen similarly to the method of analysis in geometry [43, 44]. Can this be conciliated with the model-based account of cognition?

As Hintikka and Remes [45] have argued, the method of analysis, developed in Antiquity, has been and can be understood in two ways: as a directional, propositional approach operating on propositions, and as a configurational approach operating on geometrical figures. It is the geometer who draws the figures and makes inferences based on them. From a cognitive viewpoint, the method of analysis thus suggests an interactive inquiry, going beyond intramental types of reasoning (see also [23]).³

Thus, it can be concluded that Aristotle's seminal idea continues to hold: there is a deep similarity between geometrical problem-solving and design. Also, we see that the idea of the duality of reasoning has an old pedigree, although it has not often been explicitly discussed.

Model-Based Reasoning in Design: An Empirical Case Study

Even if we can draw plausible analogies from other fields, the only reliable way of understanding how model-based abduction and model-based reasoning in general, occurs in design is to observe design itself. Here, we have selected the invention of airplanes by the Wright brothers (WB for brevity) as a retrospective case. A major reason for this selection is that the design process is generally well described, and specifically Chap. 6 in [46] gives a detailed account of the WB' design process and reasoning when trying to illustrate how model-based reasoning occurs in engineering. We briefly describe the design process, primarily based on [46] and then analyze several specific steps that are related to model-based reasoning. The findings from this case study are then discussed.

Brief Description of the WB' Design Process

The WB realized at the outset that an airplane needed three components: wings for lift, engine for propulsion, and a system for the pilot to control it. They chose

³Hatchuel et al. [31] present an opposite view, based on Gedenryd's [44] (to us) erroneous interpretation of the method of analysis as intramental.

to begin by addressing the control aspect, so initially, they focused on gliders (no propulsion yet), with some of the development effort carried out with the help of kites, specially made testing equipment, and wind tunnels. To design a control system, they incorporated a front-mounted horizontal “rudder,” or elevator, for pitching the aircraft up or down, and twisting, or “warping,” wings for banking or turning. The wings’ role as lift provider was addressed next. A minimum speed to get the glider airborne was estimated based on available data, and various launching methods were considered, finally choosing flying against strong winds in Kitty Hawk, North Carolina.

Repeated testing and modifications to the gliders’ wings were necessary for the WB to develop the required understanding of lift and drag and their relation to wing profile shape. It also turned out that several design aspects were coupled, for example, controlling the aircraft by wing warping was affected by the changes needed to increase lift, and occasionally new problems were discovered, such as controlling the location of the center of pressure. The WB’ evolving theory of flight had to be updated continuously. At last, they added a steerable rudder at the rear connected to the control wires of the warping wings, and thus established a full system of control for their glider.

Next, they turned into the propulsion system design, just to discover that there was no theory of propeller design. This led them to develop a new theory, whereby a propeller blade is regarded as a lift-producing wing moving in spirals and thus producing thrust. They also decided to use two counterrotating propellers to overcome torque effect. When they could not find a suitable engine, they designed and made their own engine, and connected its output shaft to the propellers by means of bicycle chains, one of them twisted through 180°. The powered Flyer was the WB’ fourth aircraft after three gliders and flew successfully on December 17, 1903.

The WB’ Design Strategy

The WB chose control over lift and propulsion as the most difficult aspect of designing an aircraft, and consequently as the problem to start with. This choice was abductive, based on a mental model imagined of an airplane whose engine fails (therefore, it is a glider) and the pilot trying to land it safely but loses control over it and crashes, which is exactly what happened to some aviation pioneers. But if the pilot could maintain control over the glider, he would have landed it safely, and therefore control is more important than propulsion. How about lift? Having the model of a glider in mind, the WB seem to have initially put this aspect aside, probably assuming that the knowledge to design wings was available, and therefore this task would be easier.

Johnson-Laird [46] attributes the success of the WB to their choice of control as the most important aspect of the design, something that other aviators failed to see. This is probably true, but we do not really know it for sure. They could have figured the control and then not find a way to build large enough wings that were also lightweight or find an engine that was powerful enough and lightweight. In fact, they put the engine issue to be last (and not the wings), probably because their control

system design involved also the wings. When they turned into propulsion, they built their own engine but discovered that an engine was not enough and a good propeller was also necessary. Undoubtedly, in addition to their excellent reasoning skills, they were also lucky. This connects with the notions of guessing and intuition that Peirce mentions as characteristic of abductive reasoning.

Control System Design

When faced with the anomalous situation of controlling an airplane, the WB managed to introduce several innovations that were based on model-based abductions. When their rivals were looking for stability, the WB drew an analogy (a model-based abduction) from the world of bicycles (their business) to the world of aircraft: just as bicycles are not stable, aircraft too should not be stable, but rather, be controllable by the pilot. So, just as in the bicycle world model the rider balances and controls it by steering the front wheel and leaning it to the side, pilots should steer an aircraft left or right, bank it to the side, and nose it up or down (an added dimension).

A horizontal “rudder,” or elevator, was assigned the role of controlling climbing and diving, but banking presented a new anomalous situation. An insight came from an analogy to birds: just as birds point their wingtips in opposite directions in order to turn, the aircraft wing could be twisted, or “warped,” to produce a bank or turn. This analogy to birds was concurrent with another analogy to bicycles: bicycles do not stay vertical when turning, and therefore aircraft too should not turn in a horizontal plane, but rather, bank in order to turn. All in all, two analogies were used to connect three models: birds, bicycles, and aircraft.

How would wing warping be effected? The wing needed to be both flexible in torsion and stiff laterally. This new anomalous situation was solved by an analogy that came as an insight while twisting a square-section inner tube box (a physical model, in this case) and imagining its top and bottom surfaces to correspond to the warped upper and lower wings of a biplane. A physical model of the wings, made of bamboo and tissue paper, was constructed to check the idea, followed by another physical model, a 5-foot span biplane kite with cords connected to the wingtips. It was tested and confirmed the banking ability by wing warping.

Designing the Wings for Lift

Now that the role of the wings in controlling the aircraft has been established, the WB turned into designing the wings for their other role: provision of lift. They used available knowledge (Otto Lilienthal’s data) on the lift and drag to design the wings for their man-carrying glider (no engine yet). Lift (and drag) is related to the aircraft speed, and a minimum speed is required to get the aircraft airborne. So the next problem to be solved was: how to give the aircraft the initial speed?

The WB considered the contemporaneous practice of using gravity, either jumping with the glider from the crest of a hill or dropping it from a balloon, to be too dangerous. They considered constructing a catapult to launch the glider but thought it would be too challenging. Then they had an idea that came from another way of looking at the problem, what we may call “abductive transformation”: speed the air past the glider. Instead of imagining an airplane being accelerated to produce enough speed for lift, the WB imagined the background (the air) speeding past the aircraft. This is model-based reasoning: a world model of an aircraft moving through the air is replaced by a visualization of air moving around the aircraft. But how could air be speeded past the wings? The idea was to fly against strong winds, so they chose Kitty Hawk in North Carolina as the location for flying. They built the glider and flew it with the pilot lying on the lower wing to reduce drag. They practiced controlling it.

They discovered that the wings did not produce the expected lift and identified two possible hypothetical explanations (abduction of the IBE kind): either the wing section had a too shallow camber or the wing area was too small. A second glider was constructed with modifications to the wings, but lift was still low and control difficult. The core of difficulty was attributed to controlling the center of pressure, and this led to experimenting with flying the upper wing alone as a kite (use of a physical model). Corresponding modifications were incorporated in the glider, but now the wing warping did not work well. All in all, the problems with lift and warping presented a new anomalous situation, that of a discrepancy between the WB’ mental model of aircraft wings and the empirical evidence. Again, by abduction of the IBE kind, they created possible explanations or hypotheses and went on to test them. They constructed a device mounted on a horizontal bicycle wheel to measure the lift produced by various wing profiles. They confirmed the hypothesis that Lilienthal’s data were wrong and concluded that they needed to generate their own data, which was accomplished by building wind tunnels and conducting experiments.

The WB used the new aerodynamic knowledge to design their third glider. By analogy to birds (buzzards versus eagles and hawks), they introduced a slight negative dihedral (a downward slope of the wings relative to the horizon when viewed head-on) to the wings to improve stability. Vertical tails were also added to help with the warping problem. Testing the glider still did not show good results until a new idea emerged after a sleepless night (thus, hinting that it was a sudden insight coming in a flash, as often characterizing abductive reasoning [47]), to turn the fixed tail into a steerable rudder, and connect its control wires with those that warped the wings. A full aircraft control system was thus established.

Propulsion

When addressing the propulsion aspect, the WB discovered that no theory existed for propellers. Their rivals used flat propeller blades, and the WB realized that they needed to create their own theory and knowledge about propellers. They drew an analogy that the blade is like a wing traveling in a spiral course. This analogy depended

on visualizing a mental model of a wing carrying out a rotation, and regarding the lift produced as thrust. A flat blade does not generate much lift/thrust, so the blade should be cambered like a wing, and wing theory could be used to design the propeller. They designed a propeller and the theory turned out to be very accurate. They decided on two counterrotating propellers mounted behind the wings to minimize turbulence.

The WB now turned into engines. While their rivals looked for the most powerful and lightest engines, they used their theory of flight (wing area, lift, drag, estimated weight, minimum airspeed, etc.) to estimate the minimum power requirement. They could not find a manufacturer with the right engine, so they designed, tested, and re-designed their own engine. Finally, they figured out a way to connect the engine to the two oppositely rotating propellers by another analogy to the world of bicycles: an arrangement of sprockets and chains transmitted the power.

Discussion

The findings from this case study trigger conclusions and reflections into several directions. In the following, we discuss the findings regarding the re-proposed concept of abduction in design, strategic abduction, role of models in design generally, and specifically of mental models.

The Re-proposed Concept of Abduction in Design

First, it can be stated that all the assumptions underlying the re-proposed concept of abduction turn out to be true in the case studied: abduction (and thus creativity) occurred throughout the design process; the problems and their novel solutions were deeply contextual; the outcomes of abduction were evaluated through their practical utility; and several abductive inference types, especially regressive inference, composition, analogy, transformation, and manipulation were employed. What also becomes clear is that most key abductions (many of which were analogies) were model based. Although this was somewhat expected, the force and ubiquity of model-based reasoning offered a surprise.

One type of abduction that was not separately discussed in [1] stands out, namely, strategic abduction to determine the order of design developments. It seems to belong to key types of design abductions, although it may not be needed in all design tasks. Below, the specific strategic abduction, steepest-first, is separately commented.

Moreover, the case study reminds that abduction is only one link in the chain of mental moves; creativity is important but so is the critical evaluation, the skill to find counterexamples. Without keen skills in critical evaluation, there is the risk that time is wasted in pursuing unpromising avenues in design. Finally, the case study shows

that quite often it is possible to pinpoint the source of the new idea; creativity does not need to be considered as a mysterious, unexplainable phenomenon.

Strategic Abduction: Steepest-First

The WB used what Johnson-Laird [46] calls “multi-stage” strategy, as opposed to “trial and error,” the neo-Darwinian process that is inefficient, or to the neo-Lamarckian process of reasoning governed by full knowledge of all the constraints, which was not possible due to the newness of the field and paucity of knowledge. “Multi-stage” strategy means that they cycled through generating ideas, embodying them as mental or physical models and evaluating them, and in each cycle they regenerated and reevaluated. Most importantly, this strategy requires an approach that we call “steepest-first” in the context of the PA method [40]: the most challenging aspect of the design task is addressed at any given moment in the process. This approach may be contrasted with “systematic design” methods, where all the design functionalities are handled concurrently and where all the relevant knowledge is already available [48]. If all the functions and sub-functions can be known at the beginning of the design process, and if solutions that satisfy all these functionalities can be listed, then by a sort of deductive logic, combinations of the solutions will be the desired artifact. However, in innovative design cases, the functionalities may not be fully known and decomposable, and solutions are not readily available, so another strategy is needed. Choosing to address more difficult problems first is justified by assuming that problems and solutions are coupled, and therefore it should be more efficient to add the solution of easier problems to those of the difficult ones than vice versa.

The steepest-first approach—producing a hypothesis of the best order of solving different subproblems—is an abductive strategy in the sense that it is not truth preserving, that is, it can lead to a dead end or to an inferior solution. However, it has been shown to produce good results in many innovative design situations, both in terms of the final outcome and the efficiency of the process.

The Role of Models in Design

Models that appear in design reasoning range from external (physical and representational) to internal (mental and visual), and they differ in the way they are used. Among the external models, we can list sketches, computer-aided design (CAD) models, auxiliary physical objects, and the designed artifact itself. The role of sketching in design reasoning has been studied by such scholars as Schön [49] and Goldschmidt [35], who emphasize the mental dialogue through reflection between the architect and the sketch. More concrete models—CAD images and models made of cardboard, wood, clay, 3d-printed polymers, etc.—serve a similar purpose in design: provide a means

to evaluate and test the evolving artifact and, if deemed necessary, trigger another design cycle. The WB's gliders that preceded their powered 1903 Flyer fall under this category, as does their sketches and construction plans.

Perhaps a separate kind of physical model is the auxiliary object created for demonstrating or testing a particular aspect of the design. The WB built also kites, to test the concepts of wing warping and the movement of the center of pressure. They constructed a whirling arm device mounted on a horizontal bicycle wheel to determine the lift generated by various wing sections, and they built wind tunnels for drag and lift experiments.

Internal Models in Design

While the role of physical models in design reasoning is quite clear, the underlying interest here is in internal models, especially mental models and visual imagery. We claim that an important source of design ideas and concepts that come to the designer's mind is internal models of phenomena and other artifacts. The former includes understanding of physics and other scientific principles and the ability to imagine, visualize, their expression in reality. The latter consists mainly of analogies to familiar devices and the extraction of fundamental knowledge that can be applied in a different design task. The WB showed deep understanding of the physics of flight, and even generated knowledge when they could not find it or found errors in published data (lift and drag calculations, minimum airspeed for takeoff, minimum power required for level flight, etc.). They also excelled in visualizing the flow of air over various surfaces, which led to their wing warping and tail rudder design.

Their analogies to the familiar (to them) world of bicycles (control and stability issues, power transmission, and more) and to the world of birds (banking the aircraft and wing dihedral) were profound: they did not superficially copy features from one artifact to another, but rather exhibited deep learning in one field of solution principles that were transferable to another area. The WB also drew an analogy from twisting an inner tube cardboard box to warping the wings. We can assume that bird watching was a deliberate process of gaining knowledge about flying, while the twisting box analogy was accidental. Their analogies from bicycles stem from accumulated understanding in this field, and not from direct or serendipitous observation.

Conclusion

All in all, we claim that the general use of models in design, and internal models in particular, is closely associated with abductive reasoning. In doing so, we provide first an explanation to where abductive hypotheses come from (a model formed in the abducer's mind), and second, we shift the emphasis in studying design abduction from the sentential, formal logic approach, to looking at design abduction as a char-

acteristic of an inference. Previous treatments of design abduction (e.g., [50–53]) concentrated on showing how mental moves in design correspond with various syllogistic forms of abductive inferences. However, and as already pointed in [2], design abductions should focus on the novelty of the outcome, which in turn is relative and depends on what the “reasoner” knows at the time of making the inference. In other words, the mental model of a problem triggers the retrieval of the model of a solution, by resemblance, from the memory or the perception. This is the source of the generated hypotheses (plausible design solutions), and thus the synthetic, or ampliative, (nonevaluative) activities in design should be studied from the model-based reasoning perspective.

Model-based abduction is a more encompassing notion than the sentential approach. In fact, it seems that all logical formulations of abduction can be represented as model based, and the latter form adds the important information related to the knowledge that is the source for proposing the abduction’s conclusion. Knowing the model upon which the abduction is based allows us to judge the extent of novelty (relative to the abducer’s knowledge) in the reasoning and to classify the inference accordingly.

References

1. Koskela L, Paavola S, Kroll E (2018) The role of abduction in production of new ideas in design. In: Vermaas P, Vial S (eds) *Advancements in philosophy of design*. Springer
2. Kroll E, Koskela L (2017) Studying design abduction in the context of novelty. In: *DS 87-7 Proceeding of the 21st International Conference on Engineering Design (ICED 17)*, Vol 7: *Design Theory and Research Methodology*, Vancouver, Canada
3. Johnson-Laird P (2010) Mental models and human reasoning. *PNAS* 107(43):18243–18250
4. Kapitan T (1990) In what way is abductive inference creative? *Trans Charles S. Peirce Soc* 26(4):499–512
5. Magnani L (2004) Model-based and manipulative abduction in science. *Found Sci* 9:219–247
6. Magnani L, Bertolotti T (2017) *Springer handbook of model-based science*. Springer, Switzerland
7. Johnson-Laird PN (2001) Mental models and human reasoning. In: *Language, brain, and cognitive development: essays in honor of Jacques Mehler*. MIT Press, Cambridge, MA, pp 85–102
8. Johnson-Laird PN (1998) Imagery, visualization, and thinking. In: Hochberg J (ed) *Perception and cognition at the century’s end*. Academic Press, San Diego, CA, pp 441–467
9. Johnson-Laird PN (2002) Peirce, logic diagrams, and the elementary operations of reasoning. *Think Reason* 8(1):69–95
10. Tversky B (2011) Visualizing thought. *Top Cogn Sci* 3:499–535
11. Tversky B, Kessell A (2014) Thinking in action. *Pragmat Cogn* 22(2):206–223
12. Tversky B (2015) The cognitive design of tools of thought. *Rev Phil Psych* 6(1):99–116
13. Hintikka J (1997) On creativity in reasoning. In: Andersson ÅE, Sahlin N-E (eds) *The complexity of creativity*. Kluwer, Dordrecht, pp 67–78
14. Pietarinen AV (2011) Existential graphs: What a diagrammatic logic of cognition might look like. *Hist Philos Logic* 32(3):265–281
15. Peirce CS (1909) MS 619: 8. *Studies in meaning*
16. Peirce CS (1901) CP 2.778. Notes on ampliative reasoning. *Collected papers of Charles Sanders Peirce*, vol 5. HUP, Cambridge, Mass, p 2
17. Paavola S (2011) Diagrams, iconicity, and abductive discovery. *Semiotica* 186:297–314

18. Peirce CS (1898) CP 7.498. In: Burks AW (ed) *Collected papers of Charles Sanders Peirce*, vol 7. HUP, Cambridge, Mass
19. Peirce CS (1893) *Qualitative logic*, CP 7.451–2. In: Burks AW (ed) *Collected papers of Charles Sanders Peirce*, vol 7. HUP, Cambridge, Mass
20. Peirce CS (1901) *On the logic of drawing history from ancient documents especially from testimonies (logic of history)*, CP 7.218. In: Burks AW (ed) *Collected papers of Charles Sanders Peirce*, vol 7. HUP, Cambridge, Mass
21. Gärdenfors P (2004) *Conceptual spaces: the geometry of thought*. MIT Press
22. Bruza P, Cole R, Song D, Bari Z (2006) *Towards operational abduction from a cognitive perspective*. *Logic J IGPL* 14(2):161–177
23. Magnani L (2013) *Thinking through drawing: diagram constructions as epistemic mediators in geometrical discovery*. *Knowl Eng Rev* 28(3):303–326
24. Douven I (2017) *Abduction*. In: Zalta EN (ed) *The Stanford Encyclopedia of Philosophy* (Summer 2017 edn). URL = <https://plato.stanford.edu/archives/sum2017/entries/abduction/>
25. Minnameier G (2004) *Peirce-suit of truth—why inference to the best explanation and abduction ought not to be confused*. *Erkenntnis* 60(1):75–105
26. Paavola S (2006) *Hansonian and Harmanian abduction as models of discovery*. *Int Stud Phil Sci* 20(1):93–108
27. Campos D (2011) *On the distinction between Peirce’s abduction and Lipton’s inference to the best explanation*. *Synthese* 180:419–442
28. McAuliffe W (2015) *How did abduction get confused with inference to the best explanation?* *Trans Charles S Peirce Soc* 51:300–319
29. Peirce CS (1903) *Harvard lectures on pragmatism: lecture VI*, CP 5.171–172. In: *Collected papers of Charles Sanders Peirce*, vol 5. HUP, Cambridge, Mass
30. Runco MA, Jaeger GJ (2012) *The standard definition of creativity*. *Creat Res J* 24(1):92–96
31. Hatchuel A, Le Masson P, Reich Y, Subrahmanian E (2018) *Design theory: a foundation of a new paradigm for design science and engineering*. *Res Eng Des* 29(1):1–17
32. Stacey M, Lauche K (2005) *Thinking and representing in design*. In: *Design process improvement*. Springer, London, pp 198–229
33. Maier AM, Wynn DC, Howard TJ, Andreasen MM (2014) *Perceiving design as modelling: a cybernetic systems perspective*. In: *An anthology of theories and models of design*. Springer, London, pp 133–149
34. Cross N (2004) *Expertise in design: an overview*. *Des Stud* 25(5):427–441
35. Goldschmidt G (1991) *The dialectics of sketching*. *Creat Res J* 4:123–143
36. Pahl G, Beitz W, Feldhusen J, Grote K-H (2007) *Engineering design: a systematic approach*, 3rd edn. Springer, London
37. Gero JS, Kannengiesser U (2014) *The function-behaviour-structure ontology of design*. In: *An anthology of theories and models of design*. Springer, London, pp 263–283
38. Le Masson P, Weil B, Hatchuel A (2017) *Design theory: methods and organization for innovation*. Springer, Switzerland
39. Ullah AS, Rashid MM, Tamaki JI (2012) *On some unique features of C-K theory of design*. *CIRP J Manuf Sci Technol* 5(1):55–66
40. Kroll E, Condoor S, Jansson DG (2001) *Innovative conceptual design: theory and application of parameter analysis*. CUP, Cambridge
41. Kroll E, Le Masson P, Weil B (2014) *Steepest-first exploration with learning-based path evaluation: uncovering the design strategy of parameter analysis with C-K theory*. *Res Eng Des* 25:351–373
42. Kroll E, Farbman I (2016) *Casting innovative aerospace design case studies in the parameter analysis framework to uncover the design process of experts*. *Des Sci* 2
43. Koskela L, Codinhoto R, Tzortzopoulos P, Kagioglou M (2014). *The Aristotelian proto-theory of design*. In: *An anthology of theories and models of design*. Springer, London
44. Gedenryd H (1998) *How designers work—making sense of authentic cognitive activities*. Lund University, vol 75

45. Hintikka J, Remes U (1974) The method of analysis: its geometrical origin and its general significance. Reidel, Dordrecht
46. Johnson-Laird PN (2006) How we reason. Oxford University Press, New York
47. Peirce CS (1903) Harvard lectures on pragmatism: lecture VII, CP 5.181. In: Collected papers of Charles Sanders Peirce, vol 5. HUP, Cambridge, Mass
48. Kroll E (2013) Design theory and conceptual design: contrasting functional decomposition and morphology with parameter analysis. *Res Eng Des* 24:165–183
49. Schön DA (1983) The reflective practitioner: how professionals think in action. Basic Books, New York
50. March L (1976) The logic of design and the question of value. In: March L (ed) The architecture of form. CUP, Cambridge, pp 1–40
51. Roozenburg NFM (1993) On the pattern of reasoning in innovative design. *Des Stud* 14:4–18
52. Dorst K (2011) The core of ‘design thinking’ and its application. *Des Stud* 32:521–532
53. Kroll E, Koskela L (2016) Explicating concepts in reasoning from function to form by two-step innovative abductions. *AIEDAM* 30:125–137

Ekphrasis as a Basis for a Framework for Creative Design Processes



Udo Kannengiesser and John S. Gero

This paper introduces the notion of ekphrasis in the arts as a basis for developing a framework of creative designing. Ekphrasis is the transformation of a concept from one medium or domain (e.g. sculpture) to another medium or domain (e.g. music). When used in design, ekphrasis enables the use of new processes afforded within the new domain that can produce new concepts not available in the original domain. We show how five known mechanisms of creative designing—emergence, analogy, combination, mutation and first principles—can be included in a general framework as instantiations of ekphrasis. This framework is developed based on the function–behaviour–structure (FBS) ontology and its application to affordances.

Introduction

Design researchers have sometimes drawn on the world of art as a source of inspiration for explaining or illustrating concepts of designing, mainly in the area of design creativity. Most of the metaphors presented in these studies remain on an informal level. Recently, the artistic concept of ekphrasis has been formalised and used as a basis for a computational model of creative designing [1, 2]. Ekphrasis is the transformation of a concept from one medium or domain to another medium or domain [3–8]. Take as an example the mythical story of King Arthur and Excalibur, the foundation of the rightful sovereignty of the British. The precise nature of the story and what it exemplifies is not of interest here. What is of interest is that the story is depicted in multiple other forms. It is expressed as a painting in Fig. 1a, as a sculpture

U. Kannengiesser (✉)
Compunity GmbH, Linz, Austria
e-mail: udo.kannengiesser@compunity.eu

J. S. Gero
University of North Carolina at Charlotte, Charlotte, NC, USA

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_15



Fig. 1 King Arthur and Excalibur represented as **a** a painting, **b** as a sculpture and **c** as a movie

in Fig. 1b and as a movie in Fig. 1c. All three are examples of ekphrasis where the nature of the domain of expression allows for different expressions.

Ranjan et al. [9, 10] showed that the cross-domain interpretation of artistic ideas, i.e. ekphrasis, can be tested empirically and that such a cross-domain interpretation of artistic ideas can be the basis of a form of creativity. In Gero's [1, 2] model of ekphrasis, novel design concepts are the result of two instantiations of ekphrasis: One instantiation transforms the design representation from the original domain of designing to a new domain, leading to new processes that can operate on that representation. A second instantiation transforms the results of executing the new processes back into the original domain, leading to the production of new design concepts in that domain. Here, the notion of a 'domain' is understood as an agreed area of knowledge, which may include technological domains [11] and representational domains on a symbolic level.

In this paper, we extend Gero's [1, 2] model of ekphrasis by deriving a generic framework based on the function–behaviour–structure (FBS) ontology and its application to representational affordances. These affordances are defined as the action possibilities of a designer when interacting with a design representation, e.g. calculating the area of a building when being shown a floor layout representation. We show how five known mechanisms of creative designing—emergence, analogy, combination, mutation and first principles—can be viewed as instantiations of this framework. This has the advantage that creative design processes and techniques that traditionally have been studied separately can now be treated in a uniform way. Insights in creative designing may thus be more easily transferred across different methods and different domains, as they can be described using the same foundational model.

This paper is organised as follows: Section 'Creative Designing' presents foundations of design creativity using a state space view of designing. Section 'Modelling Creative Designing Using Ekphrasis' develops an ontological framework of ekphrasis that is then extended to represent creative designing. Section 'Processes of Creative Designing as Ekphrasis: Results' applies this framework to the mechanisms of emergence, analogy, combination, mutation and first principles. Section 'Conclusion' concludes the paper with a discussion of potential future work.

Creative Designing

Boden [12] distinguished between ‘historical’ (or h-) creativity and ‘psychological’ (or p-) creativity. For h-creativity, novelty is evaluated in relation to the history of humankind. The first steam engine was an example of an h-creative design. P-creativity implies novelty only with respect to the lifetime of an individual, for example, a novice architect designing a high-rise office building for the first time.

An extension of Boden’s classification has been proposed by Suwa et al. [13] who added the notion of ‘situated’ (or s-) creativity. S-creativity is defined with respect to the situation rather than to the outcomes of the process. A design or design concept is viewed as s-creative if it is introduced for the first time in the ongoing design process. S-creativity is independent of any post hoc ascriptions of creativity to the product of designing. The concept of creativity used in this paper is one of the s-creativities.

S-creativity allows a characterisation of the design process as either routine or non-routine [14, 15]. *Routine designing* is when the state space of possible designs is well defined, fixed and bounded at the beginning of the design process. Designing then consists of finding a specific set of values for known design variables and known ranges of values. This corresponds to a view of designing as search. No creativity is involved in this idealised view. *Non-routine designing* can be either innovative and creative. *Innovative designing* assumes a well-defined, fixed and bounded set of design variables but modifies the ranges of values to be outside the norm. *Creative designing* introduces new design variables, so that the state space of possible designs is extended. Variables can be introduced additively, leading to an expanded design state space, or substitutively, leading to a shift of the design state space that may be disjoint from the original one [15]. A summary of this view of routine, innovative and creative designing is depicted in Fig. 2.

Five processes have been proposed that can lead to extensions of the design state space [15]:

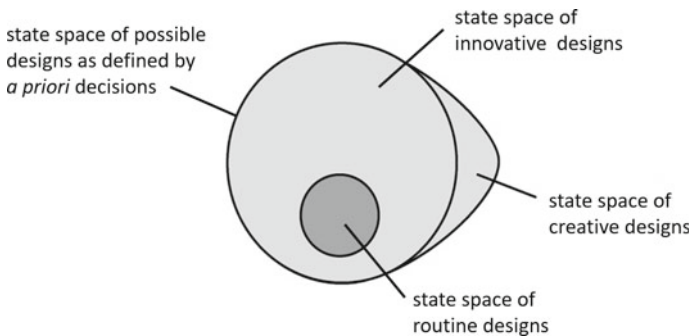


Fig. 2 The state spaces of routine designs and innovative designs as subspaces of the state space of possible designs (as defined by a priori decisions), and the state space of creative designs as its superset (here depicted for additive variable introductions)

- Emergence is the process of making implicit features in a representation explicit.
- Analogy is the process of extracting useful concepts in an existing design and introducing them into the current design.
- Combination is the process of forming a new concept from two or more separate ones.
- Mutation is the process of arbitrarily altering an existing concept.
- First principles is the process of using foundational concepts as the basis for designing.

Some of these processes have been studied separately from one another, often in different research domains and communities. Emergence is mostly the subject of research in visual cognition. Analogy, combination and mutation are studied within design creativity research. First principles are mainly used in physics-based approaches such as mechanical engineering.

This domain specificity makes the five processes difficult to apply in different domains, because every domain has its own set of representations that afford different processes operating on these representations [16]. A generic framework of creative designing that encompasses all five processes would therefore need to include transformations of representations across various domains. Ekphrasis, which is concerned with transforming concepts from one domain into another, can provide the basis for such a framework.

Modelling Creative Designing Using Ekphrasis

An ontological framework of ekphrasis can be developed based on the FBS ontology and its application to representational affordances [16].

An Ontological View of Ekphrasis

Here, we provide a brief introduction of the FBS ontology and its use in representing affordance. More detail can be found in [16]. The FBS ontology has been developed to represent a wide variety of artefacts including physical objects, software, processes and organisations [17]. Recently, this ontology has also been applied to representations in design, as the basis for a model of representational affordances [16].

Structure (S) is defined as the components of an artefact and their relationships. The structure of representations includes symbolic or iconic constructs and their relationships. For instance, a graph-based representational structure of a building may consist of nodes (representing spaces in the building) interconnected by arcs (representing topological relationships between the spaces). An iconic (geometric) representational structure of the building may consist of vectors (representing sur-

faces of the building’s shape). An evolutionary representation of the building may consist of genes (representing the layout of the rooms).

Behaviour (B) is defined as the attributes that can be derived from an artefact’s structure. External or exogenous effects may be needed to produce behaviour by interacting with the structure. These effects are often induced by the intentional actions of a user. Mental or physical operations typically establish the exogenous effects interacting with representations, producing attributes (i.e. behaviours) that describe the results of these operations. For instance, features of a graph-based representation are behaviours obtained by applying the exogenous effect of searching for specific patterns in the graph structure. The total amount of space in a geometric representation of a building is a behaviour obtained by applying the exogenous effect of using mathematical operations. A modified gene structure of the evolutionary representation of the building is a behaviour that may be obtained by applying crossover and mutation operators.

Function (F) is defined as an artefact’s teleology (‘what the object is for’). It is ascribed to behaviour by establishing a teleological connection between a human’s goals and measurable effects of the artefact. For instance, allowing compliance checking in the early stages of designing may be a function of a graph-based representation of the building. Allowing engineering simulations such as thermal analysis may be a function of a geometric representation. Exploring alternative room layouts may be a function of the evolutionary representation.

Affordances are the potential actions of a user interacting with an artefact’s structure and thereby producing artefact behaviours. In the FBS ontology, these actions can be captured as exogenous effects. Figure 3 shows two shapes symbolising affordances and behaviour, respectively. For an affordance to produce behaviour, there needs to be a ‘fit’ between the two. Conceptualising behaviour as including an ‘input port’ or ‘receptor’, which metaphorically mirrors the shape of the affordance, illustrates this fit. Relevant aspects of affordances can be defined as input parameters of behaviour, and measurable effects of these affordances can be defined as output parameters. For example, the ‘open-ability’ affordance of a door includes the amount and direction of force applied to the door. (We use the common ‘verb + -ility’ convention for labelling affordances.) The speed with which the door opens when applying the force would be an output parameter associated with this input.

In representational affordances, the input parameters describe design actions afforded by a design representation. Output parameters represent the effects of the design actions, including measures for the success of the actions with respect to a task-related goal. Take the example of the graph-based building representation;



Fig. 3 Behaviour (B) provides input parameters (X_{in}) representing relevant properties of affordances (A), and output parameters (X_{out}) representing measurable states produced

an affordance called ‘pattern search-ability’ provides the input of a behaviour that includes graph features as output. For the geometric building representation, an affordance may be called ‘space calcul-ability’, viewed as an input to a behaviour that includes the total amount of space as output. For the genetic engineering representation, the affordances may be viewed to include ‘combin-ability’ and ‘mutat-ability’.

Ekphrasis can be viewed as a transformation of a representation from an original domain to a new domain from which new representational affordances can be derived. Using the FBS ontology applied to representations, we can describe this as follows:

$$S^{dn} = \tau(S^{do}) \tag{1}$$

$$B^{dn} = \tau(S^{dn}) \tag{2}$$

where dn = new domain, do = original domain, and τ = transformation.

B^{dn} and B^{do} , and S^{dn} and S^{do} , respectively, are typically disjoint (i.e. $B^{dn} \cap B^{do} = \emptyset$, and $S^{dn} \cap S^{do} = \emptyset$) because they are based on the unique knowledge representations available in a domain. However, there may be exceptions as domains can overlap in various ways [18]. We will provide examples of such overlaps later in this paper. F^{dn} and F^{do} are non-disjoint (i.e. $F^{dn} \cap F^{do} \neq \emptyset$) because they are associated with the concept to be conveyed that transcends the different, domain-specific representations. Figure 4 uses a state space view to show the relations between the function, behaviour and structure of the representation before and after the ekphrasis.

This ontological framework can be illustrated using the example of an artwork being represented as a poem, as shown in Fig. 5. Thus, the original domain (do) in this example is art, and the new domain (dn) is poetry.

The original structure (S^{do}) of the artwork ‘Equinox’ shown in Fig. 5 is a composition of blocks, cogwheels and paint. It is transformed into the new structure (S^{dn}) of the poem ‘Autumn Window’, which can be viewed as a composition of words and sentences. The two representational structures are completely disjoint from each other, based on the disjoint types of structure elements available in the two domains.

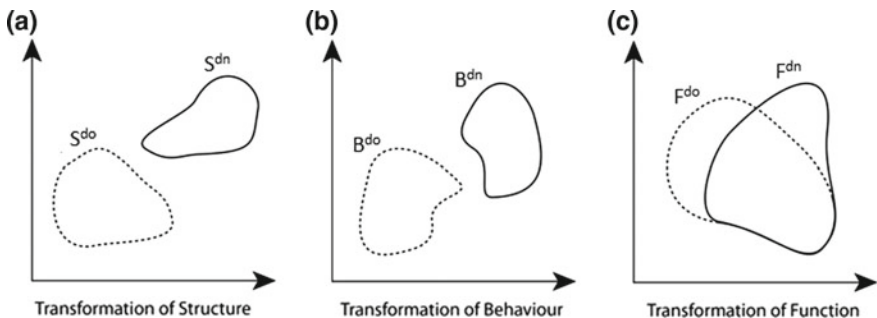


Fig. 4 Ekphrasis as transformations of structure, behaviour and function: **a** structure is transformed into a disjoint state space, **b** behaviour is transformed into a disjoint state space and **c** function is transformed into a non-disjoint state space



"Equinox"; Mixed media on wood; 19" x 19"

Autumn Window

Sun and moon
come to an even stance,
blocks of time
tell when they each
can shine,
as clocks are re-set,
equinoctial points
intersect

the lines of light streak
across the room
earlier than last week,
passive orange and yellow tones
slip through my fingers

as I sit by the window,
reminisce about youthful
moments
when time never needed
to be wound,
it simply sprung
from season to season

now it drags
as leaves on the ground
crisp and crunchy,
crumble into brown pieces.

-Suzanne Bruce

Suzanne Bruce© Autumn Window

Fig. 5 Example of ekphrasis transforming an artwork into a poem (Artwork by Janet Manalo, poetry by Suzanne Bruce; <http://www.ekphrasticexpressions.com>)

The original behaviour (B^{do}) of 'Equinox' includes attributes such as the distribution of paint and the area covered by the physical elements on the canvas. It also includes neurocognitive activities afforded by the artwork, such as spatial focussing, 3D object recognition or mental simulations. The new behaviour (B^{dn}) of 'Autumn Window' includes different attributes that are specific to the domain of poetry, such as rhymes and rhythmic patterns. Similar to the artwork, this poem can also afford various neurocognitive activities. Yet, these activities are specific to the domains of poetry and text, and thus disjoint from those afforded by the artwork. They include the syntactic and semantic interpretation of words and sentences, and the recognition of specific textual patterns and poetic styles. The original function (F^{do}) of 'Equinox' can be interpreted as the goals of conveying or evoking emotional responses related to the concept of a beginning autumn, or more generally of seasonal change, to the viewer. The new function (F^{dn}) of 'Autumn Window' is the same as the one of 'Equinox', yet the poem seems to augment it with the concept of a changing perception of time between youth and adulthood.

A Model of Ekphrasis in Creative Designing

We can develop an ontological framework of creative designing that is based on two consecutive instantiations of ekphrasis. A first ekphrasis E1 at time t_{E1} transforms the representational structure $S^{do}(t_{E1})$ in the original design domain (do) into a representational structure $S^{dn}(t_{E1})$ in a new domain (dn) and derives new representational affordances $B^{dn}(t_{E1})$ in that new domain. This is represented in Eqs. (3) and (4):

$$S^{dn}(t_{E1}) = \tau(S^{do}(t_{E1})) \quad (3)$$

$$B^{dn}(t_{E1}) = \tau(S^{dn}(t_{E1})) \quad (4)$$

Executing these affordances produces a new representation in domain dn, which is interpreted as a new representational structure (S^{dn}) to be used as a basis for a second ekphrasis E2 at time t_{E2} :

$$S^{dn}(t_{E2}) = \iota(B^{dn}(t_{E1})) \quad (5)$$

where ι = interpretation of execution results.

A second ekphrasis transforms this representational structure back into the original domain (do), as a basis for further representational affordances that allow continuing designing in the original domain:

$$S^{do}(t_{E2}) = \tau(S^{dn}(t_{E2})) \quad (6)$$

$$B^{do}(t_{E2}) = \tau(S^{do}(t_{E2})) \quad (7)$$

This model of creative designing based on double ekphrasis is shown conceptually in Fig. 6. This model can be extended using additional processes according to the FBS framework including the situated FBS framework [17]. For example, the process of evaluation can be added to address comparisons between multiple behaviours resulting from a set of transformations from structure to behaviour.

Ekphrasis could be applied iteratively to move from the original domain to a new domain and then from the new domain to a second new domain, before returning to the original domain.

Processes of Creative Designing as Ekphrasis: Results

In this section, we show the results from how the processes that can extend the design state space fit in the model of creative designing based on ekphrasis.

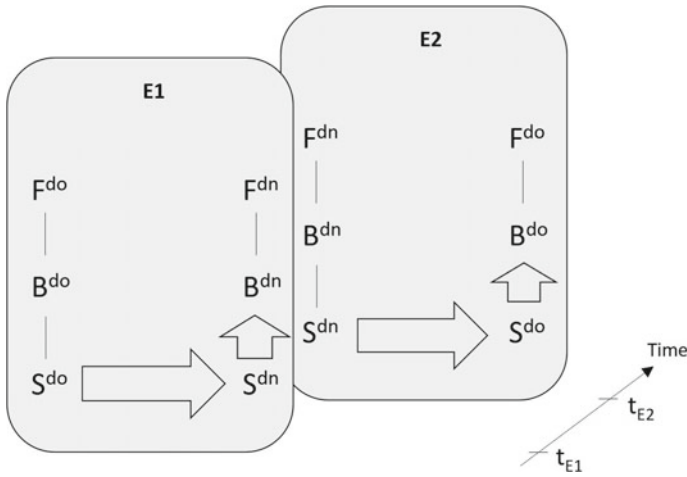


Fig. 6 Creative designing based on two instantiations of ekphrasis, E1 and E2, at times t_{E1} and t_{E2}

Emergence

Emergence makes implicit features of a representation explicit. An example of shape emergence is shown in Fig. 7. Initially, only the three triangles in Fig. 7a were drawn. Implicit in this representation is the shape of a trapezoid that in this example of emergence was made explicit in Fig. 7b.

The primary shapes (i.e. the triangles initially drawn) can be conceptualised as a representation in the domain of line segments (drawn between vertices). This representation is thus a set of line segments and vertices: $S^{do}(t_{E1}) = (\text{line segments, vertices})$. They afford perceptive activities of searching triangles in the representation: $B^{do}(t_{E1}) = (\text{searching triangles})$. This supports the designer’s goal of reasoning about two-dimensional spaces in a building design: $F^{do}(t_{E1}) = (\text{to reason about spaces in a building design})$.

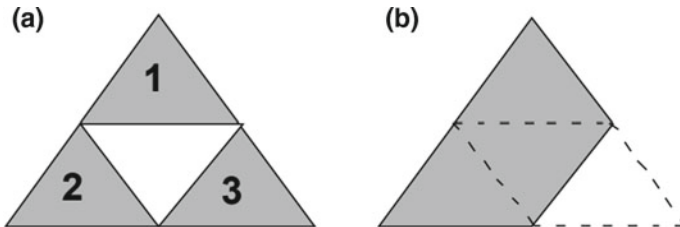


Fig. 7 An example of emergence: **a** three equilateral triangles, which are the only shapes explicitly represented; **b** one emergent form of a trapezoid moving that shape from being implicit to being explicit (image taken from [19])

Table 1 Differences between representational F , B and S across the domain of line segments (do) and the domain of maximal lines (dn) during ekphrasis 1

Ontological category	Original domain (do)	New domain (dn)
$F(t_{E1})$	To reason about spaces in a building design	
$B(t_{E1})$	Searching triangles	Searching shapes
$S(t_{E1})$	Line segments, vertices	Maximal lines

The first ekphrasis transforms $S^{do}(t_{E1})$ from its original domain—the domain of line segments—into a new domain: the domain of maximal lines [19]. Maximal lines are lines that embed at least one line segment and do not belong to the domain of line segments [20]. Consequently, the new representational structure $S^{dn}(t_{E1})$ is a set of maximal lines:

$$S^{dn}(t_{E1}) = \tau(S^{do}(t_{E1})) = (\text{maximal lines}) \tag{8}$$

The maximal lines in $S^{dn}(t_{E1})$ have various intersections that do not exist in S^{do} , which affords searching shapes that were not necessarily intended initially:

$$B^{dn}(t_{E1}) = \tau(S^{dn}(t_{E1})) = (\text{searching shapes}) \tag{9}$$

The differences between the two domains in terms of representational function, behaviour and structure are summarised in Table 1.

The trapezoid shape resulting from the search, as shown in Fig. 7b, is then interpreted as a new representational structure being used as the basis for a second ekphrasis:

$$S^{dn}(t_{E2}) = \iota(B^{dn}(t_{E1})) = (\text{emergent trapezoid}) \tag{10}$$

The second ekphrasis transforms $S^{dn}(t_{E2})$ back in the original domain, turning the intersections into vertices, and the maximal lines into line segments between these vertices:

$$S^{do}(t_{E2}) = \tau(S^{dn}(t_{E2})) = (\text{line segments, vertices}) \tag{11}$$

This affords the cognitive activity of searching trapezoids, yet now in the original domain:

$$B^{do}(t_{E2}) = \tau(S^{do}(t_{E2})) = (\text{searching trapezoids}) \tag{12}$$

Analogy

This process uses ekphrasis to express a design in domains in which similarities with the original domain can potentially be found. A popular domain for finding analogies is biology. An example of a biologically inspired design is the roof of the Munich Olympic Stadium, shown in the right-hand side of Fig. 8. Its tensile structure reminiscent of cobwebs (left-hand side of Fig. 8) was a departure from traditional stadium roofs built using massive concrete.

The initial design is represented as an optimisation problem consisting of geometric parameters of a massive stadium roof and an associated fitness function: $S^{\text{do}}(t_{\text{E1}}) = (\text{geometric parameters of massive stadium roof, fitness function})$. This representation affords the use of suitable optimisation techniques: $B^{\text{do}}(t_{\text{E1}}) = (\text{applying optimisation techniques})$. This supports the designer's goal of generating a design with optimised performance (e.g. a roof with minimal weight): $F^{\text{do}}(t_{\text{E1}}) = (\text{to generate a roof design with minimal weight})$.

The first ekphrasis transforms $S^{\text{do}}(t_{\text{E1}})$ from the domain of the built environment into the domain of biology:

$$S^{\text{dn}}(t_{\text{E1}}) = \tau(S^{\text{do}}(t_{\text{E1}})) = (\text{problem represented biologically}) \quad (13)$$

This problem representation affords search activities to find solutions in the biological world, for example, by using a biomimetics database [21]:

$$B^{\text{dn}}(t_{\text{E1}}) = \tau(S^{\text{dn}}(t_{\text{E1}})) = (\text{searching for biological solutions}) \quad (14)$$

The differences between the two domains in terms of representational function, behaviour and structure are summarised in Table 2.

The result of the search for biological solutions in this example is the cobwebs shown in Fig. 8. The phase in analogy-making that is concerned with finding such an analogous solution is commonly called 'matching' [22]. That solution is interpreted as a new representational structure to be used for a second ekphrasis:



Fig. 8 Example of analogy: the roof of the Munich Olympic Stadium was inspired by natural structures such as cobwebs

Table 2 Differences between representational F, B and S across the domain of the built environment (do) and the domain of biology (dn) during ekphrasis 1

Ontological category	Original domain (do)	New domain (dn)
$F(t_{E1})$	To generate a roof design with minimal weight	
$B(t_{E1})$	Applying optimisation techniques	Searching for biological solutions
$S(t_{E1})$	Geometric parameters of massive stadium roof, fitness function	Problem represented biologically

$$S^{dn}(t_{E2}) = \iota(B^{dn}(t_{E1})) = (\text{structure of cobwebs}) \tag{15}$$

The second ekphrasis transforms $S^{dn}(t_{E2})$ back in the domain of the built environment, formulating it as a modified optimisation problem that contains some of the design parameters describing cobwebs. This is what research in analogy in design refers to as the ‘mapping’ phase [22]. We can write

$$S^{do}(t_{E2}) = \tau(S^{dn}(t_{E2})) = (\text{geometric parameters of cobwebs, fitness function}) \tag{16}$$

This structure affords the use of standard optimisation techniques in that domain:

$$B^{do}(t_{E2}) = \tau(S^{do}(t_{E2})) = (\text{applying optimisation techniques}) \tag{17}$$

This optimisation results in the roof structure shown in Fig. 8.

Combination

Combination brings together two known concepts to form a new concept [23] that is an intersection between existing but commonly incompatible frames of reference [24]. For example, the concept of a chair can be combined with the concept of a cradle to form the new concept of a rocking chair, as shown in Fig. 9.

Using our model of ekphrasis, the process of combination in this example can be represented as follows. The initial design is assumed to be a geometrical representation of a chair: $S^{do}(t_{E1}) = (\text{geometry of a chair})$. This representation affords the use of various methods for detailing the design, such as deciding on the exact dimensions, materials, coatings and colours of the chair: $B^{do}(t_{E1}) = (\text{detailing the chair design})$. The designer’s goal associated with this representation is to produce a design that satisfies any given requirements: $F^{do}(t_{E1}) = (\text{to generate a chair design})$.

The first ekphrasis transforms $S^{do}(t_{E1})$ from the domain of chairs into the more general domain of furniture, leading to a more general representation of a system that provides support.

Table 3 Differences between representational F , B and S across the domain of chairs (do) and the domain of furniture (dn) during ekphrasis 1

Ontological category	Original domain (do)	New domain (dn)
$F(t_{E1})$	To generate a chair design	
$B(t_{E1})$	Detailing the chair design	Searching forms
$S(t_{E1})$	Geometry of a chair	Support system

$$S^{dn}(t_{E1}) = \tau(S^{do}(t_{E1})) = (\text{support system}) \tag{18}$$

The new representation affords search activities to find forms for that support structure in the new domain:

$$B^{dn}(t_{E1}) = \tau(B^{dn}(t_{E1})) = (\text{searching forms}) \tag{19}$$

The differences between the two domains in terms of representational function, behaviour and structure are summarised in Table 3.

The form of a cradle found during the search for forms is interpreted as a new representational structure providing input for a second ekphrasis:

$$S^{dn}(t_{E2}) = \iota(B^{dn}(t_{E1})) = (\text{form of a cradle}) \tag{20}$$

The second ekphrasis transforms $S^{dn}(t_{E2})$ from the domain of furniture back to the domain of chairs, by synthesising the design of a rocking chair that combines some structure features of a cradle with the initial chair design, as shown in Fig. 9:

$$S^{do}(t_{E2}) = \tau(S^{dn}(t_{E2})) = (\text{geometry of a rocking chair}) \tag{21}$$

This structure affords the use of similar detail design methods as prior to the first ekphrasis:

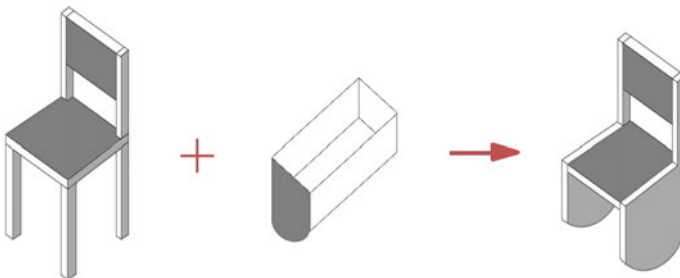


Fig. 9 Example of combination: combining a chair with a cradle to create a rocking chair [25]

$$B^{\text{do}}(t_{E2}) = \tau(S^{\text{do}}(t_{E2})) = (\text{to generate a chair design}) \quad (22)$$

Mutation

Mutation alters an existing concept. It can occur either homogeneously by changing the value of a design variable or heterogeneously by changing the class of design variable [15]. Heterogeneous mutation implies moving from one domain to another. For creative designing, it is mostly the heterogeneous type of mutation that can produce a change in the design state space. An example is the mutation of a door's hinges into a slider, which results in a different approach for opening and closing the door: from rotational to linear movement, Fig. 10.

As an instance of ekphrasis, this example of mutation can be represented as follows. The initial door design is a structure representation of a door opening mechanism using hinges: $S^{\text{do}}(t_{E1}) = (\text{structure of a rotating door})$, which affords detail design methods: $B^{\text{do}}(t_{E1}) = (\text{detailing the door design})$. The designer's goal associated with this representation is to produce a design that satisfies given requirements: $F^{\text{do}}(t_{E1}) = (\text{to generate a door design})$.

The first ekphrasis transforms $S^{\text{do}}(t_{E1})$ from the domain of physical mechanisms into the evolutionary domain, involving genes that encode various structure features of the door:

$$S^{\text{dn}}(t_{E1}) = \tau(S^{\text{do}}(t_{E1})) = (\text{genes}) \quad (23)$$

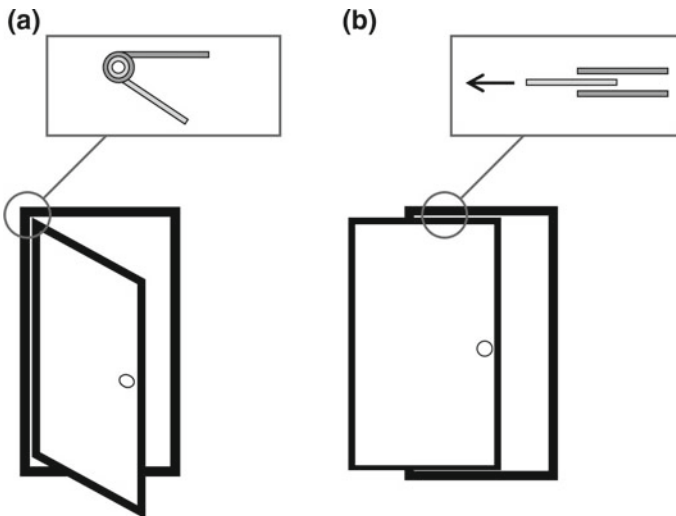


Fig. 10 Mutation of **a** a rotating door into **b** a sliding door

Table 4 Differences between representational F , B and S across the domain of physical mechanisms (do) and the evolutionary domain (dn) during ekphrasis 1

Ontological category	Original domain (do)	New domain (dn)
$F(t_{E1})$	To generate a door design	
$B(t_{E1})$	Detailing the door design	Applying mutation on genes
$S(t_{E1})$	Geometry of a rotating door	Genes

With the intention to allow random changes in the genes, the new representation affords the mutation of some of the geometrical elements:

$$B^{dn}(t_{E1}) = \tau(S^{dn}(t_{E1})) = (\text{applying mutation on genes}) \tag{24}$$

The differences between the two domains in terms of the representational function, behaviour and structure are summarised in Table 4.

The result of the mutation in the new domain is a substitution of the gene encoding ‘angle’ with a gene encoding ‘sliding length’. This is interpreted as a new representational structure providing input for a second ekphrasis:

$$S^{dn}(t_{E2}) = \iota(B^{dn}(t_{E1})) = (\text{mutated gene}) \tag{25}$$

The second ekphrasis transforms $S^{dn}(t_{E2})$ from the evolutionary domain back to the domain of physical mechanisms, by using knowledge that the new ‘sliding length’ allows moving the door in a linear direction:

$$S^{do}(t_{E2}) = \tau(S^{dn}(t_{E2})) = (\text{structure of a sliding door}) \tag{26}$$

This structure affords the use of detail design methods including the selection of a slider instead of hinges:

$$B^{do}(t_{E2}) = \tau(S^{do}(t_{E2})) = (\text{detailing the door design}) \tag{27}$$

First Principles

This process transforms a design from the physical domain into the domain of algebra, where new variables can be introduced using dimensional variable expansion [26, 27]. For example, the geometry of the beam shown in Fig. 11a is represented algebraically using two variables: length and radius. Both of these variables are then split into several variables (through a process called dimensional variable expansion), which—when transformed back into the physical world—describe beam segments of varying thickness, thus turning the original beam into a composite beam, Fig. 11b.

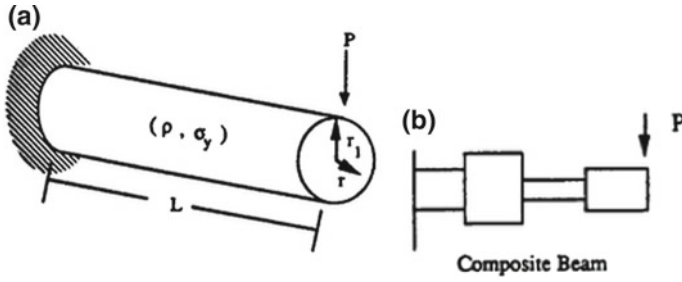


Fig. 11 Example of first principles (image from [27]): **a** original beam, **b** composite beam after dimensional variable expansion

Table 5 Differences between representational F , B and S across the domain of physics (do) and the domain of algebra (dn) during ekphrasis 1

Ontological category	Original domain (do)	New domain (dn)
$F(t_{E1})$	To generate a design for supporting/lifting loads	
$B(t_{E1})$	Searching for beam materials	Applying DVE
$S(t_{E1})$	Beam shape, load conditions	Algebraic equalities and inequalities

The initial design is represented geometrically and physically: $S^{do}(t_{E1}) = (\text{beam shape, load conditions})$. This representation may initially afford searching for materials for the beam: $B^{do}(t_{E1}) = (\text{searching for beam materials})$. This behaviour supports the designer’s goal of generating a beam design that can resist a specific load: $F^{do}(t_{E1}) = (\text{to generate a design for supporting/lifting loads})$.

The first ekphrasis transforms $S^{do}(t_{E1})$ from the domain of physics into the domain of algebra, using a set of algebraic equalities and inequalities:

$$S^{dn}(t_{E1}) = \tau(S^{do}(t_{E1})) = (\text{algebraic equalities and inequalities}) \quad (28)$$

This representation can afford various activities concerned with algebraic reasoning, one of which is dimensional variable expansion (DVE):

$$B^{dn}(t_{E1}) = \tau(S^{dn}(t_{E1})) = (\text{applying DVE}) \quad (29)$$

The differences between the two domains in terms of the representational function, behaviour and structure are summarised in Table 5.

The result of applying DVE is a set of algebraic equalities and inequalities using new variables, which is interpreted as a new representational structure to be used for the second ekphrasis:

$$S^{dn}(t_{E2}) = \iota(B^{dn}(t_{E1})) = (\text{algebraic equalities and inequalities using new variables}) \quad (30)$$

The second ekphrasis transforms $S^{\text{dn}}(t_{E2})$ back into the physical domain, representing it as a composite beam with specified load conditions:

$$S^{\text{do}}(t_{E2}) = \tau(S^{\text{dn}}(t_{E2})) = (\text{composite shape, load conditions}) \quad (31)$$

This structure affords searching for materials, possibly different materials for different beam segments:

$$B^{\text{do}}(t_{E2}) = \tau(S^{\text{do}}(t_{E2})) = (\text{searching for beam materials}) \quad (32)$$

Conclusion

Emergence, analogy, combination, mutation and first principles have been known as processes for creative designing, as they can alter the state space of possible designs. However, most of them have been studied only as instances of designing in specific domains of design and computation. This has been an obstacle for understanding their commonalities and deriving a unifying framework for them. The previous section has shown that these five creative processes can be viewed as instances of a single framework of creative designing based on ekphrasis. Such a framework facilitates communication between researchers in different design disciplines and provides a new perspective to reframe existing ways of thinking about creative processes.

The main limitation of the approach is that the notion of a domain is not formally defined in the literature. Consequently, the instantiation of the ekphrasis framework for specific examples of creative designing can be difficult, as one domain may not always be clearly distinguished from another. For example, the domains of line segments and maximal lines (see section on emergence) may not appear fundamentally different from each other although mathematically they are disjoint indicating different domains.

Other processes can be investigated to fit into this framework, such as those listed in [28]. This would test its genericity beyond the five processes examined in this paper. An example is one of the oldest and most commonly known creative processes: Wallas' [29] model of creative processes consisting of the four stages of preparation, incubation, illumination and verification. The incubation stage is akin to the first ekphrasis in our framework, as it involves the designer directing attention to an unrelated domain before a creative idea emerges and is used within the original design domain. There has been some debate as to whether it is the domain being attended to or simply the break from the original activity (via forgetting) that leads to a restructuring of the problem domain [30]. In both cases, however, incubation can be seen as a transformation of a design representation from the original domain to a new domain that affords new cognitive behaviours.

Finally, the concept of affordances used as a basis for the proposed framework provides the potential for further studies. In particular, the distinction between reflexive, reactive and reflective affordances [16] may be useful for refining the framework in

a way that considers the situatedness of designing. This may help address questions such as whether the new design variables introduced in a design state space are the result of either exploration or search in the new domain. Some of the examples in this paper suggest that search in the new domain (e.g. searching analogical solutions in a biomimetics database) may suffice to generate new design variables. The three types of affordances may be used to characterise the different modes of reasoning in the new domain.

Acknowledgements This research is supported in part by the US National Science Foundation, Grant No. CMMI-1400466. Figure 5 was used with permission from Janet Manalo and Suzanne Bruce.

References

1. Gero JS (2017) Ekphrasis as a design method. In: International conference on engineering design 2017, Vancouver, Canada (to appear)
2. Gero JS (2017) Generalizing ekphrastic expression: a foundation for a computational method to aid creative design. In: Janssen P, Loh P, Raonic A, Schnabel MA (eds) Protocols, flows and glitches. Proceedings of the 22nd international conference of the association for computer-aided architectural design research in Asia (CAADRIA) 2017, pp 345–354
3. Fowler DP (1991) Narrate and describe: the problem of ekphrasis. *J Roman Stud* 81:25–35
4. Goldhill S (2007) What is ekphrasis for? *Class Philology* 102(1):1–19
5. Knapp JA (2011) Harnessing the visual: from illustration to ekphrasis, Image ethics in Shakespeare and Spenser. Palgrave Macmillan, UK, pp 31–46
6. Leader S (2014) Ekphrasis and its reverse. Academic commons program, paper 11. http://digitalcommons.risd.edu/grad_academiccommonsprogram/11 (RISD paper)
7. Newby Z (2002) Testing the boundaries of ekphrasis: Lucian on the Hall. *Ramus* 31(1–2):126–135
8. Scott GF (1992) Ekphrasis. *Eur Romantic Rev* 3(2):215–224
9. Ranjan A, Gabora L, O'Connor B (2013) The cross-domain re-interpretation of artistic ideas. arXiv preprint arXiv:1308.4706
10. Ranjan A, Gabora L, O'Connor B (2013) Evidence that cross-domain re-interpretations of creative ideas are recognizable. arXiv preprint arXiv:1310.0519
11. Alstott J, Triulzi G, Yan B, Luo J (2017) Inventors' explorations across technology domains. *Des Sci* 3
12. Boden MA (1991) *The creative mind: myths and mechanisms*. Basic Books, New York
13. Suwa M, Gero JS, Purcell T (1999) Unexpected discoveries and s-inventions of design requirements: a key to creative designs. In: Gero JS, Maher ML (eds) *Computational models of creative design IV*. Key Centre of Design Computing and Cognition, University of Sydney, Australia, pp 297–320
14. Gero JS, Kannengiesser U (2011) Design. In: Runco MA, Pritzker SR (eds) *Encyclopedia of creativity*, vol 1, 2nd edn. Academic Press, San Diego, pp 369–375
15. Gero JS (1996) Creativity, emergence and evolution in design. *Knowl Based Syst* 9(7):435–448
16. Gero JS, Kannengiesser U (2012) Representational affordances in design, with examples from analogy making and optimization. *Res Eng Des* 23(3):235–249
17. Gero JS, Kannengiesser U (2014) The function-behaviour-structure ontology of design. In Chakrabarti A, Blessing LTM (eds) *An anthology of theories and models of design*. Springer, Berlin, pp 263–283
18. Tennis JT (2003) Two axes of domains for domain analysis. *Knowl Organ* 30(3–4):191–195

19. Gero JS (1992) Creativity, emergence and evolution in design. In: Gero JS, Sudweeks F (eds) Preprints computational models of creative design. Department of Architectural and Design Science, University of Sydney, pp 1–28
20. Stiny G (1980) Introduction to shape and shape grammars. *Environ Planning B Urban Analytics City Sci* 7(3):343–351
21. Deldin J-M, Schuknecht M (2014) The AskNature database: enabling solutions in biomimetic design. In: Goel AK, McAdams DA, Stone RB (eds) *Biologically inspired design: computational methods and tools*. Springer, London, pp 17–27
22. Gentner D (1983) Structure-mapping: a theoretical framework for analogy. *Cogn Sci* 7(2):155–170
23. Nagai Y, Taura Y, Mukai F (2009) Concepts blending and dissimilarity: factors for creative concept generation process. *Des Stud* 30(6):648–675
24. Koestler A (1964) *The act of creation*. Hutchinson, London
25. Rosenman MA, Gero JS (1989) Creativity in design using a prototype approach. Preprints modeling creativity and knowledge-based creative design. Design Computing Unit, Department of Architectural and Design Science, University of Sydney, pp 207–232
26. Cagan J, Agogino AM (1991) Dimensional variable expansion—a formal approach to innovative design. *Res Eng Des* 3(2):75–85
27. Aelion V, Cagan J, Powers G (1991) Inducing optimally directed innovative designs from chemical engineering first principles. *Comput Chem Eng* 15(9):619–627
28. Howard TJ, Culley SJ, Dekoninck E (2008) Describing the creative design process by the integration of engineering design and cognitive psychology literature. *Des Stud* 29(2):160–180
29. Wallas G (1926) *The art of thought*. Jonathan Cape, London
30. Segal E (2004) Incubation in insight problem solving. *Creativity Res J* 16(1):141–148

Notes for an Improvisational Specification of Design Spaces



Alexandros Charidis

Classical specifications for design spaces are characterized by an implicit need for a priori closure of descriptions of alternative designs before calculating. In this paper, an improvisational specification for design spaces made of shapes is presented. Shapes created visually and without prior description are recorded in a computation history. This history is read backwards to specify descriptions of recorded shapes and the space in which they are closed members. Descriptions of shapes, and the space in which they lie, are both *made* on the go as rules are applied in the course of a computation; every new visual action (rule application) redescribes the space in which the shapes obtained “thus far” belong. A reconsideration of the classical notion of a design space and its various uses in design theory is suggested, emphasizing a need to reconcile traditional formalistic pursuits that aim at “capturing” descriptions of alternative design possibilities with the open-ended, improvisational nature of creative work in architecture, the visual arts and related areas of spatial design.

Introduction

In order to specify a design space, one needs to know in advance what rules to use to construct descriptions of alternative designs and what atoms (indivisible compositional units) underlie each design description in the space. Is this foreknowledge necessary if one calculates with unanalyzed, visual shapes—with drawings, so to say? Work on design theory developed around the shape grammar formalism has shown that descriptions aren’t necessary to calculate with visual shapes (For a detailed presentation of this matter, see in [1, 2] and [3]). Instead, descriptions of shapes can be specified after rules have been applied visually. In this paper, I show that not

A. Charidis (✉)
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: charidis@mit.edu

only descriptions of shapes but also the space that includes these descriptions as closed members can be worked out backwards, after recording the generated shapes unanalyzed in a computation history.

To this end, I develop mathematical tools for an improvisational specification of design spaces made of shapes, and I present a complete specification of a space for shapes made of linear elements. The proposed improvisational specification offers a more natural framework for studying the notion of a design space in areas, such as architecture, the visual arts, and like areas of spatial design, where the open-ended, improvisational aspects of creative work are central to the design process. I articulate open questions in the last section of this paper along with new directions for the formal study of creative work.

Calculating as Improvising with Shapes

When calculating with shape grammars, shapes are treated as unanalyzed, visual—material drawings. The (visual) properties of shapes are formalized in a series of algebras U_{ij} [3], closed under the Euclidean transformations and a part relation (\leq) that includes the Boolean operations for shapes.¹ To calculate, one needs rules to manipulate shapes, to go from one shape to another in the same algebra, or to link shapes coming from different ones. Rules are defined according to certain families of *schemas* involving operators for parts of shapes, transformations (Euclidean ones but also other kinds), and boundaries. Schemas are meant to support seeing; they provide “rules of thumb” according to which shape-specific rules can be defined more or less on the fly, as a designer or composer (in the broadest sense of the word) calculates visually. Schemas—of the more unrestricted kind—are written as $x \rightarrow y$, with x and y playing the role of variables that take shapes values. For example, the rule in Fig. 1a is defined for shapes in U_{12} according to the schema $x \rightarrow t(x)$ (i.e., here $y = t(x)$), where t is a linear transformation. To distinguish between schemas and rules I will use capital letters A, B to refer to specific shapes assigned to variables x or y . The rule in Fig. 1a, for example, can be written as $A \rightarrow B$, or more elaborately, $A \rightarrow t(A)$.

For a rule to be applicable to a shape, the left part of the rule has to “match” with some part of the shape. This matching mechanism isn’t working in the same way as in generative string grammars and other formal machines that compute with symbols. A shape rule $A \rightarrow B$ is applicable to shape C whenever there exists a transformation t that makes $t(A)$ a subshape of C . Subshapes, however, are not constituents; there are no hidden or layered elements in a shape ahead of time. The matching mechanism for shapes works in the following intuitive manner: a shape is a subshape of another shape when we can trace the first in the second shape, or equivalently, when drawing the first shape on top of the second shape causes no change to the appearance of the

¹At times, notation U_i is preferred, as in U_1 or U_2 . In these cases where the “ j ” index is omitted, it is assumed that $i \leq j$.

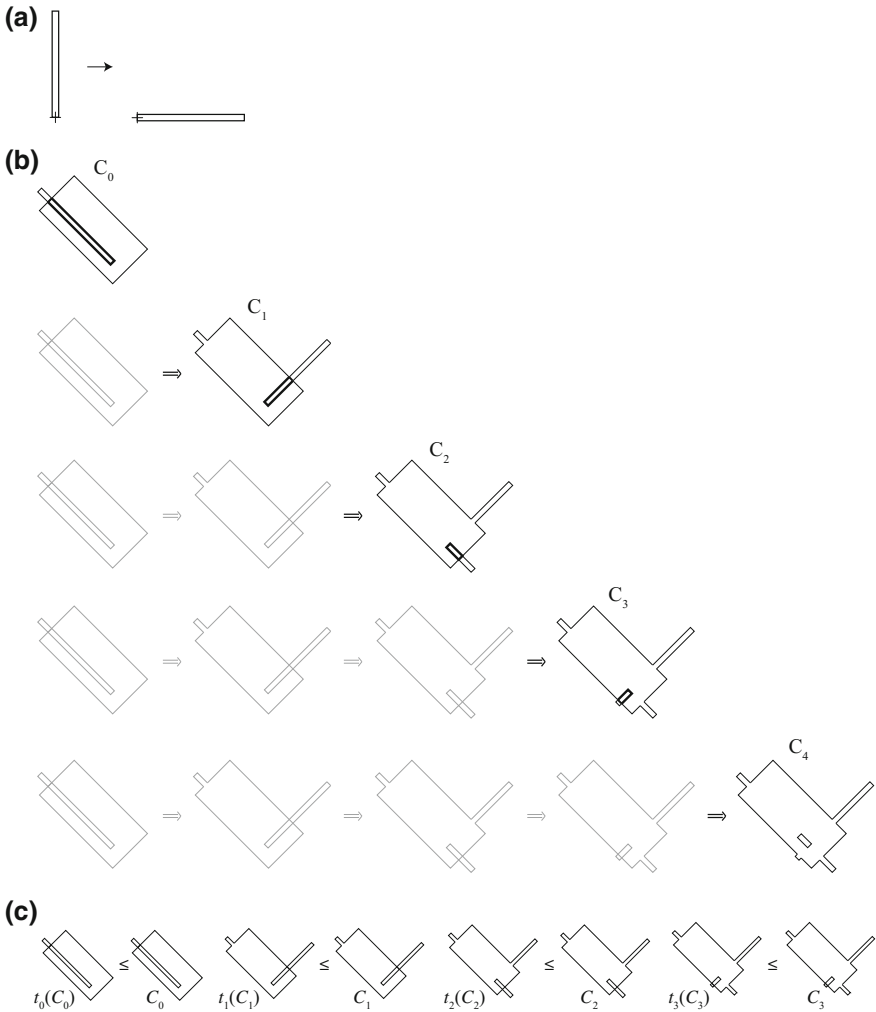


Fig. 1 **a** A rule defined for shapes in U_{12} according to the schema $x \rightarrow t(x)$. **b** A five-step computation $C_0 \Rightarrow \dots \Rightarrow C_4$ using rules in the schema $x \rightarrow t(x)$ where the “left” shape is different in each step. The rule is applied to distinguished (emergent) rectangles that satisfy the part relations given in (c)

second shape. Strings, on the other hand, are defined in algebras (monoids) with an associative operation, namely, concatenation and with symbols coming from a fixed alphabet. When composed, individual symbols preserve their integrity—symbols don’t fuse together in composition. In a string grammar, in order to apply a rule, the left side of the rule has to match identically with an existing substring of the current string—divisions and units are there to begin with. Rules defined in terms of logical atoms and their compositions as in the case of production system formalisms

in artificial intelligence and in logic [4, 5] work in an analogous manner. The only case when shapes behave precisely as symbols and atoms during a computation is when shapes are made with points (e.g. see [2]).

Rules defined in schemas together with the part relation (embedding) make calculating look like improvising with shapes: a repertoire of expressive devices (schemas) is available to make that which one sees in shapes manipulable with rules—so that “you’re free to go on as you please” [3]. “Free seeing” and descriptions of shapes are connected in an important way: the latter are byproducts of the former, not preconditions. Consider the computation in Fig. 1b for shapes in U_1 . The computation starts with shape C_0 and each new shape C_{i+1} is generated out of its preceding shape C_i according to a rule defined in a schema $x \rightarrow t(x)$ with the “left” shape now chosen on the go. The subshapes matched and changed in each step are given in a series of part relations of the form $t_i(A) \leq C_i$ for $i = 0 \dots 3$, shown in Fig. 1c (Transformation t_i embeds x to shape C_i in step i and it should not be confused with transformation t that transforms x onto $t(x)$ within the definition of the schema). Rule applications in all steps are independent of the way shapes are described. But descriptions can be imbued retrospectively, by analyzing the action of the rule used in each step.

When a rule is applied to a shape to create another one, the action of the rule implies a certain decomposition on the shape with respect to the rule application. Decompositions of shapes (descriptions) can be formalized as topologies (See prior work on topologies for shapes in [1–3]). A topology for a shape C is a set of shapes with members the shape C itself, the empty shape, and subshapes marked by a topological closure operator γ . A closure operator $\gamma: C \rightarrow C$ is a mapping which associates to every part x of C its closure $\gamma(x)$, the smallest shape in C that includes x as a part. A mapping γ is a closure operation whenever it satisfies the following properties:

- (1) $\gamma(0) = 0$;
- (2) $x \leq \gamma(x)$;
- (3) $\gamma(\gamma(x)) = \gamma(x)$;
- (4) $\gamma(x) = x$, implies x is closed;
- (5) $\gamma(x + y) = \gamma(x) + \gamma(y)$.

where x, y are any two parts of C and 0 represents the empty shape. From the above definitions, we also have that $\gamma(C) = C$, and for any two parts x, y of C if $x \leq y$ then $\gamma(x) \leq \gamma(y)$. We write as $\mathcal{T} = (C, \gamma)$ the topology of the shape C that consists of subshapes obtained by closure operator γ (The family of subshapes in a topology for a shape forms an algebra with respect to the Boolean operations of sum, multiplication and complement. Adding the operation of closure, we obtain what [6] define a closure algebra for sets; [2] defines topologies for shapes in an analogous way). The lattice diagrams in Fig. 2 show three decompositions of the same shape. Notice that only the first two satisfy the properties for a topology.

As a basic case, any shape can be described by an “all-or-none” topology with closed members the empty shape and the shape itself. But topologies can be defined more interestingly to reflect the action of the rules used in a computation. Consider, for example, the case of identity rules defined in a schema $x \rightarrow x$. One such rule is

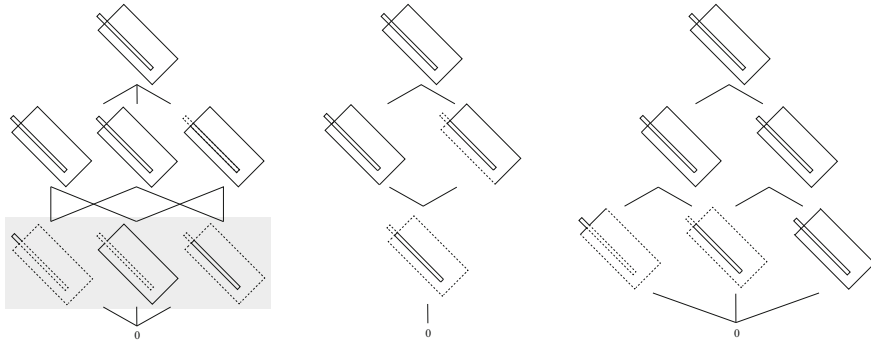


Fig. 2 Three different ways of decomposing the same shape where only the first two satisfy the properties for a topology on a shape

shown in Fig. 3a. Suppose that this identity rule is applied once under transformation t_0 to shape C_0 to create the identical shape C_1 shown in Fig. 3a. And suppose further that shape C_0 comes equipped with a closure operator γ_0 . Even if the application of the identity rule leaves the appearance of the shape untouched, it immediately implies a topology on shape C_0 that supports the action of the rule. In particular, the topology in Fig. 3b that consists of the part matched in the rule application, namely the part $t_0(A)$, and its complement $C_0 - t_0(A)$. This topology in effect shows the part(s) of shape C_0 needed to satisfy the preconditions of the visual action (rule application) after performing the action. In general, if a rule applies multiple times to the same shape, under different transformations, a topology for the shape can be such so that it includes all the parts needed to support each different rule application as long as the resulting set satisfies the basic axioms for a topology.

Back to the computation in Fig. 1b, we perform a similar analysis to obtain the topologies implied on shapes due to the rule applications. Consider the first three consecutive steps $C_0 \Rightarrow C_1 \Rightarrow C_2$. Rules applied on shapes C_0 and C_1 in a discrete fashion, immediately implying two independent topologies. In particular, the topologies in Fig. 4 where the parts matched in each rule application and their complements are kept closed. The two topologies are incompatible with respect to the rule applications: no part in the topology for shape C_0 explains the appearance of the part $t_1(A)$ matched in the rule application in C_1 . The same thing holds for the topologies of any two consecutive shapes in the computation $C_0 \Rightarrow C_1 \Rightarrow C_2 \Rightarrow C_3 \Rightarrow C_4$. The part relation makes “free seeing” and improvisation possible; a rule can always be devised on the fly in order to calculate with what we choose to see momentarily in shapes. But when we analyze the overall computation retrospectively—when we cast a visual computation to a symbolic one—we find discontinuity in the way shapes are described. In particular, the structure (topology) assigned on one shape is not preserved or altered in a continuous way (without breaks or inconsistencies) to obtain the structure for another shape. It is as if every new visual action needs a different set of atoms to describe the shape this new action generates.

Fig. 3 **a** An identity rule $A \rightarrow A$; its application to shape C_0 under transformation t_0 results in an identical shape C_1 . **b** Topology implied on shape C_0 that supports the application of the identity rule

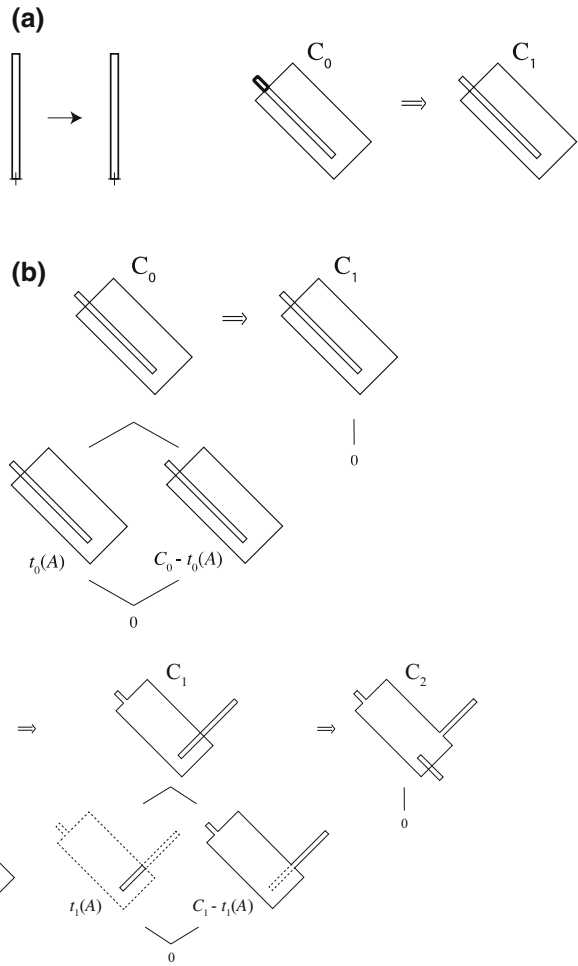


Fig. 4 Mutually incompatible descriptions of shapes with respect to the rule applications

What consequences does this entail for the classical notion of a design space where computations are meant to strictly manipulate descriptions of shapes? In classical approaches to the specification of design spaces, rules are defined in such a way so that they apply to preselected atoms, or to compositions of them [7–10]. The actual creative process, the process by which members of a space are calculated, is a matter of search through the space of possibilities (this point is emphasized in a number of places, for instance [10–12]). But if one computes with shapes bereft of atoms, and if every new visual action introduces a new set of atoms independently of past calculations, how would one specify a space of possibilities ahead of time to capture every possible description of shape that can be created visually—past, present, or

future ones? In other words, what would be the specification of a space that captures an improvisation with shapes?

In the following pages, I develop an alternative improvisational specification for design spaces made of shapes. With visual shapes, no fixed specification of a space of alternative descriptions is needed ahead of time. This space is instead constructed backwards by recording the generated shapes unanalyzed in a computation history and reconstructing the history in alternative ways. Descriptions of shapes, and the space within which they lie, are both *made* on the go; every new visual action re-specifies the space in which shapes recorded “thus far” belong.

Improvisational Specification with Shapes

A *computation (or derivation) history* for a shape grammar on a starting shape is a finite sequence of recorded shapes C_0, C_1, \dots, C_n , where, (i) C_0 is the starting shape, (ii) C_n the final shape² in the sequence, and (iii) each shape C_{i+1} follows from the preceding shape C_i according to a rule. A computation history is the complete record of unanalyzed shapes obtained when the computation is paused. In general, a computation can be paused after every new rule application, or after more than one rule applications, to interrogate the results obtained “thus far”.

Statements (i) and (ii) are straightforward; C_0 is the starting shape, where calculations begin, and C_n is the shape where calculations stop momentarily. Statement (iii) simply says that every two consecutive shapes C_i and C_{i+1} are connected according to a rule. In general, shapes in shape rules come unanalyzed—without prior decomposition into atoms. In the course of a computation, descriptions of shapes in rules and descriptions of shapes created *by* these rules are defined reciprocally—one influencing the structure of the other. By reading the computation history backwards, one can specify descriptions of shapes in such a way so that they are logically continuous with respect to the atoms in the rules that yield them. This can be done in alternative ways as the following presentation shows.

Let $A \rightarrow B$ be a rule applied to shape C under some transformation t to generate the shape C' . The action of the rule requires the part $t(A) \leq C$ to be a closed member in the topology on C (for the shape to be recognizable in the first place). More generally, any shape z can become a recognized division in C as long as it has parts shared with it, that is to say, $C \cdot z \neq 0$. If it happens that z shares parts with C , then by recognizing those shared parts as members of the topology on C we are essentially dividing C into two parts; one part is formed in the product $C \cdot z$ and the other part in the difference $C - z$. Every part x of C can be expressed in terms of z , like so:

²The term “final shape” in this case is not meant to stand as an analogy to “final configuration” in the computation history of a Turing machine or a “final string” (i.e., string without variables) in the derivation tree of a generative (string) grammar. Instead, the term final shape is meant to have a momentary flavor. It is the last shape created before we stop applying rules.

$$x = x \cdot z + x - z \quad (1)$$

Suppose γ is the closure operator of C and that, before the introduction of shape z , the only closed members in the topology of C were the shape itself and the empty shape. With the introduction of z , this topology is restructured in terms of a new closure operator γ_z that recognizes the product $C \cdot z$ as a closed member along with every other piece previously closed in terms of γ . For every part x of C^3 :

$$\gamma_z(x) = \gamma(x \cdot z) + \gamma(x - z) \quad (2)$$

The resulting topology on C with closure operator γ_z is a refined version of its previous topology defined with closure operator γ . Topology (C, γ_z) is strictly finer than topology (C, γ) ; every part in (C, γ) is also in (C, γ_z) but there exist parts in (C, γ_z) that are not in (C, γ) , for example, the part $C \cdot z$. Note that if $z = t(A)$ and $\gamma(t(A)) = t(A)$, then (2) essentially divides C into the shape $t(A)$ and its complement $C - t(A)$.

Shape C' can be calculated using the standard formula $(C - t(A)) + t(B)$. If no further rules are applied, C' has a topology with only members C' itself and the empty shape. But suppose C and C' are members of a larger computation history. Then another rule $A' \rightarrow B'$ exists which applies to shape C' under some transformation t' to generate the next shape in the sequence, say C'' . As previously, the action of the rule requires that the part $t'(A') \leq C'$ to be a member of the topology on C' . Let $z = t'(A')$ and $\gamma'(t'(A')) = t'(A')$. Then z divides C' into three parts: $(C - t(A)) \cdot t'(A')$, $t(B) \cdot t'(A')$ and $C' - t'(A')$ (C' is essentially the sum of those three parts). Now if $t'(A')$ is an emergent part, then the product $(C - t(A)) \cdot t'(A')$ requires C' to have a part in its topology whose appearance is not explained by the topology of the preceding shape C . This is precisely the case with the computation in Fig. 1 where descriptions of consecutive shapes are not continuous because rules recognize emergent parts (e.g., Fig. 4).

For two consecutive shapes C and C' to have continuous topologies, an additional condition must be met. The rule applied in C to generate C' must imply a continuous function $h: C \rightarrow C'$, from closed parts in C to closed parts in C' , so that the inequality $h(\gamma(x)) \leq \gamma'(h(x))$ holds for every part x of C [2]. The role of h is to describe the action of a rule. It can be defined in multiple ways depending on the particular rule considered. For example, the computation in Fig. 1 proceeds with rules defined in schema $x \rightarrow t(x)$, where shapes x and $t(x)$ are two independent pieces, one merely replacing the other. Hence, a mapping can be devised that describes only what $t(A)$ alters, in particular, the mapping $h: C \rightarrow C - t(A)$, defined by $h(x) = x - t(A)$. This mapping is shown pictorially as a shape rule in Fig. 5a for the first step of the computation in Fig. 1.

Mapping h takes every part of C to the shape $\hat{C} = C - t(A)$, which is also part of C' . Shape \hat{C} is the part in C that is guaranteed to stay the same in C' before $t(B)$ is added or some new part comes to be recognized due to a new rule application. In

³Using the identity: $\gamma(x + y) = \gamma(x) + \gamma(y)$, where x and y are shapes.

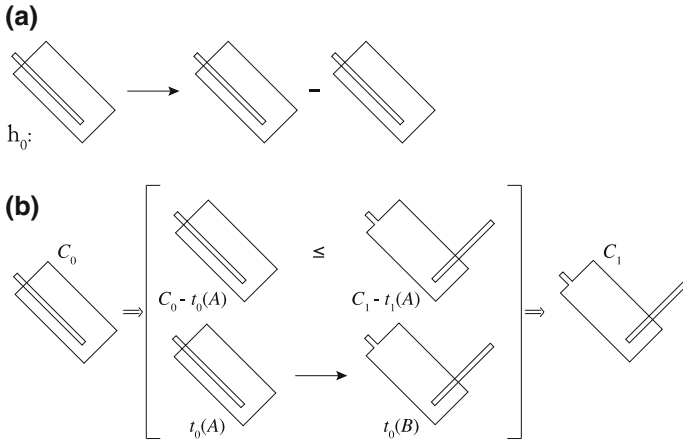


Fig. 5 **a** Shape rule represents mapping h in step $C_0 \Rightarrow C_1$. **b** Rule $A \rightarrow B$ applies to C_0 and distinguishes $t_0(A)$ and its complement $C_0 - t_0(A)$, which is also the part preserved in C_1 before $t_0(B)$ is added

Fig. 5b, for example, when the rule applies to shape C_0 under transformation t_0 to produce shape C_1 , the shape $C_0 - t_0(A) \leq C_0$ is preserved and is the same shape as $C_1 - t_0(B) \leq C_1$.

We approximate an identity relation following [2] between the topologies of C and C' so that the parts that remain unaltered in both correspond to shape \hat{C} :

$$h(\gamma(x)) = \hat{C} \cdot \gamma'(h(x)) \tag{3}$$

Using the definition of mapping h given earlier, by expanding both sides of (3) we obtain the following formula that shows how closed parts in the topologies of C and C' are related whenever the application of the rule that takes C to C' is continuous:

$$\gamma(x) = t(A) + \hat{C} \cdot \gamma'(x - t(A)) \tag{4}$$

Given a sequence of shapes recorded in a computation history, continuous topologies can be specified in a number of ways depending on what parts one wants to recognize and what parts to ignore in the recorded shapes.

Consider the sequence C_0, C_1, C_2 in Fig. 4. The topology on shape C_0 can be refined to include—in addition to the mandatory part $t_1(A)$ —a representation of the emergent piece recognized by the rule application in C_1 . This piece is formed in the product $(C_0 - t_0(A)) \cdot t_1(A)$ and is shown in Fig. 6b. This piece together with shape $t_0(B) \cdot t_1(A)$ explain the appearance of the emergent piece $t_1(A)$ in C_1 . To make the descriptions in the sequence C_0, C_1, C_2 continuous, we go backwards to shape C_0 and respecify its topology in the way shown in Fig. 6a. On the technical side, formula (4) shows how each closed part in this new topology for C is mapped to some closed

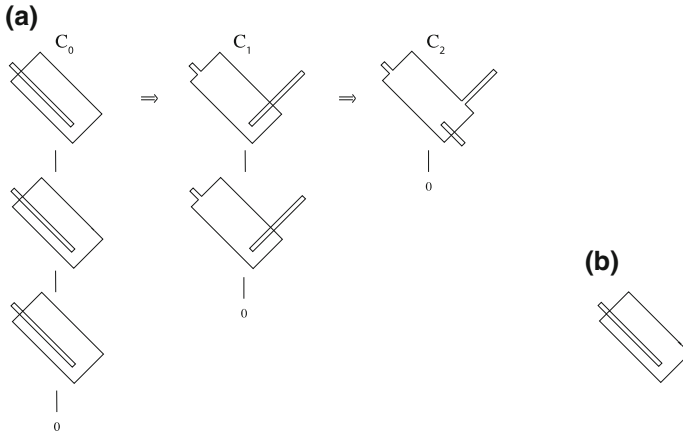


Fig. 6 **a** Continuous topologies assigned on shapes C_0 , C_1 and C_2 , **b** the non-empty part formed in the product $(C_0 - t_0(A)) \cdot t_1(A)$ due to the emergent piece $t_1(A)$

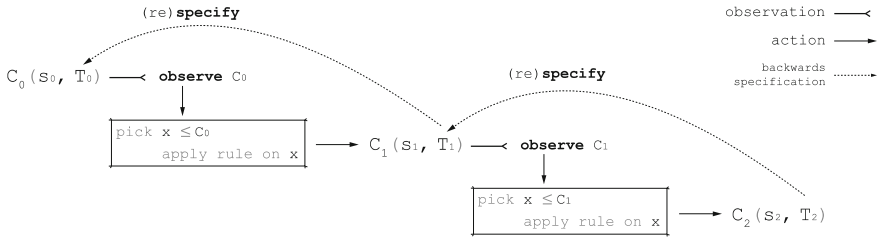


Fig. 7 Verbal sketch for (re)specifying descriptions (topologies) of shapes backwards as new rules are applied

part in the topology of C' . This overall process can be executed rule after rule, going back and forth from a forward visual action to its symbolic specification and back again. The verbal sketch in Fig. 7 explains this process diagrammatically.

Continuous topologies can be assigned to shapes in more than one ways. In sequence C_0, C_1, C_2 , there are other parts involved in the action of the rules besides the (mandatory) part $t(A)$ that makes a rule applicable to a shape. Two other series of continuous topologies are shown in Fig. 8. In the first series (Fig. 8a) $t(A)$ along with its complement $C - t(A)$ are closed in C . In the second series (Fig. 8b) $t(A)$ is closed in C and $t(B)$ is closed in C' . Each topology in the first series is a Boolean algebra; the lattices are closed under addition, multiplication, and difference. As a result, every non-empty part in the topology for shape C_0 is assembled as the union of one or more atoms. These atoms are shown at the bottom of the lattice; they are the three parts that cover the empty shape.

Continuity is still maintained in these new series insofar as the parts that belong to shape $C - t(A)$ are treated separately. Let mapping h be defined in the following way:

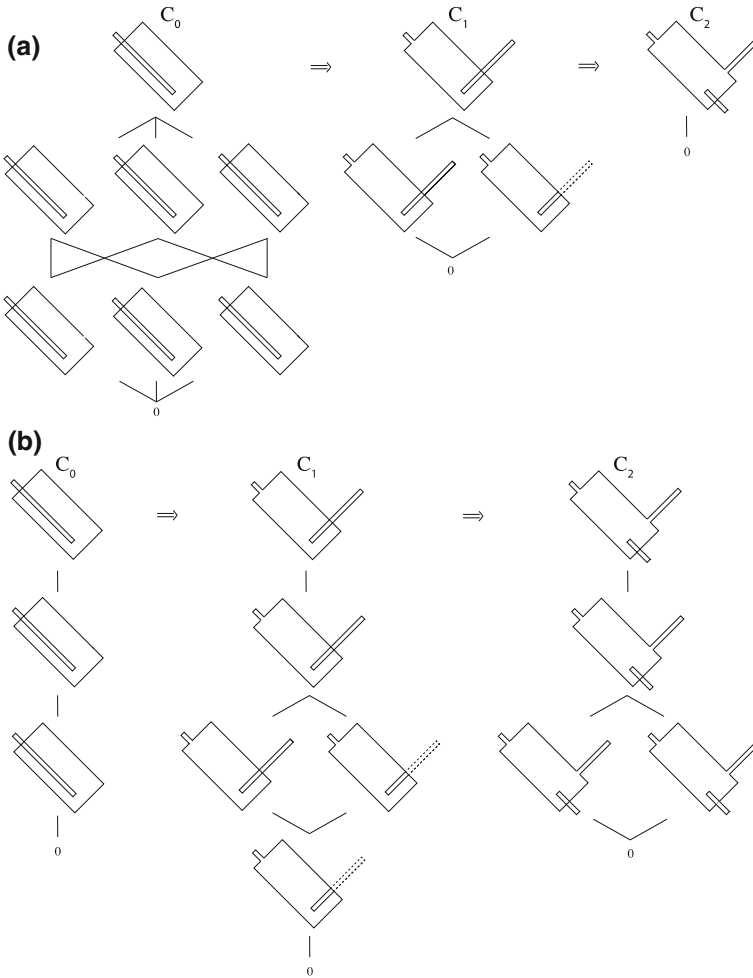


Fig. 8 Topological decomposition where **a** $t(A)$ and $C - t(A)$ are closed in C , and **b** $t(A)$ is closed in C and $t(B)$ is closed in C'

$$h = \begin{cases} x, & \text{if } x \leq C - t(A) \\ x - t(A), & \text{otherwise} \end{cases} \tag{5}$$

The identity in the first term $h(x) = x$ is closure preserving with respect to every part x of $C - t(A)$; the second term, $h(x) = x - t(A)$ is the same as previously. Mapping (5) along with Formula (3), give the following closure equations that describe how closed parts in the topologies of C and C' in Fig. 8a are related whenever the rule applications are continuous:

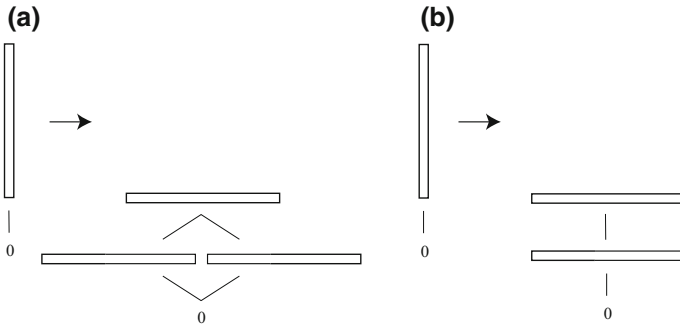


Fig. 9 Topological decomposition where **a** $t(A)$ and $C - t(A)$ are closed in C , and **b** $t(A)$ is closed in C and $t(B)$ is closed in C'

$$\gamma(x) = \begin{cases} (C - t(A)) \cdot \gamma'(x), & \text{if } x \leq C - t(A) \\ t(A) + \widehat{C} \cdot \gamma'(x - t(A)), & \text{otherwise} \end{cases} \quad (6)$$

The different topologies assigned on shapes in the sequence C_0, C_1 , and C_2 are in effect (re)structuring the shapes in the definition of the rules used to generate them. For example, in Fig. 8a, when the rule is applied on shape C_0 to create shape C_1 , the topology implied on the right shape $t_0(B)$ is a Boolean algebra made of two atoms; one atom contributes to the formation of the emergent rectangle in C_1 , and the other to its complement. This topology is shown separately in Fig. 9a. Similarly, in Fig. 8b, when the rule applies to shape C_0 to create shape C_1 , the topology implied on the right shape $t_0(B)$ includes one extra piece, which is created in the product $t_0(B) \cdot t_1(A)$. This topology is shown separately in Fig. 9b. While shapes in shape rules come initially undivided, they acquire descriptions (topologies) as a consequence of how they are used to generate shapes and according to how their actions are interpreted retrospectively.

As new shapes are added in a computation history, one is required to work backwards from the end all the way to the beginning and make appropriate adjustments to topologies already assigned. This exposes the atoms needed to describe all shapes in the recorded history “thus far” in a continuous manner, in effect constructing a specification for a space which includes every description as a closed member. This can be illustrated in a nice way in longer computations where visual entities are restructured over time in an ongoing manner. In Figs. 10 and 11, continuous topologies are specified rule after rule for the complete sequence C_0, C_1, C_2, C_3, C_4 (Fig. 1); $t(A)$ and $C - t(A)$ are kept closed in C , $t(B)$ and $C' - t(B)$ are kept closed in C' . The drawings are organized in the horizontal direction to emphasize forward continuity and the backwards restructuring of descriptions of shapes. Topologies are assigned

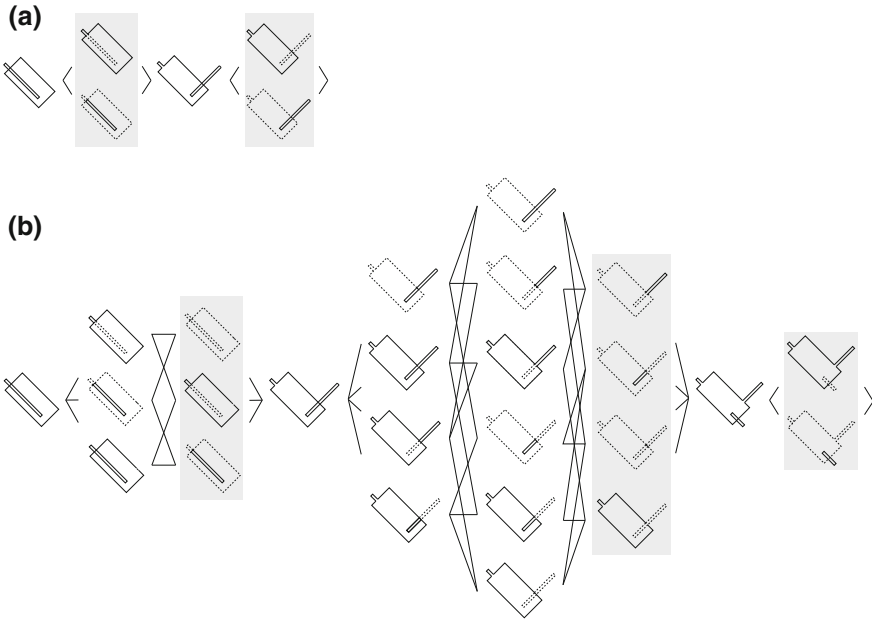


Fig. 10 Continuous topologies for **a** $C_0 \Rightarrow C_1$, and **b** $C_0 \Rightarrow C_1 \Rightarrow C_2$. Atoms that make up descriptions of shapes are shown with grey planes

to shapes after every rule application: first for step $C_0 \Rightarrow C_1$ in Fig. 10a; then for step $C_0 \Rightarrow C_1 \Rightarrow C_2$ in Fig. 10b; next for step $C_1 \Rightarrow C_2 \Rightarrow C_3$ in Fig. 11a, b.⁴

The breaking up of the computation into consecutive series helps to distinguish (momentarily) fixed sets of atoms that drive the rule applications. For every lattice diagram in Figs. 10 and 11, the corresponding set of atoms is highlighted with a grey plane. These atoms build up each topology in a combinatorial manner—the lattices provide instructions for how each closed shape can be assembled piece by piece in terms of these atoms. As an example, in the lattice in Fig. 11a, shape C_1 has a topology due to the rule applications up until the creation of shape C_4 . This lattice can be regarded as a collection of nested Boolean algebras that are stacked in increasing size. Let the atoms of this topology be represented as the collection of one element sets $\mathcal{A}_1 = (\{a_i\} \mid i = 1, 2, \dots, 5; 0 \leftarrow a_i)$ (the arrow \leftarrow means a_i “covers” the empty shape) and let $|A|$ be the cardinality of this set. The set of all possible combinations we can obtain with these atoms corresponds to the powerset of A with cardinality $32 (2^{|A|})$. \mathcal{A}_1 constitutes a basis that generates the topology on C_1 .

The particular atoms recognized in the topologies of Figs. 10 and 11 are implied in the application of rules. Likewise, rules are also restructured as a consequence of how they are applied. More specifically, Fig. 12 shows the structures implied on

⁴The last rule application $C_3 \Rightarrow C_4$ is omitted since the resulting topologies would make the drawings of the lattices significantly large.

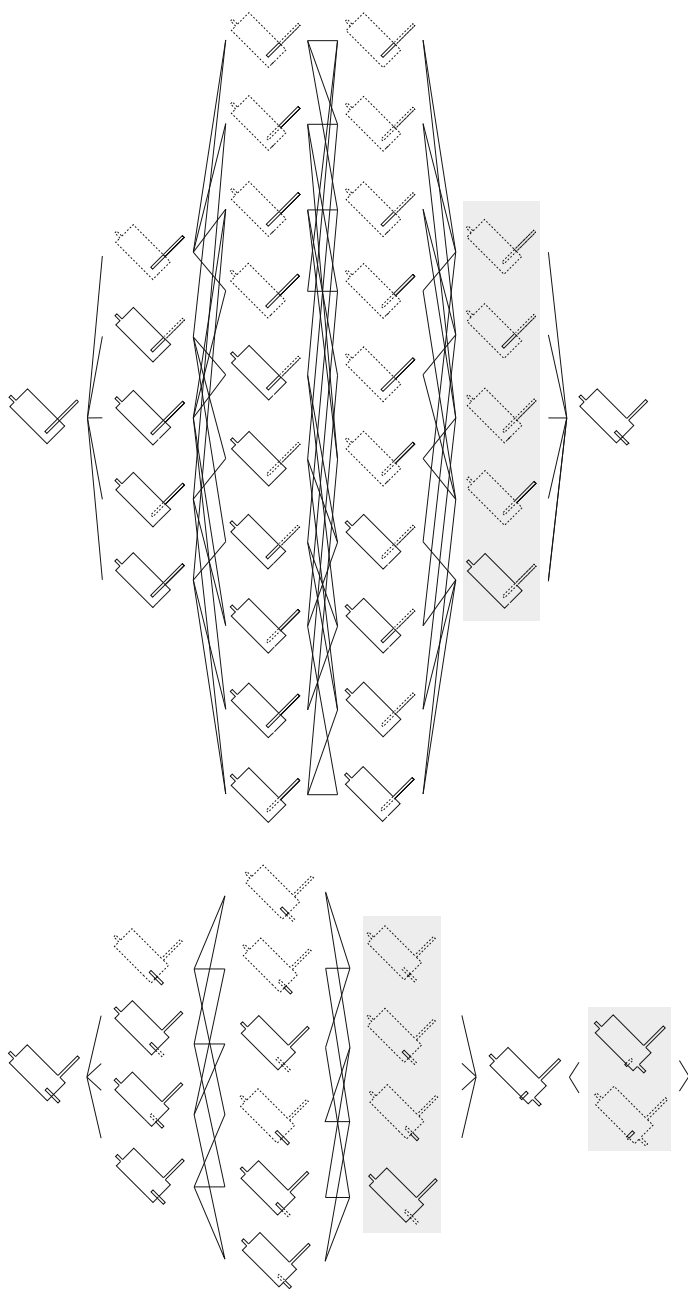


Fig. 11 Continuous topologies for $C_1 \Rightarrow C_2 \Rightarrow C_3$. Atoms that makeup descriptions of shapes are shown with grey planes

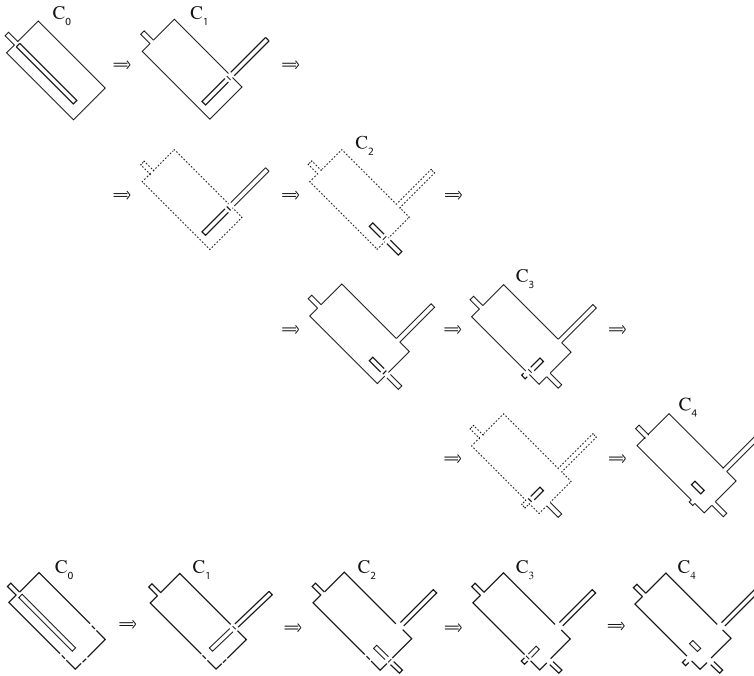


Fig. 12 Analysis of structures assigned on the left and right shapes for each rule used to generate the shapes in the sequence C_0, C_1, \dots, C_4

the left and right shape of the rules used to construct the shapes in Figs. 10 and 11. This fluid, back and forth from rules and the descriptions they imply on shapes shows that while schemas provide general mechanisms for defining rules on the go, the rules themselves are specific only to the shapes they apply. At the same time, it also shows that shapes in rules acquire descriptions as a consequence of their use over time. Shapes in shape rules and shapes generated *by* these rules are restructured reciprocally in the course of the computation.

It is worthwhile to consider analogous situations in some standard approaches to computation in design. For instance, in a parametric design space divisions of shapes must be known in advance so that all future computations are done without breaks or inconsistencies with respect to how shapes are described. Such systems require designers to have foreknowledge of the structure underlying each possible design in the space. For shapes recorded unanalyzed in a computation history, this structure is derived retrospectively by analyzing conditions for continuous mappings between topologies for shapes. The bottom series in Fig. 12 essentially shows what divisions are needed to begin with in order to run the computation forward with analyzed shapes that have sufficient commonalities between their topological structures. The indifference to atoms fixed from beforehand also distinguishes shape rules from rules defined in production systems where preconditions and conclusions for each rule

consist of sets of atoms defined ahead of time [4, 5, 13]. When shape rules are used in a creative work, we can instead leave their preconditions and conclusions without description and fix them in the course of their use over time. This facilitates structure and change and provides for a much more natural framework for studying how designers engage with materials of computation (here shapes and their perceptible parts) and how descriptions emerge as byproducts of this engagement.

Discussion

The study of how descriptions map continuously from one shape to another as a consequence of the rules used in the course of a computation provides a neat basis for crafting computation histories—and consequently specifications of design spaces—in an ongoing fashion. The notion of a design space becomes in this way a device for recapitulating what we do on the spot, as opposed to a device for prescribing a future of closed possibilities, as in the classical meaning of the term. This view of design spaces presents new directions for the formal study of the design process.

Design Spaces of Things

The emphasis in the eye and the hand of a human designer who calculates (applies rules) on a piece of paper that we often see in the literature of shape grammars serves a slower, but real-time action-oriented and visual approach to calculating that happens outside, in the real world. A drawing done by an architect or designer on a piece of paper, in a conventional way, leaves its own physical “mark,” namely, the drawing itself. Likewise, shapes generated with a shape grammar are concrete visual–physical drawings whose descriptions depend on the particular rules used—we have the ability to calculate without them being analyzed from beforehand. This paradigm of computing directly with the object of interest as opposed to having a description of it aims toward a very different direction than classical approaches to the specification of design spaces.

The paradigm then can be extended further to consider the use of an improvisational specification of design spaces in creative processes where the materials of computation aren’t only shapes made of linear elements but other concrete things that may still exhibit or preserve the properties of shapes. There exist shapes with weights, such as color or material, and can be used for the improvisational specification of spaces made of weighted shapes, such as during a painting process or in the early stages of an architectural studio project where students experiment with shapes and different materials in an improvisatory manner. This would require more work in the context of decompositions and rule continuity for different definitions of weighted shapes and their corresponding algebras. Further, research aimed at understanding making processes through computation may benefit from this particular direction. As

[14] reminds us, “materials” are the kinds of things you make with; making (creating) with shapes is just one kind of making. Whatever the choice of “things” might be for a particular area of interest—shapes, watercolors, sounds, biological materials, and physical materials—algebras can be defined in terms of these things, at minimum, by working out the arithmetic operations for calculating members of the algebra and the part relation (embedding) in rules to drive creation (computation) [5]. For example, the algebras U_{ij} , V_{ij} , and W_{ij} for shapes and generalizations of them are suggestive for how one may proceed with this endeavor.

An interesting question is how to cast an improvisational specification with shapes to an improvisational specification with things beyond shapes. What further algebras are needed to study continuity of rule applications when calculating with things? What kind of decompositions and mappings appear that are relevant when rules are specified in terms of not only the shape of objects but their physical-material presence, too? Studies toward these directions may take the present work as an entry point in order to extend the notion of a design space from a space of latently existing abstractions to spaces that are made by a designer in the moment of the creative action and are made of concrete things like shapes rather than abstract symbol representations.

Improvisation and Design Spaces in Retrospect

The term improvisation is used here in a metaphorical sense. It expresses the basic idea underlying the approach toward the specification of design spaces that is taken in this paper: design spaces can be considered as open-ended constructions, by working out the space that captures the results (in this case, generated shapes) of a fluid process of creation *backwards*. This approach towards the specification of design spaces suggests an open-ended relationship between what a designer or composer does real-time on the spot, the description of that which is created (its compositional parts) and the description of the space in which it lies. It facilitates structure as well as imagination so that both can become elements of the same formal framework.

Seeing design spaces as retrospective constructions that emerge from what designers do in the moment of the creative action as opposed to constructions specified in advance and independently of them, should offer an alternative characterization of a central concept underlying many aspects of design theory and computation, namely, the concept of a design space. While the formal counterpart of this view is developed here in the context of visual computation with shapes, it should still offer an idea for how computation can reconcile the need for open endedness and backwards reflection in areas of design, such as architecture, the visual arts, and related areas of spatial design, with the traditional, formalistic pursuit for “capturing” that which is created in the terms of some specification of a design space.

References

1. Krstic D (2016) From shape computations to shape decompositions. In: Gero JS (ed) *Design computing and cognition '16*, Springer, Netherlands, pp 361–376
2. Stiny G (1994) Shape rules: closure, continuity, and emergence. *Environ Plan* 20:359–362
3. Stiny G (2006) *Shape: talking about seeing and doing*. The MIT Press, Cambridge
4. Nilsson NJ (1982) *Principles of artificial intelligence*. Springer, Berlin
5. Charidis A (2018) *Improvisational specification of design spaces*. Master's thesis, Departments of Architecture and Electrical Engineering and Computer Science, Massachusetts Institute of Technology
6. McKinsey JCC, Tarski A (1944) The algebra of topology. *Ann Math* 45:141–191
7. Cagan J, Campbell MI, Finger S, Tomiyama T (2005) A framework for computational design synthesis: model and applications. *J Comput Inf Sci Eng* 5:171–181
8. Fensel D (2000) Problem solving methods: understanding, description, development and reuse. In: *Lecture notes in artificial intelligence 1971*. Springer, Berlin
9. Radford D, Gero J (1998) *Design by optimization in architecture, building, and construction*. Wiley, USA
10. Woodburry RF (1991) Searching for designs: paradigm and practice. *Build Environ* 26:61–73
11. Gero JS (1993) Towards a model of exploration in computer-aided design. In: Gero JS, Tyugu N (ed) *Formal methods for computer-aided design*, North-Holland, pp 315–336
12. Stouffs R (eds) (2006) *Design spaces: the explicit representation of spaces of alternatives*. *Artif Intell Eng Des Anal Manuf Special Issue* 20(2)
13. Ligeza A (2006) Logical foundations for rule-based systems, 2nd edn. In: *Studies in computational intelligence*, vol 11. Springer, Berlin
14. Knight T (2015) Shapes and other things. *Nexus Netw J* 17:963–980

Design of Transfer Reinforcement Learning Mechanisms for Autonomous Collision Avoidance



Xiongqing Liu and Yan Jin

It is often hard for a reinforcement learning (RL) agent to utilize previous experience to solve new similar but more complex tasks. In this research, we combine the transfer learning with reinforcement learning and investigate how the hyperparameters of both transfer learning and reinforcement learning impact the learning effectiveness and task performance in the context of autonomous robotic collision avoidance. A deep reinforcement learning algorithm was first implemented for a robot to learn, from its experience, how to avoid randomly generated single obstacles. After that the effect of transfer of previously learned experience was studied by introducing two important concepts, *transfer belief*—i.e., how much a robot should believe in its previous experience—and *transfer period*—i.e., how long the previous experience should be applied in the new context. The proposed approach has been tested for collision avoidance problems by altering transfer period. It is shown that transfer learnings on average had ~50% speed increase at ~30% competence levels, and there exists an optimal *transfer period* where the *variance* is the lowest and learning *speed* is the fastest.

Introduction

Collision avoidance is a common research topic in many industrial fields. In the area of robotics, research has been focused on issues related to how vehicle robots avoid obstacles as well as each other [33] and how assembly robots avoid interferences among its own arms or with those of others [17]. In transportation, self-driving cars must be able to avoid obstacles and other vehicles in various situations [28]. In the shipping industry, collision avoidance can be highly difficult, when water areas are

X. Liu · Y. Jin (✉)
University of Southern California, Los Angeles, USA
e-mail: yjin@usc.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_17

congested, due to the large ship inertia causing immovability [14]. Airplane collision avoidance [43] and even the collision with debris in space [2] have become issues due to the increasing level of congestion.

Approaches to solving collision avoidance problems can be divided into two large categories, one is vehicle control system [23], relying on traditional control theories and intelligent systems approach, and the other is traffic system development. The recent progress in machine learning, especially deep learning [21], has opened the ways to developing systems that can learn from humans' operation experiences through supervised deep learning and from machines' own experiences through reinforcement learning. The reinforcement learning approach allows an agent to learn from its experience. By interacting with the environment, the agent learns to select actions at any state to maximize the total reward. In case of deep learning, e.g., AlphaGo [3], the agent learns from the experience of human experts and apply the learned skills to solving the problems in the same domain of the experts.

One common observation about the current machine learning systems, including AlphaGo, is that they can only function within the narrow domain of the tasks that they are trained to work for. This observation manifests the limited level of "intelligence" of the current systems.

In his seminal paper, March [24] examined the organizational learning in humans and presented various features of, and relationships between, the essences of human organization learning: exploration of new possibilities and exploitation of old certainties. Allocating resources to these two capabilities represents the adaptiveness of the human organization. Based on this insight, a machine's intelligence can be considered as composed of the machine's capabilities of exploration, exploitation, and its ability to regulate the "resource" allocation between the two. This basic idea has been implemented in our research at two different layers. First, the reinforcement learning itself is based on the exploration-exploitation of the learned knowledge (i.e., the agent's current neural network) and the random choices. Second, the transfer learning allows the agent to exploit the previously learned experience (i.e., an expert's neural network obtained from the previous task context) and explore the new task context through learning and exploration. The long-term goal of this research is to develop an integrated transfer reinforcement learning technique that allows agents to learn from multiple task domains and exploit the learned knowledge in new task contexts for more effective learning and better task performance.

In this paper, we focus on the robotic collision avoidance problem and investigate how transfer learning [30], in addition to deep reinforcement learning, can be applied to allow agents to exploit and explore across task contexts. In the following, Section "Related Work" provides a critical review of the relevant work in collision avoidance and machine learning and points out the gap in the literature. Section "A Transfer and Reinforcement Learning Approach" describes our proposed approach of transfer reinforcement learning in detail. Computer-simulation-based case studies are presented in Section "Case Studies" with the results being discussed in Section "Results and Discussion." Section "Conclusions and Future Work" draws the conclusions and points to future research directions.

Related Work

Collision avoidance problems have always attracted the attention of researchers in various fields: artificial intelligence, control theory, robotics, multi-agent system, etc. The traditional practice to achieve real-time obstacle avoidance was to create an artificial potential field [19]. Fahimi [10] proposed harmonic potential functions and the panel method to address multi-robot obstacle avoidance problem in the presence of both static and dynamic obstacles. Mastellone et al. [25] designed a controller for collision avoidance based on Lyapunov-type approach and demonstrated the robustness of the system when the communication between robots was unreliable. Keller et al. [18] designed a path planner for unmanned aircraft systems to provide surveillance by combining graph search and B-spline parametric curve construction, which could successfully navigate around obstacles and provide sufficient coverage. Tang and Kumar [35] proposed the OMP+CHOP algorithm for a centralized multi-robot system, which was shown to be safe and complete, but at the cost of optimality.

For collision avoidance algorithms to be more adaptive and flexible in real-world complex environment, learning capabilities of a multi-agent system have been developed. In recent years, deep learning has achieved tremendous success in various areas such as image recognition [20], speech recognition [15], automatic game playing [27], self-driving [1], and so on. Deep learning algorithms can extract high-level features by utilizing deep neural networks, such as convolutional neural networks (CNNs) [20], multi-layer perceptrons, and recurrent neural networks (RNNs) [21]. Scaling up deep learning algorithms is able to discover high-level features in a complex task. Dean et al. [7] constructed a very large system which was able to train 1 billion parameters using 16000 CPU cores. Coates et al. [5] scaled to networks with over 11 billion parameters using a cluster of GPU servers.

Mnih et al. [27] introduced deep learning algorithm using experience replay and CNNs to learn a Q function that can play various Atari 2600 games better than human players. Experience replay allows a learning agent to randomly sample batches from past experiences to update Q -values, thus breaking the correlations between consecutive frames. By combining supervised learning and reinforcement learning, a group at DeepMind has further proven that their deep learning algorithm can outperform a world champion in the most challenging classic game Go [3, 34]. Schaul et al. [32] further developed a prioritized experience replay framework to sample more important transitions and learn more efficiently.

Chen et al. [6] developed a decentralized multi-agent collision avoidance algorithm based on deep reinforcement learning. Two agents were simulated to navigate toward their own goal positions and learn a value network which encodes the expected time to goal, and the solution was then generalized in multi-agent scenarios. Deep learning algorithms have been successful in achieving end-to-end learning. Dieleman and Schrauwen [8] applied feature learning directly to raw audio signals by training convolutional neural networks. The results showed that the system learns automatically frequency decompositions and feature representations from raw audio.

Self-driving is a promising field that heavily relies on the advances in deep learning. Since self-driving cars always require a great deal of expensive and complex hardware, Yu et al. [42] implemented a deep Q learning algorithm using dataset (images) from real-time play of the game JavaScript Racer. In a recently published paper [1], a convolutional neural network is trained to map steering commands directly from raw pixels from camera input. This end-to-end learning approach is challenging in that it requires huge number of inputs and the advantage is that it releases the reliance on the designer's prior domain knowledge.

Transfer learning refers to utilizing knowledge gained from source tasks to solve a target task. In a reinforcement learning context, transfer learning can potentially speed up the learning agent to learn a new but related task (i.e., target task) by learning source tasks first. Taylor and Stone [36] introduced a transfer algorithm called *rule transfer*, which summarizes source task policy, modifies the decision list, and generates a policy for the target task. Rule learning is well understood and human readable. The agent benefits from the decision list initially and continues to refine its policy through target task training. It was shown that *rule transfer* can significantly improve learning in robot soccer using learned policy from a grid-world task.

Fernandez and Veloso [11] proposed two algorithms to address the challenges of *policy reuse* in a reinforcement learning agent. The major components include an exploration strategy and a similarity function to estimate the similarity between past policies and new ones. The PRQ-learning algorithm probabilistically biases an exploration learning process using a *policy library*. In the second algorithm called PLPR, the policy library is created when learning new policies and reusing past policies.

Torrey [37] introduced the induction logic programming for analyzing previous experience of source task and transferred rules for when to take actions. Through an advice-taking algorithm, the target task learner could benefit from outside imperfect guidance. A system AI^2 (Advice via Induction and Instruction) for transfer learning in reinforcement learning was built, which creates relational *transfer advice* using inductive logic programming.

In transfer learning within deep neural networks, a base network on a base dataset and task is first trained, and the learned features are then transferred to the target network to be trained on a target dataset and task, commonly by copying the first n layers of the base network to the first n layers of the target network. A task-driven deep transfer learning framework for image classification was designed [9], where the features and classifiers are obtained at the same time. Parisotto et al. [31] proposed a transfer reinforcement learning approach (Actor-Mimic) to mimic expert decisions for multi-task learning, which adopts the concept of *policy distillation* [16].

To date there has been little literature aiming to combine deep reinforcement learning and transfer learning to solve robotic collision avoidance problems, because (1) it is difficult to directly learn from raw pixel or distance sensory inputs and (2) it requires large amount of training data, which is not easy to generate in real life. This research aims to close the gap between real-world collision avoidance and deep learning by proposing a combined transfer and reinforcement learning approach to learn a new task more efficiently.

A Transfer and Reinforcement Learning Approach

Before moving into details of the mechanism for collision avoidance, we first introduce the basic idea and our overall goal of research on integrated machine learning for developing intelligent systems.

Reinforcement learning has the advantage of learning from the agent's own experience and the agent learns to choose actions at any state to maximize the total rewards by interacting with the environment. Although reinforcement learning allows agents to acquire collision avoidance skills [12, 13, 26], one challenge is that it requires a large amount of training data, which is usually hard to obtain in real life considering the cost of building physical systems and conducting experiments.

On the other hand, recent progress in self-driving car research [1, 29] and deep learning, e.g., AlphaGo [3, 4, 39] have demonstrated that the experience of human "experts" represents a highly valuable source of intelligence and can be learned by machines through deep learning. However, in many situations, the access to human expertise data can be very limited, since it is difficult, if not impossible, to acquire human experience data in all possible situations. How to effectively and efficiently combine human expertise with machine self-learning remains to be a challenge.

In this research, we consider that a machine's "intelligence" is dependent on three fundamental capabilities given below:

- First, it must be able to "exploit" the existing knowledge or expertise to the maximum extent so that all the known situations can be dealt with. This capability corresponds to transfer learning at a macro-scale and deep learning mechanisms at a micro-scale.
- Second, the machine must be able to "explore" the unknown territories and develop new knowledge or expertise from its own experience. Reinforcement learning is a candidate for this capability.
- Lastly, depending on the level of dynamics of the task domain and environment, the machine must be able to "adapt" the ratio of exploitation over exploration in order to stay effective. More dynamic or changeable domains require more exploration. Human design or meta-level learning mechanisms are needed to deal with this issue.

Our long-term goal is to develop an integrated machine learning technology that can (1) learn from multiple experts from diverse domains, (2) apply the learned expertise to explore new domains (e.g., requiring multiple domain expertise or more complex), and (3) manage its own learning processes (i.e., exploitation and exploration) according to the change in task domains. The "domains" can be knowledge domains, such as mechanical design, and technical domains, such as robotic (e.g., robot, car, ship) collision avoidance. Our current focus is on technical domains.

As the first step in the research, we seek to develop an integrated learning mechanism that can take advantage of existing steering experience from either humans or other robots to learn about actions in new and more complex situations. More specifically, we propose a *transfer reinforcement learning*, or TRL for short, approach built

on deep reinforcement learning algorithms. By combining the experience from the “expert,” the agent can reduce trial time and learn about more complex tasks faster.

Deep Reinforcement Learning

The attempt was made to use reinforcement learning algorithms to train the system so that it automatically learns to solve tasks only from sensory inputs and a scalar reward signal. However, it was difficult to collect the sufficient amount of data as the training input, especially in real life, by only relying on sensory inputs. In addition, the state–action space is always continuous which makes it impractical to build a look-up Q -table. To overcome the curse of dimensionality, deep neural networks are used as functional approximators to replace the Q -table and approximate Q -values.

We began this research by implementing the deep reinforcement learning algorithm with experience replay as proposed in [27]. We first consider standard Q learning [41] which can be formulated as a tuple of $\langle S, A, P, R, \gamma \rangle$. S is the state space, which consists of all the agent’s possible states in the environment. A is the action space consisting of all the possible actions that the agent can take. P is the transition matrix (usually unknown in a model-free environment), R is the reward function, and γ is the discount factor. At any given time t , the agent’s goal is to maximize its future discounted return $R_t = \sum_{t'}^T \gamma^{t'-t} r_{t'}$, where T is the time when the game terminates. Like many other reinforcement learning algorithms, the agent estimates at each time step the action-value function $Q(s, a)$, using Bellman equation as an update. Such value iteration algorithms converge to the optimal value function.

$$Q_{i+1}(s, a) = \mathbb{E} \left[r + \gamma \max_{a'} Q_i(s', a') | s, a \right]$$

In order to adapt to tasks involving infinitely large state/action space where building the Q -table is impractical, deep Q learning with experience replay uses a neural network as a function approximator (Q-network). A Q-network with weights θ_i can be trained by minimizing the loss function $L_i(\theta_i)$ at each iteration i ,

$$L_i(\theta_i) = \mathbb{E}[(y_i - Q(s, a; \theta_i))^2]$$

where $y_i = \mathbb{E} \left[r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) \right]$ is the target Q-network for iteration i . The gradient is calculated by the following:

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{s,a,r,s'} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i) \right) \nabla_{\theta_i} Q(s, a; \theta_i) \right]$$

The deep Q learning algorithm utilizes a technique called *experience replay* where the agents’ experiences, $e_t = (s_t, a_t, r_t, s_{t+1})$, are stored into a *replay memory*, $D =$

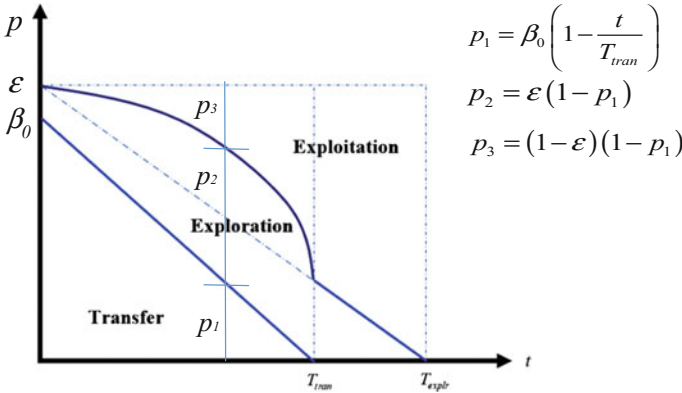


Fig. 1 Exploration/exploitation with transfer

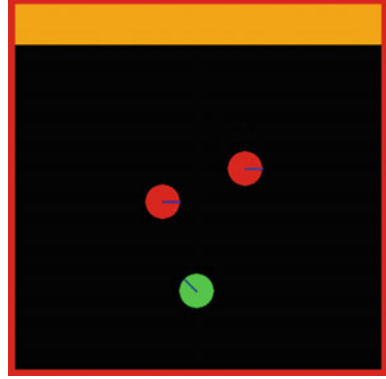
e_1, e_2, \dots, e_N (N is the capacity of the replay memory). Then, mini-batches are randomly sampled from D and applied to Q learning updates. The agent selects an action according to the ϵ -greedy policy.

Various approaches have been addressed to stabilize learning process, such as deep Q-network (DQN) [27], double DQN [38], and dueling DQN [40]. In this research, our base network is built by combining these three approaches.

Transfer Reinforcement Learning

The goal of transfer learning is to transfer “expert” knowledge into a learning agent (student) for new tasks which are more complex. The expert network is first obtained by training through the source task, and then used for initialization in the student’s network for the target task. In order to utilize the expert experience more efficiently, a new *transfer phase* is added to the traditional ϵ -greedy policy (Fig. 1), where the agent selects *transfer action* according to the expert network. The transfer action is defined as one of the **three** actions with the top **three** values of the expert network. This new policy is called ϵ_T -greedy policy, which is defined as the following:

- (a) *Transfer*: With probability $p_1 = \beta_0\left(1 - \frac{t}{T_{tran}}\right)$, the agent selects the transfer action—i.e., the action suggested by the expert neural network. T_{tran} shown in Fig. 1 is the *transfer period*, during which the agent is influenced by the expert network (transfer period is shorter than the exploration period T_{expl} , where ϵ is annealed close to 0.1); β_0 is the initial *transfer belief*, which measures how much confidence the agent puts in the expert knowledge.
- (b) *Exploration*: With probability $p_2 = \epsilon(1 - p_1)$, the agent selects a random action.

Fig. 2 Environment setup**Table 1** Agent actions

Action	v	ω
a_1	5	0.35
a_2	5	0.2
a_3	5	0.1
a_4	10	0
a_5	5	-0.1
a_6	5	-0.2
a_7	5	-0.35

(c) *Exploitation*: With probability $p_3 = (1 - \varepsilon)(1 - p_1)$, the agent selects the current best action produced by its own learned knowledge/network.

Agent Learning Behavior

A computer game environment was created to conduct case studies for transfer reinforcement learning for collision avoidance. The game environment consists of a learning agent (green), static obstacle (red), and a goal area (orange), as shown in Fig. 2. The simpler *source task* has only a single static obstacle, whereas the more complex *target task* always has two obstacles. The obstacle is randomly generated at the beginning of each collision avoidance episode.

- The state in the case studies is defined as the pixel values of the game window. Figure 2 shows an example.
- The action space is composed of seven actions, a_1 through a_7 , as indicated in Table 1.
- The reward function is defined as

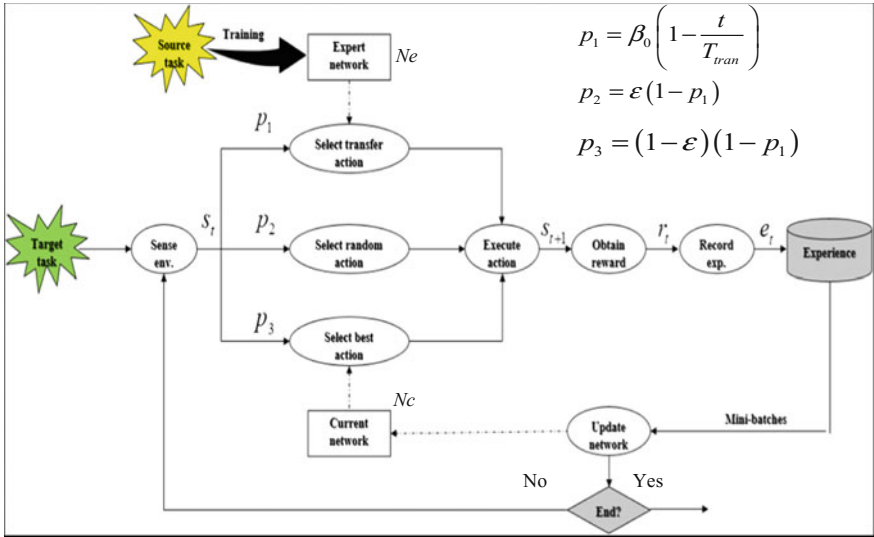


Fig. 3 Agent learning behavior

$$r = \begin{cases} 200 & \text{if reach goal position} \\ -900 & \text{if hit any obstacle} \\ -1 & \text{otherwise} \end{cases}$$

Figure 3 illustrates the proposed transfer reinforcement learning process. An expert network N_e is first obtained by training through the source task, which involves a single obstacle. In the target task, the agent follows ε_T -greedy policy to select actions with probabilities p_1 , p_2 , and p_3 as described in Section “Transfer Reinforcement Learning.”

After receiving a reward r_t from the environment, the agent stores the current experience e_t into the experience replay memory. The currently learned network N_c is then updated by sampling mini-batches from the experience replay, as shown in Fig. 3.

Case Studies

Collision Avoidance Game System Architecture

The collision avoidance game system consists of two modules: a visualization module (Pygame) and a machine learning module (TensorFlow). The visualization module creates graphical display for the system, where it reads the current environment state

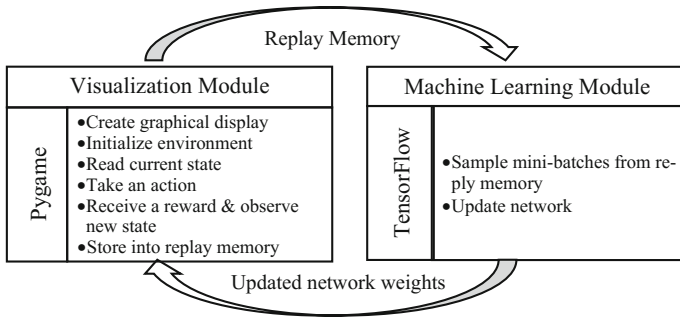


Fig. 4 Collision avoidance game system architecture

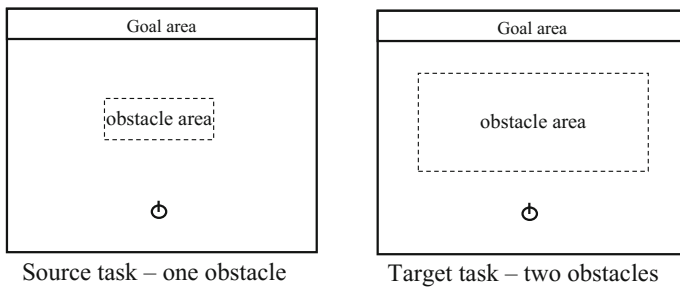


Fig. 5 Case study task situations

and simulates kinematics and dynamics. After taking some action, the agent will receive a reward, based on which a replay memory is constructed and sent to the machine learning module. TensorFlow deals with the heavy lifting to sample batches of experience and update the network weights, and then sends the updated weights back to the visualization module, as shown in Fig. 4.

Case Parameters

Two task situations are used for the case studies, namely, “Source task—one obstacle” and “Target task—two obstacles,” as shown in Fig. 5. As indicated in the figure, the source task has a smaller obstacle area and only one obstacle can randomly appear in the obstacle area. The target task situation, however, has a much wider obstacle area and there are always two obstacles that can appear in any random relative positions within the large obstacle area.

The network structure is the same as the original DQN paper [27] with an array of 84*84 pixels input and an output of seven actions. All our case studies were trained using Adam optimizer with a learning rate of 0.001. The discount factor γ is 0.99. The agent follows either ϵ -greedy policy in the source task or ϵ_T -greedy policy in

Table 2 Hyperparameters

	Source task One obstacle	Target task Two obstacles
Replay memory size	50,000	50,000
Mini-batch size	32	32
Discount factor	0.99	0.99
Learning rate α	0.001	0.001
Total training episodes	50,000	50,000
ε	1 \rightarrow 0.1	1 \rightarrow 0.1
Annealing frames	1 million	1 million
Transfer period (frames)	N/A	150 K/300 K/700 K/1 M
Initial transfer belief	N/A	0.9

the target task with ε annealed from 1.0 to 0.1 over the first 1 million frames, with 1 frame = 1 state. The replay memory consists of 50,000 recent frames, and 50,000 episodes were trained in total, with 1 episode = 1 game play. The transfer period could be the first 150, 300, 700 K, or 1 million frames. The choice of hyperparameters is summarized in Table 2.

Results and Discussion

After an expert network is obtained by training through the source task with single obstacle, the agent is then given a more complex target task, which has two obstacles and a larger obstacle field. The results of learning efficiency and effectiveness are shown in Figs. 6, 7, and 8. All the curves in Fig. 6 are the average of 10 learning performances by running 10 random seeds. The curves are also smoothed using exponential smoothing with the dumping factor set to 0.9. In all three figures, the unit of the x -axis is the number of 100 episodes. Each episode is defined as the period from agent starting movement to arriving at the goal, as shown in Fig. 5.

The colors in the figures indicate different lengths of the transfer period, which is measured in number of frames. For example, the blue line in Fig. 6 shows the performance of transfer learning with transfer period = 300 K frames. Roughly, 1 million frames = 115 (\times 100) episodes. The two red-colored baseline cases, discussed below, do not use ε_T -greedy decision policy.

The y -axis of Figs. 6 and 7 is the total reward value. Since the reward function is set to heavily penalize the collision with the obstacle and very small positive values are for reaching the goal, the final value of the total reward is close to zero. In Fig. 8, the y -axis shows the standard deviation of multiple learning runs at different number of episodes, measured as total reward value.

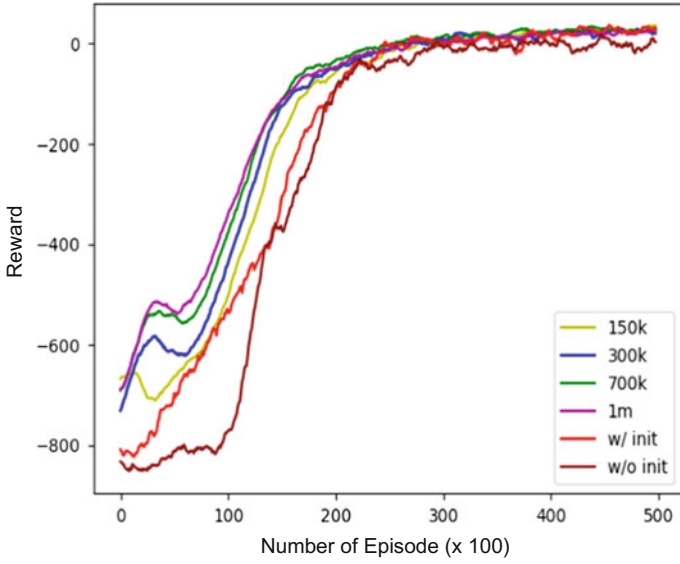


Fig. 6 Average learning performance of each transfer period

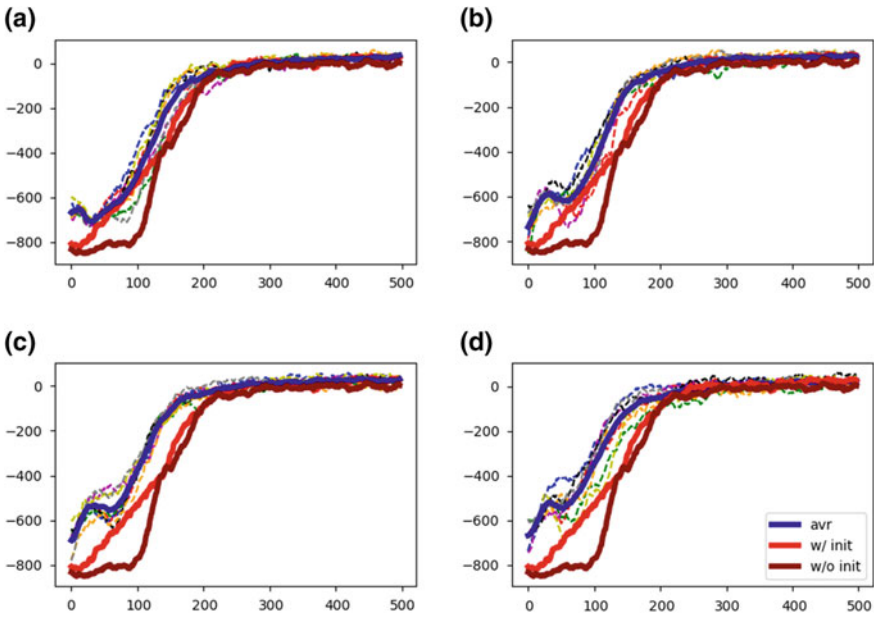


Fig. 7 Different transfer periods **a** 150 K frames, **b** 300 K frames, **c** 700 K frames, and **d** 1 million frames

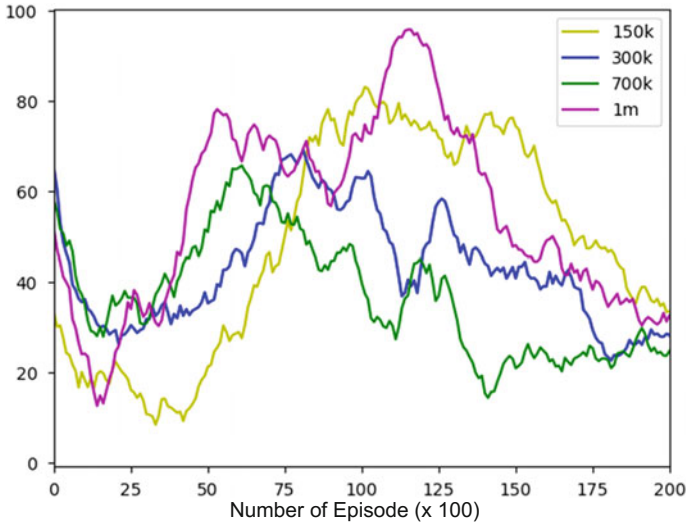


Fig. 8 Standard deviation plot of various transfer periods

Baseline Cases

For the purpose of comparison, we established two baseline cases. The first baseline case is for an agent to learn about the “target task—two obstacles” by “*bootstrap*”—i.e., the neural network is randomly initialized. The dark red lines shown in Figs. 6 and 7 indicate the learning performance of this baseline case. As the figures show, starting from scratch requires more time for the agent to learn about the task. Especially, it takes much longer training for the agent to become capable of dealing with the two-obstacle collision avoidance.

Another baseline case is “*copy expert*”—i.e., the weights of the expert network learned from the source task are copied into the learning agent as the initial neural network for the “target task.” After initialization, the agent starts its regular reinforcement learning: the copied expert network weights are updated by following the ε -greedy policy (i.e., ε starts from 1 and annealed to 0.1) to select actions. The red line shown in Figs. 6 and 7 indicates the learning performance of this baseline case.

Learning Speed

Baseline case *bootstrap*: As shown by the dark red line in Fig. 6, without any input from the expert knowledge, it takes much longer for the agents to learn about the *target task*. Huge lag appears until around 11 K frames point. However, it catches

up very fast after that point. The final learning effectiveness within the 50 K frames range is inferior.

Baseline case *copy expert*: In this case, as shown of the red color in Fig. 6, the starting point for the learning agent is a complete copy of the expert network. Since immediately after the learning process starts, the “expert network” will be updated by following ε -greedy policy, the “expert supervision” does not really exist. As a result of *copy expert*, the learning picks up faster than *bootstrap* case with almost the constant speed. We believe that the difference in learning speed between these two baseline cases indicates the level of similarity of the source task and target task domain. A detailed discussion of the similarities between source and target task domains will be presented in a separate publication [22].

Transfer reinforcement learning (TRL) cases with varying transfer period: Our primary simulation runs of TRL processes have revealed that the transfer period plays a key role in affecting learning speed. Figure 6 illustrates the learning performance of varying transfer period from 150, 300, 700 K, to 1 M frames with yellow, blue, green, and pink colors, respectively. As shown in Fig. 1, shorter transfer period T_{trans} means shorter period of *expert supervision*—i.e., to use expert network N_e to select actions (also see Fig. 3). From a learning speed point of view, the results in Fig. 6 indicate that longer transfer periods lead to better learning performance, with the effect diminishing as it becomes sufficiently long (after 700 K frames). When the transfer period is getting closer to 1 million frames—i.e., the annealing time when ε decreases to 0.1—the performance decreases. Comparing with the two baseline cases discussed above, the positive impact of *expert supervision* is considerably large, especially until the 200 K episodes range.

Learning Variance

In addition to learning speed, we identified the variance as an important measure of learning performance since in most intelligent engineering systems, the consistency of learning performance is very much demanded. Figure 7 illustrates the learning variance multiple learning runs with different transfer periods of first 150 K, 300 L, 700 K, and 1 M frames. Each color represents an independent trial. Each transfer period has 10 trials in total. The red curves are the two baseline cases. The standard deviation of each transfer period case before convergence (t from 0 to 200) is shown in Fig. 8. It can be seen that the variances of different transfer periods share a similar pattern: decreases at beginning, then increases, and finally decreases again as the learning converges. The width of the exploration (see Fig. 1) played a role in determining such a pattern.

A careful examination of Fig. 8 indicates that the overall variances are larger for both short transfer period case (150 K frames) and long transfer period case (1 M frames), while the 300–700 K transfer period cases appear to have less variance for different learning trials, exhibiting more consistent learning performance of the system. Further research is needed to investigate this interesting phenomenon.

Conclusions and Future Work

Collision avoidance is a common research topic in various industrial fields. Recent progress in machine learning has made it possible to train robots or agents to acquire collision avoidance knowledge. Although in the engineering research community, design is still focused on *static* and *dynamic*, *mechanical* and *structural* systems, future demands on intelligent engineering systems call for methods for designing *intelligent and learning* systems. In this research, we approach the problem of collision avoidance from an intelligent and learning systems perspective. By considering machine intelligence as the capabilities of exploitation and exploration together with adaptation, we developed a transfer reinforcement learning approach that can be tuned to exploit past experience of human experts and other robots and explore the new domain through deep reinforcement learning. Following is a summary of our findings.

- The proposed transfer reinforcement learning approach has been tested in a game environment and proved useful to solve similar complex collision avoidance tasks.
- The transfer period is a crucial component that needs to be adjusted. Our transfer learning scheme has two effects: learning *speed* and *variance*. Compared to the *bootstrap* case, the *copy expert* strategy performed better. Comparing with *bootstrap*, the transfer learnings on average had a ~50% increase at ~25% competence level and ~30% increase at 75% competence level. As transfer period increases, the learning speed increases. However, transfer period being too long may slow down the learning, but still faster than the baselines.
- The standard deviation plot shows that variance starts to decrease, and then increases, and finally decreases as learning converges. The longer the transfer period, the earlier the variance starts to increase. As learning proceeds, either short or long transfer period leads to high variance, whereas medium transfer period has low variance.
- There exists an optimal length of transfer period (700 K frames) when the variance is low and learning is fast. This optimal transfer period is believed to be task dependent, which is relevant to the inter-task similarity of source and target tasks.

Our ongoing research investigates task similarities and transfer strategies in transfer reinforcement learning (including varying transfer beliefs) and exploring multi-robot collision avoidance problems mixed with more complex fixed and moving obstacles.

This paper was based on the work supported by Monohakobi Technology Institute (MTI) and Nippon Yusen Kaisha (NYK). The authors are grateful to MTI and NYK for their support.

References

1. Bojarski M et al (2016) End to end learning for self-driving cars. arXiv: 1604.07316 [cs.LG]
2. Casanova D, Tardioli C, Lemaître A (2014) Space debris collision avoidance using a three-filter sequence. *Mon Not R Astron Soc* 442(4):3235–3242
3. Chen JX (2016) The evolution of computing: AlphaGo. *Comput Sci Eng* 18(4):4–7
4. Churchland PS, Sejnowski TJ (2016) *The computational brain*. MIT Press, Cambridge
5. Coates A, Huval B, Wang T, Wu D, Ng A (2013) Deep learning with COTS HPC systems. In: *International conference on machine learning*
6. Chen YF, Liu M, Everett M, How JP (2016) Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. arXiv: 1609.07845 [cs.MA]
7. Dean J et al (2012) Large scale distributed deep networks. In: *International conference on neural information processing systems*. Curran Associates Inc., New York
8. Dieleman S, Schrauwen B (2014) End-to-end learning for music audio. In: *IEEE international conference on acoustics, speech and signal processing*
9. Ding Z, Nasrabadi N, Fu Y (2016) Task-driven deep transfer learning for image classification. In: *IEEE international conference on acoustics, speech and signal processing*
10. Fahimi F, Nataraj C, Ashrafiun H (2009) Real-time obstacle avoidance for multiple mobile robots. *Robotica* 27(2):189–198
11. Fernandez F, Veloso M (2006) Probabilistic policy reuse in a reinforcement learning agent. In: *International joint conference on autonomous agents and multiagent systems*, vol 58, pp 720–727
12. Frommberger L (2008) Learning to behave in space: a qualitative spatial representation for robot navigation with reinforcement learning. *Int J Artif Intell Tools* 17(03):465–482
13. Fujii T, Arai Y, Asama H, Endo I (1998) Multilayered reinforcement learning for complicated collision avoidance problems. In: *Proceedings 1998 IEEE international conference on robotics and automation*, vol 3, pp 2186–2191
14. Goerlandt F, Kujala P (2014) On the reliability and validity of ship–ship collision risk analysis in light of different perspectives on risk. *Saf Sci* 62:348–365
15. Hinton G, Deng L, Yu D, Dahl GE, Mohamed A, Jaitly N (2012) A senior, deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Sig Process Mag* 29(6):82–97
16. Hinton G, Vinyals O, Dean J (2015) Distilling the knowledge in a neural network. arXiv: 1503.02531v1 [stat.ML]
17. Hourtash AM, Hingwe P, Schena BM, Devengenzo RL (2016) U.S. Patent No. 9,492,235. U.S. Patent and Trademark Office, Washington, DC
18. Keller J, Thakur D, Gallier J, Kumar V (2016) Obstacle avoidance and path intersection validation for UAS: a B-spline approach. In: *IEEE international conference on unmanned aircraft systems*, pp 420–429
19. Khatib O (1986) Real-time obstacle avoidance for manipulators and mobile robots. *Int J Robot Res* 5(1)
20. Krizhevsky A, Sutskever I, Hinton G (2012) ImageNet classification with deep convolutional neural networks. *Commun ACM* 60(2)
21. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. *Nature* 521(7553):436–444
22. Liu X, Jin Y (2018) Transfer reinforcement learning: task similarities and transfer strategies (in preparation)
23. Machado T, Malheiro T, Monteiro S, Erhagen W, Bicho E (2016) Multi-constrained joint transportation tasks by teams of autonomous mobile robots using a dynamical systems approach. In: *2016 IEEE international conference on robotics and automation (ICRA)*, pp 3111–3117
24. March JG (1991) Exploration and exploitation in organizational learning. *Organ Sci* 2(1):71–87
25. Mastellone S, Stipanovic D, Graunke C, Intlekofer K, Spong M (2008) Formation control and collision avoidance for multi-agent non-holonomic systems: theory and experiments. *Int J Rob Res* 27(1):107–126

26. Mataric MJ (1997) Reinforcement learning in the multi-robot domain. In: Robot colonies. Springer, US, pp 73–83
27. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing Atari with deep reinforcement learning. arXiv:1312.5602v1 [cs.LG]
28. Mukhtar A, Xia L, Tang TB (2015) Vehicle detection techniques for collision avoidance systems: a review. *IEEE Trans Intell Transp Syst* 16(5):2318–2338
29. Ohn-Bar E, Trivedi MM (2016) Looking at humans in the age of self-driving and highly automated vehicles. *IEEE Trans Intell Veh* 1(1):90–104
30. Pan SJ, Yang Q (2010) A survey on transfer learning. *IEEE Trans Knowl Data Eng* 22(10):1345–1359
31. Parisotto E, Ba JL, Salakhutdinov R (2016) Actor-mimic: deep multitask and transfer reinforcement learning. arXiv:1511.06342v4 [cs.LG]
32. Schaul T, Quan J, Antonoglou I, Silver D (2016) Prioritized experience replay. *International Conference on Learning Representations*, 2016
33. Shiomi M, Zanlungo F, Hayashi K, Kanda T (2014) Towards a socially acceptable collision avoidance for a mobile robot navigating among pedestrians using a pedestrian model. *Int J Soc Robot* 6(3):443–455
34. Silver D et al (2016) Mastering the game of Go with deep neural networks and tree search. *Nature* 529(7587):484
35. Tang S, Kumar V (2015) A complete algorithm for generating safe trajectories for multi-robot teams. In: *International symposium on robotics research*
36. Taylor M, Stone P (2007) Cross-domain transfer for reinforcement learning. In: *International conference on machine learning*, ACM
37. Torrey L, Shavlik J, Walker T, Maclin R (2006) Skill acquisition via transfer learning and advice taking. In: *European conference on machine learning*. Springer, Berlin
38. van Hasselt H, Guez A, Silver D (2015) Deep reinforcement learning with double Q-learning. arXiv:1509.06461v3 [cs.LG]
39. Wang FY, Zhang JJ, Zheng X, Wang X, Yuan Y, Dai X, Zhang J, Yang, L (2016). Where does AlphaGo go: from Church-Turing thesis to AlphaGo thesis and beyond. *IEEE/CAA J Automatica Sin* 3(2):113–120
40. Wang Z, School T, Hessel M, van Haselt H, Lanctot M, de Freitas N (2016) Dueling network architectures for deep reinforcement learning. arXiv:1511.06581v3 [cs.LG]
41. Watkins C (1989) *Learning from delayed rewards*. Doctoral dissertation, University of Cambridge, Cambridge
42. Yu A, Palefsky-Smith R, Bedi R (2016) Deep reinforcement learning for simulated autonomous vehicle control
43. Zou X, Alexander R, McDermid J (2016) On the validation of a UAV collision avoidance system developed by model-based optimization: challenges and a tentative partial solution. In: *2016 46th annual IEEE/IFIP international conference on dependable systems and networks workshop*, pp 192–199

Part V
Design Cognition—Design Behaviors

Building a Social-Cognitive Framework for Design: Personality and Design Self-efficacy Effects on Pro-design Behaviors



Hristina Milojevic and Yan Jin

The purpose of this work is to offer a framework that analogously considers factors significant for engineering design and industrial organization, borrowing from literature in domains of cognition and social theories. We conducted two studies: at Shanghai Jiao Tong University and University of Southern California, that allowed us to investigate personal, environmental, cognitive, and behavioral traits and processes, as affected by design self-efficacy, in engineering designers and non-technical designers in training. Through a social-cognitive framework for design, we explore the kind of influence that occurs among person, environment, and behavior reciprocally. We found that the rational mode of thinking was particularly highly associated with design self-efficacy, and intuitive mode particularly insufficiently associated with design self-efficacy. Design self-efficacy was further positively associated with big five personality conscientiousness, and highly negatively associated with neuroticism, where some significance is seen in specific correlations with design self-efficacy in personality domains. The comprehensive findings call for a repetition study and further theoretical considerations for findings in the framework's domain.

A Social-Cognitive Take on Design Creativity

The previous research of the authors had studied design creativity from a standpoint of idea generation and exploration (e.g., [9]), creative stimulation (e.g., [18]), and collaborative stimulation [26], largely focusing on more than one designer. While the research thus far had focused on observing how design thinking and operation processes occur and how various patterns of such thinking processes impact design outcomes, little attention was paid to identifying various influencers that contribute

H. Milojevic · Y. Jin (✉)
University of Southern California, Los Angeles, USA
e-mail: yjin@usc.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_18

323

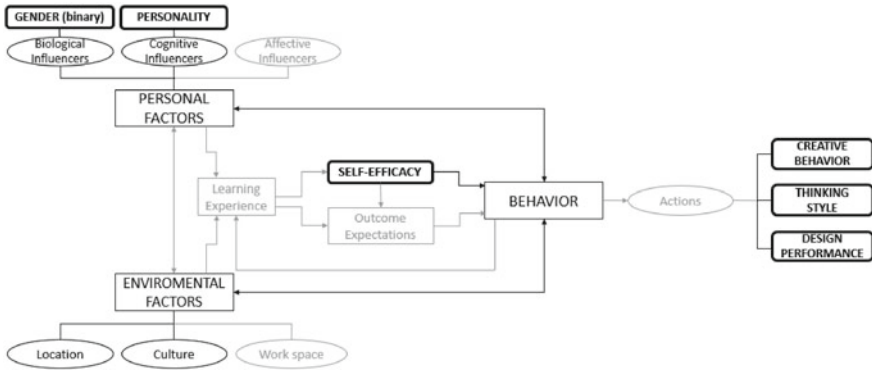


Fig. 1 Framework for building social-cognitive design perspective, studied partially, with respect to personality as a single personal influencer on design self-efficacy, and behavior per effects of design self-efficacy

to the formation of the designers' thinking and operation behaviors. The larger scope of present research focuses on the designer as an individual, treating their cognition, behavior, environment, competences, motivation, actions taken towards completing design-related tasks, and their own design outcomes, as a system of interest. More specifically, we introduce a concept called *pro-design behavior* to indicate the largely habitual *thinking* and *doing* behaviors that potentially lead to higher design creativity and better design performance. Pro-design behavior involves thinking style, creative behaviors, and design performance, later depicted in Fig. 1. A general research question to be addressed is: “*what are important influencers that shape someone's more pro-design behaviors?*”

Limiting the research system of interest steadily to an individual designer, there are fewer ways to conduct research interventions. While one might be able to displace an engineer into a new environment, placing them on, for example, a particularly crafted team of designers would not be an intervention of interest. As such, one of the larger goals of this research is to identify and propose an intervention that would allow for designer's most effective use of their dual process thinking [14, 31] behind creative design processes.

Early on, the project began with an outlook on proposing a duality to thinking behind creative engineering design. One way to do so was to rely on Epstein's cognitive-experiential self-theory [14], which proposes human mind as governed by two modes of processing: (i) rational (need for cognition), and (ii) experiential (faith in intuition). The preliminary results indicated that in order for one to be creative and demonstrate creativity with design outcomes, he or she must be approximately equally rational and experiential in their thinking [23]. In this case, the research remains within the domain of pure cognition.

In addition to the cognitive-experiential self-theory [14, 33], which aims to study humans from a spectrum across rational and intuitive thinking, the dual-process theory [31] closely compares in its division onto implicit and explicit processes, with the classification emerging based on the level of consciousness each process carries [15].

Investigating potentially important influencers requires expanding the scope of study on both mental and social horizons by including more aspects into consideration. Some social and mental aspects could be *personal*, such as gender, height or weight, or personality traits. Others could be *environmental*, such as the country or town one lives in, the type of culture they possess, or the type of space they spend their days in. Lastly, they could be *behavioral* and involve habits or actions.

These three social and mental categories are known as *influencers* in studies of social, social-cognitive, and social learning theories [2, 4, 5]. Within the influencers that pertain to design creativity, some useful allocations involve

1. *Personal* influencers: gender, personality
2. *Environmental* influencers: country of residence, professional and academic culture
3. *Behavioral* influencers: thinking styles, behavioral creativity, design performance.

While the three categories of influencers have mutual effects among themselves, the central variable that affects all three, and being affected by them, is *self-efficacy*, defined as “the belief that one can master a situation and produce positive outcomes” [3]. Considering self-efficacy is not a field-uniform measure, we study the effects of *design self-efficacy* in this particular case [7]. Self-efficacy scales for many different processes have either been published and opened up for use or can be self-made [2]. Carberry [7] relied on a Massachusetts science and technology/engineering curriculum framework and identified the eight steps of a design process for design self-efficacy estimate [21].

Model: A Social-Cognitive Framework for Design

Prior research efforts of the authors had generated a design thinking styles framework [22], demonstrating relationships between *thinking style* [14, 24] as a class of independent variables, and three other classes of dependent variables: *personality* [16, 33], *behavioral creativity* [30], and *design performance* [19, 28]. The framework demonstrated a significant and consistent correlation between rational thinking and the creative class of variables. The design thinking style framework was created in basic terms in order to initiate a study of dual thinking processes for early-stage engineering design and further explore the role of *perspective taking* in idea generation in engineering design [17, 20]. One direction is to study influencers accessible to a designer. Detecting, studying, and analyzing sets of *influencers* [8, 25] accessible to a designer, serve the greater goal of proposing new training methods and supporting tools for engineering designers, aimed to make them think in a manner best suited for their available design task [10].

To further explore ways duality of thinking could be built upon towards an engineering design duality of processing, in cognitive or practical domains, the relationship between the designer’s performance, e.g., creative [8], or professional [27], and

the designer's social environment should be considered. Thus begins the exploration of various social theories in domains of psychology and organization.

Based in social-theory driven studies of creativity [8, 27], organization [6], or design [1], the concepts of *motivation* and *self-efficacy* embedded in particular domains (e.g., creative domain, design domain, and learning domain) quickly emerge as the most considered and least defined. Hence, the research briefly abandons its consideration for specific domains, exploring most purely how one learns the social-cognitive rules and adopts beliefs about oneself.

The process of learning is commonly defined as a change, in cognition, behavior, or competence. This change can be continuous [29], persisting [13], or relatively permanent [32], according to different definitions. In this study, we adopt the definition of learning as a *relatively permanent change caused by an experience or action*. This change can occur within particular domains of interest, e.g., cognitive, behavioral, and constructivist. Ultimately, one is capable of learning in very many ways. The specific ways of interest are social-cognitive learning, self-regulated learning, and cognitive apprenticeship learning. Each of these learning strategies can be analyzed in social-cognitive theory (SCT) and social-cognitive learning theory (SCLT) terms. The social theories commonly share the triadic reciprocity (Fig. 1) in a form similar to the original triad proposed [5]. An example of such related triad is a visual representation of Cognitive Apprenticeship model [11]. The triadic model communicates reflexive affects between *personal*, *environmental*, and *behavioral* factors. When considering the effect of a person on the environment and their behavior, it occurs by understanding and observing their environment, as well as adjusting behavior for that expected to yield a positive outcome.

The process of triadic social-cognitive influencing is closely related to self-regulated learning, self-management, and self-efficacy. Self-regulation involves *self-monitoring*, *self-judgement*, and *self-reaction*. While these concepts will not be integrated in the social-cognitive design framework, they are the drive-concepts that make self-efficacy scoring in the form of a scale accessible [34].

In order to form the model proposed in Fig. 1, titled Social-Cognitive Framework for design, proposing the SCT triad with attributes adequately assigned to the three main factor categories, would suffice. However, in order to ensure the model is being understood from its affective standpoint, we rely on the expanded, social-cognitive career theory (SCCT), driving concepts, such as learning experiences, outcome expectations, and actions, while self-efficacy remains present for all social-cognitive domains [27].

Personal factors are intrinsic to a person within the social-cognitive framework, and divided into *biological* (assigned at birth), *cognitive*, and *affective* (changes in cognition). In this case, the *personal factors* studied will be biological (gender/sex) and cognitive (Big Five Personality). The environmental factors studied are culture (discipline) and country (location). Finally, the behavioral factors studied are Creative Behavior (*biographical creativity*, *behavioral creativity*, and *domain creativity*), Thinking Style (*rational* and *intuitive*), and Design Performance (*novelty* and *usability*), as indicated in Fig. 1.

Methods: Assessing Design Self-efficacy and Its Effects on Pro-design Behaviors

The proposed framework of social-cognitive framework for design (Fig. 1) is an expansive triad of personal, environmental, and behavioral influencers, which constantly drive one another, drive and are being driven by design self-efficacy, and offer the potential for further propositions of categorical and relational development within. Considering it is an early stage emergence from bringing social, learning, career, and cognitive theories into the realm of design in engineering and interdisciplinary domains, the social-cognitive framework for design can be unveiled into a more intricate theoretical framework driving a more intricate set of outcomes caused by pro-design behaviors of higher complexity. For purposes of this preliminary study, however, the framework is kept at little to no deviance from the Bandura-proposed social-cognitive triad, with categorical attributes assigned to each influencing category, so as to offer the greatest insight into the social-cognitive effects on engineering design, in domains of design cognition and design outcomes, with a potential for application in industrial organization, methodology creation, and artificial intelligence developments.

The research behind the social-cognitive design framework aims to compare design self-efficacy based on its characterization by sets of influencers assumed as mutually exclusive, and, in this case also binary. For example, the concept of *Gender* is assumed as gender binary, either female or male, contrary to the adopted view that gender identity and expression may transcend the binary biological sex [12]. The other two influencers were named *Country* and *Culture*, and are also proposed as binary, in order to define, respectively, the geographic location of the subjects studied (the United States or China) and the academic culture subjects identify and professionally growing in (Engineering or Non-Engineering).

Following suitable framework developments, the following hypotheses were formed, for purposes of this study.

- **H1:** Design self-efficacy will reflect differences within attributes to SCT triadic model's influencers studied: gender, location, culture, and personality.
- **H2:** High design self-efficacy scores are associated with high intuitive thinking scores.
- **H3:** High design self-efficacy scores are associated with high behavioral creativity scores; high design self-efficacy scores are also associated with high design performance scores.

Subjects

Total of 60 students, pursuing coursework in engineering, design, or both, participated in the study, from their home universities of the University of Southern California (Los Angeles, USA) and Shanghai Jiao Tong University (Shanghai, China). The

sample gender distribution was 18 female students (30%) to 42 male students (70%). Majority of the sample (75%) was based in China, consisting of 45 students, while the remaining 25% consisted of 15 students based in the United States. All were undergraduate students, distributed across class years: 31 students of the first year (51.7%), 10 students of the second year (16.6%), 3 students of the third year (5%), and the remaining 16 students of the fourth year (26.7%). Majority of the sample identified as an engineering student, 46 out of 60 (77%), and 24 (23%) were pursuing a variety of majors, and referred to as the non-engineering students, in this study. Per location, sample based in China had 33.3% female and 66.7% male students, 68.9% of engineering and 31.1% of non-engineering students. The sample based in the U.S. had 20% of female and 80% of male students, and was entirely comprised of students in mechanical and aerospace engineering. The U.S. sample yielded one-quarter of the entire sample, while the Chinese sample yielded the remaining three quarters.

Assessment Procedures

All students were asked to complete the following surveys: rational-experiential inventory (REI), big five personality inventory (BFI), biographical inventory of behavioral creativity (BICB), creative behavior inventory (CBI), and revised creative domain questionnaire (CDQ-R), as well as the design self-efficacy survey, which were then considered in the context of students' social-cognitive influencers.

Rational-experiential inventory (REI) is a measure of thinking style preferences, for **rational** (need for cognition) or **experiential** (faith in intuition) mode of processing in thinking [33]. *Big Five Personality Inventory (BFI)* is a measure of personality, commonly used in psychological and psychiatric diagnosing of personality disorders, alas also beneficial in merely communicating how a person is, through five specific personality traits being assessed: **extraversion**, **agreeableness**, **conscientiousness**, **neuroticism**, and **openness** [16]. *Biographical inventory of creative behaviors (BICB)* is a measure of behavioral creativity which considers the number of different habitual, everyday creative activities an individual has engaged in the last 12 months, and it defines the proposed variable of **biographical creativity** [30]. *Creative behavior inventory (CBI)* is a measure of behavioral creativity which considers the number of times an individual has engaged in a tangible, craft or art-driven creative activity, and it defines the variable of **creative accomplishment** [30]. *Revised Creativity Domain Questionnaire (CDQ-R)* is a measure of behavioral creativity which considers how one perceives oneself in a variety of areas creativity plays a key role, such as acting, leadership, computer science, or solving personal problems, and it defines the variable of **creative ability** [30]. **Design Self-Efficacy** survey is a self-efficacy measure, as it pertains to design tasks and design skills, as well as confidence one exercises in one's ability to perform highly in the areas asked about [7].

The non-questionnaire-defined variables are those of design assessment, which feature design novelty and design usability. **Design novelty** assesses functional cre-

ativity of a design solution, relative to the frequency of said function being proposed within the set of design solutions being evaluated [28]. **Design usability** is an expert panel-assessed measure of how effectively design addresses user-needs [19].

Results of surveys are found using standard scoring methods proposed by each survey's author. For surveys that needed to be correlated with one another across many categories, it is important to observe that their most concise form is presented in Table 1, contents of which will be discussed further on.

Results: Mutual Influences

The quantified variables described in the methods section, and previously studied in contexts of correlation to thinking styles assessed through REI [22], are now being considered within the expanded, social-cognitive framework proposed in Fig. 1. Within this framework, the triadic social-cognitive influencing model, where each relationship of influencers (**person** ↔ **behavior**, **behavior** ↔ **environment**, and **person** ↔ **environment**) is driven by self-efficacy, encompasses elements from the original design thinking styles framework proposed in Fig. 1. As such, the analysis of the results is done with respect to **two personal influencers** (**gender** considered *male* or *female* is a *biological personal influencer*, and **university class standing** considered a *first-year* and *upper class* is an *affective personal influencer*) and **two environmental influencers** (**location** considered *China* or *the U.S.* is a *cultural environmental influencer*, and **field of study** considered as *engineering* or *non-engineering* is also a cultural environmental influencer) [5].

In addition to the proposed influencers considered to extend an association to relationships studied among the variables discussed in the methods section, we also consider *personality*-based variables as attributes of the **personal influencer** category and *behavioral creativity* variables as attributes of the **behavioral influencer** category [2].

In this study, we had four attributes to the social-cognitive influencing categories. The personal category was attributed *gender* as a biological cognitive influencer, and *personality* as a cognitive personal influencer. The environmental category is attributed *location* and (academic) *culture*. Following are some of the results.

- An average **design self-efficacy** of **73.8** was found for the entire sample, on a scale from 0 to 100.
- Average **personality scores** are, for extraversion **3.12**, for agreeableness **3.82**, for conscientiousness **3.40**, for neuroticism **2.76**, and for openness **3.44**, on a scale of 1–5.
- Average **rational mode** score was **3.71**, while the average **intuitive mode** score was **3.09**, on a scale from 1 to 5.
- Average **creativity score** for biographical creativity was **0.31** on a scale from 0 to 1, for creative behaviors was **1.74** on a scale A–D enumerated 1–4, and for domain creativity was **2.98** on a scale from 1 to 5.

Table 1 Table of correlations of listed scores with respect to design self-efficacy score, per each category of influencers within the larger sample

Dep. Variable	Rat. (REI)		Int. (REI)		BICB		CBI		CDQ-R		D. Novelty		D. Usability	
	Corr.	<i>p</i>	Corr.	<i>p</i>	Corr.	<i>p</i>	Corr.	<i>p</i>	Corr.	<i>p</i>	Corr.	<i>p</i>	Corr.	<i>p</i>
Influencer														
Gender														
Female	0.53	0.03	0.27	0.27	0.25	0.32	0.32	0.20	0.59	0.01	0.10	0.72	0.34	0.21
Male	0.46	0.00	0.11	0.51	0.25	0.12	0.15	0.35	0.22	0.16	0.20	0.29	0.13	0.52
Location														
China	0.45	0.00	0.02	0.88	0.10	0.50	0.13	0.38	0.39	0.01				
US	0.15	0.58	0.16	0.58	0.33	0.23	0.23	0.41	0.18	0.52				
Field														
Engineer	0.42	0.00	0.05	0.73	0.32	0.03	0.22	0.15	0.24	0.11	0.38	0.04	0.16	0.38
Non-engineer	0.39	0.17	0.03	0.92	0.12	0.69	0.12	0.69	0.55	0.04	0.25	0.39	0.44	0.11
BFI														
Agreeableness	0.58	0.00	0.13	0.49	0.13	0.47	0.19	0.29	0.24	0.19	0.02	0.93	0.28	0.17
Conscientiousness	0.63	0.10	0.39	0.34	0.61	0.11	0.55	0.15	0.22	0.59	0.91	0.03	0.36	0.56
Extraversion	0.06	0.91	0.45	0.37	0.82	0.04	0.52	0.29	0.61	0.20	0.43	0.72	0.98	0.12
Neuroticism	0.42	0.35	0.20	0.67	0.32	0.48	0.37	0.41	0.44	0.33	0.08	0.86	0.35	0.44
Openness	0.69	0.04	0.18	0.64	0.03	0.94	0.39	0.30	0.64	0.06	0.48	0.33	0.52	0.29

Findings which are significant are marked in bold

- Within the Chinese-based sample that completed a design challenge as well, the measured **design novelty** had an average of **8.21**, with the range from 0 to 10. The average **design usability** was **3.05**, rated on a scale from 1 to 5.

In the analysis of the results, the first consideration was given to purely design self-efficacy scores within the context of influencers available, then consideration was given to three factors of behavior: *thinking styles*, *creative behavior*, and *design performance*, as influenced by design self-efficacy, with some context placed upon the previously studied influencers.

Considering the volume of the analysis presented here on, it is important to highlight that correlations were calculated between design self-efficacy and each of *thinking styles*, *behavioral creativity*, and *design performance*, with respect to each suitable set of influencers. Such findings are summarized in Table 1, and reveal many insignificant relationships found. We will use this information to better analyze data in the upcoming sections.

Relying on the information listed in the table, we may state that the following correlation values with respect to design self-efficacy are found significant:

- Rationality (REI) correlation with respect to both genders, Chinese location, engineering field, and personality traits of agreeableness and openness.
- Biographical creativity (BICB) correlation with respect to the engineering field and extraversion.
- Domain creativity (CDQ-R) correlation with respect to the female gender, Chinese location, and non-engineering fields.
- Design novelty (N) correlation with respect to the engineering field and conscientiousness.

Design Self-efficacy Relationship with Personal and Environmental SCT Influencers

Design self-efficacy, with listed associated scores, is

- 5% higher in **Men (74.9)**, than women (71.2);
- 14% higher in **American-based individuals (82.4)**, than Chinese-based ones (70.9);
- 15% higher in **Engineers (76.5)**, than non-engineers (65.0);
- Negative 42.6% associated with **Big Five Neuroticism**.
- Positive 42.4% associated with **Big Five Conscientiousness**.
- Positive 23% associated with **Big Five Openness**.
- Positive 13% associated with **Big Five Extraversion**.
- Positive 4.7% associated with **Big Five Agreeableness**.

What these findings report is that the most impactful influencers under consideration are *location*, *discipline*, *neuroticism (personality)*, and *conscientiousness*

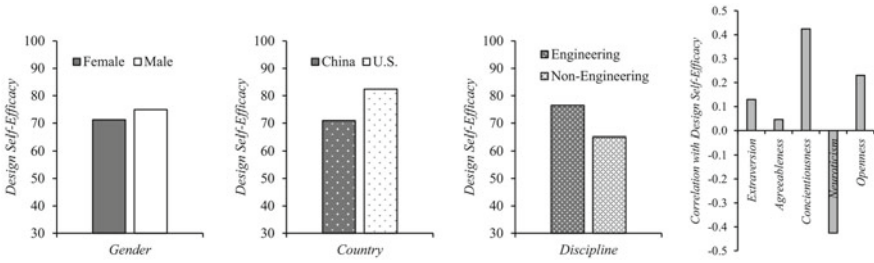


Fig. 2 Design self-efficacy with respect to personal and environmental influencers; left to right: *gender*, binary (female/male), *country* (China/United States), *discipline* (engineering/non-engineering), and *personality* (extraversion, agreeableness, conscientiousness, neuroticism, and openness)

(*personality*). Namely, the more favorable location is the U.S., and the more favorable discipline is engineering (Fig. 2).

Design Self-efficacy Relationship with Intuitive Thinking

Thinking styles were assessed per standard scoring of Rational-Experiential Inventory (REI), generating two separate scores, for the rational and intuitive mode. These scores were then analyzed in terms of how design self-efficacy scores associate with them, as well as how this association is guided by the available influencers from the previous section.

To address the second hypothesis, we first find the correlations between the overall design self-efficacy and rational mode, as **0.49**, and the correlation between design self-efficacy and intuitive mode as **0.02**.

These relationships, contextualized by the influencers gender, location and discipline in Fig. 3 and personality in Fig. 4, demonstrate the following observations for rational and intuitive modes.

Rational mode of thinking is associated with design self-efficacy

- Most positively for subjects located in China
- Least associated for subjects located in the U.S.
- Associated no differently for male or female subjects (association is positive across the board)
- Most positively associated for subjects with the highest personality scores being conscientious, open, neurotic, or agreeable (in that order)
- Not associated for subjects with the highest personality score for extraversion.

Intuitive mode of thinking is associated with design self-efficacy

- Positively for female subjects
- Negatively for male subjects

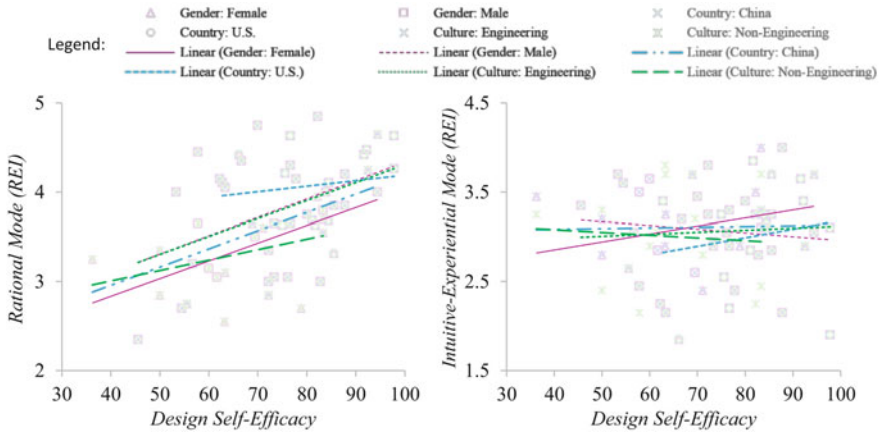


Fig. 3 Rational mode of thinking and intuitive mode of thinking with respect to design self-efficacy, contextually studied with respect to the gender, location and discipline of subjects

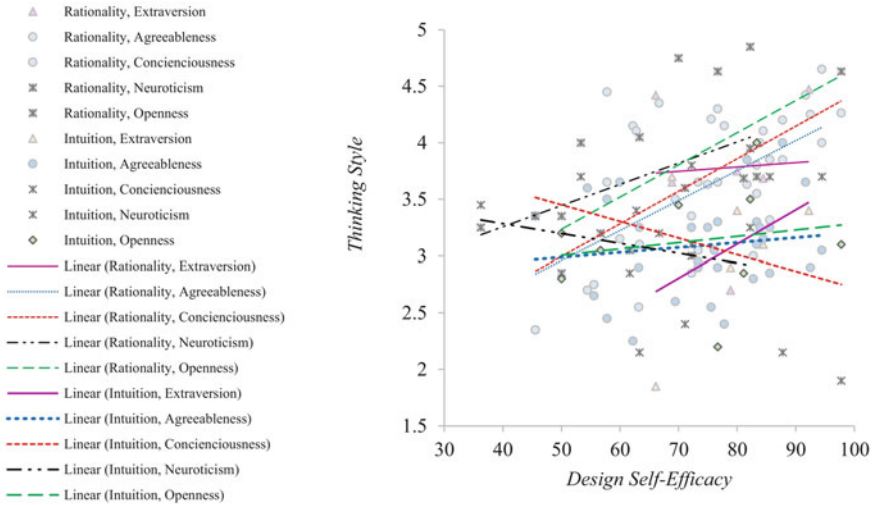


Fig. 4 Rational mode of thinking and intuitive mode of thinking with respect to design self-efficacy, contextually studied with respect to big five personality traits: extraversion, agreeableness, conscientiousness, neuroticism, and openness

- Positively for subjects located in the U.S.
- Least associated for subjects located in China
- Associated no differently for engineering or non-engineering disciplines (association is close to none across board)
- Most positively associated for subjects with the highest personality score for extraversion

- Not associated for subjects with the highest personality score for agreeableness or openness
- Most negatively associated for subjects with the highest personality scores for conscientiousness or neuroticism.

The ultimate finding is that the rational mode is better associated with design self-efficacy than the intuitive mode, which contradicts our hypothesis. Figures 3 and 4 visualize in detail these preliminary findings, yet per Table 1 p -values, any findings regarding the intuitive mode of thinking are insignificant, and rational mode of thinking has a great deal of significant findings, across domains of both genders, Chinese location, engineering field, and personality traits of agreeableness and openness.

Design Self-efficacy Relationship with Creative Behavior

Creative Behavior was scored using the three designated measures of behavioral creativity

- (1) BICB: Biographic Index of Creative Behaviors, to measure biographic creativity
- (2) CBI: Creative Behavior Inventory, to measure creative behavior
- (3) CDQ-R: Creative Domains Questionnaire, Revised, to measure domain creativity.

To address the third hypothesis, we found the correlations between the overall design self-efficacy and each of these three variables, resulting in correlations of **0.23 for biographic creativity**, **0.15 for creative behavior**, and **0.36 for domain creativity**.

In the context of gender, location, and discipline—influencers, these variables were studied with respect to design self-efficacy, as depicted in Fig. 5.

Biographical Creativity (from BICB) was associated with design self-efficacy

- Most positively associated for the location being the U.S., discipline engineering, and gender male.
- Not associated for subjects based in China.
- Most negatively associated for subjects in non-engineering disciplines.

Creative behavior (from CBI) was associated with design self-efficacy:

- Most positively associated for the location being the U.S., discipline being engineering, and gender being female
- Not associated for subjects in non-engineering disciplines.

Domain creativity (from CDQ-R) was associated with design self-efficacy:

- Most positively associated for gender being female
- Not associated with non-engineering disciplines.

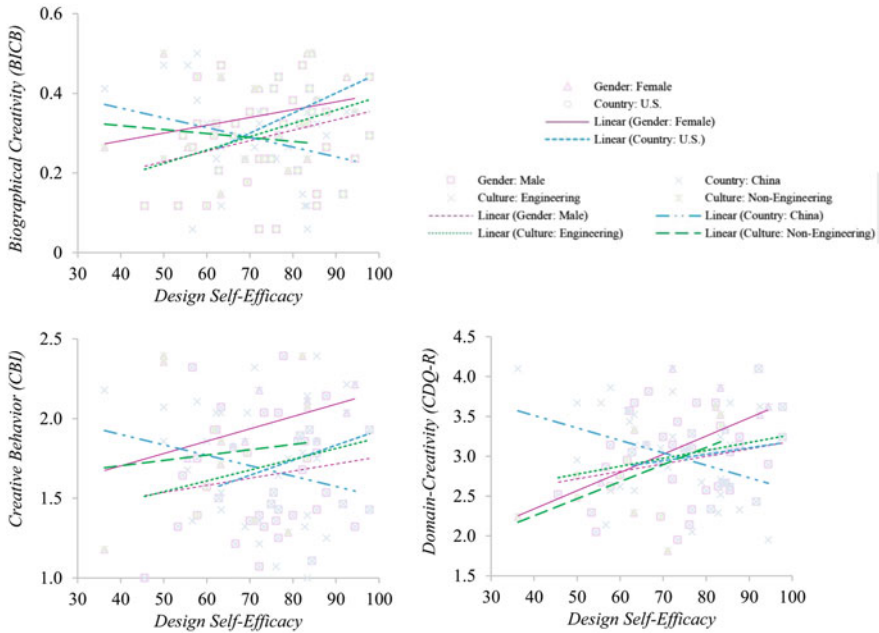


Fig. 5 Behavioral creativity scores of BICB, CBI and CDQ-R, studied with respect to design self-efficacy, in the contexts of gender, location, and discipline

Findings on the association of design self-efficacy with behavioral creativity are inviting for further studies in the domain of our proposed hypothesis of their association being high. Figure 5 visualizes the preliminary findings for creative behavior to design self-efficacy relationship. From Table 1, we can state that none of CBI-related findings are significant, while the BICB findings are significant in domains of the engineering field and extraversion. CDQ-R findings are significant in the domains of the female gender, Chinese location, and non-engineering fields.

Design Self-efficacy Relationship with Design Performance

Design performance was assessed relying on two established variables: design novelty and design usability. These scores had design self-efficacy correlations of **0.11 for design novelty**, and **0.24 for design usability**.

These two variables were then studied in the context of influencers of gender, discipline, and personality, as depicted in Figs. 6 and 7.

Design novelty was associated with design self-efficacy

- Most positively associated when discipline is engineering
- Not associated with gender
- Most negatively associated when discipline is non-engineering
- Most positively associated for subjects with the highest personality scores in conscientiousness and openness
- Not associated for subjects with the highest scores in agreeableness and neuroticism

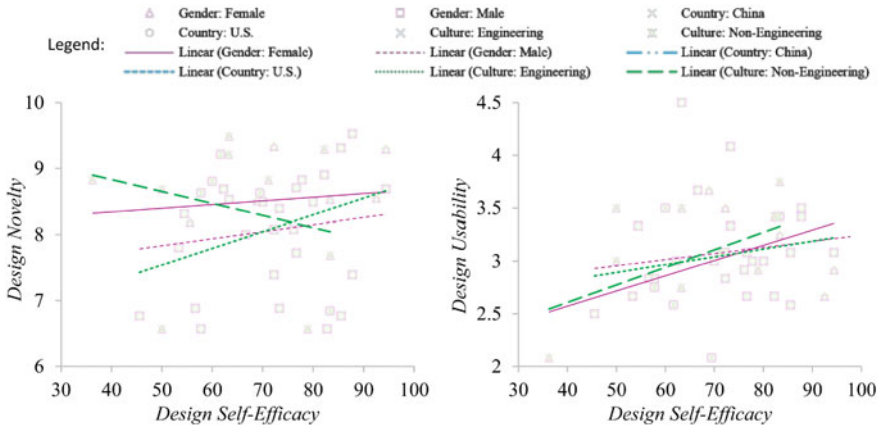


Fig. 6 Design novelty and design usability scores, studied with respect to design self-efficacy, in the contexts of gender, location, and discipline

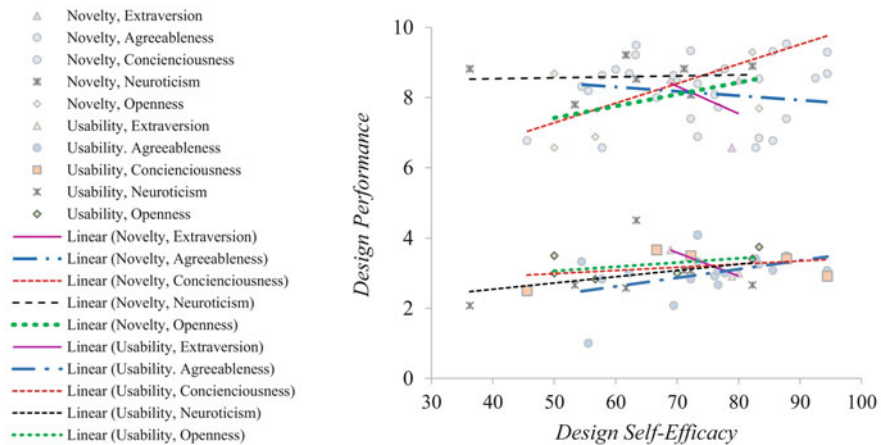


Fig. 7 Design novelty and design usability scores, studied with respect to design self-efficacy, in the contexts big five personality traits: extraversion, agreeableness, contentiousness, neuroticism, and openness

- Most negatively associated for subjects with the highest personality score in extraversion.

Design usability was associated with design self-efficacy

- Most positively associated with gender being female
- Not associated with the discipline
- Not associated with gender being male
- Most positively associated for subjects with the highest personality scores of agreeableness, conscientiousness, neuroticism, and openness
- Most negatively associated for subjects with the highest personality scores of extraversion.

The findings for usability are not significant in Figs. 6 and 7, while some of the findings for novelty are, specifically in domains of engineering field and conscientiousness.

Conclusions and Further Recommendations

Bridging design research with social influencing, and thus social-cognitive, and other social theories, while remaining within our original domain of dual process theory and dual process framework for early stage engineering design, has posed a considerable challenge, and is something that few have done before to this extent. While our findings show one disproven hypothesis and two hypotheses that require further considerations, we are of the belief that this preliminary work sets ground for further exploration of social and behavioral contexts for design.

We have, in the end, found that the highest correlation with design self-efficacy exists for the rational mode of thinking, at 0.49. No other studied quantity gets even close to correlating this well with design self-efficacy. Rationality also lends itself to the highest number of significant findings among the preliminary ones reported. One way to describe this would be that those who exhibit high rational scores also approach their knowledge acquisition of design steps and methods more rationally, thus being more able to claim that they are highly confident about completing the breakdown of design tasks. Another way to interpret this finding would be that the more rational subjects would have found themselves in more situations where they would need to conduct engineering design, thus building greater expertise and thus greater confidence and motivation for completing the process repeatedly.

To make our second hypothesis strikingly disproven, we should note that out of the entire set of behavioral variables, the correlation found for intuitive thinking mode to design self-efficacy was by far the lowest, and did not carry any significance. The low correlation and very high p -values call for larger sample study or an alternative method for studying intuition.

While the influencing of binary factors like gender, discipline, and location was simpler to analyze and deduce findings on, we propose a greater exploration of a

much challenging influencing process that goes on between Big Five personality traits and the studied behavioral variables. In our analysis, we could only complete plots of this relationship by selecting the most dominant personality trait (the highest scoring one) and ascribing it as the sole personality influencer for the subject in question. While this offers assistive graphics and a large deal of contextual analysis, one of our next step includes finding a better method of complete this analysis more wholesomely. Additionally, of the five traits, neuroticism never yields any significance across different correlations studied.

Lastly, we hope to expand our model proposed in Fig. 1, in directions of studying creativity from more cognitive or personality-driven standpoints, assessing design through different sets of methods and variables, and finding further organizational influencers that could aid or stifle design processes on individual level. We expect to expand our thinking style variable beyond its current domain, separately study abilities of subjects, and investigate what creative processes assist the stylistic use of one's abilities in most successful ways.

This research was supported in part by DeTao Masters Academy Research Institute through a summer research grant. The authors are grateful to DeTao for their support. The authors also extend their thanks to Professor Zhinan Zhang of Shanghai Jiao Tong University for helping the process of surveying students at the university.

The human subject data acquisition was approved by the University of Southern California University Park Institutional Review Board (UPIRB) under the study ID number: UP-16-00701, and subjects were supplied with Certified Information Sheet linked.

References

1. Baird FM (2000) An ethnographic study of engineering design teams at Rolls-Royce. *Des Stud* 21(4):333–355
2. Bandura A (1977) Self-efficacy: toward a unifying theory of behavioral change. *Psychol Rev* 84(2):191–215
3. Bandura A (1999) Social cognitive theory: an agentic perspective. *Asian J Soc Psychol* 2(1):21–41
4. Bandura A (2001) Social cognitive theory: an agentic perspective. *Annu Rev Psychol* 52(1):1–26
5. Bandura A (2005) The evolution of social cognitive theory. In: Smith KG (ed) *Great minds in management*. Oxford University Press, Oxford, pp 9–35
6. Bechtoldt MN (2010) Motivated information processing, social tuning, and group creativity. *Interpersonal Relat Group Processes* 99(4):622–637
7. Carberry AR (2010) Measuring engineering design self-efficacy. *J Eng Educ* 99(1):71–79
8. Choi JN (2011) Individual and contextual predictors of creative performance: the mediating role of psychological processes. *Creativity Res J* 16(2 & 3):187–199
9. Chulisp P, Jin Y (2006) Impact of mental iteration on concept generation. *J Mech Des* 128:14–25
10. De Dreu CK (2008) Motivated information processing in group judgement and decision making. *Pers Soc Psychol Rev* 12(1):22–49
11. Dennen VP (2007) The cognitive apprenticeship model in educational practice. Florida State University, Florida, pp 426–439

12. Diamond M (2002) Sex and gender are different: sexual identity and gender identity are different. *Clin Child Psychol Psychiatry* 7(3):320–334
13. Driscoll MP (1994) *Psychology of learning*. Allyn & Bacon, Needham Heights, MA
14. Epstein S (2003) Cognitive-experiential self-theory of personality. In: Milton MJ (ed) *Comprehensive handbook of psychology*. Wiley, Hoboken, pp 159–184
15. Evans JS (2013) Dual-process theories of higher cognition. *Perspect Psychol Sci* 8(3):223–241
16. Goldberg LR (1993) The structure of phenotypic personality traits. *Am Psychol* 48(1):26–34
17. Grant AM (2011) The necessity of others is the mother of intention: intrinsic and prosocial motivations, perspective taking, and creativity. *Acad Manage J* 54(1):73–96
18. Jin Y, Benami O (2010) Creative patterns and stimulation in conceptual design. *Artif Intell Eng Des Anal Manuf* 24(2):191–209
19. Kudrowitz BM (2010) The play pyramid: a play classification and ideation tool for toy. *Int J Art Technol* 3(1):36–56
20. Lamm CB (2007) The neural substrate of human empathy: effects of perspective-taking and cognitive appraisal. *J Cogn Neurosci* 19(1):42–58
21. Massachusetts Department of Education (2001/2006) *Massachusetts science and technology/engineering curriculum framework*. Massachusetts Department of Education, Malden, MA
22. Milojevic HG (2016) Influence of thinking style on design creativity. In: *The fourth international conference on design creativity*. The Design Society, Atlanta, GA
23. Moore DJ (2014) A dual-process analysis of design idea generation. In: *International design engineering technical conferences and computers and information in engineering*. ASME
24. Pacini R, Epstein S (1999) The relation of rational and experiential information processing styles to personality, basic beliefs, and the ratio-bias phenomenon. *J Pers Soc Psychol* 76(6):972–987
25. Perry-Smith JE (2003) The social side of creativity: a static and dynamic social network perspective. *Acad Manage Rev* 28(1):89–106
26. Sauder J, Jin Y (2016) A qualitative study of collaborative stimulation in group design thinking. *Des Sci* 2(4):1–25
27. Schaub M, Tokar DM (2005) The role of personality and learning experiences in social cognitive career theory. *J Vocat Behav* 66:304–325
28. Shah JM (2012) Applied tests of design skills—Part I: divergent thinking. *J Mech Des* 1–10
29. Shuell TJ (1986) Cognitive conceptions of learning. *Rev Educ Res* 56(4):411–436
30. Silvia JP-P (2012) *Assessing creativity with self-report scales: a review and empirical evaluation*. Univeristy of Nebraska Omaha Digital Commons@UNO
31. Stanovich KE (2000) Individual differences in reasoning: implications for the rationality debate. *Behav Brain Sci* 23(5):645–665
32. Weinstein C, Mayer RE (1986) The teaching of learning strategies. In: Wittrock ME (ed) *Handbook of research on teaching*. Macmillan, New York, pp 315–327
33. Witteman CV (2009) Assessing rational and intuitive thinking styles. *Eur J Psychol Assess* 25(1):39–47
34. Zimmerman BJ (1990) Self-regulated learning and academic achievement: an overview. *Educ Psychol* 25(1):3–17

Cognitive Style and Field Knowledge in Complex Design Problem-Solving: A Comparative Case Study of Decision Support Systems



Yuan Ling Zi Shi, Hyunseung Bang, Guy Hoffman, Daniel Selva and So-Yeon Yoon

Cognitive differences between how people perceive and process information have been broadly studied in the fields of education and psychology. Previous findings show that comprehension is optimized when information presentation aligns with the cognitive abilities and preferences of an individual. On the other hand, the possession of field knowledge has also been studied to influence learning outcome and perception. This paper aims to understand the effects of individual's information processing styles and field knowledge on design decision-making, specifically focusing on designer learning and user experience. Two distinct decision support systems interfaces were developed to better examine the effect using a mixed model design. A total of 48 college students participated in the experimental study and interacted with the two different interfaces of a satellite design system in a randomized order. Analysis results show significant impacts of field knowledge and visual processing style on both learning and user experience. Potential interaction effects with the design support system interface type and cognitive styles were also observed.

Introduction

Advancing technologies allow more intelligent and powerful functionalities in decision support systems for complex problems. Decision support systems (DSS) are designed to facilitate the decision process by providing manipulable, current, timely information that is accurate, relevant, and complete [1]. They allow better decision-making by expanding the human capacity to completely and accurately assess available information [2]. This expansion of information processing capacity is needed for tackling complex design problems that many industries are facing today. Complex

Y. L. Z. Shi · H. Bang · G. Hoffman · D. Selva · S.-Y. Yoon (✉)
Cornell University, Ithaca, NY, USA
e-mail: sy492@cornell.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_19

design problems often are large in scale and multidisciplinary; to tackle such problems, it is necessary to determine possible interactions among the subsystems and their parts [3]. DSS's simulation and optimization abilities allow users to manipulate these subsystems and components to examine the interactions. With the need to process and present such complex information, the design of these DSS interfaces can be crucial to the success of their implementations. As multidisciplinary approaches are increasingly valued in problem-solving, teams are becoming more diverse with people coming from different professional and academic backgrounds. To facilitate such collaboration, the DSS interfaces need to support a variety of users who may exhibit very different cognitive processes. Therefore, it is important to research the factors that potentially influence human–DSS interaction experience and performance. In addition, such exploration can assist underrepresented or disadvantaged populations where decision support systems can be designed to be more inclusive and equitable. Ultimately, the long-term goal of the authors is to identify operable design principles to improve user performance and satisfaction for a variety of users. This paper contributes towards that effort by exploring the effects of individual cognitive style and field knowledge on user performance and experience during complex design problem-solving with DSS.

Cognitive Style

One of the most critical components in decision-making process is the human decision-maker; thus, it is important to consider the ways decision-makers acquire information to make judgments such as individual cognitive styles [4]. Cognitive style is often defined as consistencies in one's acquisition and processing of information, including the considerations of perception, thought, memory, imagery, and problem-solving [4]. Furthermore, Sternberg and Grigorenko [5] defined cognitive style as people's typically preferred modes of processing information. The field of cognitive style gained its popularity starting in the early 1950s, and since then, different dimensions of cognitive styles emerged over the years, such as sharpener versus leveler [5], field dependent versus field independent [6], holist versus serialist [7], and verbalizer versus visualizer [5]. Around the 1980s and 1990s, many studies examined these dimensions to study the potential influence of cognitive styles on DSS user performance.

Benbasat and Schroeder [8] examined the interaction effect of cognitive styles, presence of a decision aid, and different forms of information presentation on decision performance assessed by a decision-making game. Analysis showed an interaction effect between cognitive style and the presence of decision aid on the number of reports needed while making decisions. High analytic thinkers with the help of the decision aid used fewer reports during decision-making than their counterparts without decision aids, and vice versa for low analytic thinkers. Benbasat and Dexter [9] further explored this relationship and found an overall better performance measured by profit gained among the high analytic thinkers. An interaction effect of cognitive

style and the presence of decision aids on decision time were also revealed. High analytic thinkers with decision aids took more time than those without, whereas low analytic thinkers with decision aid took a similar amount of time as those without.

Different dimensions of cognitive styles have been examined. In a study focused on the interaction effect of cognitive style and graphical representation of problem elements on DSS user performance in terms of decision quality, higher field dependency was associated with longer decision time with no influence on percent error; similarly, higher need for cognition was also associated with longer decision time, but with a higher percentage error [10]. Davis and Elnicki [11] also found an interaction effect between cognitive styles assessed by the Myers–Briggs Type Indicator (MBTI) and data format on decision quality where high sensing-feeling cognitive styles were associated with better performance with tabular data and high experiential-feeling scores were associated with better performance with graphical-raw data. Using similar methods, Green and Hughes [12] added the element of training type and analysis revealed that performance was optimized when heuristic managers received seminar training and when analytic managers received hands-on workshop training.

Studies have also considered the effects of user characteristics and user experience. Ramamurthy et al. [13] studied how user characteristics influenced DSS effectiveness both in terms of performance and user satisfaction. Individuals with higher sensing and thinking scores outperformed individuals with higher intuitive and feeling scores also had higher scores in performance and efficiency; they also responded with less perceived difficulty and displayed more favorable attitudes towards the DSS. Within the context of specific DSS models, higher sensing scores were associated with more positive attitudes with reference to perceived usefulness and willingness to use; however, higher thinking scores were associated with more negative attitudes [13].

There had been concerns regarding the ability of cognitive style research in the field of DSS design to produce operable design guidelines. Huber [14] argued that there were inadequate theories in cognitive styles, poor operationalization, and insufficient research designs, which contributed, to stagnation in the field. Furthermore, reviews of existing studies showed that cognitive style explained very little of DSS user performance [14]. However, new efforts were made to unify the field of cognitive style in the 1990s [15]. Sternberg and Grigorenko [5] further stated that the study of cognitive styles does show promise in terms of predicting school and other kinds of performances. Moreover, new advances in the field also provide exciting opportunities for new research areas in the context of DSS design. In this paper, we will be focusing on two dimensions of cognitive styles that had recent developments in theory and instrumentation: the rational-experiential cognitive style, and the object-spatial visualization style.

Rational and Experimental Cognitive Style

The cognitive-experiential self-theory describes two parallel and interacting modes of information processing: the rational cognitive style and the experiential cognitive style [16]. In this theory, the rational cognitive system is described as analytic and logical whereas the experiential system is attributed to being holistic and affective. Thus, rational thinkers are characterized by the ability and reliance on thinking in a logical and analytic manner; experiential thinkers have the ability and preference to rely on one's intuition and feelings in making decisions [17]. These two dimensions were chosen, as there were a great number of previous studies that examined the relationship between DSS usage and the holistic-analytic cognitive styles. We aim to build upon the literature, however, by taking on a slightly different theoretical perspective by using the rational-experiential cognitive styles instead of the holistic-analytic cognitive styles.

Object and Spatial Visualization Style

Studies had supported the existence of a visualizer-verbalizer dimension of cognitive style where visualizers primarily rely on imagery when performing cognitive tasks and verbalizers primarily rely on verbal-analytical strategies. Within the visualizer cognitive style, newer findings suggested two qualitatively different types of visualizers, object versus spatial visualization [18]. They are related but distinct dimensions [19]. Object visualization refers to processing visual information in terms of physical appearances like shape, color, and texture; spatial visualization refers to processing visual information in terms of spatial relationships such as location, movement, transformation, and other spatial attributes [20]. In addition, object visualizers have a tendency to encode images globally as a single perceptual unit, which they process holistically, whereas spatial visualizers have a tendency to encode and process images analytically, in sequence of components, and use spatial relations to arrange and analyze them [18]. A study has examined the interaction of cognitive style and information presentation format on comprehension specifically considering the object-spatial visualization styles in addition to the visualizer-verbalizer dimension of cognitive style [21]. The information was distributed in three different forms: text only (verbal); text + picture (object visual); and text + schematic diagram (spatial visual). Results showed an optimization of comprehension when the information presentation matched with the cognitive style of the individual. In this study, we wish to apply this finding in the context of DSS interface design to explore its implications on DSS user performance and user experience.

Field Knowledge

In this study, field knowledge, also known as domain knowledge, is defined as understanding of the context of the materials; it does not necessarily refer to design expertise. In terms of hypermedia learning, domain knowledge has shown significant influence both in terms of navigation behavior and disorientation problems. Studies have reported superior performance in navigation among individuals who exhibit domain knowledge [22, 23] and more disorientation in hypermedia systems among those who have little to no domain knowledge [24]. Furthermore, prior knowledge experts were also found to show more positive perceptions of their learning processes [25, 26]. The prior knowledge for multimedia-learning environments expressed that design principles that assist low-knowledge learners may not benefit or may even hinder high-knowledge learners [27]. Researchers [28] also found that individuals with lower domain knowledge benefit more from tutorials and examples than those with higher domain knowledge. We wish to explore these relationships in the context of DSS interactions.

DSS Performance and User Experience

The outcomes of DSS interaction can be categorized into user performance and user experience. In this paper, DSS user performance is measured by learning outcomes. The ultimate goal of the proposed DSS in this study is to allow users to better understand the design problem and how to manipulate different parameters to achieve better outcomes. Learning outcomes consist of evaluations of comprehension and application of the information gained while interacting with the assigned DSS. In the context of our experiment, user experience refers to subjective perception of the interaction process using DSS for design problem-solving processes in terms of perceived performance, affect, and sensation.

Research Design and Methodology

iFEED DSS

The Interactive Feature Extraction for Engineering Design system (iFEED) was the decision support system used in the experiment [29]. It addresses a real-world system architecture problem with a goal to design a constellation of satellites to provide operational observation of the Earth's climate.

In the context of the design problem, satellites are being launched into five orbits around the Earth and each satellite can carry up to 12 instruments. The main objective of this task was to optimize the design of this system consisting of up to 5 satellites to

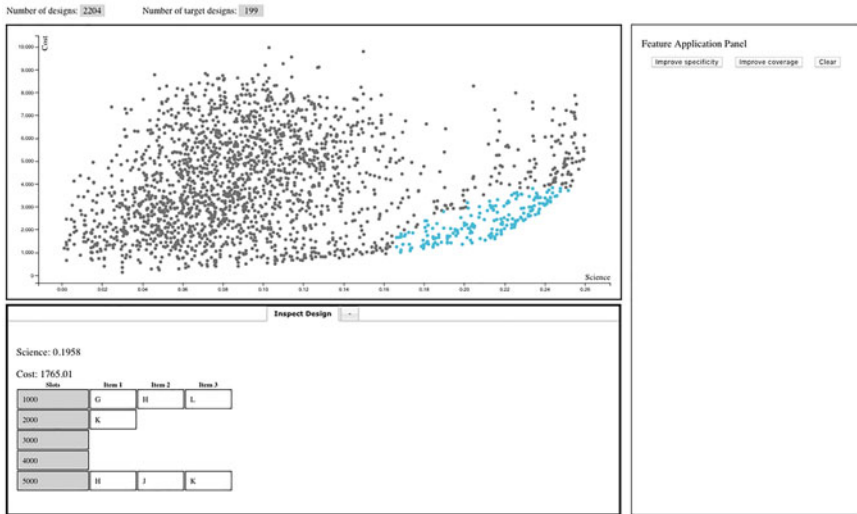


Fig. 1 Object (literal) space of DSS interface

maximize the scientific benefit and minimize the lifecycle cost. The iFEED interface provided two main capabilities: to inspect individual designs and to run data mining algorithms to extract common features shared by a selected group of designs. Design inspections were done in the Objective Space, which consists of an interactive scatter plot of satellite constellation designs and a window that displays the configuration of the selected designs (Fig. 1). The user could hover over any data point on the scatter plot and the window below would live update information on the data point highlighted. We consider this display as a literal representation of the data where a design's scientific benefit and cost are directly indicated on the plot and the literal configurations of each satellite are displayed. Additionally, a Feature Space is used to extract feature information shared by a selected group of designs using data mining (Fig. 2). It also includes an interactive scatter plot with each data point representing a shared feature. The plot has two axes of coverage and specificity where coverage expresses how many designs in the desired region share this feature whereas specificity expresses how few designs outside of the region exhibit this feature. The feature details are displayed in a logic tree diagram on the right. We characterize this as an abstract representation of the data as it is showing information on a more conceptual level.

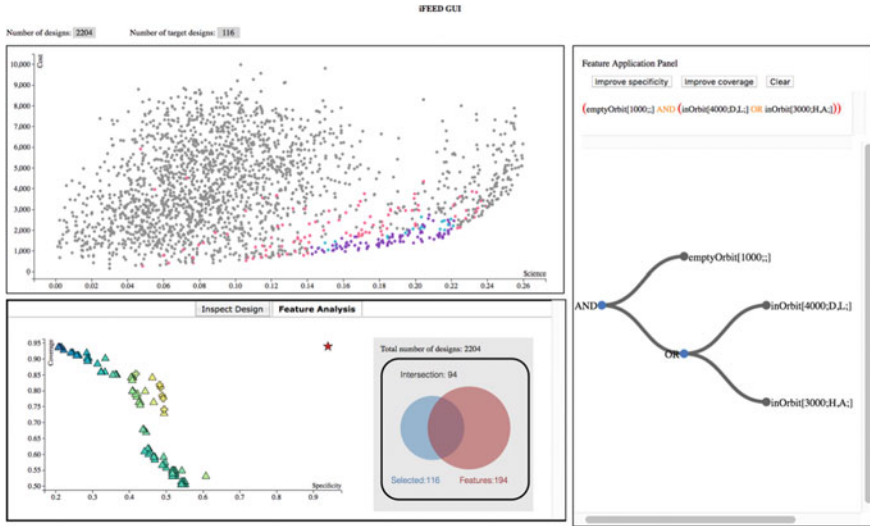


Fig. 2 Feature (abstract) space of DSS interface

Participants

Undergraduate and graduate students enrolled in Cornell University were recruited by online and in class advertisements with participation incentives with either a 15 dollars Amazon gift card or extra credit. A total of 48 students, of which 20 (41%) were females, participated in the study. Thirty-four (70%) students were STEM majors: Science, Technology, Engineering, and Mathematics. Table 1 summarizes the sample demographics. None of the students had any prior interaction with the iFEED interface. Ethical approval was provided by the Cornell University Office of Research Integrity and Assurance, Institutional Review Board for Human rights. Thus, the IRB protocol was followed and informed consents were obtained from all participants.

Variables and Measures

An online survey using the Qualtrics system was sent to the participants prior to the study to gather demographic information such as gender, ethnicity, major field, and school year. Field knowledge was determined by whether the student was in STEM majors or not as the context of the design task was heavily rooted in the field of engineering and mathematics. Different cognitive style scales were also included. The Object-Spatial Imagery Questionnaire (OSIQ) was used to measure individual visualization styles. The questionnaire consists of 40 Likert items that allowed par-

Table 1 Participant demographics

Characteristics		<i>n</i>
Major field	STEM	34
	Non-STEM	14
Year	Sophomore	7
	Junior	4
	Senior	5
	Graduate	28
Gender	Male	28
	Female	20
Ethnicity	Caucasian	21
	Asian	24
	Hispanic	2
	Others	1
Total (<i>N</i>)		48

ticipants to rate their level of agreement for each statement on a 5-point scale. In terms of internal consistency, the questionnaire has a Cronbach alpha of .79 for spatial measures and .83 for object measures. The questionnaire was also tested against established measures with acceptable convergent validity coefficient ranges [20]. In terms of rational-experiential cognitive styles, the Rational-Experiential Inventory (REI) [30] was used and it has been tested to have satisfactory validity and reliability [31].

A post-experiment questionnaire was used to assess learning and user experience from the given tasks with DSS, dependent variables of the study. Learning was evaluated by a quiz containing 25 items asking if specific satellite constellation designs would reside in the target region studied during the experiment. Then the User Experience Questionnaire (UEQ) [13] was administered to measure user experience. The UEQ has 26 sets of opposing adjectives and the participants would rate their experience on a 7-point scale within each of the 26 dimensions.

Research Design

The study followed a mixed model design with three independent and two dependent variables. The independent variables include cognitive styles, field knowledge, and DSS interface variations. Cognitive style and field knowledge were between-subject variables whereas the DSS interface variable was a within-subject variable. The dependent variables include learning and experience.

Procedure

First, the pre-experiment online survey was sent via email to all participants after they have signed up for individual experiment timeslots online. They were instructed to complete the survey before their scheduled experiment times. Upon arrival at the lab, the participant was asked to sign a consent form and was directed to the computer station. They were first walked through an interactive tutorial regarding the iFEED system and the objectives of their tasks were given at the end. Then the participants interacted with one of the two DSS (literal versus abstract interfaces) at a randomized order for 10 min and the post-experiment questionnaire was administered. Then, the second interface was introduced, and the same procedures were repeated.

Hypothesis

To explore the effect of individual cognitive style and field knowledge on user performance and experience during complex design problem-solving with DSS, we built upon previous literature and five hypotheses were generated:

- H1: Prior field knowledge predicts DSS learning and user experience such that higher field knowledge is associated with better DSS user performance and user experience with interface design being a moderator.
- H2: Rational-experiential cognitive style predicts DSS learning and user experience and this relationship is moderated by field knowledge and interface design.
- H3: Object-spatial visualization style predicts DSS learning and user experience and this relationship is moderated by field knowledge and interface design.
- H4: Spatial visualizers and rational thinkers have more positive learning outcomes and user experience when information is presented in an abstract manner (Fig. 3).
- H5: Object visualizers and experiential thinkers have better learning outcomes and higher ratings of user experience when information is presented in a literal manner.

Results

A series of mixed model analyses were performed to examine the effects of cognitive styles and field knowledge on user experience and user learning using two types of DSS. A mixed model was used because the experiment included both within-subject (two DSS interface types) and between-subject (STEM vs. non-STEM) independent variables; this would allow us to control for the random effects from individual characteristics.

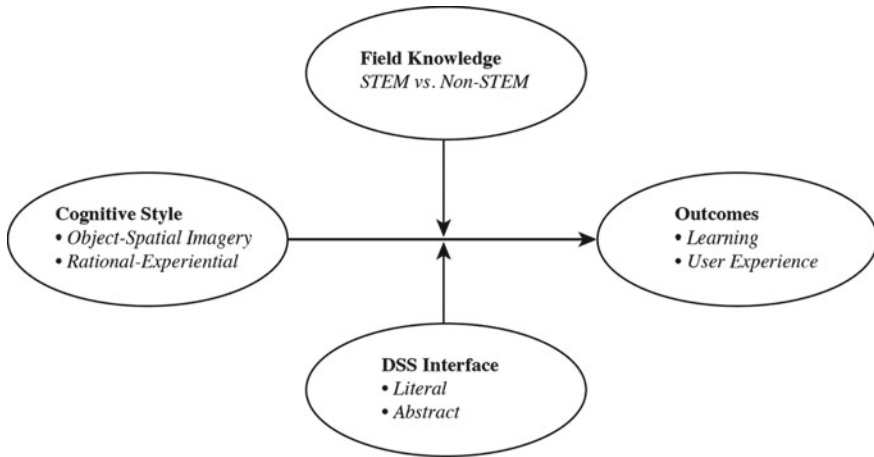


Fig. 3 Research concept diagram

The above correlation matrix (Table 2) shows a significant correlation between the rational style and UEQ scores, $r(41) = .32, p < .05$, within the objective space condition. On the other hand, there were significant correlations between objective imagery style and learning ($r(33) = -.39, p < .05$), and spatial imagery style and learning ($r(33) = .52, p < .001$). This provided a rough overview of the potential relationships to be examined.

Results from comparisons of DSS usage outcomes between field knowledge groups are reported in Table 3. When comparing DSS learning outcomes between STEM and non-STEM students, overall, STEM students performed better on the learning test than non-STEM students. However, this relationship is only significant for scores regarding the feature space ($t(12) = 2.57, p = .02$). In regard to user experience, analysis showed a main effect of being in STEM majors on rating on the UEQ questionnaire such that STEM students gave higher ratings of user experience for both the objective space ($t(23) = 4.25, p < .001$) and feature space conditions ($t(33) = 3.33, p = .002$). This mostly confirms H1 where prior knowledge predicts DSS learning and user experience, however only learning outcomes were moderated by interface design.

Looking at cognitive style by field knowledge groups (Table 4), the analysis showed significant differences in object-spatial cognitive style between STEM and non-STEM students, which aligned with findings from previous studies [20]. On average, STEM students have significantly higher scores on the spatial imagery scale ($t(30) = 2.41, p = .022$) and lower scores on the object imagery scale ($t(31) = -4.31, p < .001$), and the opposite relationships were observed for non-STEM students. We also looked at cognitive style dominance, meaning if an individual scored higher on one dimension than another or if the individual scored equally on both dimensions. Analysis showed that STEM students tend to be more Spatial dominant whereas

Table 2 Correlation matrix

DSS	Measures	Outcomes		Cognitive styles			
		1. Learn.	2. UEQ	3. Object.	4. Spatial	5. Rational.	6. Exp.
Objective space	1. Learning	–	–				
	2. User exp. (UEQ)	.28					
	3. Object I. style	–.16	–.00	–			
	4. Spatial I. style	.10	.25	–.24	–		
	5. Rational style	–.07	.32*	–.21	.24	–	
	6. Experiential style	–.14	–.03	.17	–.11	–.14	–
Feature space	1. Learning	–					
	2. User exp. (UEQ)	.16	–				
	3. Object I. style	–.39*	.17	–			
	4. Spatial I. style	.52***	.10	–.24	–		
	5. Rational style	.27	.11	–.21	.24	–	
	6. Experiential style	–.21	–.04	.17	–.11	.24	–

* $p < .05$, ** $p < .01$, *** $p < .001$

non-STEM students tend to be more object dominant. No significant differences in Rational-Experiential cognitive styles were observed.

Furthermore, four mixed model analyses were conducted to assess the effects of cognitive styles and field knowledge on learning (Table 5), and four additional mixed model analyses were conducted to assess their effects on user experience (Table 6). Analysis showed partial support for H1 and H3. Within model 1, there was a main effect of object visualization style on overall learning $F(60) = -2.20, p = .02$, such that object visualizers tend to have lower learning outcomes. There also seemed to be an object visualization x field knowledge interaction effect such that lower object scores within STEM fields have higher learning outcomes, $F(60) = -2.41, p = .01$, whereas lower object scores within non-STEM fields did not have a significant effect on learning outcomes. Looking at model 2, two main effects were found to be significant. There was a main effect of spatial visualization ($F(60) = 1.75, p = .02$)

Table 3 Comparisons of design support system (DSS) usage outcomes by field knowledge groups

Outcome	Field knowledge (n)	Design support system		t-test
		Objective space (literal)	Feature space (abstract)	
Learning M(SD)	STEM (10)	18.79 (2.64)	19.67 (2.12)	$t(28) = 1.39, p = .17$
	Non-STEM (24)	17.20 (2.97)	16.80 (3.22)	$t(20) = -.21, p = .83$
	<i>t-test</i>	$t(17) = 1.46, p = .16$	$t(12) = 2.57^*, p = .02$	
User experience M(SD)	STEM (31)	4.81 (.85)	4.50 (.06)	$t(57) = -1.29, p = .20$
	Non-STEM (12)	3.69 (.75)	3.65 (1.02)	$t(21) = -.14, p = .89$
	<i>t-test</i>	$t(23) = 4.25^{***}, p = .0003$	$t(33) = 3.33^{**}, p = .002$	

* $p < .05$, ** $p < .01$, *** $p < .001$, **** $p < .0001$

Table 4 Comparisons of cognitive styles by field knowledge groups

		Field knowledge		t-test
		STEM n = 32	Non-STEM n = 13	
Cognitive styles M(SD)	Object imagery	3.03 (.57)	3.67 (.40)	$t(31) = -4.31^{***}, p = .0001$
	Spatial imagery	3.71 (.56)	3.34 (.42)	$t(30) = 2.41^*, p = .022$
	Δ Obj.-spat.	-.68 (.84)	.33 (.59)	$t(32) = -4.60^{***}, p < .0001$
	Rational	3.86 (.489)	3.56 (.53)	$t(21) = 1.75, p = .095$
	Intuitive	3.07 (.57)	3.64 (.57)	$t(20) = -.91, p = .37$
	Δ Ratio.-intu.	.78 (.78)	.29 (.91)	$t(20) = 1.71, p = .10$

* $p < .05$, ** $p < .01$, *** $p < .001$, **** $p < .0001$

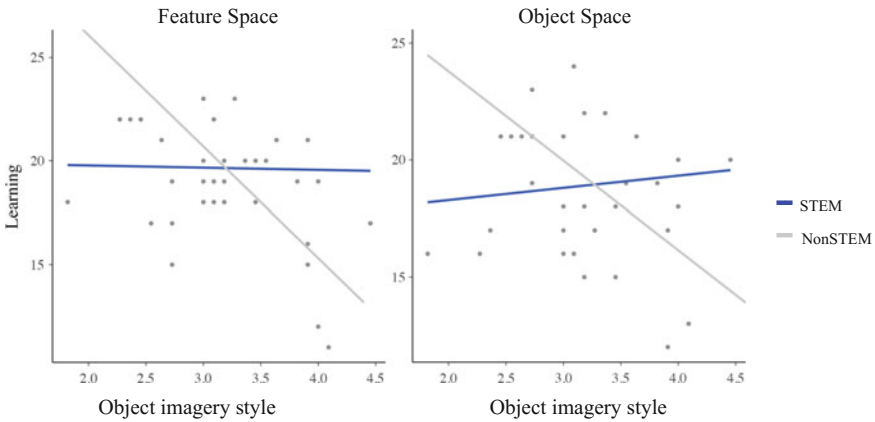


Fig. 4 Regression plots of object imagery scores and learning outcomes for both feature (left) and object space (right)

and field knowledge ($F(60) = -.68, p = .08$) on general learning with no interaction effects were found in this model. Regarding model 3, only field knowledge was found to have a main effect on learning, $F(60) = -1.24, p = .002$. Similarly, in model 4, only field knowledge’s main effect on learning was found, $F(60) = -1.09, p = .003$. Within model 5, two main effects were found for object imagery ($F(39) = 2.19, p = .03$) and field knowledge ($F(39) = .60, p = .0001$) on user experience which provided partial support for H1 and H3. In models 6, 7, and 8, only field knowledge shows the significant main effect on user experience ($F(39) = -.42, p = .006, F(39) = -4.7, p = .001, F(39) = -.52, p = .0002$, respectively) further partially supporting H1.

Regression analyses were conducted to assess the effects of cognitive style on learning and user experience between the two DSS interfaces. Results show partial support for H3 such that only the effects of object-spatial cognitive style on learning within the feature space showed any significance and no support for H2 as no significant findings were found for rational-experiential cognitive styles. The regression plots in Fig. 4 show interaction effects between object imagery scores and field knowledge on learning outcomes. In both plots, higher object scores predict lower learning outcomes among non-STEM majors; however, this relationship is only significant within the feature space ($p = .0044$). However, among STEM majors, higher object scores show better learning outcomes within the object space, which suggest partial support for H5. In this case, object imagery was a significant predictor learning outcomes within the feature space, $b = -1.82, t(32) = -2.39, p = .023$; spatial imagery was a stronger predictor with $b = 2.54, t(32) = .52, p = .0015$; lastly, when considering imagery dominance, the difference in object and spatial imagery scores also significantly predicted learning within the feature space, $b = -1.72, t(32) = -.57, p < .001$ (Table 7).

Table 5 Effects of cognitive styles and field knowledge on learning: mixed model results

<i>Mixed model 1: Object, STEM, DSS</i>				<i>Mixed model 2: Spatial, STEM, DSS</i>					
Fixed effects	B	SE	F	p	Fixed effects	B	SE	F	p
[Object]	-2.20	.94	5.49	.02*	[Spatial]	1.75	.72	5.89	.02*
[STEM]	.05	.57	.01	.93	[STEM]	-68	.38	3.13	.08
[DSS]	-.29	.93	.26	.62	[DSS]	-.08	.38	.05	.83
[Object × STEM]	-2.41	.94	6.58	.01*	[Spatial × STEM]	1.17	.72	2.66	.11
[Object × DSS]	.55	.94	.34	.56	[Spatial × DSS]	-.69	.72	.92	.34
[STEM × DSS]	.08	.57	.02	.88	[STEM × DSS]	.22	.38	.33	.57
[Object × STEM × DSS]	.24	.94	.06	.80	[Spatial × STEM × DSS]	.46	.72	.41	.53
<i>Mixed model 3: Rational, STEM, DSS</i>				<i>Mixed model 4: Experiential (Exp), STEM, DSS</i>					
Fixed effects	B	SE	F	p	Fixed effects	B	SE	F	p
[Rational]	-.34	.67	.26	.61	[Experiential]	-.48	.64	.57	.45
[STEM]	-1.24	.38	10.59	.002**	[STEM]	-1.09	.36	9.29	.003**
[DSS]	-.00	.38	.00	.99	[DSS]	-.16	.36	.21	.65
[Rational × STEM]	-.68	.67	1.03	.31	[Exp × STEM]	.67	.64	1.11	.30
[Rational × DSS]	-.67	.67	1.02	.32	[Exp × DSS]	.10	.64	.03	.87

(continued)

Table 5 (continued)

[STEM × DSS]	.25	.38	.43	.52	[STEM × DSS]	.29	.36	.66	.42
[Rational × STEM × DSS]	.83	.67	1.55	.22	[Exp. × STEM × DSS]	.42	.64	.43	.51

* $p < .05$, ** $p < .01$, *** $p < .001$, **** $p < .0001$, DV: Learning (test score), random effect: participants' individual effects

Table 6 Effects of cognitive styles and field knowledge on user experience: mixed model results

<i>Mixed model 5: Object imagery, STEM, DSS</i>							<i>Mixed model 6: Spatial imagery, STEM, DSS</i>							
Fixed effects	B	SE	F	p	Fixed effects	B	SE	F	p	Fixed effects	B	SE	F	p
[Object]	2.19	.95	5.11	.03*	[Spatial]	.27	.28	.92	.34	[STEM]	-.42	.14	8.48	.006***
[STEM]	.60	.27	18.17	.0001***	[DSS]	.10	.10	1.07	.31	[Spatial × STEM]	.32	.28	1.30	.26
[DSS]	.06	.12	.29	.59	[Object × DSS]	-.06	.20	.08	.77	[Spatial × DSS]	.12	.19	.37	.55
[Object × STEM]	-.07	.27	.07	.79	[STEM × DSS]	-.06	.12	.24	.63	[Object × STEM × DSS]	.09	.20	.20	.66
[Object × DSS]	-.06	.20	.08	.77										
[STEM × DSS]	-.06	.12	.24	.63										
[Object × STEM × DSS]	.09	.20	.20	.66										

<i>Mixed model 7: Rational, STEM, DSS</i>							<i>Mixed model 8: Experiential (Exp), STEM, DSS</i>							
Fixed effects	B	SE	F	p	Fixed effects	B	SE	F	p	Fixed effects	B	SE	F	p
[Rational]	.17	.24	.49	.49	[Experiential]	.14	.20	.50	.50	[STEM]	-.52	.12	17.32	.0002***
[STEM]	-.47	.13	12.15	.001**	[DSS]	.08	.09	.84	.36	[Exp. × STEM]	.32	.20	2.52	.12
[DSS]	.07	.09	.59	.45	[Rational × STEM]	.03	.24	.02	.90	[Exp. × DSS]	.03	.13	.16	.69
[Rational × STEM]	.03	.24	.02	.90										
[Rational × DSS]	.13	.16	.68	.42										

(continued)

Table 6 (continued)

[STEM × DSS]	-.06	.09	.42	.52	[STEM × DSS]	-.07	.09	.72	.40
[Rational × STEM × DSS]	-.17	.16	1.15	.29	[Exp. × STEM × DSS]	.08	.14	.29	.59

* $p < .05$, ** $p < .01$, *** $p < .001$, **** $p < .0001$, DV: User Experience, random effect: participants' individual effects

Table 7 Significant effects of cognitive styles on learning in feature design support systems: regression analysis results

IV: Cognitive style	DV: Learning with feature DSS			
	<i>b</i>	β	<i>t</i>	<i>p</i>
Object imagery	-1.82	-.39	-2.39	.023*
Spatial imagery	2.54	.52	3.46	.0015**
Δ Obj.-spat. imagery	-1.72	-.57	-3.94	.0004***

* $p < .05$, ** $p < .01$, *** $p < .001$, **** $p < .0001$

Conclusion and Discussion

The main findings of the study show a significant effect of field knowledge on complex design DSS learning and user experience, which also aligned with previous findings that were discussed in the literature review. Higher field knowledge, which in this case was determined by STEM major fields, predicts higher learning outcomes and better user experience scores, which supports H1. In terms of cognitive style, only the dimensions of object-spatial visualization styles were found to have significant effect whereas rational-experiential cognitive styles did not have any significant influence. Thus, H2 was not supported and H3 was partially supported. Object scores were found to have a main effect on learning and user experience; spatial scores were found to have a main effect on learning alone. Furthermore, there is an interaction effect of visualization style, field knowledge and DSS interface design where higher object imagery scores can predict lower learning outcomes among non-STEM majors within the feature space condition. However, within the STEM majors, higher object imagery scores may predict better learning outcomes in the object space condition and thus suggest partial support for H5. There was no empirical support for H4.

These preliminary findings present exciting opportunities for potential applications in the field of DSS interface design. The interaction effect between cognitive style, field knowledge and DSS interface on learning and user experience suggests that information on these user characteristics can inform designers to optimize learning and user experience. Ultimately, we aim to contribute to the effort of closing the gaps between different underrepresented groups in engineering and design through researching how systems can be better designed to accommodate different cognitive styles, skill sets, and experiences. This would also better support collaboration across more diverse groups of engineers and designers, as diverse teams often are more successful at tackling complex issues. Given the resources, there were limitations to the study. We explored a very specific satellite design context among a small sample of university students, thus decreasing external validity. Moreover, we were unable to delve into the details of how individuals interacted with the interface and thus only general relationships were observed and no mechanisms could be concluded. However, the findings paved ways for future studies to further examine the underlying

mechanisms as well as to explore the effects of different cognitive style dimensions, DSS interface designs, and design problem contexts.

References

1. Power DJ, Sharda R (2000) Supporting business decision-making. In: Decision support systems hyperbook. DSS Resources, Cedar Falls, IA, pp 356–374
2. Rogers JL (1996) DeMAID/GA: an enhanced design manager's aid for intelligent decomposition. In: 6th AIAA/USAF/NASA/ISSMO symposium on multidisciplinary analysis and optimization, pp 1497–1504
3. Todd P, Benbasat I (1999) Evaluating the impact of DSS, cognitive effort, and incentives on strategy selection. *Inf Syst Res* 10(4):3
4. Benbasat BI (1977) Cognitive style considerations in DSS design. In: Proceedings of a conference on decision support systems, pp 37–38
5. Sternberg RJ, Grigorenko EL (1997) Are cognitive styles still in style? *Am Psychol* 52(7):700–712
6. Witkin HA, Moore CA, Goodenough D, Cox PW (1977) Field-dependent and field-independent cognitive styles and their educational implications. *Rev Educ Res* 47(1):1–64
7. Pask G (1976) Styles and strategies of learning. *Br J Educ Psychol* 46(2):128–148
8. Schroeder RG, Benbasat I (1975) An experimental evaluation of the relationship of uncertainty in the environment to information used by decision makers. *Decis Sci* 6:556–5567
9. Benbasat I, Dexter AS (1982) Individual differences in the use of decision support aids. *J Account Res* 1–11
10. Crossland MD, Herschel RT, Perkins WC, Scudder JN (2000) The impact of task and cognitive style on decision-making effectiveness using a geographic information system. *J Organ End User Comput* 12(1):15–25
11. Davis DL, Elnicki RA (1984) User cognitive types for decision support systems. *Omega* 12(6):601–614
12. Green GI, Hughes CT (2015) Effects of decision support systems training and cognitive style on decision process attributes. *J Manage Inf Syst* 3(2):83–93
13. Ramamurthy K, King WR, Premkumar G (1992) User characteristics–DSS effectiveness linkage: An empirical assessment. *Int J Man Mach Stud* 36(3):469–505
14. Huber GP (1983) Cognitive style as a basis for MIS and DSS designs: much ADO about nothing? *Manage Sci* 29(5):567–579
15. Blajenkova O, Kozhevnikov M (2008) The new object-spatial-verbal cognitive style model: theory and measurement. *Appl Cogn Psychol* 20:239–263
16. Epstein S (1994) Integration of the cognitive and the psychodynamic unconscious. *Am Psychol* 49(8):709–724
17. Shiloh S, Salton E, Sharabi D (2002) Individual differences in rational and intuitive thinking styles as predictors of heuristic responses and framing effects. *Pers Individ Dif* 32(3):415–429
18. Kozhevnikov M, Kosslyn S, Shephard J (2005) Spatial versus object visualizers: a new characterization of visual cognitive style. *Mem Cognit* 33(4):710–726
19. Höffler TN, Koć-Januchta M, Leutner D (2017) More evidence for three types of cognitive style: validating the object-spatial imagery and verbal questionnaire using eye tracking when learning with texts and pictures. *Appl Cogn Psychol* 31(1):109–115
20. Blajenkova O, Kozhevnikov M, Motes MA (2006) Object-spatial imagery: a new self-report imagery questionnaire. *Appl Cogn Psychol* 20:239–263
21. Thomas PR, McKay JB (2010) Cognitive styles and instructional design in university learning. *Learn Individ Differ* 20(3):197–202
22. Florance V, Marchionini G (1995) Information processing in the context of medical care. In: *Sigir '95*, pp 158–163

23. Patel AD, Gibson E, Ratner J, Besson M, Holcomb PJ (1998) Processing syntactic relations in language and music: an event-related potential study. *J Cogn Neurosci* 10(6):717–733
24. McDonald S, Stevenson RJ (1998) Effects of text structure and prior knowledge of the learner on navigation in hypertext. *Hum Factors J Hum Factors Ergon Soc* 40(1):18–27
25. Ghinea G, Chen SY (2003) The impact of cognitive styles on perceptual distributed multimedia quality. *Br J Educ Technol* 34(4):393–406
26. Chen S (2002) A cognitive model for non-linear learning in hypermedia programmes. *Br J Educ Technol* 33(4):449–460
27. Mayer RE (2002) Multimedia learning. *Psychol Learn Motiv* 41:85–139
28. Mitchell TJJ, Chen SY, Macredie RD (2005) The relationship between web enjoyment and student perceptions and learning using a web-based tutorial. *Learn Media Technol* 30(1):27–40
29. Bang H, Selva D (2016) iFEED: interactive feature extraction for engineering design. In: Proceedings of the ASME 2016 international design engineering technical conference & computers and information in engineering conference IDETC/CIE 2016, pp 1–11
30. Pacini R, Epstein S (1999) The relation of rational and experiential information processing styles to personality, basic beliefs, and the ratio-bias phenomenon. *J Pers Soc Psychol* 76(6):972–987
31. Björklund F, Bäckström M (2008) Individual differences in processing styles: validity of the rational-experiential inventory. *Scand J Psychol* 49(5):139–446

What Do Experienced Practitioners Discuss When Designing Product/Service Systems?



Abhijna Neramballi, Tomohiko Sakao and John S. Gero

This paper presents empirical results aimed at increasing the understanding of conceptual activities of Product/Service Systems (PSS) design by experienced designers from industry. Results are derived from a protocol analysis of five PSS design sessions, using the Function–Behavior–Structure coding scheme. Sessions included five pairs of professional designers and the task was to redesign a concept for an existing PSS to improve its resource efficiency. The results show (i) the distribution of design issues during PSS design sessions, (ii) on average 47% of the overall cognitive design effort spent by the designers is related to behavior, and (iii) all the design issues except requirements are constantly focused on during the entirety of the design sessions. Major differences compared to product design are the average occurrence of function for PSS design (23%) for product design (4%) and of structure for PSS design (22%) compared to the product design (35%).

Introduction

Today, a large number of companies increasingly provide a combination of products and services, both in the manufacturing (e.g., [1]) and service industries (e.g., [2]). Such an offering is called a Product/Service System (PSS), which is defined as “tangible products and intangible services designed and combined so that they jointly are capable of fulfilling specific customer needs” [3]. The services here include consultation, user support, inspection, maintenance, refurbishing, repair, product take-back, and upgrade (see, e.g., [4]). Those companies focus on value [5] to be

A. Neramballi (✉) · T. Sakao
Department of Management and Engineering, Linköping University, Linköping, Sweden
e-mail: abhijna.neramballi@liu.se

J. S. Gero
University of North Carolina at Charlotte, Charlotte, USA

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_20

created, rather than products or services as such. The products and services are a means for the value and are exchangeable with each other in designing a PSS [6]. For instance, a PSS with aircraft engines may provide high engine availability as a value for the user, which can be realized by increased durability of critical components or by arranging maintenance engineers to provide more timely and efficient maintenance services. This industrial phenomenon creates sources of innovation, as it provides possibilities to design new offerings.

However, it is typical that a traditional manufacturer provides services by first carrying out product design and then service design in a sequential manner [7]. This is not an adequate way of designing combined systems, as products and services in a PSS depend on each other [8] and such a sequential design requires multiple iterations, thus decreasing the efficiency of the design. As a consequence, PSS design needs to be different from product design (e.g., [9]), service design (e.g., [10]), or a sequential combination of them. What is needed is to design products and services in an integrated manner for the value to be created.

Some prescriptive models and methods for PSS design have been proposed (e.g., [11]). However, scientific understanding of how PSS design is conducted, especially regarding insights based on empirical research, has rarely been documented. For instance, what is discussed during PSS design and what processes occur are yet to be adequately explored. This paper targets this lack of detailed knowledge about PSS design.

Aim

In order to fill the knowledge gap identified above, this paper aims at increasing the understanding of the activities during the conceptual phase of a PSS design. Conceptual design is targeted in this paper because it involves activities characteristic to PSS design such as utilizing exchangeability between products and services [6]. Exchangeability here means being able to exchange efforts for service and product design to improve the overall PSS characteristics of interdependent products and services (ibid). In addition, conceptual design is an early phase of design, and thus more influential on the overall performance of the design. Understanding conceptual design activities are expected to contribute more to the understanding of PSS design than other phases.

Designing a physical product is utilized as a reference to compare characteristics of PSS design with product design. There is already substantial literature showing the characteristics of product design (e.g., [12–16]).

Significance

This paper is based on a limited dataset derived from a small cohort of practicing, professional PSS designers sufficient to answer research questions and to form the basis of hypotheses that can be tested with a statistically significant cohort size. A

major contribution of the increased understanding expected from this paper will lie in the effective design of further research deriving and testing a set of hypotheses regarding distributions of design issues in the conceptual design of a PSS. This further research will result in more grounded, generalizable knowledge about PSS design.

There is an increasing need in our societies for PSS design. Customers' needs for servitized offerings [8] and fierce competition are often the major motivations for providing a PSS. However, a PSS is also expected to contribute to decreased environmental impacts, for example, in resource consumption [17]. The PSS is reported to have the potential to increase product life [18] and to decrease lifecycle environmental impacts [19]. It has indeed been heralded as one of the most effective instruments for enhancing resource efficiency [20]. Therefore, improving PSS design is often demanded by industry and by society at large.

Research Questions

This section analyses the PSS literature to derive characteristics of a PSS compared to pure physical products. Based on these derived characteristics, it further reasons about their implications on the conceptual design of a PSS compared to that of products.

Pure physical products are material intensive and most of their added value is derived from the manufacturing processes that transform raw materials into the final product [21]. In contrast to the design of a PSS, when a product is designed its potential service aspects in the use phase of a product lifecycle are less emphasized (*ibid*). Furthermore, previous research has shown that the structure of the design object recurs frequently in the cognitive activities of designers during product design, and was found to be the dominant design issue [12, 22, 23]. This dominance of structure over other design issues such as behavior and function could be attributed to the lack of a systems perspective in the product design [23].

On the other hand, a major property of a PSS is its open process systems [24]. This means that a PSS is a system with input and output flows. Output flows are determined by processes in a PSS, which can be used to describe functions and human activities. The human activities [25] are characterized by heterogeneity inherited from generic characteristics of pure service [26]. Pure physical products cannot be considered as open process systems in the same way as a PSS, due to the absence of output flows characterized by human activities and services, which is prevalent in a PSS.

Further, a PSS is characterized by interdependency between product and service [8] and thus interaction between them [27]. This means that the conceptual design of a PSS requires simultaneous and conflicting product and service engineering [8]. It is more complex than the product component or the service component within the PSS. The complexities in conceptual design of a PSS highlight the need for systems thinking [28]. For designing a system, behavior as a system needs to be analyzed. Behavior of elements is relevant to design in general [29], however, the higher level of complexity of a PSS makes the behavior as a system especially relevant in the conceptual design of a PSS. This may be applied to function as well as behavior.

These lead to the research questions (RQs) below. RQ1 is an elucidatory question that provides the basis for further research. RQ2 focuses on the behavior based on the discussion above. RQ3 refers to product design and is found on the notion that since there are two disparate subsystems in PSS will there be more discussion about functions that are distributed between them.

- RQ1: What are the distributions of design issues and design processes in the conceptual design of a PSS?
- RQ2: Is behavior the dominant design issue in the conceptual design of a PSS?
- RQ3: Is function as a design issue more dominant in the conceptual design of a PSS than in the design of a product alone?

Methods

Protocol analysis is adopted in this research to examine the research questions, as it provides empirically based quantitative evidence as well as rich qualitative information. Protocol analysis is a rigorous methodology for eliciting verbal reports of thought sequences as a valid source of data on thinking. It is also a well-developed, validated method for the acquisition of data on thinking [30, 31]. It has been used extensively in design research, for example, in exploratory studies and hypothesis testing, to help the development of the understanding of the cognitive behavior of designers [32–38].

This research utilizes a method for determining and describing design cognition using a coding scheme based on the Function–Behavior–Structure (FBS) ontology [39]. This is a design ontology that is independent of the design task, the designer's experience and the design environment, and hence produces commensurable results from different experiments [14, 40–43]. It is, therefore, suitable for use in analyzing PSS design in comparison with other types of design. The FBS ontology provides a uniform framework for classifying cognitive design issues and cognitive design processes and includes higher level semantics in their representation.

The FBS ontology [39] models designing in terms of three classes of ontological variables, namely function, behavior, and structure, plus two variables that are expressible in terms of requirements and design description. In this view, the goal of the design is to transform a set of functions, driven by the client requirements (R), into a set of design descriptions (D). The *function* (F) of a designed object is defined as its intended purpose, expectations, or teleology, while the *behavior* (B) of that object is either derived (Bs) or expected (Be) from the structure, where *structure* (S) represents the components of an object and their relationships.

The FBS ontology has been referenced extensively as an ontology of design that has been used in various disciplines, and one that transcends individual designers, the design task, the design environment, and whether the design is done individually or in teams [14, 44–49].

Materials from the PSS Design Case

The data for the protocol analysis was derived from design sessions conducted in a laboratory environment with experienced practitioners as participants who were given a PSS redesign task. The aim of this task was to redesign a concept for an existing PSS to improve its resource efficiency in terms of material and energy use. A conceptual design was chosen for this study as it is characteristic of a great deal of PSS design, for example, by permitting exchangeability between products and services of the systems which are essential for PSS design [6].

A laboratory environment was chosen over an industrial setting, partly because the prospect of shadowing and monitoring the participants as they carry out design activities in their own industrial setting was ruled out due to confidentiality reasons. In addition, the participants are employees of different manufacturing companies working with different types of PSSs. This would increase the variability of the unit of analysis and characteristics of data, potentially reducing the internal validity of the study. Furthermore, previous research suggests that the design activities of products and services are usually separated in industrial environments and are not integrated at the required level [7]. Hence, all the participants were asked to perform the same design task in a controlled laboratory environment, thus providing the opportunity for unrestricted collaboration between product and service design.

A design brief (see [Appendix](#)), which included the information necessary to carry out the task, was provided to the participants beforehand. The PSS utilizes a coffee machine used in offices provided by a hypothetical firm that develops, manufactures, and delivers the product and related services to its clients. The machine utilized is an actual model available on the real market, but the details of the real provider are not disclosed to the participants. A detailed specification of the machine and the services was provided in the design brief, and the participants were also allowed to visually inspect the physical model. The firm's service portfolio included activities such as installation of the machines, replenishment of the consumables, maintenance, repair, and overhaul.

There were 10 participants in total, and they were instructed to perform the task in pairs. In each pair, one participant was instructed to assume the responsibility of a service designer, while the other took the role of a product designer. This instruction was given based on their background and work experience. No information regarding design tools or methods was given to the participants to prevent external influence on the design process. They were provided with a poster-sized sheet of blank paper, post-its, and pens in different colors for the design task. The tasks of all the pairs were documented simultaneously using both audio and video recording devices in parallel sessions to obtain richer cognitive information from both verbal and nonverbal actions of the participants [23]. The data collected was later subjected to protocol analysis. The language used during the sessions was Swedish, which was the language spoken daily by all the participants.

The participants are experienced product or service designers with an average experience of 9 years, with a standard deviation of 5.34 years. They work for leading

Table 1 Excerpt of the segmented protocols (translated into English)

Designer	Segment	Code
A	(Writes)	D
A	Service report	S
B	When it is used	Be
A	And what it is used for I saw that it was	F
B	Which I interpret as each one has his own card	Be
B	And then you go and take coffee	Bs

manufacturing industries that provide PSSs in different sectors such as automotive, electronics, composite components, and grinding machines.

The audio and video recordings of the sessions were initially transcribed before being segmented and coded using the FBS ontology by two independent coders. The results of both coders were compared and arbitrated by a third coder, who also made the final decision regarding the codes. The length of the sessions on average was 75 min, with a standard deviation of around 10 min.

Results of Protocol Analysis

Overview

The utterances were segmented and coded, the segments that did not belong to any of the design issues of the FBS ontology were considered as noise and were removed. After the removal of the noise, an average of 958 segments per session, with a standard deviation of around 214 segments, were subjected to further analysis. Table 1 illustrates an excerpt of the segmentation and coding of the transcribed data collected from one of the design sessions, aiming mainly to show examples of utterances for most of the design issues. The two coders carried out their coding independently and then a third person carried out an arbitration. A simple statistical measure of agreement between each coder's codes and the final arbitrated codes are used as the reliability measure since Cohen's kappa was not applicable in this setting. This was done by calculating the ratio of the sum of number of agreements between the individual coders coding and the arbitrated codes, over the total number of codes. The two coders had an average of 71% agreement with the final arbitrated codes for all the design sessions.

Table 2 Design issue distribution [%]

	S1	S2	S3	S4	S5	Mean	σ	CV
Function (F)	20.5	26.2	26.2	18.0	23.7	22.92	3.23	0.14
Expected Behavior (Be)	18.2	21.6	15.1	17.5	25.4	19.56	3.58	0.18
Behavior of structure (Bs)	33.3	26.3	24.2	30.8	22.8	27.48	3.97	0.14
Structure (S)	19.4	20.1	28.7	19.6	23.3	22.22	3.53	0.17
Design description (D)	8.4	5.5	5.0	13.8	4.7	7.48	3.42	0.45

Note “Sn” means “Session n”, σ standard deviation, CV Coefficient of variation

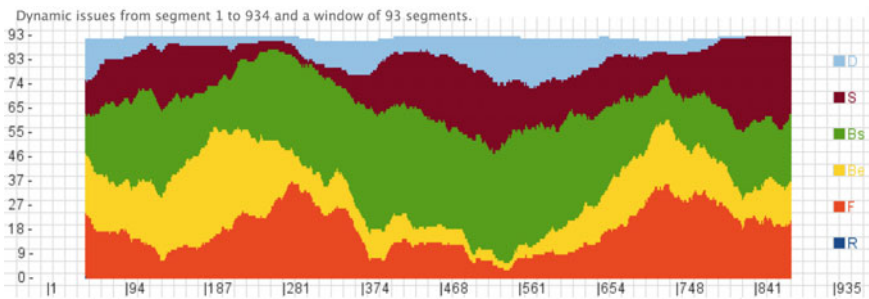


Fig. 1 Moving average of cognitive effort expended on design issues, Session 1

Design Issue Distribution

The distribution of occurrence of design issues over the five design sessions is shown in Table 2. The data for R are not reported here as the number of occurrences was too low for this analysis. Design issues, Be and Bs, have an average percentage of occurrence of 19.56 and 27.48%, respectively, over the five design sessions. These two design issues together represent behavior, having an average of 47.04% of the overall design cognitive effort spent by the designers during all five sessions. Almost half of the design issue occurrence is accounted for by behavior in comparison with F (22.92%), S (22.22%), and D (7.48%). The low values of coefficient of variations of all the design issues over different sessions indicate low levels of variance within this limited data set.

The moving averages of the cognitive design effort expended on design issues over the five design sessions are presented in Figs. 1, 2, 3, 4, and 5. The figures are generated using LINKODER, a publicly available software application (linkoder.com). Moving average window lengths of 93, 96, 84, and 135 segments were used, which correspond to 1/10th of their respective, complete sessions. This normalizes the data in the figures.

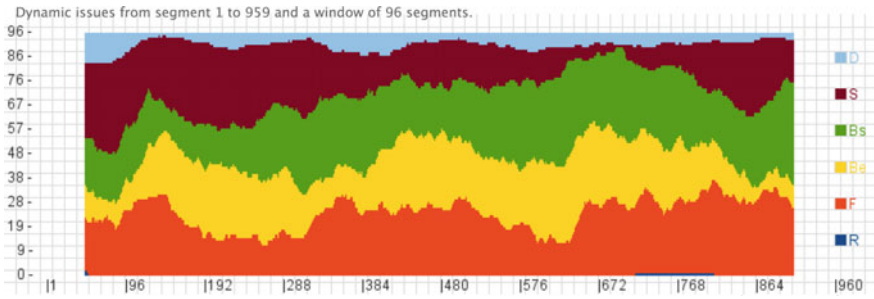


Fig. 2 Moving average of cognitive effort expended on design issues, Session 2

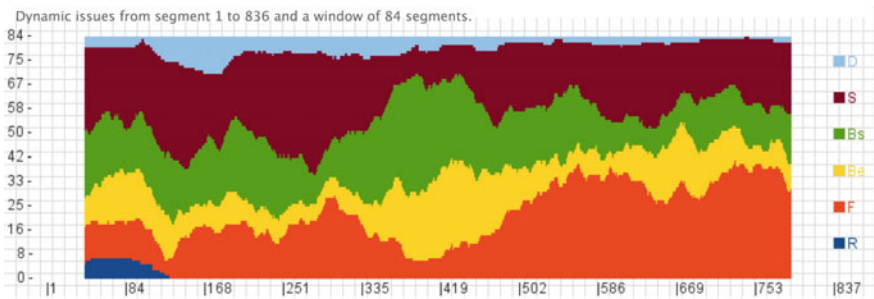


Fig. 3 Moving average of cognitive effort expended on design issues, Session 3

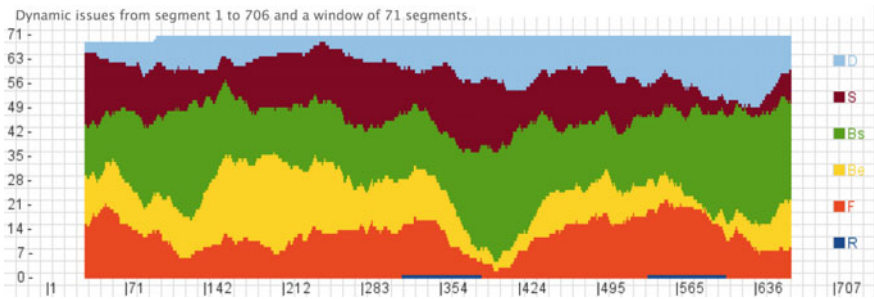


Fig. 4 Moving average of cognitive effort expended on design issues, Session 4

These visualizations qualitatively indicate that the cognitive design effort for the design issues vary significantly over time. These graphical figures provide opportunities for qualitative interpretation of the results, and they visualize the dominance of the design issue behavior (Be and Bs) during the transition over the different segments. All the figures illustrate that the cognitive effort expended on behavior increases during the middle of the sessions.

The cumulative occurrences of design issues in the protocol of the first session are presented in Fig. 6. An analysis of the data supporting Fig. 6 provides quantitative

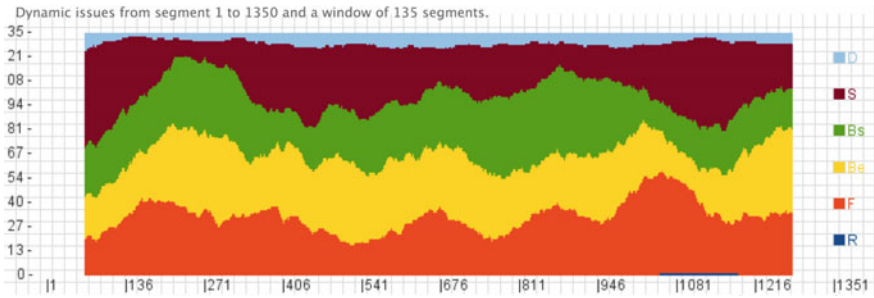


Fig. 5 Moving average of cognitive effort expended on design issues, Session 5

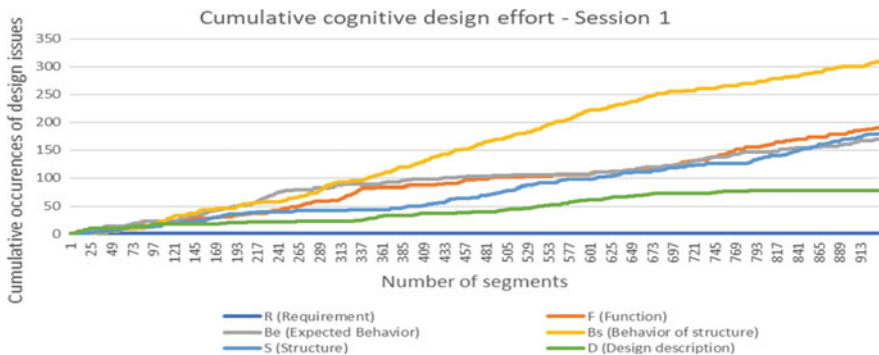


Fig. 6 Cumulative cognitive effort expended on design issues in Session 1

and qualitative insights into the distributions of design issues. It is evident that at the end of the design session the designers had expended the highest cognitive design effort on Bs.

To quantify the shape of each graph, a linear approximation was conducted for each design issue’s cumulative occurrence across each session. Figure 7 illustrates such an example of the linear approximation of the cumulative occurrence of the design issue Bs for Session 1.

The coefficient of determination for Bs in this session is 0.9910, which indicates a high degree of linearity. The coefficients of determination of the cumulative occurrence of the design issues of all the sessions with the exception of R, since there is insufficient data to carry out the calculation reliably, are shown in Table 3. Design issue Bs has the highest linearity with an average coefficient of determination (R^2) of 0.9918 over the five sessions. It is closely followed by F (0.9833), S (0.9797), Be (0.9758), and D (0.9685), which are higher than the threshold of linearity, and which requires R^2 to be equal to or greater than 0.95. This suggests that all the other design issues are regularly focused on by the designers during the sessions.

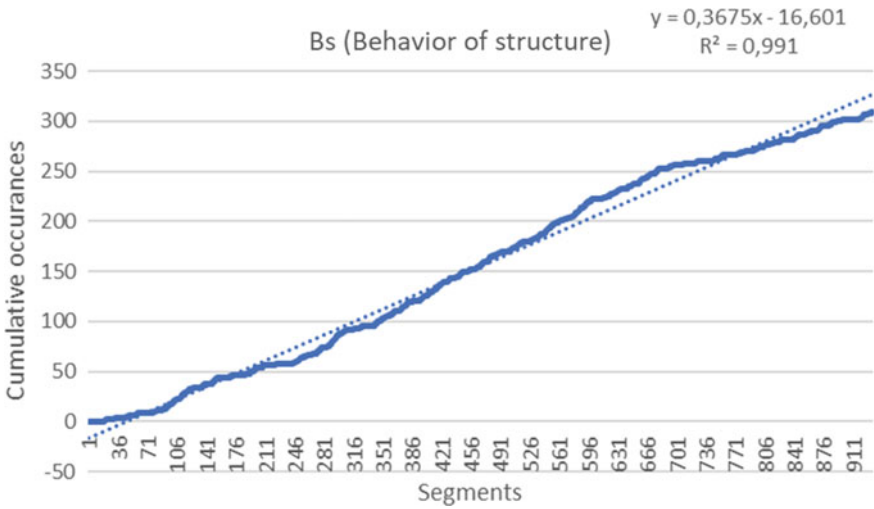


Fig. 7 Linear approximation of the cumulative occurrences of design issue Bs, Session 1

Table 3 Coefficients of determination from linear approximation of the transition per design issue

	Session 1	Session 2	Session 3	Session 4	Session 5	Average
F	0.9818	0.9934	0.9573	0.9911	0.9929	0.9833
Be	0.9486	0.9926	0.9853	0.9568	0.9959	0.9758
Bs	0.9910	0.9915	0.9870	0.9944	0.9953	0.9918
S	0.9760	0.9521	0.9907	0.9864	0.9933	0.9797
D	0.9641	0.9889	0.9277	0.9693	0.9925	0.9685

Syntactic Design Process Distribution

The distribution of the syntactic design processes defined by the FBS ontology of all the sessions is shown in Table 4. This distribution is given in terms of percentage of the ratio of occurrence of each process over that of all the eight processes. A unidirectional process between the design issues is represented by “→”, while a bidirectional process between the design issues is represented by “-”.

“Reformulation 1” has the highest average percentage of occurrence with 22.0%. This is followed by “Evaluation” with 21.12%, which is a bidirectional syntactic design process between Be and Bs. The low values of coefficient of variations of all the design issues over different sessions indicate low levels of variance within this limited data set.

The moving averages of syntactic design processes of all the sessions are presented in Figs. 8, 9, 10, 11, and 12. These figures are also generated using LINKODER. Moving average windows of lengths of 234, 240, 209, 177, and 337 segments are used which correspond to a quarter of their respective complete sessions, to normalize the

Table 4 Syntactic design process distributions, expressed as percentages

	S1	S2	S3	S4	S5	Mean	σ	CV
F → Be	14.6	9.8	5.8	11.5	13.8	11.1	3.14	0.28
Be → S	9.9	8.3	6.1	9.9	10.9	9.02	1.68	0.18
S → Bs	17.0	13.2	16.0	18.5	12.9	15.52	2.17	0.13
Be–Bs	20.4	22.8	18.6	21.8	22.0	21.12	1.47	0.06
S → D	6.8	3.4	4.9	5.8	2.9	4.76	1.45	0.30
S → S	18.3	24.6	33.4	16.0	17.7	22.0	6.40	0.29
S → Be	7.7	8.9	7.0	8.2	9.9	8.34	0.99	0.11
S → F	5.3	8.9	8.1	8.2	9.9	8.08	1.53	0.18

Note “Sn” means “Session n”, σ standard deviation, CV Coefficient of variation. F → Be: Formulation, Be → S: Synthesis, S → Bs: Analysis, Be–Bs: Evaluation, S → D: Documentation, S → S: Reformulation 1, S → Be: Reformulation 2, S → F: Reformulation 3

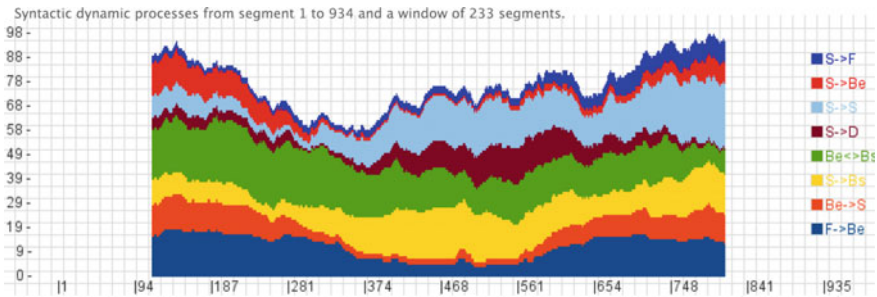


Fig. 8 Moving average of syntactic design processes, Session 1

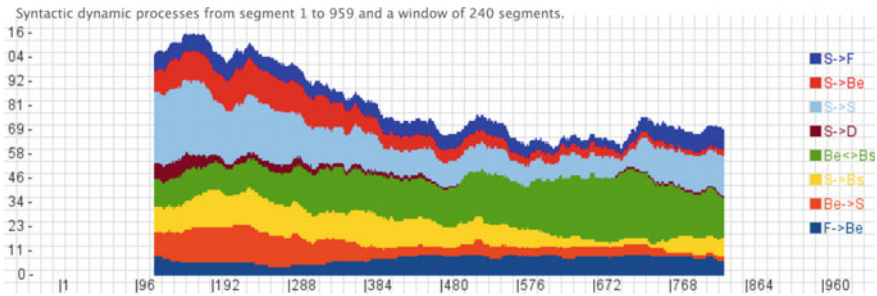


Fig. 9 Moving average of syntactic design processes, Session 2

data in the figures. These figures show that the syntactic design processes change over time and also qualitatively confirm the dominance of Reformulation 1, Evaluation, and Analysis over the five sessions established by the quantitative findings presented in Table 4.

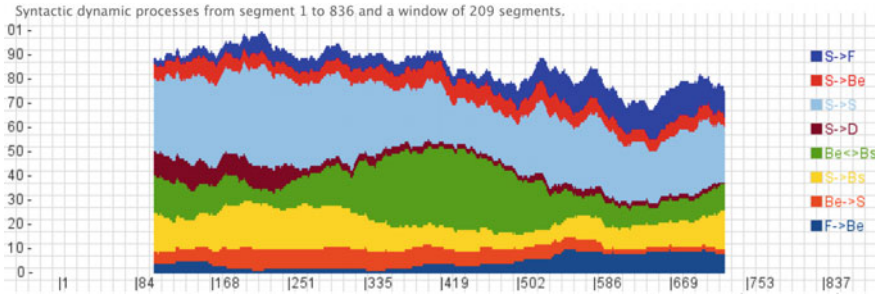


Fig. 10 Moving average of syntactic design processes, Session 3

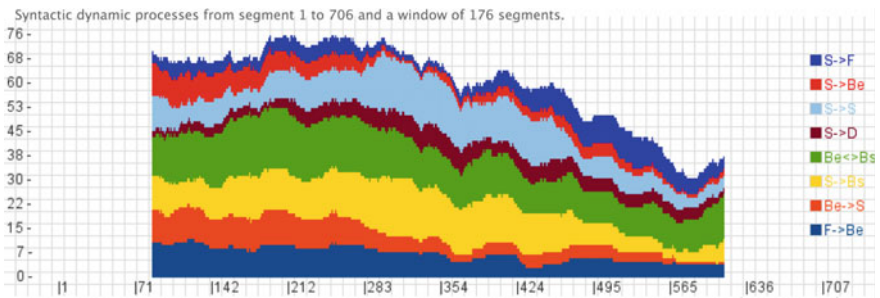


Fig. 11 Moving average of syntactic design processes, Session 4

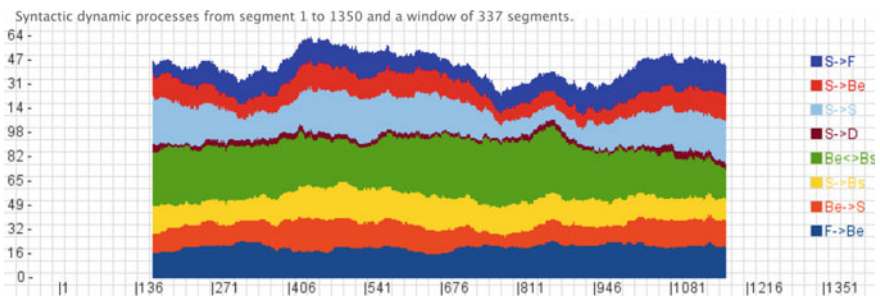


Fig. 12 Moving average of syntactic design processes, Session 5

Comparative Analysis of PSS and Product Design Sessions

Multiple empirical studies of product designs have utilized the protocol analysis method with the FBS-based coding scheme [12–15, 23, 43]. As a consequence, the results from all of these and related studies are commensurable with the results of the study of PSS design reported here. Published studies have been selected for comparison [50–52]. The first is from a brainstorming session in the industry [50]. The second is from 10 design sessions with undergraduate engineers studying mechani-

Table 5 Design issue distributions [%] from multiple studies of product design as compared to this study (of PSS design)

Study	Refs.	R	F	Be	Bs	S	D
This study	–	0.4	22.9	19.5	27.4	22.2	7.4
Brainstorming	[50]	1.8	4.3	17.3	28.5	37.1	11.0
Engineering Design major	[52]	2.5	6.2	5.9	30.0	37.2	15.5
Engineering Mechanics major	[52]	5.2	2.7	8.2	32.0	33.2	18.0
Software	[51]	0.2	0.0	30.1	19.4	30.3	15.9

cal design [52]. The third is from 10 undergraduate engineers studying engineering mechanics [52]. The fourth is from a software design session in the industry [51]. This produces results of a range of designers being studied. From these results, it should be possible to observe that product design sessions are similar to each other and that the PSS design session exhibits one or multiple significant differences.

The average percentage distribution of each of the six design issues for these four studies is presented in Table 5.

What can be observed from the results in Table 5 is that for all the product design sessions, the design issue of S is the dominant issue. However, for the PSS design session, the dominant design issue is Bs. In all of the four-product design results, F sits in the lower half of the distribution of design issues wherein the PSS design session it is the second most dominant.

Discussion

The results presented above form the basis for examining the research questions. Concerning RQ1—What are the distributions of design issues and design processes in the conceptual design of a PSS?—the highest design issue distributions from this study were given to Bs, F, S, and Be (in the descending order as shown in Table 2). These four issues in total received more than 90%, and each of them received a substantially high percentage (more than 18%). In addition, F, Be, Bs, S, and D are consistently focused on by the designers during the sessions. The highest process distributions were for Reformulation 1 (S → S), Evaluation (Be–Bs), and Analysis (S → Bs) (in the descending order as in Table 4). These three processes accounted for around 60% of all process activity. Each of all the eight syntactic processes received more than 5%.

Regarding RQ2—Is behavior the dominant design issue in the conceptual design of a PSS?—behavior was dominant (47.04% in total of Bs and Be in average as in Table 2). In addition, the processes involving behavior also exhibited high percentage

occurrences (Evaluation (Be–Bs) 21.12% and Analysis (S → Bs) 15.52% as shown in Table 4).

For RQ3—Is function as a design issue more dominant in the conceptual design of a PSS than in the design of a product alone?—the results of the comparative analysis above are used. These comparisons are qualitative at this stage since the experiment group results reported in this paper are based on a small cohort. The function was more dominant in PSS design of this study than in of the product design studies (see Table 5). The average percentage for PSS design was 22.9%, while the percentages for product design were between 0.0 and 6.2%.

Compared with product design in the industry (as shown by “Brainstorming” in Table 5), this study of PSS design in industry shows clear similarities and differences. The percent occurrences for R, Be, and Bs are relatively close to each other, while those for F and S show noticeable differences.

Since this study is based on only five sessions, the external validity of the results is limited. Due to the availability of limited data sets, only mean values are considered. However, these issues are countered by documenting the research in a standard and transparent manner, to accommodate the possibility of reproducing the results with larger data sets in the authors’ immediate future work. Also, standard deviation and coefficient of variation are used to measure the variability of the results of different sessions, which provides some insights into the statistical reliability of the limited data sets. These results establish the premises for the formulation of the following two hypotheses, which will be addressed in the future work: (i) the design issue ‘Behavior’, is dominant during conceptual design of PSS, (ii) ‘Analysis’ and ‘Evaluation’ are the dominant processes during conceptual design of PSS. More sessions from the same setting are required to derive statistically reliable results to test these preliminary hypotheses.

Other future works are planned as follows. First, transitions of cognitive efforts over time will be investigated more. For instance, questions such as “which parts of a design session received more efforts on a specific design issue?” may be addressed, as Figs. 1, 2, 3, 4, and 5 seem to exhibit differences in efforts on a design issue between parts. Second, differences between product designers and service designers will be examined in terms of design issues and processes. The authors began to make relevant hypotheses for this examination: e.g., presented in their own work [53] was “in PSS design, service designers address the customer more than the provider”. Third, the quality of different sessions will be analyzed, while this paper with the immediate future work mentioned in the last paragraph focuses on their quantitative aspect. The quality may include that of design solutions obtained from each session. A possible aim is then to extract patterns of design processes that produce solutions with higher quality. Finally, further comparisons are needed between PSS design when the design moves beyond conceptual design with product design to verify the differences seen in the results above.

Conclusion

This paper aims to increase the understanding of conceptual activities of Product/Service System (PSS) design by professional designers from industry. The exploratory study described in the paper adopted the protocol analysis method using the Function–Behavior–Structure coding scheme and presented empirical results. The results show that almost half of the overall cognitive design effort spent by the designers is related to behavior as a design issue. The design issues discussed most frequently were behavior derived from structure, function, structure, and behavior expected in descending order. In addition, a primary contrast was observed between the focus on function during PSS design sessions and that on structure in product-only design sessions. These results demonstrate that it will be possible to test the preliminary hypotheses by carrying out an experiment that produces statistically reliable results.

This study does not provide generalizable results due to the lack of a statistically reliable cohort size. To obtain additional data from an increased cohort size that enables reliable statistical analysis is an immediate future work by the authors.

Acknowledgements This research is supported in part by the Mistra REES (Resource Efficient and Effective Solutions) program funded by Mistra (The Swedish Foundation for Strategic Environmental Research) (grant number DIA 2014/16). It is also supported in part by a grant from the US National Science Foundation Grant No. CMMI-1400466 to the third author.

Informed consent was obtained from the participants of the design sessions on a voluntary basis and is in line with the regulations as described by Linköping University's Centre for Applied Research Ethics. The authors acknowledge their contribution to this research.

Appendix: Design Brief Used for the PSS Design Session

This appendix shows major parts of the design brief. The complete design brief is to be uploaded on the first author's ResearchGate page.

The company

The design is carried out for a company who develops, manufactures, and delivers coffee machines and related services. This hypothetical firm is named *Jobbkaffe* and is based in Sweden. Instructing on use, installing the machines, supplying consumables, and carrying out MRO (maintenance, repair, and overhaul) are part of the company's service portfolio.

Client of the company

University A, whose employees are mainly professors, PhD students, and administrative staff, is the client of *Jobbkaffe*. The employees and their guests want to get something warm to drink, typically early in the morning as well as during a morning break and an afternoon break.

Fig. 13 Photo of the product



Design object

The design object addressed is one of their major offerings, including both product and service. The product model is named Spengler PSL 50 BTC (a model existing on the real market—see Fig. 13) and is provided by *Jobbkaffe*. Instead of only providing a physical product, *Jobbkaffe* also provides service.

Design task

A redesign task is to be completed in a group of two practitioners working in a cooperative manner. Each group is demanded to derive a concept with the highest potential for the offering that improves resource efficiency within approx. 1 h. In the end, the group must describe the concept on a blank paper with text (drawings can also be used for clarification). Resource here means a natural resource such as material and energy, but not a human resource. In case specific information is not available, the group to make an assumption, e.g., material types. The language is preferred to be in English, but you may choose Swedish.

Deliverable of the task

The deliverable is a concept for the offering containing products and services described on a blank paper. The concept should be derived from the group discussion, including choices and reasons for the developed concept. The concept needs to have

sufficient information before detailed design begins. The improvement can be all on the products, the services, and the payment model, but could be on one or two of them.

Data of product model

Beverage types:

The product is equipped with a total of 7 different beverages with a counting system for selectable strength and size. The main ingredients are; freshly grinded coffee beans, chocolate powder and milk powder, which are used in different combinations.

How the product works:

A distinctive touch display instructs on the different beverage choices. The coffee machine brews coffee through a steel filter in 30 seconds. For beverages containing milk or chocolate the powder instantly blends with the water. The machine is equipped with a cup sensor that is used to pour out drink only when a cup is positioned. The coffee machine has a system that takes care of any residual liquid and coffee grounds and indicates when these have to be emptied. The residual liquid and coffee grounds are gathered in two separate dispensers and has to be emptied manually when full.

How the product is installed:

The machine is connected to the regular water system as e.g. a dishwasher and uses a regular wall socket for electricity.

Peripherals:

Cups made of paper are offered beside the machine, but mugs can also be used. There is a cup holder that smaller cups can be put on or it can be turned to the side in order to fit larger mugs. The machine is filled with coffee beans, milk powder, and chocolate powder. Other consumables e.g. tea bags and sugar has to be provided next to the machine.

Materials:

Outer casing: Painted or brushed stainless steel plates
Beverage dispenser: Styrene Acrylonitrile (SAN)
Lid to beverage dispenser: Polystyrene (PS)
Smaller plastic details: Polyamide (PA)
Others: Not marked

Other specifications of the product:

Height: 800 mm (1750 mm including base cabinet)
Width: 450 mm
Depth: 455 mm
Weight: 45 kg
Electrical connection: 230 Volt
Maximum power: 2300 W

Capacity: (Consumable; Amount per cup; In total):

Coffee beans; 2500 g; 15 g/150 ml
Hot chocolate; 1700 g; 21 g/150 ml
Milk; 1200 g; 6 g/150 ml

Data of related services

Jobbkaffe delivers warranty of quality, early information on the next maintenance (in case pre-ordered), and telephone support service. Daily check of machines, cleaning, waste removal, etc. and supply of filling in coffee beans, etc. is carried out by a cleaning service company working for Linköping University.

Payment model

The customer buys the coffee machine (the initial installation is included in the price). *Jobbkaffe* provides additional options on demand by the customer: buying consumables, as well as regular service and support in case of failure.

Reference :

The product model: <http://www.Jobmeal.se/sv/automater/kaffeautomater/p/psl-50btc>.

References

1. Baines TS, Bigdeli AZ, Bustinza OF, Ridgway K (2017) Servitization: revisiting the state-of-the-art and research priorities. *Int J Oper Prod Manage* 37(2):256–278
2. Sawhney M (2016) Putting products into services. *Harvard Bus Rev* 94(9):82–89
3. Tischner U, Verkuil M, Tukker A (2002) First draft PSS review. Econcept, Cologne
4. Sakao T, Napolitano N, Tronci M, Sundin E, Lindahl M (2008) How are product-service combined offers provided in Germany and Italy? Analysis with company sizes and countries. *J Syst Sci Syst Eng* 17(3):367–381
5. Ulaga W, Chacour S (2001) Measuring customer perceived value in business markets. *Ind Mark Manage* 30:525–540
6. Sakao T, Lindahl M (2015) A method to improve integrated product service offerings based on life cycle costing. *CIRP Ann Manuf Technol* 64(1):33–36
7. Matschewsky J, Kambanou ML, Sakao T (2017) Designing and providing integrated productservice systems—challenges, opportunities and solutions resulting from prescriptive approaches in two industrial companies. *Int J Prod Res* 56(6):2150–2168
8. Meier H, Roy R, Seliger G (2010) Industrial product-service systems—IPS². *CIRP Ann Manuf Technol* 59(2):607–627
9. Pahl G, Beitz W, Feldhusen J, Grote KH, Wallace K, Blessing LTM (2006) *Engineering design: a systematic approach*, 3rd edn. Springer, London
10. Verma R, Fitzsimmons J, Heineke J, Davis M (2002) New issues and opportunities in service design research. *J Oper Manage* 20(2):117–120
11. Morelli N (2003) Product-service systems, a perspective shift for designers: a case study: the design of a telecentre. *Des Stud* 24(1):73–99
12. Kannengiesser U, Gero JS (2015) Is designing independent of domain? Comparing models of engineering, software and service design. *Res Eng Des* 26:253–275
13. Gero JS, Jiang H, Vieira S (2013) Exploring a multi-meeting engineering design project. In: Chakrabarti A, Prakash RV (eds) 4th international conference on research into design (ICORD'13), Springer, Chennai, India, pp 73–84
14. Jiang H (2012) Understanding senior design students' product conceptual design activities. National University of Singapore, Singapore
15. Song T (2014) Expert vs. novice: problem decomposition/recomposition in engineering design. Utah State University, Utah

16. McDonnell J, Lloyd P (eds) Design meeting protocols. DTRS 72007, University of the Arts, London
17. Umeda Y, Takata S, Kimura F, Tomiyama T, Sutherland JW, Kara S, Herrmann C, Duflou JR (2012) Toward integrated product and process life cycle planning—an environmental perspective. *CIRP Ann Manuf Technol* 61(2):681–702
18. Roy R (2000) Sustainable product-service systems. *Futures* 32:289–299
19. Lindahl M, Sundin E, Sakao T (2014) Environmental and economic benefits of integrated product service offerings quantified with real business cases. *J Clean Prod* 64:288–296
20. Tukker A (2015) Product services for a resource-efficient and circular economy—a review. *J Clean Prod* 97:76–91
21. Mont OK (2002) Clarifying the concept of product–service system. *J Clean Prod* 10(3):237–245
22. Kan J, Gero J (2008) Acquiring information from linkography in protocol studies of designers. *Des Stud* 29(4):315–337
23. Lammi M, Becker K (2013) Engineering design thinking. *J Technol Educ* 24(2):55–77
24. Durugbo C, Tiwari A, Alcock JR (2011) A review of information flow diagrammatic models for product–service systems. *Int J Adv Manuf Technol* 52:1193–1208
25. Alonso-Rasgado T, Thompson G (2006) A rapid design process for total care product creation. *J Eng Des* 17(6):509–531
26. Regan WJ (1963) The service revolution. *J Mark* 47(July):57–62
27. Komoto H, Tomiyama T (2008) Integration of a service CAD and a life cycle simulator. *CIRP Ann Manuf Technol* 57(1):9–12
28. Baines TS, Lightfoot HW, Evans S, Neely A, Greenough R, Peppard J, Roy R, Shehab E, Braganza A, Tiwari A, Alcock JR, Angus JP, Bastl M, Cousens A, Irving P, Johnson M, Kingston J, Lockett H, Martinez V, Michele P, Tranfield D, Walton IM, Wilson H (2007) State-of-the-art in product-service systems. *Proc Inst Mech Eng B* 221:1543–1552
29. Love T (2000) Philosophy of design: a metatheoretical structure for design theory. *Des Stud* 21:293–313
30. Ericsson KA, Simon HA (1993) Protocol analysis verbal reports as data, revised edition. MIT Press, Cambridge, MA
31. van Someren, MW, Bardard YF, Sandberh JAC (1994) The think aloud method: a practical guide to modelling cognitive processes. Academic Press, London
32. Atman CJ, Bursic KM (1996) Teaching engineering design: can reading a textbook make a difference? *Res Eng Des* 8:240–250
33. Purcell T, Gero JS (1998) Drawings and the design process: a review of protocol studies in design and other disciplines and related research in cognitive psychology. *Des Stud* 19(4):389–430
34. Mc Neill T, Gero JS, Warren J (1998) Understanding conceptual electronic design using protocol analysis. *Res Eng Des* 10(3):129–140
35. Suwa M, Purcell T, Gero JS (1998) Macroscopic analysis of design processes based on a scheme for coding designer’s actions. *Des Stud* 19(4):455–483
36. Tang H-H, Gero JS (2002) A cognitive method to measure potential creativity in designing. In: Workshop 17—creative systems: approaches to creativity in AI and cognitive science (ECAI-02), Lyon
37. McDonnell J, Lloyd P (eds) About designing: analysing design meetings. CRC Press, USA
38. Hay L, McTeague C, Duffy AHB, Pidgeon LM, Vuletic T, Grealy M (2016) A systematic review of protocol studies on conceptual design cognition. In: Gero JS (ed) 7th international conference on design computing and cognition, Springer, Chicago. pp 135–153
39. Gero JS (1990) Design prototypes: a knowledge representation schema for design. *AI Mag* 11(4):26–36
40. Gero JS (2010) Generalizing design cognition research. In: DTRS8: interpreting design thinking, DAB documents, Sydney
41. Gero JS, Kannengiesser U (2014) The function-behaviour-structure ontology of design. In: Chakrabarti A, Blessing L (eds) An anthology of theories and models of design. Springer, Berlin, pp 263–283

42. Kan WT (2008) Quantitative methods for studying design protocols. The University of Sydney, Sydney
43. Kan WT, Gero JS (2017) Quantitative methods for studying design protocols. Springer, Dordrecht
44. Branki CN (1995) The acts of cooperative design. *Concurrent Eng* 3(3):237–245
45. Kruchten P (2005) Casting software design in the function-behavior-structure framework. *IEEE Softw* 22(2):52–58
46. Visser W (2006) The cognitive artifacts of designing. CRC Press, USA
47. Hofmeister C, Kruchten P, Nord RL, Obbink H, Ran A, America P (2007) A general model of software architecture design derived from five industrial approaches. *J Syst Softw* 80:106–126
48. Robin V, Rose B, Girard P (2007) Modelling collaborative knowledge to support engineering design project manager. *Comput Ind* 58(2):188–198
49. Van Wie M, Bryant CR, Bohm MR, McAdams DA, Stone RB (2005) A model of function-based representations. *Artif Intell Eng Des Anal Manuf* 19(2):89–111
50. Kan JWT, Gero JS (2007) Using the FBS ontology to capture semantic design information. In: McDonnell J, Lloyd P (eds) *DTRS7 (Design thinking research symposium)*, University of the Arts, London, pp 155–165
51. Kan JWT, Gero JS, Sarkar S (2010) Using a generic method to study software design cognition. In: van der Hoek A, Petre M, Baker A (eds) *Workshop on studying professional software design*, pp 1–7
52. Williams CB, Lee Y, Paretti M, Gero JS (2011) Effects of design education on design cognition: a preliminary comparison of engineering students. In: *41st ASEE/IEEE frontiers in education*
53. Widgren M, Sakao T (2016) Unanswered questions in conceptual design towards circular economy. In: *The 14th international design conference—(DESIGN 2016)*, Dubrovnik, pp 571–578

Visual Behaviour During Perception of Architectural Drawings: Differences Between Architects and Non-architects



Canan Albayrak Colaço and Cengiz Acartürk

Architectural design is not just a technical process based on problem-solving and representation, but also a social process distributed between architect and non-architect stakeholders. In this study, visual behaviour differences between architects and non-architects during perception, interpretation and evaluation of architectural drawings are analysed. An eye tracking experiment was conducted on two groups of participants: 19 graduate-level students of the Department of Architecture and 19 students from other faculties. Eye tracking data were analysed according to three categories: means of gaze duration, gaze count and gaze plot patterns.

Introduction

Development of new tools and methods contribute to new understandings in design cognition. Different models have emerged in the design research based on different ways of using theories and approaching design situations [1]. Early studies related to the phenomenon of design emerged in the 1960s, with a focus on design as problem-solving. The rational and explicit handling of design, so called the Design Methods Movement, was heavily criticized in the design world for being a very limiting and disrespectful aspect of the design ability [2]. However, the outcomes of this approach helped to develop research and a knowledge-based view of the design discipline [3]. With the possibility of rationalization and transparency in design and design methods, the existing professional monopolies in design expertise between designers and users could be broken by wider sections of society participating in the design process [4].

C. A. Colaço (✉)

Architecture Department, Middle East Technical University, Ankara, Turkey

C. Acartürk

Middle East Technical University, Informatics Institute, Ankara, Turkey

© Springer Nature Switzerland AG 2019

J. S. Gero (ed.), *Design Computing and Cognition '18*,

https://doi.org/10.1007/978-3-030-05363-5_21

The first international conference on Design Participation was held in 1971 in order to discuss the possibilities of blurring the present distinction between designer and user. The conference brought together designers, teachers and researchers that shared similar interests in user participation who were mostly followers of the Design Methods Movement [4].

Later decades after Design Methods Movement, research on “representation and recall” opened new directions in design cognition [5]. Researchers have conducted experiments, empirical studies and protocol analysis in order to analyse fundamental aspects of design cognition and representation. These studies are mainly based on scrutinizing what designers do when they come up with creative insights or concepts in the early stages of design and lack to provide insight for the direct input of layman in design process.

In this work, a theoretical grounding in which architectural design is understood as not just a technical process based on problem-solving and representation, but also a social process distributed between architect and non-architect stakeholders is adopted. Our focus is to analyse differences between architects and non-architects during perception, interpretation and evaluation of architectural drawings. Architects and non-architects are expected to exhibit different perceptual and cognitive processes. Thus, clarifying such differences is critical for further development of methods and tools supporting communication between different design parties and especially empowering non-professional parties. Empirical research is needed in order to test developing methods and tools; and compare these with traditional ones.

The state-of-the-art research is being developed in architecture with the aim of supporting distributed modes of architecture. These mainly ground on formal methods such as ontologies, Building Information Modelling (BIM), Virtual Reality (VR), Augmented Reality (AR), Virtual Design to Construction (VDC), etc. [6]. Despite the growing interest in these new tools, their usage in architectural practice is still very limited [7]. 2D technical drawings and 3D computer-generated images are still the main conventional representation techniques in architectural practice.

In order to provide empirical data related to these conventional representations and to later compare them with recently developed techniques, within the context of this paper, we presented the participants architectural drawings of two alternative buildings on a computer screen. The participants were asked to choose one to be constructed in their city and express the reasons for their selection. Accordingly, the experimental context is designed so that the participants would need to perceive, analyse and evaluate the architectural design through the presented 2D technical drawings and 3D computer-generated images.

The experiment was conducted using eye tracking devices on two groups of participants: graduate-level students of the Department of Architecture and students from other faculties. Eye tracking is a research methodology in cognition that is mainly used to study attention and perception [8]. The ongoing research in a variety of domains of cognitive science and psychology demonstrated that eye movements reveal significant information about human thoughts, intentions and cognitive processes [8]. Studying eye movements is a methodology that is gaining interest to trace expertise differences in perception, especially through the use of domain-specific rep-

resentations in experimental setups. Accordingly, eye tracking methods are herein used in order to study differences between architects and non-architects during perception, interpretation and evaluation of architectural drawings.

Rationalization of Design and Design Participation

Research into the phenomenon of design was systematically carried out in the 1960s Design Methods Movement by theorists, scientists, architects and engineers among various disciplines and perspectives, in order to understand, analyse, develop and discuss the fundamental aspects of design activity. The studies that are related to the understanding of design have led to a shift in the focus of design, from the end product to the methods of its inception, processes and production.

In the 1960s, the mathematician Christopher Alexander was among the first researchers to define design as a form of problem-solving. He argued that design is shaped by our definition and therefore by the structure of the problem. Alexander's Notes on the Synthesis of Form [9], which is highly focused on the components of physical structures of design, and A Pattern Language [10] are early interpretations of this shift in the focus of design from the end product to the process. Alexander says that "these notes are about the process of design; the process of inventing physical things which display physical order, organization of form..." [9].

Herbert Simon introduced the notion of design as a way of thinking [11]. According to Nigel Cross, the decade culminated with Simon's work and his specific plea for the development of "a science of design: a body of intellectually tough, analytic, partly formalizable partly empirical, teachable doctrine about the design process" [12].

This initial work on design thinking was based on the earlier works of problem-solving in cognitive science and artificial intelligence [5]. Design was treated as a type of problem-solving, as an investigation of a space of possible solutions for the finest or a satisfying solution, in an attitude similar to studies of chess, crypto-arithmetic, and puzzle solving [5]. Almost until the late 90s, it had been common to use the language of cognitive science studies and its concepts for the problem-solving behaviour. However, research in design thinking suggests that design is not normal problem-solving [13]. Approaching design as a normal problem-solving has been criticized in the design world: The early efforts to restructure the process of design into something more traditional and methodical were seen at the beginning as lack of respect for the natural design ability, which is something inherent within human cognition and is a key part of what makes us human [2]. Rationalization and formalization of design hide some complexity, but in order to adequately address this complexity mode, reduction of complexity is required for a closer examination [3]. The shift of focus from the end product to the process opened up possibilities of design process transparency. Starting from the 1971 Design Participation Conference, new design methods and models have been researched for a reorientation of knowledge and power during the design process between stakeholders, a real transfer of power

on design decisions, a redefinition of the relationship between designer and its users [4]. This approach adopted herein provides a theoretical lens in which architectural design can be treated as a social process distributed between architect and non-architect stakeholders.

Architectural Representations

In subfields of design cognition, such as architectural design, industrial design, engineering design or software design, cognitive processes have both similarities and differences. Within the scope of the present paper, the focus of design cognition will be on architectural design. The protocol analyses and other empirical studies of architectural design have shaped the groundwork in order to understand the nature of architectural design cognition [14].

A major difference between architecture and other design areas is the richness of representations. The range and scope of representations used during the different stages of architectural design process are broader than in any other design area [15]. In the architectural design process there are many stages, ranging from the conceptual to the construction stage. These different stages heavily rely on a wide range and scope of representations, from sketches and diagrams to 1/10 scale details. Additionally, architecture operates in various heterogeneous dimensions, including functional, ergonomic, social, psychological, cognitive, climatic, economical aspects. Consequently, architectural design problems are required to be represented through a wide variety of parameters [15]. Unlike any other design field, in architectural design the introduction of novel and diverse sets of representations is encouraged. Therefore, a saturation of representational formats and media is observed. Moreover, architecture is socially situated and it must respond to a variety of non-specialist parties that are also involved. All these facts make architecture a 'representation-saturated problem domain' that relies on different types of representation [15].

Empirical research studies of sketching in early design periods were pioneered in the 1990s. Gabriela Goldschmidt investigated the process of sketching through thinking aloud sessions, where the participants were asked to define their action verbally during sketching [16]. The sessions were recorded during the design act and then transcribed. Transcriptions along with the sketches made by the participants comprised the protocol which served as data. Goldschmidt's studies show a symbiotic relationship between internal (cognitive) and external (analogue) representations, and how the interchange between them is critical for pushing design forward [16]. Masaki Suwa and Barbara Tversky conducted protocol analysis indoors to examine how architects think and read off from their own freehand sketches and how they perceptually interact and benefit from their sketches. They suggest that freehand sketches are external representations that are essential for design ideas during the early design process [17].

These studies have become one of the main resources in design research aiming to understand the role of sketching in the early design process. They demonstrated

that there is a cognitive boundary to the quantity of complexity that can be handled internally. By sketching, a temporary external store for creative ideas is provided [2]. Designers use drawings as a means of imagining or discovering something that they cannot construct in the mind, and as a means of communicating with others [2].

Despite the ongoing research of new tools and representational formats, the main conventional representation in architectural design still consists of architectural drawing (plans, sections, elevation) and 3D models. These may be drawn by hand or computer. Within the scope of the present paper, the focus of representation is on architectural drawings rather than informal and personal types of drawing such as sketching and doodling.

Architectural drawing has been the subject matter of basic courses in architectural curricula. Students are expected to read and interpret architectural drawings and generate architectural drawings of plans, elevations and sections during the early stages of architectural education. Eastman suggests that this requires a mental mapping from 3D to 2D plans, sections and elevations [5]. Plans and sections are significant abstractions of a building, such as space allocation, horizontal and vertical circulation, structural and construction systems, etc. Images generated from 3D models provide representation for other more perceptual aspects of buildings. Students gain many skills related to architectural drawings. For instance, they learn to select appropriate 2D and 3D architectural drawings and to map between different representations. Students are expected to be able to make deductions regarding several aspects of buildings from their drawings, i.e. the ability to fluently read and write architectural drawings [5].

Most of research conducted on architectural design representation has been restricted to the early stages of design, such as preliminary and conceptual processes. Besides, it is focused on scrutinizing what designers do in the early stages when they come up with creative insights or ideas. For a more complete study of design cognition, new questions need to be raised from a design cognition perspective in order to move beyond a mere focus on the earlier stages of design. This is especially relevant since the design process does not end up when the ideas are stored in the designer's head, on paper or in digital format. Through multidisciplinary team work, architectural design evolves in complexity not just until the construction phase but even throughout its interaction with the user. As mentioned above, empirical research on design cognition is mainly based on the stage of design inception, neglecting further processes and production as well as the involvement of new actors.

In this respect, not only the scope but also the understanding of the differences between novice-layman and professional/non-professional performance in design is still rather limited, despite being a significant research area, and considering that architectural design involves both internal and external parties. Internal parties are specialists, a design team of different professional groups, such as urban planners, interior architects, landscape architects, and engineers; furthermore, external parties are non-specialists, such as contractors, investors, clients, users, public administration, etc.

Expertise Studies and Eye Tracking

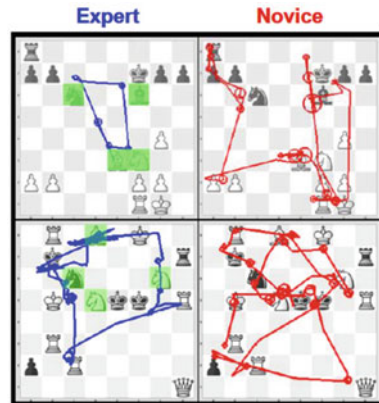
Expertise studies is an interdisciplinary research area covering multiple domains of study: professional domains (medicine, surgery, design, software design, etc.); arts, sports and motor skills (driving, dance, etc.); games (chess, video games, Tetris, etc.) and many other [18]. Expert studies provide insight into teaching, learning methods and skill training techniques in these areas [18]. Some expertise studies use a proficiency scale with seven levels: naïve (one who is out of domain), novice (new member of domain), initiate (a novice who had an initiation), apprentice (a student undergoing learning), journeyman (who can already perform unsupervised), expert (distinguished journeyman), and master (an expert who is also qualified to teach) [18].

One of the early works on learning and cognitive capacities was carried out by cognitive psychologist George A. Miller in the 1950s. He proposed that a mental process, called ‘chunking’, is necessary for cognition, by recoding information into larger units and arranging it into structures [19]. Miller pointed out that memory span is a fixed number of chunks, but not a fixed number of bits of information [19]. Learning increases the bits per chunk; the expert chunking mechanism reveals differences from that of novices. It is possible to increase the number of bits of information that memory span contains by building larger chunks.

In 1973, William G. Chase and Herbert Simon became pioneers in studying expertise through analysing eye movements. They studied perception in chess by analysing the varying strength of chess players’ eye movements [20]. Their findings revealed that chess masters automatically chunk the board into a set of identified patterns, which enable them to sort through the details of the game faster and easier than less experienced players [20]. In chess playing, chess masters survey chess board positions and recognize patterns and strategies for resolving them. On the other hand, novices cannot see beyond the next one or two moves and exhibit more complex gaze patterns. In the later years of developing of technologies to record, measure and analyse eye movements, eye tracking methods have been used to prove empirical data supporting these early expertise studies. The eye movement patterns of professional and amateur chess players, obtained by means of eye tracking are shown in Fig. 1.

Eye tracking proved to be an empirically valid method to trace expertise differences in perception, especially through the use of domain specific representations in experimental setups [22, 23]. Increasing number of disciplines are exploiting these developing techniques in their research methodologies and education techniques: e.g. medicine specialities that rely on analysis of imagery, such as radiology, pathology and dermatology. The research conducted in these fields also confirm that experts perform different visual behaviour and stereotypical scan patterns when compared to novices [22]. Instead of passively photocopying the representation that carry visual information, perception actively interprets the information based on experience and goals [22]. These studies suggest that providing training information to novices on how to look for relevant image features, as well as eye movement monitoring, could assist students in developing techniques to make decisions more quickly and accu-

Fig. 1 Eye tracking patterns of professional and amateur chess players [21]



rately [23]. Accordingly, such training could prevent potential medical errors that occur during visual analysis of medical images by reducing the delay in capturing information [23].

Expertise Studies in Architecture

In current architectural design research, eye tracking is not a commonly used methodology. However, researchers who conducted expertise studies in the architectural domain also refer to cognitive differences related to expertise [24–26]. In the domain of architecture, Akin [24] discovered direct evidence of pattern chunking into hierarchical representations. Cross [25] mentions that experts are more capable of storing and accessing information in ‘larger cognitive chunks’ than novices. Experts are also capable of recognizing the underlying principles rather than focusing on the surface features of problems [25]. Following the proficiency scale used in expertise studies, degrees of design expertise are grouped into seven levels of distinction with a slight difference such as: naïve, novice, advanced beginner, competent, expert, master and visionary [1]. This study focuses on the differences between non-architect students (graduates from backgrounds other than design, who represent the ones who are out of domain), and graduate-level architecture students (who are eligible to perform architecture by themselves).

Experiment

Participants, Materials and Procedure

The experimental context of this study was conducted at the Human Computer Interaction Research and Application Laboratory at the Middle East Technical University

(METU), Ankara. The research study was approved by the ethics committee of METU Applied Ethics Research Center (UEAM). 19 students from METU Department of Architecture and 19 students from other departments participated in the experiments. The participants provided informed consent at the beginning of the experiment session. All students were graduate-level students (either master or doctorate), the group of architecture students included those who had completed their Bachelor's degree in architecture, whereas non-architecture students were chosen among those with backgrounds other than design.

All subjects participated in an eye tracking experiment. Their eye movements were recorded by a non-intrusive 120 Hz eye tracker (Tobii T120). The participants were presented architectural drawings of two alternative buildings on computer screen. These alternative buildings were the proposals for a monumental grave/museum building to be constructed in their city. The participants were asked to choose one of them and write a reason for their selection, so that the participant would need to perceive and analyse the given drawings and images, as illustrated in Fig. 2. As two alternatives, two competition entry projects were used for a monumental grave/museum building [27, 28]. The two alternatives were presented, in a random order, as alternatives A and B, in order to eliminate possible presentation order bias effects on the results. Hence, in this paper the building proposal with a rectangular building form is referred unambiguously as "rectangular building" and the proposal with a triangular building form is referred as "triangular building".

Each building alternative was presented in two consecutive stimuli screens, each consisting of two 2D technical drawings and two 3D computer-generated images. The 2D technical drawings and 3D computer images, which were presented in the same screen, were selected specifically to convey similar information. In the first screen, the technical drawing of the site plan was presented next to an image of the site and the technical drawing of the floor plan was presented next to an image from the interior of the building. In the second screen, the technical drawing of a section and façade was presented next to two images of the exterior of the building. The participants were not given a limited amount of time to inspect the stimuli, but instead they were asked to press any keyboard key whenever they wanted to move to the next screen.

The dimensions of each stimuli screen were 1024×768 pixels, which matched the size of the eye tracker screen. The architectural drawings on each screen were 454×321 pixels. The technical drawings and the computer-generated images were presented in a single stimuli screen, instead of each screen being composed of one single image. This stimuli design allowed the participants to compare the information provided by different drawings and images on the same screen.

In the final slide, an image of each building was presented and the participant was asked to submit his/her choice. Finally, the participants were asked to write down the reasons for their selection.



Fig. 2 Stimuli and decision screen used in the experiment

Analysis

The eye tracking data were analysed by Tobii Studio software. For each stimuli alternative (triangular building and square building) and on each stimuli screen, four areas of interest (AOI) are defined with equal sizes on each drawing or image. Defining the AOI is necessary for instructing the software to calculate the statistics for each stimuli screen. The eye tracking metrics were then calculated in relation to those AOIs. On the first stimuli screen, four AOIs were specified: Site Plan Technical, Plan Technical, Site Plan 3D, Plan 3D. On the second screen, four AOIs were specified:

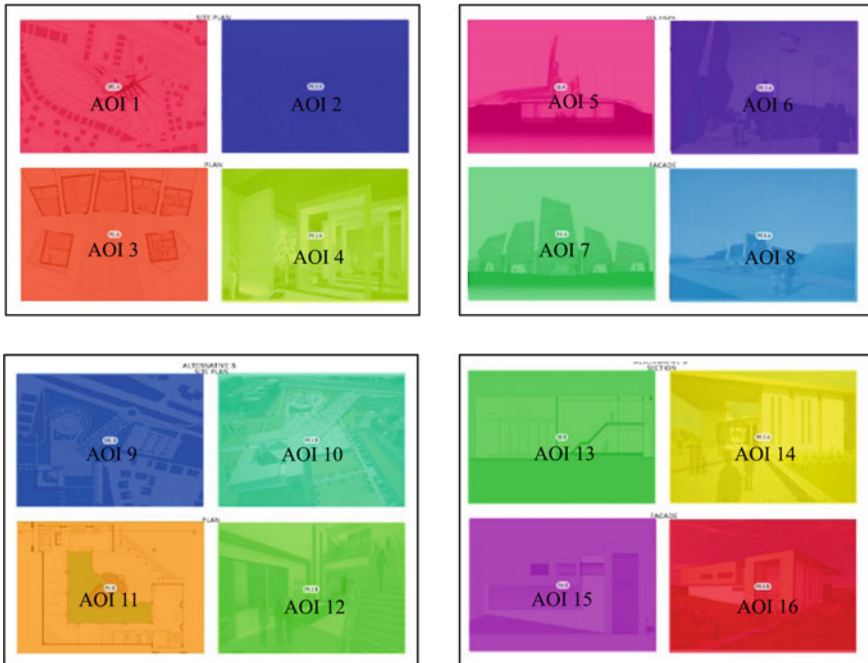


Fig. 3 The specification of the Areas of Interest (AOIs)

Section Technical, Façade Technical, Section 3D, Façade 3D. These make eight AOIs in total for each building alternative, corresponding to eight images and drawings for each building; in total 16 AOIs, as illustrated in Fig. 3.

Eye tracking metrics that are used in the analysis are: mean and total fixation duration, mean and total visit duration, number of fixations (fixation count) and number of visits to the AOIs (visit count). Metric results were analysed time wise (fixation duration and visit duration) and amount wise (fixation count and visit count), in order to compare the differences in speed with which chunks were perceived and in the size of the chunks between non-architects and architects during perception of architectural drawings.

‘Fixation duration metric’ measures the durations of each individual fixation within an AOI. ‘Fixation count metric’ measures the number of times the participant fixates on an AOI. If, during the recording, the participant leaves and returns to the same AOI, the new fixations on this area of the slide are included in the calculations of the metric. Conversely, ‘Visit duration metric’ measures the duration of each individual visit within an AOI; and ‘Visit count metric’ measures the number of visits within an active AOI.

Eye tracking measures are analysed in relation to both within-subject and between-subject factors. The between-subject variable is analysed in relation to expertise factor (non-architecture students vs. architecture students). The within-subject variables are

building form (triangular building vs. square building) and drawing type (site plan, plan, section and façade).

Results

The results revealed that most of the eye movement measures showed significant differences between the conditions of the expertise variable (non-architecture students vs. architecture students). The building form (comparison of triangular vs. square building) was a significant factor in a few eye movement measures (mean fixation duration and fixation count). The different drawing types showed statistically significant differences in the comparison of the four conditions of plan, site plan, façade, and section. The results are presented below.

Expertise: Non-architecture Students Versus Architecture Students

The two groups of the participants were different in terms of their expertise of architectural knowledge, as stated above. For a number of dimensions, the difference between the groups revealed statistically significant differences in gaze measurements (see Table 1). The participants in the non-architect group exhibited longer mean fixation duration (236.9 ms), compared to that of architect participants (215.8 ms). Non-architecture participants also spent longer inspection time (3848 ms), namely total fixation duration on the stimuli compared to architect participants (2848 ms). A similar result was obtained for the fixation counts on the stimuli between two groups of participants. The mean number of fixations of non-architect participants (16.1) was higher than that of the architects (11.4).

The mean duration of each visit to an AOI in the non-architect group (1123 ms) was also higher than for the architect group (982 ms). Likewise, the total visit duration to the AOIs was longer in non-architect students (4639 ms) compared to architects (3629 ms). On the other hand, the mean number of visit counts to each AOI was lower in the non-architect group (3.77) than in the architect group (4.69).

To sum up, the duration results so far show that non-architect participants spent more time on the stimuli, both in terms of the duration of each fixation and in terms of the total gaze time on the stimuli. The architect participants inspected the stimulus spending less time but with more visits on AOIs as one by one, and less shift between AOIs. Non-architect participants performed more inordinate eye movement patterns as seen in Figs. 4 and 5. In other words, during the perception of architectural drawings, their eyes made more moves between different AOIs than architects. These findings show different inspection strategies exhibited by the two different groups by means of duration, count and gaze patterns.

Table 1 Statistic results showing differences between non-architecture students and architecture students

	Non-architecture students	Architecture students
Mean fixation	236.9 ms	215.8 ms
Duration	SD = 71.2	SD = 75.6
Total fixation	3848 ms	2848 ms
Duration	SD = 3848	SD = 3434
Mean fixation count	16.1	11.4
	SD = 14.5	SD = 13.3
Mean visit duration	1123 ms	982 ms
	SD = 830	SD = 837
Total visit duration	4639 ms	3629 ms
	SD = 4606	SD = 4076
Mean visit count	3.77	4.69
	SD = 2.45	SD = 5.03

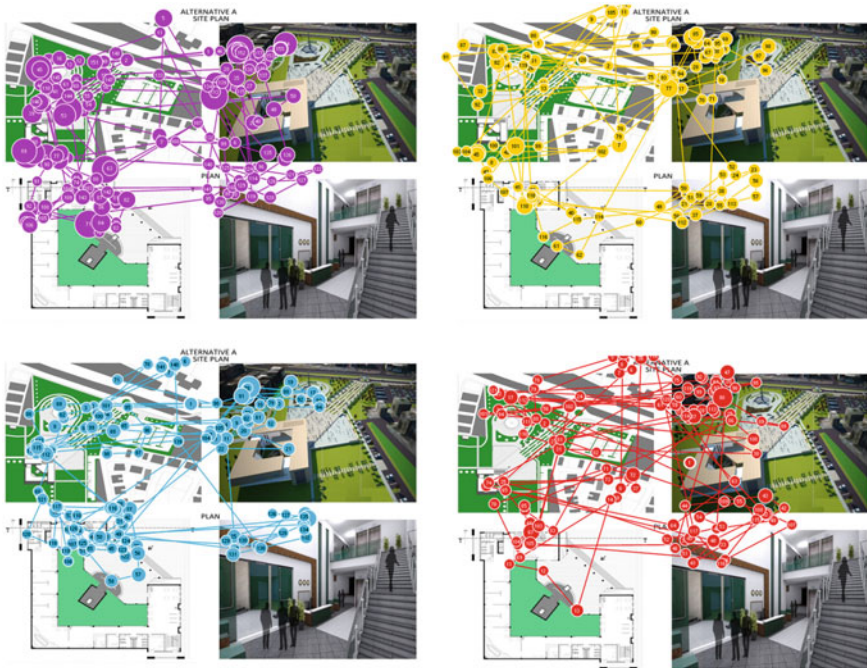


Fig. 4 Gaze plots of non-architect participants

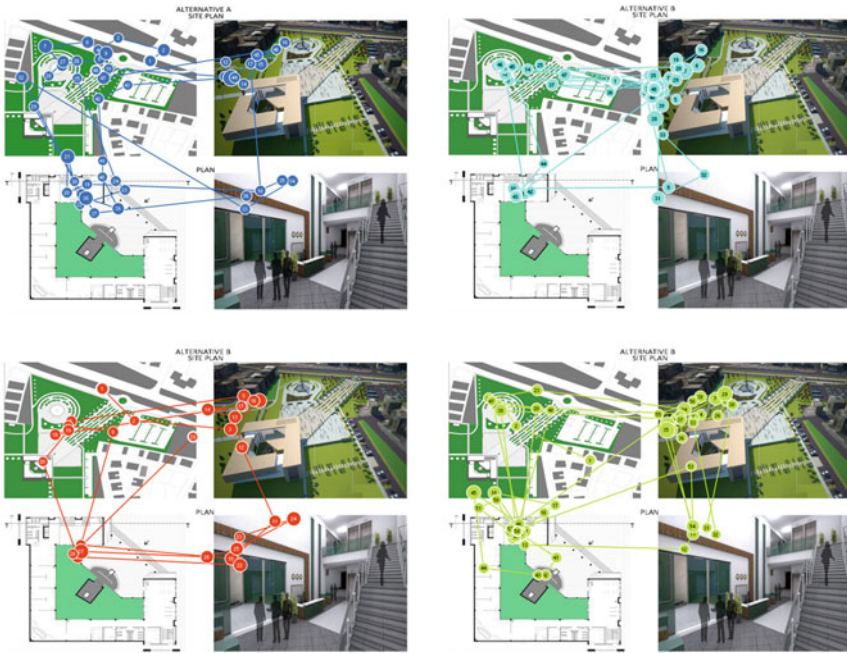


Fig. 5 Gaze plots of architect participants

Building Form: Triangular Versus Square Building

The participants were presented two different forms: an alternative building with a triangular form and another alternative with a rectangular form. The results of the analysis did not show significant differences between the two for all eye movement measures (see Table 2). The mean fixation duration on the triangular building was longer (238.3 ms) than the mean fixation duration on the rectangular building (216.4 ms). No significant differences were obtained in total fixation duration and mean fixation count between the two building alternatives. There may be various factors that influenced the inspection patterns between the two building alternatives, such as complexity of form, arrangement of plan and section layout, different design concepts, etc.

Nonetheless, a significant relation between Expertise and the Building Form variable was obtained. For instance, non-architect participants exhibited a closer inspection of the rectangular building by means of longer time and bigger number of fixations. In contrast, the pattern was reversed for the architect participants.

Table 2 Statistic results showing differences between two building alternatives

	Triangular building	Rectangular building
Mean fixation	238.3 ms	216.4 ms
Duration	SD = 79.8	SD = 64.9
Total fixation	3187 ms	3607 ms
Duration	SD = 3834	SD = 3525
Mean fixation count	12.8	15.2
	SD = 13.9	SD = 11.7

Table 3 Statistic results showing differences between drawing types

	Site plan	Plan	Section	Façade
Total fixation	5281 ms	3434 ms	2859 ms	1967 ms
Duration	SD = 4871	SD = 4152	SD = 2287	SD = 1594
Total fixation	20.4	14.3	12.7	8.52
Count	SD = 16.2	SD = 14.8	SD = 8.8	SD = 6.78

Drawing Type: Plan, Site Plan, Façade and Section

The participants were presented four different images of each of the two buildings. These images represented the plan, the site plan, the façade and the section of the buildings. Analysis of the eye movement measurements showed significant differences among those four (see Table 3). In particular, the participants (independently from expertise) spent the longest time (viz. total fixation duration) on the site plan, followed by the plan, and then the section. They spent the shortest time on the façade. The analysis of fixation counts revealed a similar picture; the highest being on the site plan, followed by the plan, then the section, and finally the façade. No meaningful relation was encountered between Expertise and the Drawing Types variable, neither for total fixation duration, nor for the mean fixation count on the AOIs.

The Final Preferences

The final preferences of the two alternatives by architecture and non-architecture students are also categorized for the two participant groups. In the group of non-architects, 13 subjects preferred the square building to be constructed in their city to the triangular building; 5 subjects preferred the triangular building. Then again, for the architect subjects, only 5 preferred the square building to be constructed in their city; 10 subjects preferred the triangular building. It is interesting to observe that the two different groups of subjects preferred different buildings. However, the architectural preference for shapes is scope of another line of research topic, where currently cognitive scientists are searching for a relation between visual aesthetic preferences and brain activation [29].

Conclusion and Future Works

Studies of differences in design cognition between architects and non-architects are still very limited. Variances are explicitly expected, but it remains important to categorize and evaluate the differences in cognitive functions between professional and non-professional within the context of socially distributed aspects of design. Clarifying the differences between architects and non-architects during perception, analysis and evaluation of representation is critical for further development of methods and tools supporting communication between different design parties, especially when empowering non-professional parties. In order to provide empirical data related to conventional representation techniques in architecture (2D technical drawings and 3D computer-generated images) the present study was conducted using eye tracking devices on two groups of participants: graduate-level students of the Department of Architecture and students from other faculties.

Three categories of variables are analysed: means of duration, count and gaze plot patterns. The results show that architecture and non-architecture students exhibit different cognitive processes during perception of architectural drawings. The duration, count and inordinateness of gaze patterns of architect subjects are lower than those of non-architect subjects. When gaze plots of architect and non-architect subjects are analysed, it is observed that architects exhibit some systematic scan patterns. This is due to the fact that architects execute knowledge and task-driven gaze control with the aim of a quick understanding of the two design alternatives and selecting one, which requires them to read and interpret architectural drawings and images, to perform mental mappings from 3D to 2D plans, sections and elevations. Hence, architects exhibited a purposeful scanning pattern. In other words, unlike non-architects, architects did not perform inordinate scanning patterns with random eye movements from one image to another. Their eyes mostly moved between site plan-plan and site plan-3D site view. Instead, from the gaze plot results, it is seen that non-architects performed more eye movements between images without making meaningful connections between drawings and images.

The results of the eye tracking experiment conducted herein showed similar results to expertise studies conducted in other abovementioned domains such as: experts require less fixation time and count, and their gaze maps are less inordinate. The difference in inordinateness of scan patterns of architect and non-architect subjects is very similar to the difference of the eye movement patterns of professional and amateur chess players.

In conclusion, significant differences are observed between architect and non-architect participants during the perception, analysis and evaluation of architectural drawings. It is not surprising that architect and non-architect participants exhibited different behaviours, since architect participants had already completed 5–7 years of their studies in the architectural field. Yet, the contribution of this study is the empirical data and proof it provided for the two facts mentioned above. First, the fact that architects require less fixation time and count, is in line with results of expertise studies that are done in other disciplines that also require domain specific

representations. Second, it is significant that the gaze pattern of the architect and non-architect groups are distinguished into two different scan patterns.

The results of this experiment point out that the requirements for a deeper communication between design parties with different cognitive faculties go beyond conventional 2D and 3D drawings. The necessity for a convergence platform for professionals and non-professionals in the field of architectural design practice gains more importance within the context of decaying professional monopolies in design expertise between designers and users; and changing relationships between architects and non-architect stakeholders.

The outcome of this study primarily provides empirical data for the comparison of different visual representation techniques that aims to empower non-architect stakeholders in the design process. This study is proposed to be further developed by conducting the same experimental setup for other fields of visual representation beyond conventional 2D and 3D drawings and to assess whether the differences between architects and non-architects described in this paper could be overcome. Besides, further expertise experiments will be conducted by means of empirical methods other than eye tracking, which is expected to provide comparative insight into the validity of different empirical methods in design research. Finally, expertise studies in architecture could be further developed in order to provide insight for design education.

Acknowledgements The authors would like to acknowledge Prof. Dr. Zeynep Mennan for her guidance and feedback; to the three DCC reviewers for their valuable comments and discussions; and to the anonymous group of students who participated in the eye tracking experiments.

References

1. Dorst K (2011) The core of 'design thinking' and its application. *Des Stud* 32(6):521–532
2. Cross N (2011) *Design thinking: understanding how designers think and work*. Berg Publishers, Oxford, New York
3. Feast L, Melles G (2010) Epistemological positions in design research: a brief review of the literature. In: 2nd international conference on design education, Sydney
4. Cross N (1972) *Design participation: proceedings of the design research society's conference*. Academy Editions, London
5. Eastman C (2001) New directions in design cognition: studies of representation and recall. In: Eastman C, McCracken M, Newstetter W (eds) *Design knowing and learning: cognition in design education*. Elsevier, pp 147–198
6. Viana DL, Morais F, Vaz JV (2018) *Formal methods in architecture and urbanism*. Cambridge Scholars Publishing, Newcastle upon Tyne
7. RIBA The Fees Bureau (2017) RIBA business benchmarking 2017 <https://www.architecture.com/-/media/gathercontent/core-cpd/additional-documents/ribabenchmarking2017executivesummary.pdf>
8. Grossmann T (2017) The eyes as windows into other minds: an integrative perspective. *Perspect Psychol Sci* 12(1):107–121
9. Alexander C (1964) *Notes on the synthesis of form*. Harvard University Press, Cambridge, Massachusetts

10. Alexander C, Silverstein M, Ishikawa S (1977) *A pattern language: towns, buildings, construction*. Oxford University Press, New York
11. Simon HA (1969) *The sciences of the artificial*. The MIT Press, Cambridge, Massachusetts
12. Cross N (2006) *Designerly ways of knowing*. Springer, London
13. Cross N (1999) Natural intelligence in design. *Des Stud* 20(1):25–39
14. Cross N (2001) Design cognition: results from protocol and other empirical studies of design activity. In: Eastman C, McCracken M, Newstetter W (eds) *Design knowing and learning: cognition in design education*. Elsevier, 79–103
15. Akin O (2001) Variants in design cognition. In: Eastman C, McCracken M, Newstetter W (eds) *Design knowing and learning: cognition in design education*. Elsevier, pp 105–124
16. Goldschmidt G (1991) The dialects of sketching. *Creativity Res J* 4(2):123–143
17. Suwa M, Tversky B (1997) What do architects and students perceive in their design sketches? A protocol analysis. *Des Stud* 18(4):385–403
18. Ericsson KA, Charness N, Feltovich P, Hoffman R (2006) *The Cambridge handbook of expertise and expert performance*. Cambridge University Press, Cambridge
19. Miller GA (1956) The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychol Rev* 101(2):343–352
20. Chase WG, Simon H (1973) Perception in chess. *Cogn Psychol* 4:55–81
21. Bilalić M et al (2010) Mechanisms and neural basis of object and pattern recognition: a study with chess experts. *J Exp Psychol Gen* 139(4):728–742
22. Li R, Pelz J, Shi P (2012) Learning image-derived eye movement patterns to characterize perceptual expertise. *Eye Tracking Res Appl Symp Proc* 393–396
23. Anderson B, Shyu CR (2011) Studying visual behaviors from multiple eye tracking features across levels of information representation. *AMIA Annu Symp Proc* 72–79
24. Akin O (1980) *Models of architectural knowledge*. Pion, London
25. Cross N (2004) Expertise in design: an overview. *Des Stud* 25(5):427–441
26. Lawson B (2005) *How designers think*, 4th edn. Routledge, New York
27. Altinişik M, Derinboğaz A (2012) Rauf Denktas Anit Mezari ve Muzesi Uluslararası Proje Yarismasi. <http://www.arkitera.com/proje/1502/esdeger-mansiyon-rauf-raif-denktas-anit-mezari-ve-muzesi-uluslararasi-proje-yarismasi>
28. Hañçerli M, Olguner O (2012) Rauf Denktas Anit Mezari ve Muzesi Uluslararası Mimari Proje Yarismasi http://www.kolokyum.com/yazi/81965__rauf_raif_denktas_anit_mezari_ve_muzesi_uluslararasi_mimari_proje_yarismasi
29. Bar M, Neta M (2007) Visual elements of subjective preference modulate amygdala activation. *Neuropsychologia* 2191–2220

Part VI
Design Grammars

On John Portman's Atria: Two Exercises in Hotel Composition



Heather Ligler and Athanassios Economou

Two formal exercises in hotel composition are presented. In both, the hospitality work of the architect John Portman is the focus. His language of hollow forms is addressed following his unique claim on the organizing principles found in his 1964 house, Entelechy I. The first exercise outlines a generative specification for his atrium hotel language in a parametric shape grammar informed by the logic of the house that generates an atrium hotel prototype. The second exercise speculates with a sketch on how transformation grammars can yield various configurations to explore Portman's atrium hotel language for a series of initial shapes. The overall goal of the research is to progress an ongoing effort to build a constructive theory on Portman's architectural language as explored for a variety of scales and contexts.

Introduction

John Portman's work is characterized by his atria that captivate the popular imagination. These compelling spaces have set the tone for principles of architectural hospitality imitated worldwide. Still, both Portman's contributions and subsequent replication inspired by his work remains difficult to assess, with no coherent theory to differentiate the subtlety and value of the originals, or even to distinguish between a copy and the real thing. Here, the subject of Portman's hotel composition is taken on to begin to systematize his approach in a logical way. Uniquely, Portman's own inventive narrative describes the origins of this work related to the precedent of his 1964 personal residence, Entelechy I. He describes the project as the design generator informing his entire corpus, a mythology he maintained in reflections throughout his life. This productive myth is one that can be engaged today in a vital way to

H. Ligler (✉) · A. Economou
Georgia Institute of Technology, Atlanta, GA, USA
e-mail: h.ligler@gatech.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_22

research and remix Portman's architectural language by looking and looking again at Entelechy I to constructively experiment with its resourcefulness as an innovation engine. This starting point is intriguing to map Portman's claims, but more importantly to engage his work in another way and likewise inform additional applications. This study focuses on the atrium hotel language and its implicit relation to the house to begin structuring this effort.

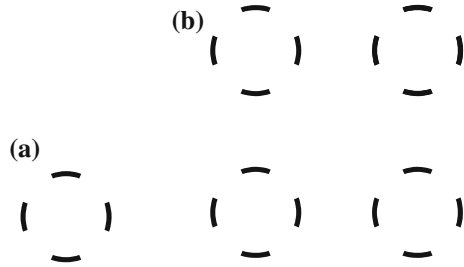
Hollow Containers

John Portman's hospitality corpus begins with his first atrium hotel, the emblematic and speculative 1967 Hyatt Regency in Atlanta, Georgia, USA. In this work, Portman essentially altered the architecture of the hotel building type in his efforts to "(re)invent the atrium," while simultaneously creating an interior urbanism defined by his distinctive hollow forms [1, 2]. However, in Portman's own conclusions and reflections on the evolution of his architectural language, he begins the story not with his infamous hotels, but with his 1964 personal residence, Entelechy I. Portman is explicit about the house as a generator in his 1976 book, *The Architect as Developer*:

... much of my later work is implicit in that house. It contains the basis for my architectural philosophy: organizing principles that work for a room or a restaurant, a building or a group of buildings. [3]

The organizing element of Entelechy I is the hollow column and even more importantly for this study, it provides the conceptual and formal basis to describe the evolution of Portman's language from house to hotel. The hollow column is a spatial, structural, and connective device composed of eight panels so that the four aligned with the cardinal axes are load-bearing and the remaining four panels can be flexibly arranged for a desired use, connection, or aesthetic. Within this system, two basic spatial types are contained by the hollow column: a minor space is defined within the singular hollow column (Fig. 1a); while a major space is the emergent field defined by four hollow columns arranged in a square grid (Fig. 1b). The plans of the house illustrate how these systems are deployed as shown in Fig. 2. The hollow containers in the house are relentlessly repeated to create emergent modules of major spaces bounded by minor spaces, a coordinate unit that characterizes the domestic spatial arrangement. Functionally, the house is separated so that the west side caters to private family use with major spaces coordinated as bedrooms, bathrooms, informal living and dining rooms, service areas, and the kitchen; while the east side has a public entertaining emphasis for welcoming guests so that it contains the entry bridge and foyer as well as major spaces that function as the formal living, dining, and music rooms. These distinctions are both functional and spatial, with the private family side characterized for a daily retreat to more intimate spaces and the public entertaining side opened to double-height volumes alongside atmospheric lily-pad-like spaces (including the central dining island) occupying the water garden that flows within the house. Supporting the major spaces, minor spaces are utilized to provide structure

Fig. 1 Hollow columns: **a** a single column defining a minor space; **b** an arrangement of columns to define an emergent major space



and horizontal circulation, light wells, vertical circulation, studies, libraries, closets, half bathrooms, and even more possibilities including additional vestibules that are often filled as micro art galleries for the display of collections including painting and sculpture (Fig. 3).

In his explanation of the development of the hollow, or exploded, column, Portman is clear that this element is related to his concept of 'space within space' and his subsequent expansion of the idea in his atrium hotels:

In both, the exploded column and the atrium, space within space was created while moving structure to the exterior skin of the circumference – integrating functional space, structure and circulation – while creating and exposing a unique spatial element – breaking the mold of compressed architecture. [4]

From the house to the hotel, or the hollow column to the hollow atrium, the fillings for the giant container are as varied as those in the house, but they take on different forms. For the hotel, as characterized by Portman's first atrium hotel (the 1967 Hyatt Regency in Atlanta, GA, USA), three plans are shown for comparison in Fig. 4. The key functional parts of the hotel are the private guestrooms and the public amenities, whose drastic differences in scale and spatial requirements are the major challenge of the composition. In the hollow hotel, this is easily and efficiently resolved as the perimeter guestroom towers contain all the private guestroom spaces, allowing amenities to be organized around the atrium lobby as well as above or below the guestroom floors so that the larger spaces can more easily avoid conflicts with the structural bays and wet walls of the guestrooms. This leaves the void of the atrium free for a variety of interventions including the characteristic exploded core with glass observation elevators that animate the volume, while also enabling unique attractions like the panoramic rotating restaurant to freely move above the atrium and terminate the vertical core.

These early models in Entelechy I and the Hyatt Regency can be seen as prototypes of Portman's language in the context of residential and hospitality design problems, each, respectively, experimenting with the possibilities of a hollow space and the device that contains it. In the house, the spatial device is multiplied as 24 hollow columns to carry the full load of the structure, each capped by a skylight, essentially creating the series of mini-atria that have inspired an interpretation of the house as a modern hypostyle hall [5]. In the hotel, the spatial device is analogously constructed as a single giant hollow container defined by the rotational symmetry of the figural

Fig. 2 The plans of the 1964 residence, Entelechy I in Atlanta, GA, USA, from bottom to top: the lower level (L_1); the upper level (L_2); and the roof level (L_R)

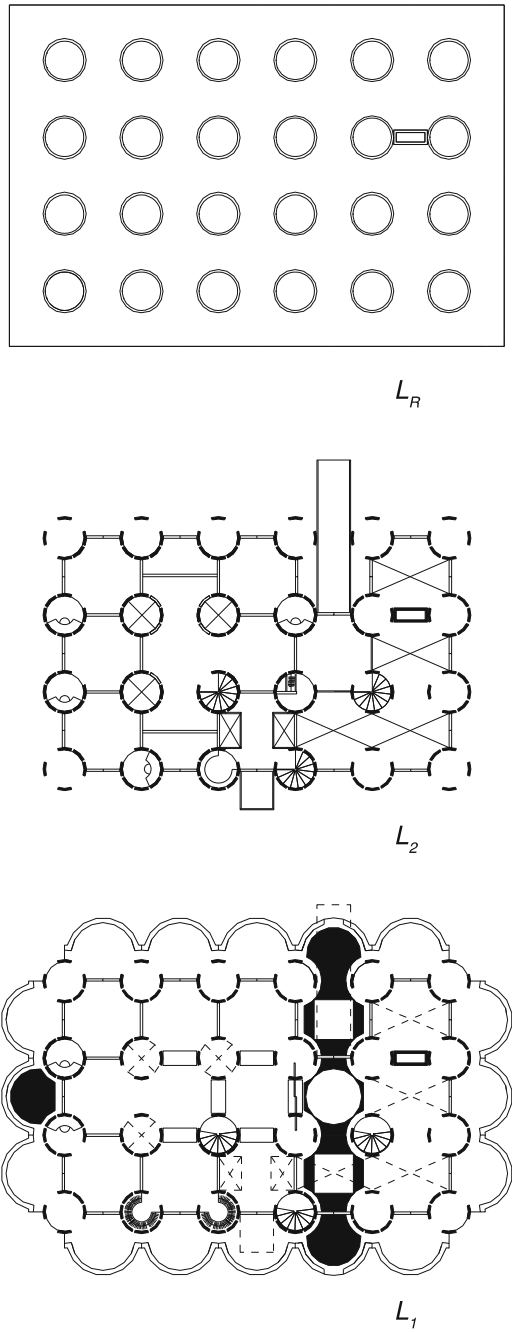




Fig. 3 Hollow columns defining minor spaces as utilized in Entelechy I

perimeter guestroom towers and capped by a skylight to essentially characterize an atrium hotel type. Altogether, the formal and functional organization is systematic and clear in both hollow house and hollow hotel.

Exercise One: From House to Hotel

A first effort to address Portman's architectural language formally is given in the Entelechy grammar, a parametric shape grammar specified to generate the 1964 house that Portman built for his family and distinguishes as crucial to the development of his design principles ever since [3, 6]. Shape grammars appeal to the ongoing ambition of this research, because the formalism offers a visual way to engage relationships in design so that a theory of architectural principles can be materialized through productive shape computations [7–9]. In this first exercise in hotel composition, the connection between house and hotel is explored to define an atrium hotel grammar to generate a prototype based on the Atlanta Hyatt Regency. Distinctive to the atrium hotel grammar is an initial transformation stage to generate the organizing principles for the initial shape. This transformation stage is one way to make connections between the house and the hotel, so that transformation rules are applied to the original design to begin a new constructive interpretation [9, 10]. This starting point begins with the house, to look again at the structure of the hollow column in the original work to set up the production of the hollow hotel. To more precisely map this process, three transformation rules are shown in Fig. 5. Transformation rule 1 starts with a single hollow column from the house and applies a scalar transformation to increase the scale of the hollow column to encompass an atrium at the hotel scale. Transformation rule 2 defines an ordering diagram to label an initial shape to organize the hotel. The ordering relationships include the shape boundary and labeled axes organized to create perpendicular intersections at each side of the shape radiating from the centroid. Transformation rule 3 generates an initial shape approximated by the circle of the original hollow column configuration and specified as an n -sided polygon defined by the same conventions. These transformation rules are applied to set up the organizational guidelines for the various initial shape(s) that can be used in the grammar. The square configuration ($n=4$) will be the main prototype for the detailed illustration of the grammar in the paper and is shown in the initial shape rule of Fig. 6.

The grammar is organized in three stages that map precisely to the house grammar for Entelechy I to facilitate ease of comparison for the interested reader. The stages

Fig. 4 The plans of the 1967 Hyatt Regency in Atlanta, GA, USA, from bottom to top: the lobby level (L_L); a typical guestroom level (L_G); and the roof level (L_R)

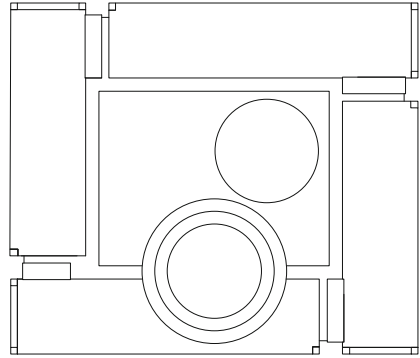
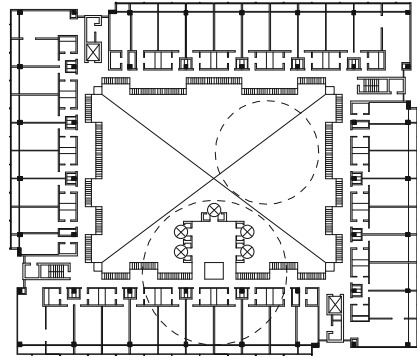
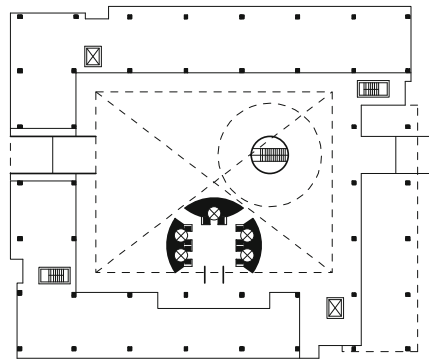
 L_R  L_G  L_L

Fig. 5 Initial transformation rules $t1-t3$

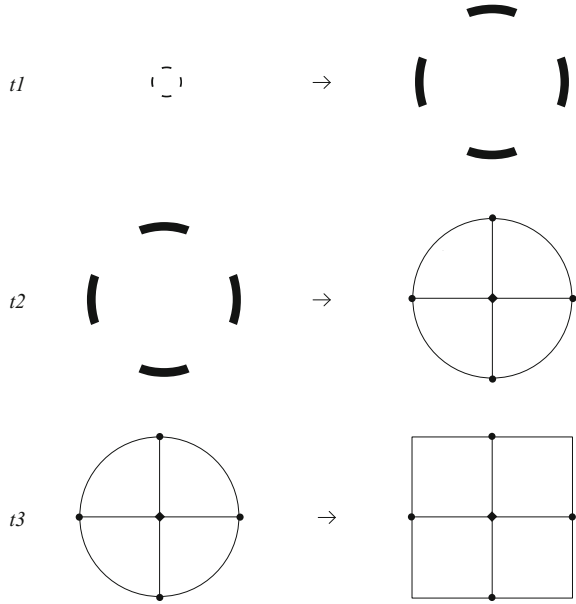
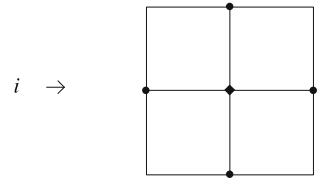


Fig. 6 The initial shape rule of the grammar

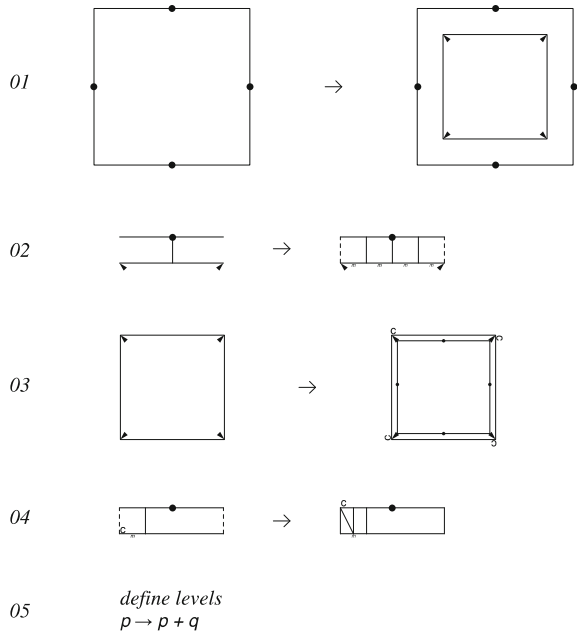


work through the design procedurally from basic concepts to developed design to mirror processes in architectural design. The three stages are Stage 1: Framework; Stage 2: Configuration; and Stage 3: Style. The first stage initializes the process and sets up the spatial canvas by developing the key boundary elements and labeling for organizational symmetries. The second stage builds the sculptural structural elements on this canvas to develop the basic configuration of a design including the consideration of natural light, structure, and circulation. The third stage continues the development to the details of the exterior and interior articulation and clarifies functional arrangements to complete the process and yield a design in the atrium hotel language.

Stage 1: Framework

Stage 1 includes five rules to complete the framework as shown in Fig. 7. Rule 1 subdivides the initial shape by an offset dimension representing the depth of the

Fig. 7 Shape rules 01–05 for stage 1: framework



guestroom module. Rule 2 subdivides each side of a given shape into a series of n guestroom modules as defined by the designer, where each module represents a hotel structural bay, which typically means two standard guestrooms or one suite. In the square configuration demonstrated here, the corner modules are emergent after this rule is applied. Rule 3 redefines the inner boundary to accommodate a perimeter corridor at the guestrooms. This rule also includes the label “ c ” at corners to define future circulation networks that will connect to the corridor. At least four c labels must be applied in any production to plan for the minimum of two egress stairs and two service elevators. For the square configuration, the c label is used to assign these locations in a configuration guided by rotational symmetry. Rule 4 uses this c label to further mark the ordering for the circulation system across a half-module of the guestroom bay. The final operation in this stage is Rule 5 that defines the number of levels the grammar will generate in a production with the description rule $p \rightarrow p + q$, where $p = 1$ for the minimum of one floor level, and q adds any additional levels. In the derivations illustrated here, the level definition includes three floors in total so that an output of three plans can be generated. The three levels necessary for this outcome include the lobby level (L_L), a typical guestroom level (L_G), and the roof level (L_R). For simplicity and clarity in the rules, only one level is shown although the application of the rule may affect multiple levels, which can be seen in the derivation at each stage. Figure 8 shows the derivation at this stage in the grammar to define the framework.

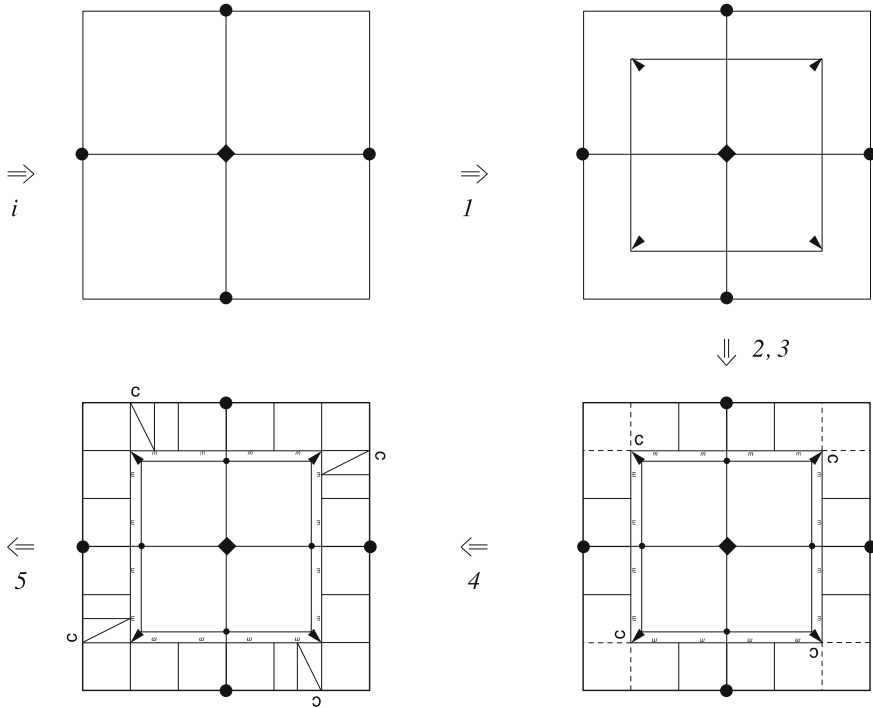
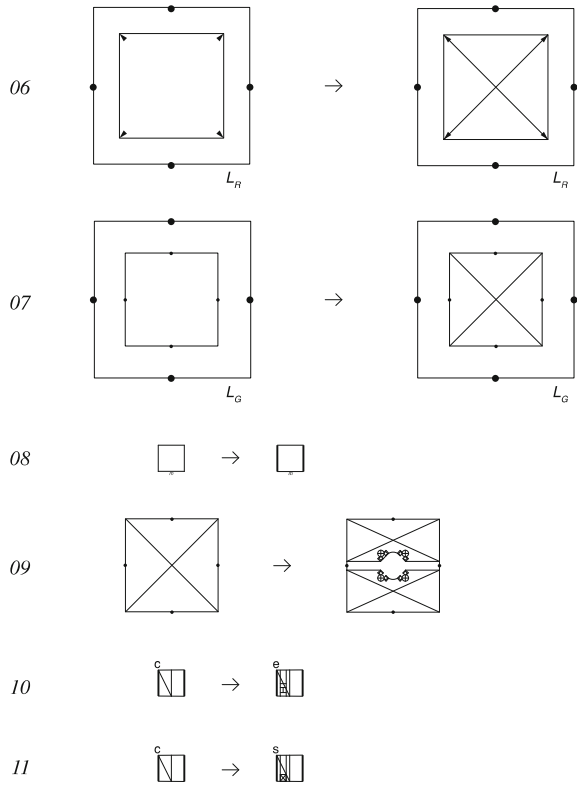


Fig. 8 Derivation of the framework for a square atrium hotel

Stage 2: Configuration

The second stage of the grammar includes six rules to develop a basic configuration in the language that foregrounds a vertical connection to the sky for natural light as well as the primary structural and circulation systems of a design as shown in Fig. 9. Rule 6 defines the roof slab at the roof level so that the central hole is left open. Rule 7 defines the atrium slab and Rule 8 adds the vertical structure at each guestroom module. Rules 9, 10, and 11 all focus on circulation. Rule 9 is applied to add the hollow guest core, which takes on many forms in Portman's hotels based on varied circulation and spatial requirements, yet always depends on the visual and experiential kinetics of the exploded elevator. Rule 10 is applied to add vertical egress circulation, which can be coordinated with code requirements to specify its required application. In the grammar, it is assumed that it must be applied at least two times for any production to allow for two exit stairs. Rule 11 completes this stage of the grammar and is applied to add service circulation for the operating needs of the hotel and follows the minimum specification of two elevators as in the Hyatt Regency. Figure 10 shows the derivation to illustrate how the design is developed from a framework to a basic configuration for the hotel.

Fig. 9 Shape rules 06–11 for stage 2: configuration



Stage 3: Style

The third and final stage of the grammar includes eight rules to articulate and fill the basic configuration of the design to complete the generation of an atrium hotel as shown in Fig. 11. Rule 12 clarifies a joint at the exterior enclosure involving the translation and depression of a segment of the perimeter. This coordinate expansion joint is organized by the circulation systems of the design and key to the extension of those systems for future development, so that additions and adjacent sites can be connected with extended circulation corridors and sky bridges. Rule 13 adds a guestroom balcony as an extension of the basic form of the massing, which stops short of the full length so that the corners of the overall form can be emphasized. Rule 14 is applied to the lobby level to create a recessed entry and its inverse, the projecting balcony, which are on opposite sides of the lobby, related either axially, symmetrically, or asymmetrically in a design. Both the recessed entry and the projected balcony impact the exterior enclosure at the lobby level too. The recessed entry breaks the perimeter at the primary entry side of the building and the projected balcony does the same, while also stimulating an inset translation of the perimeter back toward

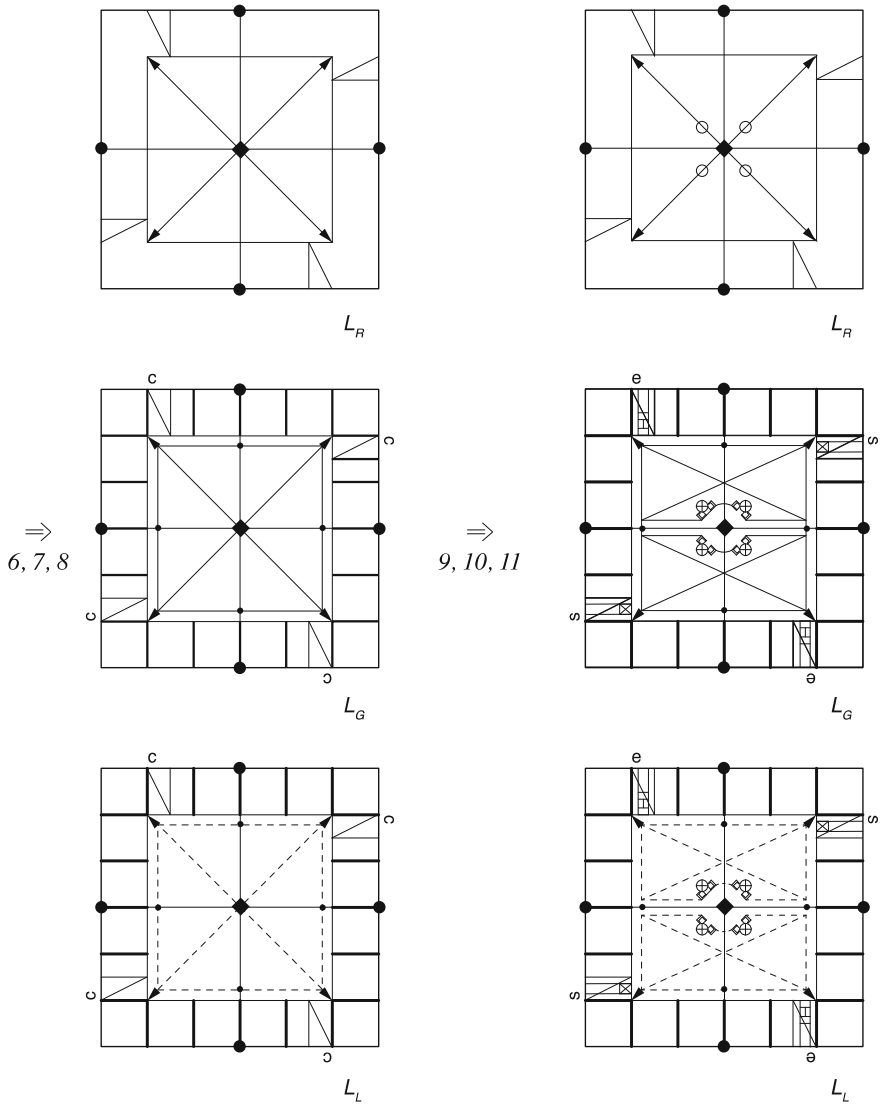


Fig. 10 Derivation of the configuration for a square atrium hotel

the interior, aiming to tie the lobby to an exterior feature of a plaza, water fountain, garden, or in the case of the Hyatt Regency, to overlook the pool deck one level below the lobby. Rule 15 adds the rotating restaurant at the top of the guest elevator core to facilitate ease of access for hotel guests and visitors as well as the best and most comprehensive panoramic views. Rule 16 is applied to create a vertical coordinate joint to further amenity levels and services below grade, typically including meeting

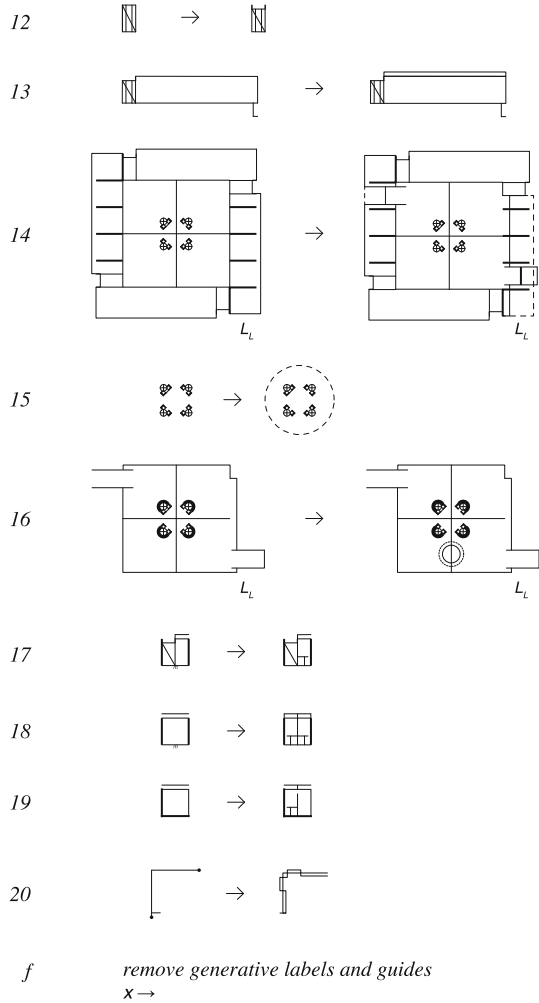
spaces and ballrooms for a variety of events. This joint might be developed with an escalator or a monumental stair to physically connect the levels. Rules 17, 18, and 19 add the guestroom layouts to the modules to detail the most private spaces in a hotel design. Rule 17 defines the guestroom layout at half-modules defined by the circulation system. Rule 18 defines the typical guestroom layouts. Rule 19 defines the guestroom layouts at the corners, where suites occupy the full bay of the module. Rule 20 develops the details of the articulated railing at the interior boundary of the atrium on each guestroom level. The railing plays on the rhythms of a design to emphasize the modulation of space between the atrium, the guest elevator core, and the guestrooms. In addition, this rhythm allows for the perimeter corridor to expand and contract, while also acting as its own container for the inorganic and organic atmospheric elements, primarily including plants and vines that provide live ornament to the atrium. The final rule of the grammar is the finishing rule, f , which removes the procedural guides and labels to yield a clean output of the design as described by the erasing schema, $x \rightarrow \dots$. Figure 12 shows the derivation at this stage to illustrate how the design is developed from a basic configuration to a design in Portman's atrium hotel language.

Exercise 2: From Hotel to Hotel

The second exercise in hotel composition aims to address the challenges of variation in a language, suggesting an algorithmic tool like Louis Sullivan's *System of Architectural Ornament*, where the architect laid out a process for developing inorganic primitives into organic designs [11]. On that basis, the starting point for this sketch goes back to the initial transformation stage of the previous exercise, where the circular exploded column from the house was used to generate the initial shape. While previous work has explored the compositional potential of the exploded column as a circular architectonic element in Portman's system [6], here the organizing shapes for the column are expanded to include both the circle and the square of the previous exercise as well as three new shapes: triangle, pentagon, and hexagon (Fig. 13). Two subdivisions are shown per side for each shape so that the cyclic symmetry of the 1967 Hyatt Regency can be immediately derived to relate the conceptual organization of the expanded hollow column to the hollow hotel (Fig. 14). While the square prototype was explored in the first exercise, this exercise aims to sketch the potential of the grammar beyond square and rectilinear forms. To do so, a series of initial shapes are shown in Fig. 15 as potential candidates to extend the grammar, with the square included as a reference. To proceed with the exercise, another approach to transformation grammars is engaged so that a series of rule transformations [10] can be generated from the grammar of the first exercise to accommodate the changing context of each new initial shape.

More precisely, each rule transformation used to accomplish a family of designs in the extended hotel language can be generated for new shape assignments—both anticipated and emergent—to carry the modified logic through every rule. A simple

Fig. 11 Shape rules 12–20 for stage 3: style



example is shown in Fig. 16 to illustrate how rule 09 in the grammar for a square hotel can be rewritten as a rule for the triangular hotel.

A shape rule schema is a convention that can be used to describe this parametric rule transformation in another way. The schema foregrounds the logic in a parametric shape rule so that it can be generalized, simplified, and applied for any shape predicate acting as an assignment to the schema. The original definition of the shape rule schema was particularized by shape-specific parameters, so that values could be assigned to variables within a shape definition for a parametric shape rule [12]. More recently, the schema has been extended to accommodate a shape assignment rather

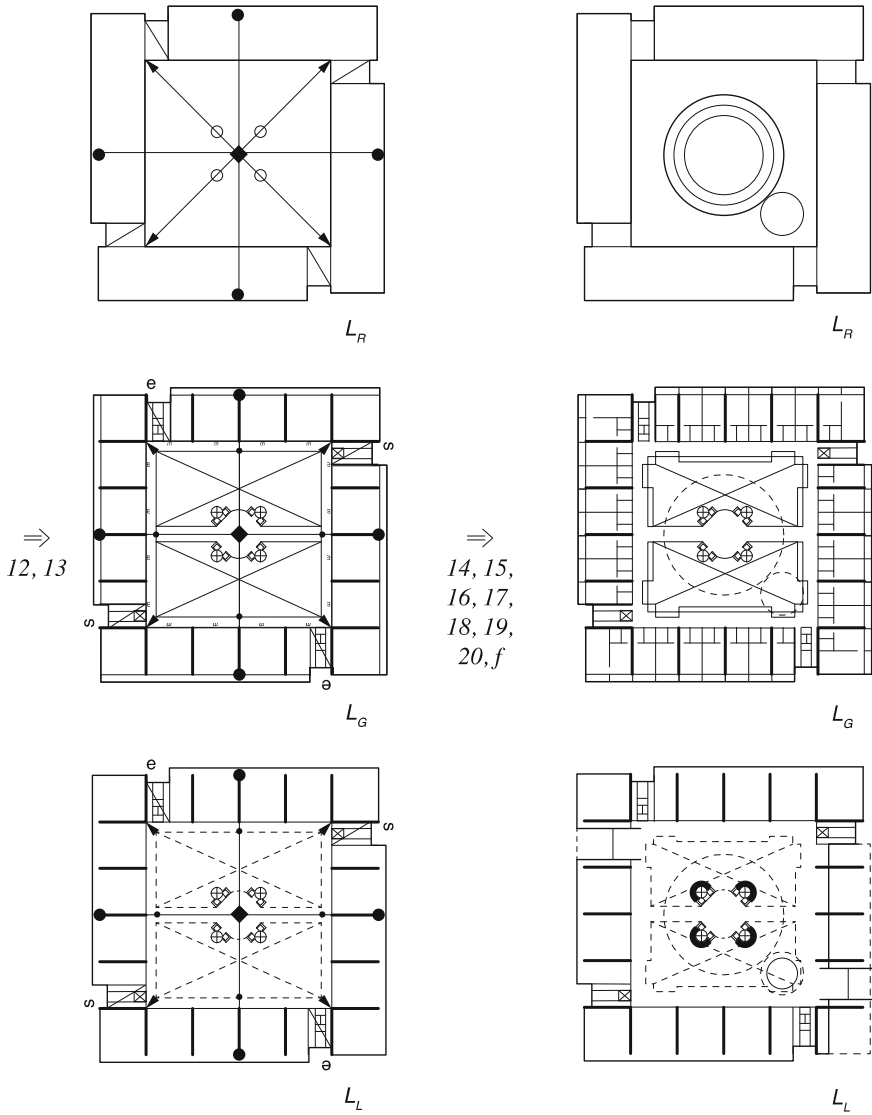


Fig. 12 Derivation of the style for a square atrium hotel



Fig. 13 Five hollow column configurations defined by a variety of shapes



Fig. 14 Five hollow column configurations organized by the cyclic symmetry group

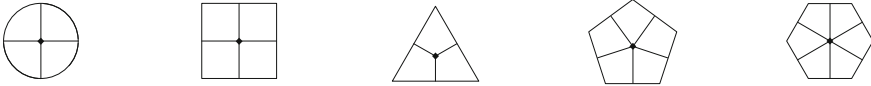
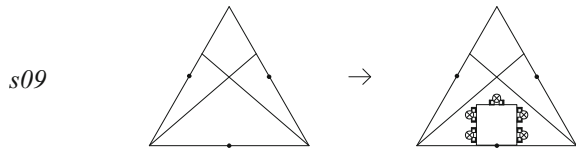


Fig. 15 Five initial shapes for atrium hotels

Fig. 16 Rule transformation s09 for a triangular atrium



than a real value assignment, so that any parametric shape could be the visual assignment to any schema to generate a parametric shape rule [8, 13]. This redefinition of the schema opens research in shape grammars up to whole new ways of description, interpretation, evaluation, adaptability, and reuse [14] that is only beginning to be addressed. Following this understanding of the schema, the rule transformation s09 can be understood in terms of changing shape assignments. For the particularities of this rule, the shape assignments that created a circular elevator core with four elevators in the center of a square atrium in the main production of the first exercise are reassigned to generate a shape rule with a square elevator core with five elevators at the edge of a triangular atrium. To describe this change, we can use the simple schema, $x \rightarrow x + y$, so that the square atrium of the original rule 09 is reassigned as the triangular atrium and labeled following the same conventions to define the x variable on the left-hand side. Then, the square elevator core with five elevators is the shape assignment for the y variable on the right-hand side to generate the new shape rule.

Though the complete details are left out here for the sake of space in the paper, the potential flexibility enabled by rule transformations described by the schema is immediately appreciated in Figs. 17, 18, and 19. Figures 17 and 18 illustrate alternative atria hotels based on this exercise in hotel composition, and Fig. 19 sketches the derivation of the typical guestroom level for all five initial shapes of Fig. 15. This final image aims to show the ambition of this exercise: to use schemas to generate rule transformations that test the logic of a grammar for a variety of different contexts, here simulated with new shape assignments. The redescription with the schema models how parametric shape rules and rule schemas can inform one another supporting a seamless loop in design process between visual rules and abstract principles shaped

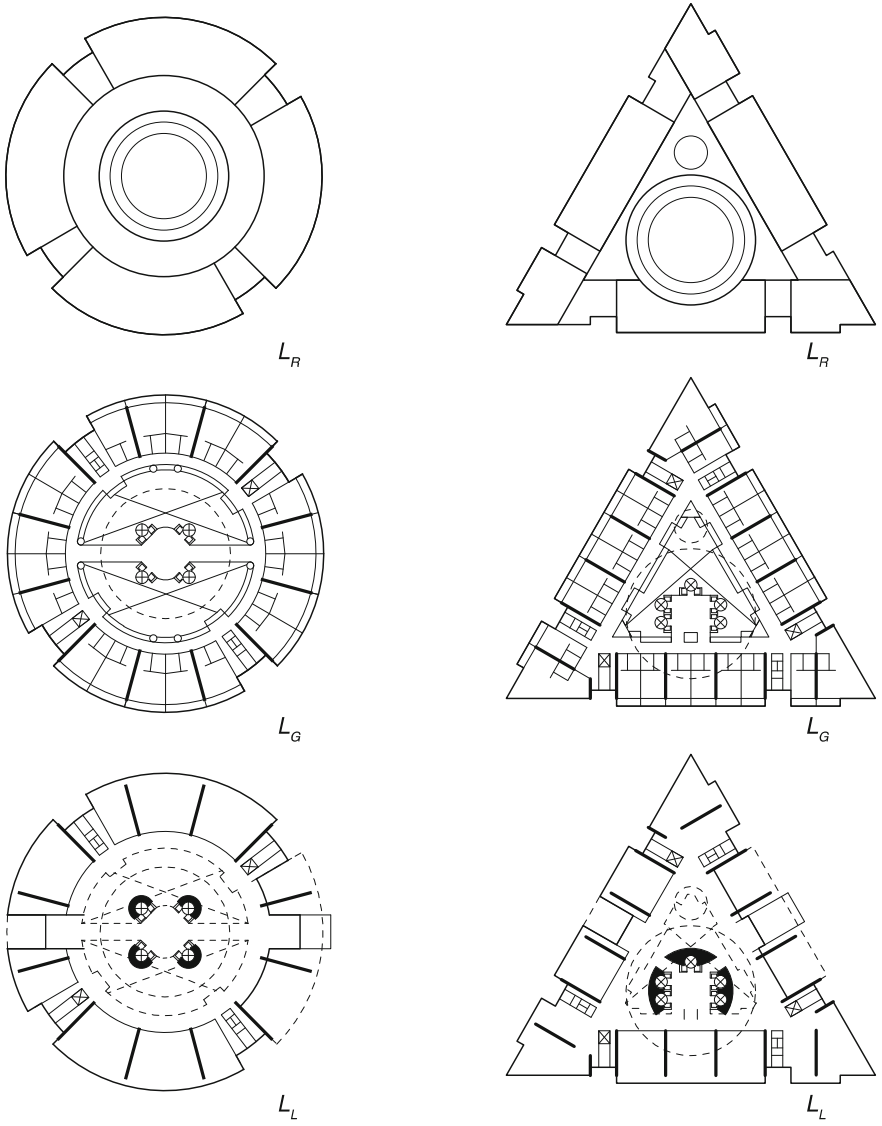


Fig. 17 Alternative atria, from left to right: a circular hotel; a triangular hotel

and formed from one another. This interplay emphasizes the flexibility proposed by visual calculating “at the quick of experience,” a familiar reality in a designer’s world and an increasingly central emphasis in shape grammar discourse [15].

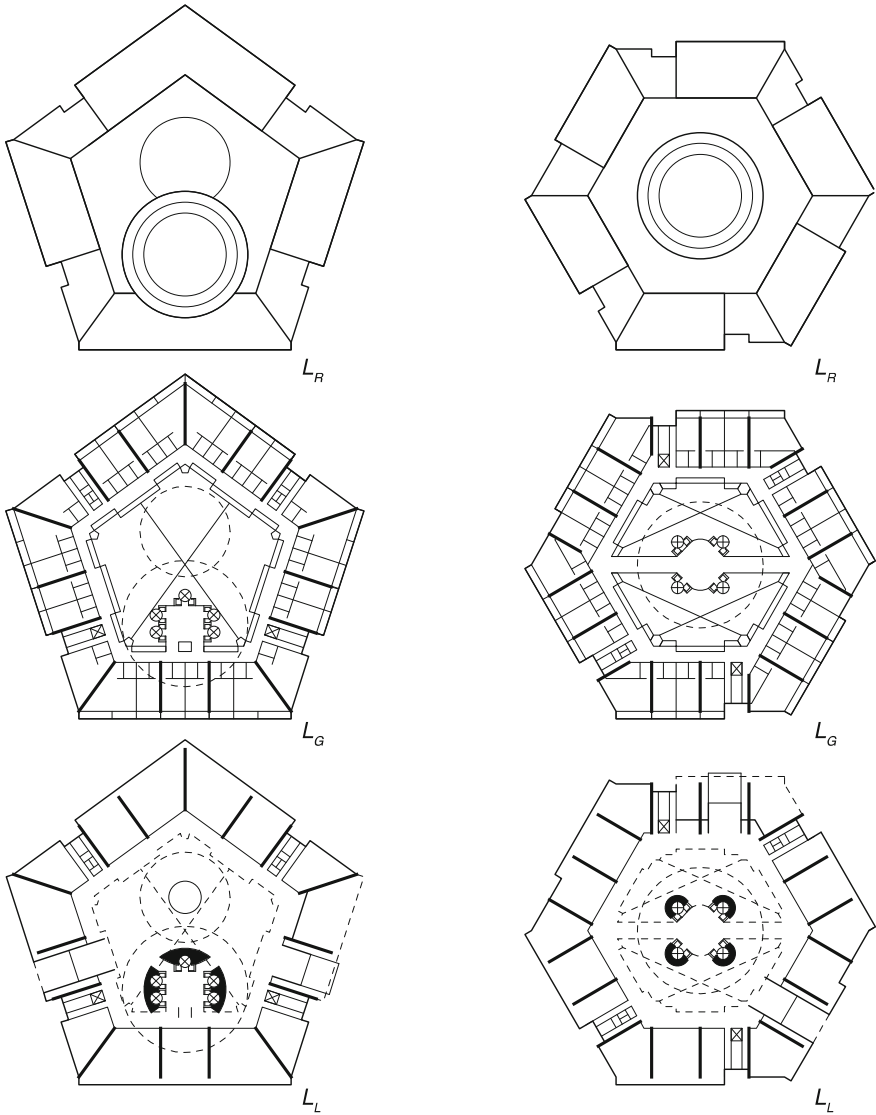
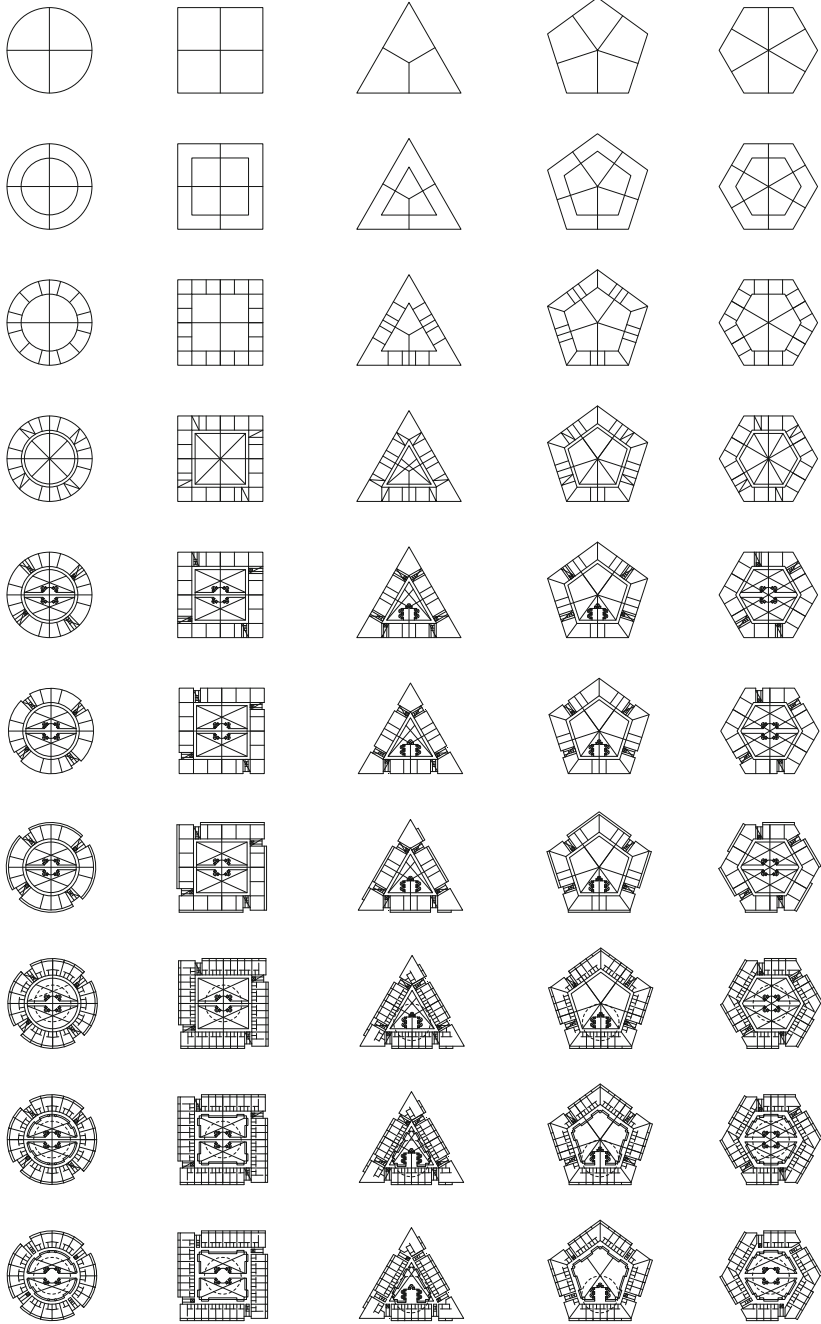


Fig. 18 Alternative atria, from left to right: a pentagonal hotel; a hexagonal hotel

Discussion

Shape grammars provide a way to constructively interpret how a design language shifts, experiments, and changes over time to add to the understanding that design is calculating [8, 9]. The strength of this generative approach is its basis in shape, promoting alignment with the inherently uncertain, exploratory, ambiguous, and



◀**Fig. 19** A sketch for the derivation of the typical guestroom level for five initial shapes in the atrium hotel language

visual environment of a design world [16]. More generally, these exercises provide a straightforward way to describe and interpret a theory of architectural composition as a logical and accessible visual calculation [17]. And more specifically in the context of John Portman's work, shape computation provides a testing ground to both formalize rule sets of best practices and experiment with the myth of Entelechy I as a design generator to interpret how organizing principles can be applied across scales and settings.

The atrium hotel grammar outlined here describes the beginning of Portman's hospitality language, but only the beginning. Immediately following the 1967 Hyatt Regency, the atrium was quickly contrasted by the 1971 addition of a dense cylindrical tower of guestrooms at the same site. This adjacency is interesting precisely because the condensed form of the tower opposes the expansive hollowness of the atrium and suggests another design approach within the hotel language. Future work will continue to develop a constructive interpretation of Portman's hotel language with this inversion, its ongoing transformation, and more to build on this project with the intent to map the developments of an architectural language in terms of influence, evolution, and contribution.

References

1. Koolhaas R, Mau B, Werlemann H (1998) *S, M, L, XL*. New York, Monacelli
2. Rice C (2016) *Interior urbanism: architecture. John Portman and Downtown America*, Bloomsbury, London
3. Portman J, Barnett J (1976) *The architect as developer*. McGraw-Hill, New York
4. Portman J (1997) *An island on an island*. *l'Arca*, Milan
5. Craig R (1989) *Mythic proportions: the portman home*. *Southern Homes*, Jan/Feb, pp 90–99
6. Ligler H, Economou A (2016) Entelechy revisited: on the generative specification of John Portman's architectural language. *Environ Plann B Plann Des* 0265813516676489
7. Stiny G (1980) Introduction to shape and shape grammars. *Environ Plan* 7(3):343–351
8. Stiny G (2006) *Shape: talking about seeing and doing*. MIT Press, Cambridge
9. Knight TW (1994) *Transformations in design: a formal approach to stylistic change and innovation in the visual arts*. Cambridge University Press, Cambridge
10. Knight TW (2014) Regarding rules: from Rimini to Rio. *Joelho*. *Revista de Cultura Arquitectónica* 5:12–27
11. Sullivan L (1990) *A system of architectural ornament, according with a philosophy of man's powers*. Rizzoli, in cooperation with the Art Institute of Chicago, New York
12. Stiny G (1977) Ice-ray: a note on the generation of Chinese lattice designs. *Environ Plan* 4:89–98
13. Stiny G (2011) What rule (s) should I use? *Nexus Netw J* 13(1):15–47

14. Economou A, Kotsopoulos, S (2014) From shape rules to rule schemata and back. design computing and cognition DCC'14. In: Gero JS, Hanna S (eds) Springer 2014, pp 419–438
15. Stiny G (2015) The critic as artist: Oscar Wilde's prolegomena to shape grammars. *Nexus Netw J* 1–36
16. Mitchell WJ (1990) *The logic of architecture: design, computation, and cognition*. MIT Press, Cambridge
17. Stiny G (1976) Two exercises in formal composition. *Environ Plan 3*:187–210

Monitoring China's City Expansion in the Urban–Rural Fringe: A Grammar for Binjiang District in Hangzhou



Ruichen Ni and José P. Duarte

This study uses an analytical shape grammar to encode the general principles behind the structure of a fringe district and describe how a rural area evolved into an urban district. The study shows that both rural and urban development patterns rule-based and, therefore, urban growth could be regulated using synthetic grammars to control the transformation of rural into urban patterns.

Introduction

Planning in China is focused on urban growth. Along with the rapid growth in cities in recent decades, urban expansion in urban–rural fringes has exacerbated conflicts between formal urban policies and informal rural developments. Urban form in fringe areas is a result of the dynamic interplay between government and local communities, leading to the formation of urban villages. This study uses an analytical shape grammar to encode the general principles behind the structure of a fringe district. Examining how a rural area evolved into an urban district, the study shows that both patterns of development are rule-based. Urban growth relies on existing rural patterns that could be coordinated using shape grammars.

China has experienced strong urban growth over the last decades. According to the National Bureau of Statistics of China, only 19.4% of its population lived in cities in 1980, but in 2011 this value exceeded 50% for the first time. Moreover, experts estimate that the urbanization rate in China will surpass 70% by 2050 [1].

Given the need for city expansion, land use conflicts in urban–rural fringes have become commonplace. According to Pryor, an urban–rural fringe emerges between a growing urban center and the surrounding rural hinterland [2]. It is where urban-

R. Ni · J. P. Duarte (✉)
The Pennsylvania State University, State College, USA
e-mail: jxp400@psu.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_23

ization progresses with the emergence of heterogeneous land use. In addition, land conversion in an urban–rural fringe stimulates the transition of social and demographic characteristics [2]. Based on Maoist’s commitment to develop the urban proletariat class, the strict classificatory residential system into “urban personnel” and “rural personnel” has formed a distinct urban–rural division in the country [3]. This distinction has originated a dualistic urban–rural structure of land development. The central metropolitan authority is responsible for city planning, while rural autonomous land distribution by local communities has been encouraged by the implementation of the household responsibility system within the dual (public and private) land ownership structure created to boost agricultural production since the 1980s. Thus, land use conversion in fringe areas does not correspond just to a shift of a transition zone between the countryside and the city. It also consists of a shift in social development where the traditional social networks of villages collide with modern industries and city lifestyles. In this scenario, major conflicts in the urban fringes include the following:

- Changing agricultural milieu of urban–rural fringe. There is a vast conversion of agricultural land into urban tracts in the suburbs for commercial, industrial, residential, and educational uses. Local dwellers are allocated to residential high-rise buildings. Communities lose their identity due to uniform urban constructions and a lack of human-scale spacing.
- Formation of urban villages. In the rapid urbanization context, the informal development of rural areas by local communities has formed another category of spaces and residents called urban villages. These villages are part of the urban area in terms of physical location but retain rural characteristics with regard to building composition and residential demography [4]. Although urban villages preserve some local culture and provide cheap housing options for people, they exert threats to the wellness of public. The complex networks within urban villages pose a danger to public security, presenting lack of public facilities, deteriorated infrastructure systems, and informal living conditions, in sharp contrast with nearby thriving modern industry [5].
- Simultaneous forces of urban and rural expansion. Urban form in fringe areas is the result of a dynamic interplay between the government and rural communities [4]. However, the importance of the latter has been downplayed by the central planning paradigm. The formal planning system is regarded as the only legitimate mode of planning by the majority [6]. This view exacerbates conflicts between the two, with the needs of rural dwellers not acknowledged by the authorities.

Nonetheless, it is important to consider both types of development in the study of how a rural area evolves into an urban land use. With this alternative view in mind, we focused our study on the Binjiang District in Hangzhou, China. Since 2000, the district has undergone massive development that transformed it from an agricultural area into an industrial-oriented urban district (Fig. 1). In 1990, building a high-tech industrial zone south to the Qiantang River became a goal in the governmental agenda for the area. The three towns near the river—Puyan, Changhe, and Xixing—were put under the jurisdiction of Hangzhou and in 1996 the Binjiang district came into being



Fig. 1 Aerial view showing land use change in the period 2000–2016 [21]

encompassing an area of 28.19 square miles. The whole district accelerated its city-centric urbanization in 2000, with the construction of Binjiang Higher Educational Park [7].

In the city plan for 2020, it is foreseen the district to be endorsed by a central high-tech industry park with a university cluster, two commercial blocks, surrounded by nine residential. The natural area with ecological value will be preserved as recreational land at the periphery of the district [8].

However, at least 15 urban villages were formed under rapid development and autonomous rural expansion [9]. Few agriculture activities remain within the urban villages, due to the loss of suitable land. Major financial support for those dwellers comes from rental business, factory jobs, and governmental subsidies. The central planning authority regards urban villages as stains in the emerging industrial highland. Starting in 2015, the government managed to redevelop some villages by allocating and dismantling them completely, leaving space for new internet industries [9].

Research Objective

The study investigates how the rural environment evolved into an urban area in the selected urban–rural fringe using an analytical shape grammar. Two types of built environments are examined: (1) new urban districts that have developed based on the city-centric planning policy and (2) local villages that have evolved following rural-centric growth in response to urban land conversion over time. By looking into shape rules and the sequence of their application, we aim to reexamine the relationship between rural and urban, challenging the usual assumption that the two types of development are unrelated. We focus on the Binjiang District in Hangzhou, the capital city of Zhejiang Province, China, due to the availability of historical aerial images and planning information. By pinpointing potential ways to plan rural land to accommodate future urban expansion, the study might provide a basis for decision-making regarding urban expansion and redevelopment of urban villages.

Theoretical Support

Shape grammars have been used as an effective tool for spatial analysis and design synthesis in parametric urban design processes [10]. By combining established theories such as Alexander's pattern language [11], Stiny's shape grammars [12], and Hillier's space syntax [13], urban grammars aim to develop a framework for urban design. The idea was tested with the development of a specific urban grammar for Praia that extracted urban program descriptions from a GIS platform and then used parametric shape grammar rules to generate the urban grid [14]. The Praia grammar was later incorporated in the generation module of the City Induction project. By extending the concept of discursive grammar [15] to urban design, the project proposed a generic urban grammar to create a platform for site-scale urban design that included modules for formulating, generating, and evaluating design alternatives [16]. The generic grammar could be used by designers to generate new specific urban grammars for diverse urban contexts. The City Induction project also introduced the concept of Urban Induction Patterns (UIP) [17] based on Alexander's pattern language [11] and Gamma et al. design patterns [18]. Each UIP captures a recursive design move that is used in urban design processes in response to a recurrent problem [18].

The present study uses the concept of generic grammar to encode the general principles behind the structure of the Binjiang district and then to generate alternative design solutions in the way proposed in the City Induction Project. Inspired by the pattern concept in the City Induction project, we rely on rural induction patterns (RIPs) to describe rural-centric growth and urban induction patterns (UIPs) to describe city-centric development, and then, compare both. RIPs and UIPs encode urban design operations regarding rural road formation, rural parcel distribution, urban street delineation, and urban block division. These design moves are extracted from the examination of annual aerial photographs from 2000 to 2017, available planning documentation, and policy information. The transformation from a rural to an urban structure is explained using induction patterns. To specify semantical and contextual information, a set of description rules are included in the generic patterns. The growth of rural and urban areas is compared, from the formation of the basic axial structure to its subsequent detail into rural or urban fabric. Table 1 lists the subset of rules that make up the induction patterns.

To describe the types of urban streets transformed by the rules, the generation module includes an urban ontology. According to Gruber, ontology refers to a "formal, explicit specification of a shared conceptualization" [19]. Defined in the City Induction project, an urban ontology serves to define types of objects and features found in the urban environment and elucidate the significant relations among them [20]. For the purpose of this study, the specification of the hierarchy of urban streets based on the ontological structure is enough to describe street transformations in the shape grammar. Transportation Network (TN) is the object class for the urban street system. Street definition (SD) refers to the street sections as compositions of profile components [20]. Table 2 shows the components included in the ontology

Table 1 Main classes of rules in RIPs and UIPs

Rural induction pattern (RIP)	Urban induction pattern (UIP)
RIP-001: rules to draw “parallel” rural division lines	UIP-001: rules to determine main geometrical axes
RIP-002: rules to draw “perpendicular” rural streets	UIP-002: rules to draw an urban grid based on axes
RIP-003: rules to transform the rural grid	UIP-003: rules to modify the street hierarchy
RIP-004: rules to subdivide the rural grid into rural plots	UIP-004: rules to subdivide the urban grid into city blocks and then plots
RIP-005: rules to subtract and concentrate rural neighborhoods	

Table 2 Detail of the transportation network and street definitions classes showing the object types within each class—the table has been simplified

Groups of entities—object classes	Entities (components of urban space)—object types
α2—TN—transportation network	Street types: R1—Ring Road/R2—Primary Structural Street/R3—Secondary street system/S1—Local Distribution/S2—Informal Urban Street
α4—SD—street definitions (street profile components)	Sp—Street profiles: ①—street parking/②—sidewalks/③—bicycle lane/④—car lane/⑤—green island (width varies)

of the street system. Extracted from actual street data on the physical forms and transportation planning information for the Binjiang district, the left column shows utilized classes, while the right column shows object types within each class. Each street type in TN is described as a collection of street profile components from the SD class [20]. Every actual street in the area can be placed into one of the street types in TN.

Each type of TN is a composition of street profile components from the SD class. Each street type includes component requirements and a maximum street width value. The requirements for TN types are as follows (refer to the street profile components in Table 2):

- R1:** ②③|④ × 3 ⑤④ × 3|③② (maximum width: 50 m)
- R2:** ②③|④ × 2 ⑤④ × 2|③② (maximum width: 40 m)
- R3:** ②③①|④⑤④|①③② (maximum width: 30 m)
- S1:** ②③①|④|①③② (maximum width: 25 m)
- S2:** ②①|④|①② (maximum width: 15 m).

Other street profile components can be added to street types, as long as the street width does exceed the maximum value allowed. Standards (widths and forms) for street profile components are consistent in each TN type, except for ⑤. The green

island width varies according to the street width and so does the planting pallet. This qualitative relationship among objects sets criteria for shape rule abstraction, making it easier to represent the streets planned and constructed within the Binjiang district.

Induction Patterns and Derivations in the Period 1990–2017

Rural Induction Patterns

The rural-centric development starts by extracting and offsetting lines from the pre-existing limits, such as river, streams, hills, and rail tracks, to create parallel axes (RIP-001). Then, “perpendicular” rural roads are drawn in a roughly straight angle to the axes, spaced according to the number of households that will manage the divided land (RIP-002). The rural grid formed by axes and perpendicular roads are further subdivided and rural fields are distributed to each individual household (RIP-003 and 004). Commonly, each household gets 1.1 mu (a Chinese unit of area) on average. This is the physical implementation of the household responsibility system for agricultural production. Facing mass industrialization resulting from the thriving of private-owned business in the new century, rural housing in the district started to concentrate in clusters (RIP-005-02 and 03). Demolition of existing agricultural components proceeds during the period between 2000 and 2017 (RIP-005-01).

The sequence of Rural Induction Patterns needed to generate the rural landscape and to subtract rural lands for latter urbanization is applied as follows (Figs. 2 and 3).

Following are some of the used RIPs and their respective rules.

Urban Induction Patterns

Similarly, the city-centric development also starts by extracting, offsetting, and extending lines from pre-existing elements to create axial roads (UIP-001-01&02). Pre-existing elements include bridges, railway tracks, river, streams, and rural roads. However, during calibration operations, only selected axes regarded as extensions of two major bridges to remain (UIP-001-03&04). They serve as references to draw “parallel” and “perpendicular” axes to form an urban grid (UIP-002). For each urban axis and road generated, rules for urban street classification were operated on the object to assign and modify its features (UIP-003). Detailed subdivision of the urban grid is based on the areas of the subdivided blocks and their proportions relative to the grid, so operates the process for subdividing urban blocks into parcels (UIP-004).

The sequence of Urban Induction Patterns used to account for the generation of urban form after the year 2000 is as follows (Fig. 4).

Following are some of the used UIPs and their respective rules (Fig 5).

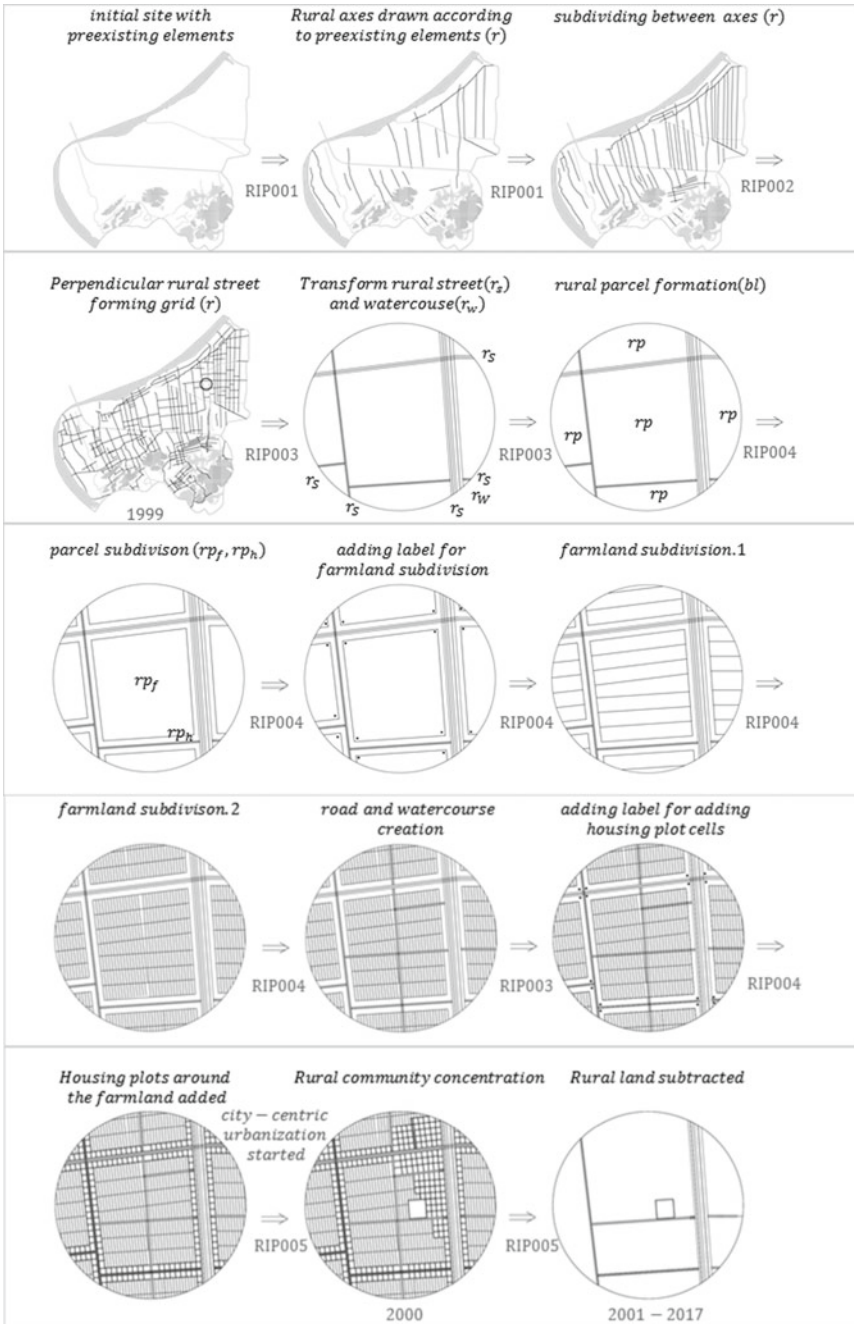


Fig. 2 Derivation sequence depicting rural growth. The derivation is simplified to the essential steps

Following are some of the used RIPs and their respective rules.

RIP-001: Rules to draw "parallel" rural structure	
Rule (number)	Rule (representation)
<p>Rule 001-01a draw rural axes that is parallel to the pre-existing limits</p> <p>$\forall p1: p1 \in \{R_{ef}\}, p1$ includes railway, stream, hill $s \in p1 \wedge s$ is maximal</p> <p>$\alpha1: \emptyset \rightarrow r$ $r \parallel s, d \in (200m, 300m)$</p>	
<p>Rule 001-01b draw rural axes that is parallel to the pre-existing limits</p> <p>$\forall p1, A, B: p1, A, B \in \{R_{ef}\}, p1$ includes railway, stream, hill</p> <p>$\alpha1: \emptyset \rightarrow r$ $\angle \gamma \in (70^\circ, 120^\circ)$</p>	
<p>Rule 001-01c Extend rural axes</p> <p>$\forall A, B: A, B \in \{R_{ef}\}$</p> <p>$\alpha1: r \rightarrow r'$</p>	
<p>Rule 001-01d Draw parallel rural axes</p> <p>$\forall A: A \in \{R_{ef}\}$</p> <p>$r \parallel r, d \in (400m, 600m)$ $\alpha1: \emptyset \rightarrow r$</p>	
RIP-002: Rules to draw "perpendicular" rural street, forming rural grid	
Rule (number)	Rule (representation)
<p>Rule 002-01 "Perpendicular" axis formed from a pre-existing limit</p> <p>$\forall A, B, C, D: A, B, C, D \in \{R_{ef}\}$ $\forall p1: p1 \in \{R_{ef}\}, p1$ includes railway, stream, hill d refers to the width of the pre-existing limit</p> <p>$\alpha1: \emptyset \rightarrow r$</p>	
<p>Rule 002-02a Draw a "perpendicular" axis from a parallel axis</p> <p>$\forall A: A \in \{R_{ef}\}$ $\forall p1: p1 \in \{R_{ef}\}, p1$ includes railway, stream, hill</p> <p>$\alpha1: \emptyset \rightarrow r$ $80^\circ \leq \angle \gamma \leq 110^\circ$</p>	
<p>Rule 002-02b Draw a "perpendicular" axis from a parallel axis</p> <p>$\forall A: A \in \{R_{ef}\}$</p> <p>$\alpha1: \emptyset \rightarrow r$ $80^\circ \leq \angle \gamma \leq 110^\circ$</p>	
<p>Rule 002-03 Erase an undesirable label point</p> <p>$\forall A, B, C: A, B, C \in \{R_{ef}\}$ $x_1 > x_2$</p> <p>$\alpha1: C \rightarrow \emptyset$</p>	
<p>Rule 002-04 Determine the width of a rural block based on the number of households living within</p> <p>$\forall A: A \in \{R_{ef}\}, x \propto n, d$ refers to the width of block n households living and managing fields within the block</p> <p>$\alpha1: \emptyset \rightarrow B$ $k_1 n \leq x \leq k_2 n$</p> <p>r006: $R_{ef} \rightarrow \emptyset$</p>	

Fig. 3 Rules for rural-centric growth

RIP- 003: Rules to transform rural grid	
Rule (number)	Rule (representation)
Rule 003-01 create rural street	<p>$\alpha 1 : r \rightarrow r_s$ $w \in \{5m, 10m\}$</p>
Rule 003-02 create artificial watercourse	<p>$\alpha 1 : r \rightarrow r_w$ $w \in \{5m, 10m, 30m\}$</p>
Rule 003-03 define rural block	<p>$\alpha 1 : \emptyset \rightarrow rp$</p>
RIP- 004: Rules to subdivide rural block into rural plots	
Rule (number)	Rule (representation)
A	
Housing plot cell	<p>h and v are respectively the cell length and width c and k are the construction periphery for housing $h = c + z + z$, $v = k + z + z$ z is $\frac{1}{2}$ the width of paths surround the housing plot</p>
B	
Rule 004-01a Subdivide rural grid into housing plot and agriculture field	<p>$\alpha 1 : rp \rightarrow rp_h, rp_f$ f refers to agriculture fields h refers to housing plot with cells,</p>
Rule 004-01b Subdivide rural grid into housing plot and agriculture field	<p>$\alpha 1 : rp \rightarrow rp_h, rp_f$ f refers to agriculture fields h refers to housing plot with cells,</p>
Rule 004-02 Place 4 labels in the outer corner of bi_h to start generation process	
Rule 004-03a Insert a housing cell, erases the label at the corner, inserts two more labels to continue the generation	<p>$\alpha 1 : \emptyset \rightarrow ph$ h and v are respectively the cell length and width</p>
Rule 004-03b Insert a housing cell, erases the label on the block side, inserts two more labels to continue the generation	<p>$\alpha 1 : \emptyset \rightarrow ph$ h and v are respectively the cell length and width</p>
Rule 004-03c Insert a housing cell, erases the label on the block side, inserts two more labels to continue the generation	<p>$\alpha 1 : \emptyset \rightarrow ph$ h and v are respectively the cell length and width</p>

Fig. 3 (continued)

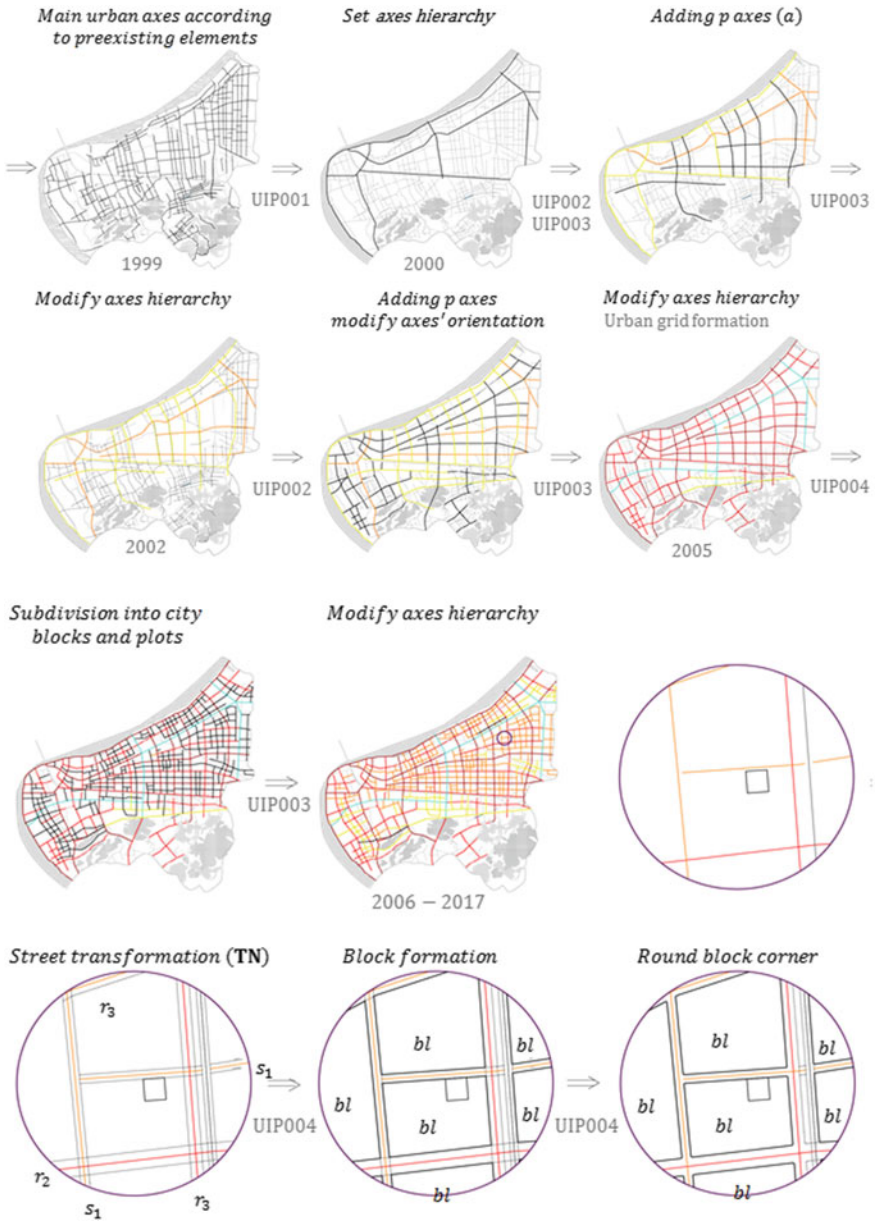


Fig. 4 Derivation sequence for the city-centric urbanization in Binjiang district. The derivation is simplified to the essential steps

Following are some of the used UIPs and their respective rules.

UIP- 001: Rules that determine the main geometrical axes	
Rule (number)	Rule (representation)
Rule 001-01a draw axes that is parallel to the pre-existing limits	<p> $\forall p1: p1 \in \{R_{ef}\}, p1$ includes railway, stream, rural street, hill $s \in p1 \wedge s$ is maximal $\alpha 1: \emptyset \rightarrow a, a \in \text{TN}$ $a \parallel s, 0 \leq d \leq 400 \text{ m}$ </p>
Rule 001-01b extend axes	<p> $\forall A, B: A, B \in \{R_{ef}\}$ $60^\circ \leq \angle \gamma \leq 180^\circ$ $\alpha 1: \emptyset \rightarrow a, a \in \text{TN}$ </p>
Rule 001-01c draw parallel axes	<p> $\forall A: A \in \{R_{ef}\}$ $a \parallel a, 0 \leq d \leq 200 \text{ m}$ $\alpha 1: \emptyset \rightarrow a, a \in \text{TN}$ </p>
Rule 001-02 Draw street that extends from bridge	<p> $0^\circ \leq \angle \gamma \leq 70^\circ$ $\forall A, B: A, B \in \{R_{ef}\}, B$ is a bridge $\alpha 1: \emptyset \rightarrow a, a \in \text{TN}$ </p>
Rule 001-03a Select axis drawn	<p> $\forall a_n \in \text{AN}$ $\exists a_{sel} \in \text{TN}: [a_{sel}] \parallel [p1] < 80\%$ $\alpha 1: a_n \rightarrow a_{sel}$ </p>
Rule 001-03b Selected axis that meet criteria remains as axis	<p> $\forall a_{sel} \in \text{TN}$ $0^\circ \leq \angle \gamma \leq 60^\circ$ $\forall A, B, C: A, B, C \in \{R_{ef}\}, B$ is a bridge $\alpha 1: a_{sel} \rightarrow a_n, n \in [3, 4, 5]$ $a_{sel}, a_n \in \text{TN}$ </p>
Rule 001-03c Else, erase the axes	<p> $0^\circ \leq \angle \gamma \leq 60^\circ$ $\alpha 1: a_{sel} \rightarrow \emptyset$ </p>

Fig. 5 Rules for city-centric growth

UIP-002: Rules to draw urban grid based on axes	
Rule (number)	Rule (representation)
<p>Rule 002-01 Draw "perpendicular" axes(p)</p>	<p> $\forall A, B: A, B \in \{R_{ef}\}, a_n \in \mathbf{TN}$ $70^\circ \leq \angle \gamma \leq 90^\circ$ </p> <p> $\alpha 1: a_n \rightarrow p_n, p_n \in \mathbf{TN}$ $\alpha 1: \emptyset \rightarrow p_n, p_n \in \mathbf{TN}$ </p>
UIP-003: Rules to modify street hierarchy	
Rule (number)	Rule (representation)
<p>Rule 003-02a Street categorization</p>	<p> $\forall B: B \in \{R_{ef}\}, B \text{ is a bridge}$ $a_n \in \mathbf{TN}$ </p> <p> $\alpha 1: a_n \rightarrow r_1, r_1 \in \mathbf{TN}$ refers to expressways with high-speed tunnel and highway </p>
<p>Rule 003-02b street categorization</p>	<p> $a_n \in \mathbf{TN}$ </p> <p> $\alpha 1: a_n \rightarrow r_2, r_2 \in \mathbf{TN}$ refers to the primary city street system </p>
UIP-004: Rules to subdivide grid into city block, to subdivide block into plots	
Rule (number)	Rule (representation)
<p>Rule 004-01 block insertion</p>	<p> $\forall a_n: n \in \{1,2,3\}$ </p> <p> $\alpha 1: \emptyset \rightarrow bl$ $d^1 = \frac{a_n width}{2}, d^2 = \frac{a'_n width}{2}, d^3 = \frac{a''_n width}{2}, d^4 = \frac{a'''_n width}{2}$ </p>
<p>Rule 004-02 block division by pre-existing limit</p>	<p> $\forall pl: pl \in \{R_{ef}\}, pl \text{ includes railway, stream, rural street, urban street}$ </p> <p> $\alpha 1: bl \rightarrow bl, bl'$ </p>

Fig. 5 (continued)


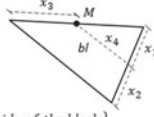
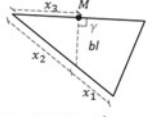
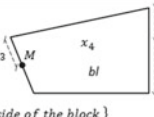
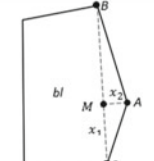
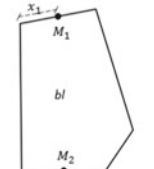
<p>Rule 004-04 block merge</p>		<p>$\alpha 1: bl, bl' \rightarrow bl$</p>
<p>Rule 004-05a block subdivision</p>		<p>$\forall M: M \in \{R_{ef} \text{ on any side of the block}\}$ $x_3 = n \times m_{\text{length of the side}}, n \in \left\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\right\} \wedge x_3 \geq 100m$ if $x_4 \parallel$ one side of the block, $x_1, x_2 \geq 100m$</p> <p>$\alpha 1: \emptyset \rightarrow q_n$ $q_n \parallel$ any of the other two sides of the block</p>
<p>Rule 004-05b block subdivision</p>		<p>$\forall M: M \in \{R_{ef} \text{ on any side of the block}\}$ $x_3 = n \times m_{\text{length of the side}}, n \in \left\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\right\} \wedge x_3 \geq 100m$ if $\angle \gamma = 90^\circ, x_1, x_2 \geq 100m$</p> <p>$\alpha 1: \emptyset \rightarrow q_n$ $q_n \perp$ the side of the block that M locates</p>
<p>Rule 004-05c block subdivision</p>		<p>$\forall M: M \in \{R_{ef} \text{ on any side of the block}\}$ $x_3 = n \times m_{\text{length of the side}}, n \in \left\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\right\} \wedge x_3 \geq 100m$ if $x_4 \parallel$ one side of the block, $x_1, x_2 \geq 100m$</p> <p>$\alpha 1: \emptyset \rightarrow q_n$ $q_n \parallel$ any of the other two sides of the block</p>
<p>Rule 004-05d block subdivision</p>		<p>$\forall A, B, C: A, B, C \in \{R_{ef} \text{ on corner of the block}\}$ if $x_2 \perp BC , \frac{1}{3} BC \leq x_1 \leq \frac{2}{3} BC \wedge x_1 \geq 100m$</p> <p>$\alpha 1: \emptyset \rightarrow q_n$ $q_n \parallel$ any of the other two sides of the block</p>
<p>Rule 004-05e block subdivision</p>		<p>$\forall M_1, M_2: M_1, M_2 \in \{R_{ef} \text{ on any side of the block}\}$ $x_1, x_2 = n \times m_{\text{length of the side}}, n \in \left\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\right\} \wedge x_1, x_2 \geq 100m$</p> <p>$\alpha 1: \emptyset \rightarrow q_n$ $q_n \parallel$ any of the two other sides of the block</p>

Fig. 5 (continued)


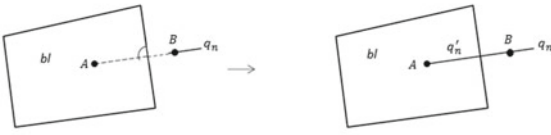

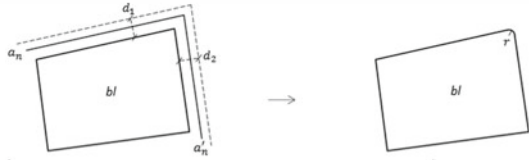
<p>Rule 004-05h block subdivision</p>	 <p>$\forall A, B: A, B \in \{R_{ef} \text{ on any corner of the block}\}$ $x_1, x_2, x_3, x_4 \geq 100m$</p> <p>$\alpha 1: \emptyset \rightarrow q_n$</p>
<p>Rule 004-06 extend the line of subdivision</p>	 <p>$\forall A, B: A, B \in \{R_{ef}\}$</p> <p>$\alpha 1: \emptyset \rightarrow q'_n, q_n \parallel q'_n$</p>
<p>Rule 004-07 assign label to the street</p>	 <p>$\alpha 1: q_n \rightarrow a_n, n \in \{4, 5\}$ $q_n, a_n \in \mathbf{TN}$ $r005: R_{ef} \rightarrow \emptyset$</p>
<p>Rule 004-08 round block corner</p>	 <p>$\text{if } d_1 > d_2, r = \frac{1}{2} d_1 (\text{width of } a_n)$ $\text{if } d_2 > d_1, r = \frac{1}{2} d_2 (\text{width of } a'_n)$</p> <p>$\alpha 1: r = 0 \rightarrow r = \frac{1}{2} d$</p>

Fig. 5 (continued)

Results and Discussion

Departing from the preconceived idea that rural and urban developments are not inter-related, by comparing the corresponding rules, subset by subset, we show that RIPs and UIPs yield very similar sequences of rural and urban growth, respectively. Both begin by placing major axes based on pre-existing limits, including river, streams, hill foote, and rail tracks (RIP-001, UIP-001). Both RIPs and UIPs contain rules to form a grid-like pattern by creating “perpendicular lines” (RIP-002, UIP-002), and rules for further subdivision. Nonetheless, small differences exist between the two, which lead to significant variations in the appearance of the two grids.

UIPs contain rules for orientation calibration (UIP-001-03a, b, c, 04a, b) while RIPs do not. The urban grid is planned at the site scale beforehand by the municipality, making sure the grid is consistent within each subdistrict. The rural pattern is more spontaneously generated by the local community without negotiation across the whole area. The calibration rules select axes that are connected to bridges and, at the same time, oriented north–south. Axes that do not meet these requirements are erased. This results in a city grid that is tilted relative to the rural fabric.

The configuration of grids that result from RIPs varies more widely because of the variable x_n , as the length of the grid (x) is positively correlated with the number of households living in the block (n) (RIP-002-04). On the other hand, city-centric urbanization requires equal-sized urban blocks with an equal number of lots. Also, urban blocks tend to be larger than rural plots.

The biggest difference in rules can be found in the grid subdivision class (RIP-004, UIP-004). Subdivision in RIPs operates in a bottom-up fashion: individual housing plots are inserted into the housing tracts (RIP-004-A, 01a, b, 02, 03a, b, c, d, e) and farmland plots are divided equally and assigned to each household (RIP-004-04, 05a, b, c, d). By contrast, the urban grid is further subdivided based on points located at $1/3$, $1/2$, or $2/3$ of the length of block edges (UIP-004-05a, b, c, d, e, f, g, h). This results in the urban grid being divided in a top-down fashion according to the size of urban blocks.

The urban street system is more complex than the network of rural roads, and the street width and profile components in TN classes vary considerably according to the urban context (RIP-003, UIP-003). Rural roads contain two choices of width [5, 10 m] without an obvious hierarchical system, but they might be transformed into irrigation watercourses based on the locations of plots and houses. With the rapid increase in city-centric urban development, land subtraction rules and village clustering rules are introduced the rural grammar (RIP-005-01a, b, c, d) because of city-centric urbanization (RIP-005-02, 03a, b, c, d). Our analysis shows that urban growth is still strong, not presenting signs of slowing down. Table 3 summarizes the similarities and differences between the two types of patterns.

Table 3 Rural and induction patterns comparison

	RIPs	UIPs
Sequence of inductions patterns	Major structure → Grid formation → Plot subdivision → Road transformation → Subtraction and concentration	Major axes → Grid formation → Block subdivision → Hierarchy transformation
Grid	Yes	Yes
Axes based on pre-existing limits	Yes	Yes
Orientation and calibration	No	Yes
Land cover	Concentrating, decreasing	Expanding, increasing
Subdivision	Bottom-up	Top-down
Grid scale	Smaller	Larger
Grid configuration	Irregular, variable: household	More equal-sized grids
Street classification	No hierarchy	With street object class

Conclusion and Future Work

This study analyzes the growth of an urban fringe in a city in China over a period of 17 years using shape grammars. The analysis resulted in the inference of Rural and Urban Induction Patterns that describe the conversion of rural into urban areas. These patterns show that urban expansion does not evolve from scratch but rather relies on the fabric of the existing territory. They also show that although people usually preconceive rural development as informal and chaotic, it is a rule-based process that shares similarities with urban development. This suggests that rural areas can be more seamlessly incorporated into city-centric planning.

As rural areas evolve into urban areas, the quality of future urban environments will depend on the quality of rural environments from which they originate. To accomplish this in the dualistic land management paradigm, we need to plan rural areas as much as we plan urban ones, so that urban villages can coexist within new urban districts in a more balanced way. This requires policymakers in China to change the idea that the city-centric process is the only legitimate mode of urbanization. This unilateral view of planning should be substituted by a collaborative planning process between city authorities and local rural communities. Then administration-driven planning could eliminate the disparity between the urban and rural landscapes. Rural village development driven by locals could preserve the old family household culture, guaranteeing social cohesion and a sense of attachment to the land. Potential reformation of planning strategies in the administrative structure needs therefore to be discussed.

The study also affirms the feasibility of using grammars to plan how rural patterns may be preserved in urban grid designs. Shape grammars could be utilized to coordinate both types of development. The generic induction patterns for the Binjiang

district provide a generative system that does not restrict development but rather offers planning flexibility [12]. This system is composed of several generic grammars that describe the land use change and urban grid generation in the district from 2000 to 2017, but it could be used by designers to synthesize new specific grammars by manipulating the induction patterns and associated rules.

By identifying key elements in land development, induction patterns can be used to control urban growth. Designers can either change existing rules by modifying rule parameters to restrict design variations or add new rules to explore new design ideas. By addressing both urban and rural growth, grammar-based patterns can be used to design new districts and redevelop existing urban villages in fringe areas. It could support new policies in which rural communities could remain as desired by locals in the rational city-centric planning process and coexist with industrial development.

References

1. Zhang Y (2014) Study on the development of urban and rural integration and its development strategy in the urbanization of nanjing. *Sichuan Archit* 6
2. Pryor R (1968) Defining the rural-urban fringe. *Soc Forces* 47(2):202–215. <https://doi.org/10.2307/2575150>
3. Potter S, Potter J (1990) *China's peasants: the anthropology of a revolution*. Cambridge University Press, Cambridge. <https://doi.org/10.1017/CBO9780511607783>
4. Wang YP, Wang Y, Wu J (2009) Urbanization and informal development in china: urban villages in Shenzhen. *Int J Urban Reg Res* 33:957–973 <https://doi.org/10.1111/j.1468-2427.2009.00891.x>
5. Knight J, Song L (1999) *The rural-urban divide. Economic disparities and interactions in China*. Oxford University Press, Oxford
6. Yang X, Day J, Han SS (2015) Urban peripheries as growth and conflict spaces: the development of new towns in china. In: Wong TC, Han S, Zhang H (eds) *Population mobility, urban planning and management in china*. Springer, Cham
7. Unknown (2017) Binjiang District overview. Hangzhou archives. Retrieved from 14 Dec 2017 http://www.hzarchives.gov.cn/infocenter/government/overview/t20110107_30927.htm
8. Unknown (2007) Overall planning guideline. Living in Hangzhou. Retrieved from 12 Sep 2007 <http://zzhz.zjol.com.cn/05zzhz/system/2007/09/12/008791497.shtml>
9. Tang J (2016) The city successfully completed the entire village relocation in 15 urban village in Binjiang district Hangzhou. *Zhejiang daily*. Retrieved from 6 Jun 2016 <http://hangzhou.zjol.com.cn/system/2016/06/06/021178335.shtml>
10. Gil J, Duarte JP (2008) Towards an urban design evaluation framework. In: *Architecture in Computro [26th eCAADe conference proceedings/ISBN 978-0-9541183-7-2]* Antwerpen (Belgium), 17–20 Sep 2008, pp 257–264
11. Alexander C, Ishikawa S, Silverstein M (1977) *A pattern language: towns, buildings, construction*. Oxford University Press, New York
12. Stiny G (1980) Introduction to shape and shape grammars. *Environ Plan* 7:343–351
13. Hillier B (1996) Space is the machine—a configurational theory of architecture. In: Nikolopoulou M (ed) *Designing open spaces in the urban environment: a bioclimatic approach*. Centre of Renewable Energy Resources, Greece, 2004. Cambridge University Press, UK
14. Beirão JN, Duarte, JP, Stouffs R (2009) An urban grammar for Praia: towards generic shape grammars for urban design. In: *Computation: The New Realm of architectural design [27th eCAADe conference proceedings/ISBN 978-0-9541183-8-9]* Istanbul (Turkey), 16–19 Sep 2009, pp 575–584

15. Duarte JP (2005) A discursive grammar for customizing mass housing: the case of Siza's houses at Malagueira. *Autom Construct* 14(2):265–275
16. Beirão JN, Duarte JP, Gil J, Montenegro N (2009) Monitoring urban design through generative design support tools: a generative grammar for Praia. In: Proceedings of the 15th APDR congress on networks and regional development, Cidade da Praia, Cabo Verde
17. Beirão JN, Mendes G, Duarte JP, Stouffs R (2010) Implementing a generative urban design model: grammar-based design patterns for urban design. In: Future cities [28th eCAADe conference proceedings/ISBN 978-0-9541183-9-6] ETH Zurich (Switzerland), 15–18 Sep 2010, pp 265–274
18. Gamma E, Helm R, Johnson R, Vlissides J (1995) Design patterns: elements of reusable object-oriented software. Addison-Wesley, Reading, MA
19. Gruber TB (1993) A translation approach to portable ontology specifications. *Knowl Acquisition* 5(2):257–267
20. Beirão JN, Duarte JP, Stouffs R (2011) Nexus Netw J 13:73. <https://doi.org/10.1007/s00004-011-0059-3>
21. Figure 1. 2000–2016 Land Use Change in Aerial View. Google Earth, Hangzhou, (eds) Author

Composite Shape Rules



Rudi Stouffs and Dan Hou

Generally, non-terminal symbols such as labeled points are used to constrain rule application and, thereby, guide rule selection in the application of shape grammars. However, distinguishing between salient rules that offer the user design choices and deterministic rules that together and in a certain order (mechanically) complete a specific design transformation, may require other means of guiding rule selection that better reflect on the logic of the rule derivation process. We present a concept of composite shape rules embedding algorithmic patterns for rule automation. We denote these composite shape rules *flows*, and adopt a notation from regular expressions. In this paper, we describe the context that led to the conception of this approach, describe the sequencing mechanisms, and present a case study. We conclude with a brief discussion disclosing additional potential of the notation.

Introduction

Shape grammars are a formal rewriting system for producing languages of shapes [1]. At a minimum, a shape grammar consists of a set of productions, or shape rules, operating over a vocabulary of spatial elements, e.g., line or plane segments, optionally augmented with qualitative attributes, e.g. line thicknesses or colors. Then, a shape is defined as any composition of (augmented) spatial elements, and a shape rule as any combination of a left-hand-side shape and a right-hand-side shape, where the former cannot be empty. A shape rule applies to a shape if a transformation can be determined under which the left-hand-side of the shape rule is a part of the given

R. Stouffs (✉)
National University of Singapore, Singapore, Singapore
e-mail: stouffs@nus.edu.sg

D. Hou
Tianjin University, Tianjin, China

shape. Rule application involves replacing this part with the right-hand-side of the shape rule under the same transformation.

Traditionally, a distinction is made between the terminal vocabulary of (augmented) spatial elements and a non-terminal vocabulary of symbols or markers, e.g., labeled points [1, 2]. In this case, a shape and, by extension, both sides of a shape rule may combine both spatial elements and symbols from the respective vocabularies. A shape grammar conceived in this way generally includes an initial shape as the starting point in the productive (generative) process. The language defined by such a shape grammar is the set of shapes generated by the grammar from the initial shape that do not contain any non-terminal symbols.

While this traditional approach may seem overly formalistic in comparison to simply ‘calculating with shapes’ [3], most shape grammars presented in literature do adopt a notion of non-terminal symbols. Often, shape grammars emanate from an analysis of a particular body of architecture or are designed in order to generate a particular type or style of building. Any restrictions posed on the generative language generally necessitate constraining the application of rules. Non-terminal symbols are commonly used to constrain rule application and, thereby, guide rule selection. In addition, a shape grammar may need to specify a large number of rules, further exacerbating the problem of constraining rule applications. While rules may be collected into stages, stages often are defined to rely on distinct non-terminal symbols.

The downside of using non-terminals is that they clutter the shape rule and make both the rule and its role in the derivation process more difficult to understand. Such an understanding must necessarily include the role of these non-terminals. Unfortunately, there is usually little relationship between the specification of non-terminals (including their naming) and the logic of the rule derivation process, beyond the identification of stages.

Our motivation for this study comes from an active development of a design grammar using railway station design as a demonstration study. Design grammars, also termed ‘grammars for designing’ [4], denote grammars that are progressively developed by designers for a new design context. They are distinct from analytical grammars that are developed from a specific body of designs, e.g., similar buildings by the same architect, and are constrained to only generate designs from that body or designs that can be assessed as belonging to the same body. Developing an analytical grammar involves systematically determining all possible rule variations and encoding these into a grammar. Rule variations are necessarily finite and the encoding is done by the developer of the grammar, not by the user. Rule complexity is therefore not much of an issue and non-terminal symbols can be used to guide rule selection and derivation. For a design grammar, however, the designer is both the developer and the user of the rules; rule development becomes an important issue and reducing the need for using non-terminal symbols in the specification of shape rules a critical objective.

In this paper, we consider an algorithmic approach to rule sequencing, defining composite rules composed of shape rules or other composite rules and including sequencing directives. Firstly, a brief overview of the literature on rule sequencing is presented. Subsequently, the selected sequencing mechanisms are described

and explained. Next, a case study is presented. Finally, a brief discussion discloses additional potential of the notation.

Rule Sequencing

Knight [5] defines a deterministic grammar as a grammar imposing a restriction on rule ordering, without any restriction on rule format. She considers at least two different ways to define a deterministic grammar. Firstly, a function can be associated with any grammar to determine for each step of a derivation which rule to apply next (and under which transformation). Secondly, rules are distinguished (e.g., using non-terminal symbols) in such a way that at most one rule is applicable (under only one transformation) in each step of a derivation [5].

Liew [6] similarly considers deterministic rules as rules that together and in a certain order (mechanically) complete a specific design transformation. He distinguishes deterministic rules from salient rules, which offer the user design choices. Specifically, Liew [6] conceives of an explicit sequence of rules, denoted a 'directive' or 'macro'. This directive takes the form of an if-then-else control structure where the condition is a rule application that either succeeds (true) or fails (false) and the consequence (then or else) is another rule application, possibly applying the same rule. Either the then (success) or else (failure) part of the control structure can be omitted, allowing the rule sequence to end. The first rule in the rule sequence is denoted the primary rule, the others secondary rules. It is understood that primary rules are salient rules, whereas secondary rules belong to the deterministic category. Whereas Knight deems deterministic as emergent from the fact that in each step of the derivation at most one rule applies under a single transformation, Liew considers a prescriptive, algorithmic approach.

Grasl and Economou [7] present different approaches to automating rule selection using rule selection agents that implement a sensor-logic-actuator mechanism. The sensor allows an agent to inquire about the shape at hand or apply 'control' rules to determine whether a specific condition is met. These control rules do not make any changes to the shape, that is, their left-hand-side and right-hand-side are identical. The actuator allows an agent to apply a specific rule or undo the previous rule application. Grasl and Economou [7] consider different agents implementing different rule selection approaches. For example, one agent performs a depth-first search over a set of rules in order to enumerate all possible derivations. Another agent randomly picks a rule, though its performance may be improved using sequencing or weighted randomness. Variant approaches include a genetic algorithm approach and a rule-based approach. In the latter case, a backward chaining technique is used to find the appropriate rule sequence, using control rules to decide which action to take next [7].

Grasl and Economou do not distinguish salient from deterministic rules, assuming that all rules are salient to some extent. This is very much in line with Stiny's [3] concept of calculating with shapes, which amounts to having a grammar with a limited

number of rules that can generate an unlimited number of designs. Unfortunately, shape grammars that—assist to—generate architectural designs often require a large number of rules and, while the creative process of salient rule development and selection is obviously of primary interest to the designer, deterministic rules may need to complement salient rules in order to achieve a desired result. As such, this work is motivated by the desire to be able to automate the application of deterministic rules following the selection and application of a salient rule. That is, we recognize that while design grammars are developed by the user of the grammar, thereby necessitating simpler rules, parts of a design grammar may need to be automated or semi-automated. Instead of suggesting the use of non-terminals to guide automated rule selection, we propose the use of composite shape rules embedding algorithmic patterns for rule automation.

Flows

We denote composite shape rules embedding algorithmic patterns as *flows*, in order to clearly distinguish shape rules from composite shape rules, where necessary. At the basic level, we support sequence, iteration and selection as algorithmic patterns for *flows*. Given a rule r specified in the form $lhs \rightarrow rhs$, with lhs and rhs denoting the left-hand-side and right-hand-side of the rule, respectively, rule r applies to a shape s if there exists a transformation $t \in T$ such that $t(lhs) \leq s$. The set T is generally considered to contain all similarity transformations, that is any composition of translation, rotation, reflection and uniform scaling. Then, the application of rule r to s under transformation t yields the shape $s \leftarrow t(lhs) + t(rhs)$.

Two rules r_1 ($lhs_1 \rightarrow rhs_1$) and r_2 ($lhs_2 \rightarrow rhs_2$) apply in sequence if upon a successful application of r_1 , r_2 is applied to the shape resulting from the application of rule r_1 . Considering Liew's [6] 'directive', this can be graphically represented as in Fig. 1(left). Algorithmically, this may be written in the form:

```

if  $\exists t_1 \in T : t_1(lhs_1) \leq s$  then
   $s \leftarrow s - t_1(lhs_1) + t_1(rhs_1)$ 
  if  $\exists t_2 \in T : t_2(lhs_2) \leq s$  then
     $s \leftarrow s - t_2(lhs_2) + t_2(rhs_2)$ 
  end
end
end

```

A single rule r_1 can be applied iteratively if upon every successful application of r_1 , a new application is attempted (Fig. 1(center)). Algorithmically, this can be expressed using a while-do construct:

```

while  $\exists t_1 \in T : t_1(lhs_1) \leq s$  do

```

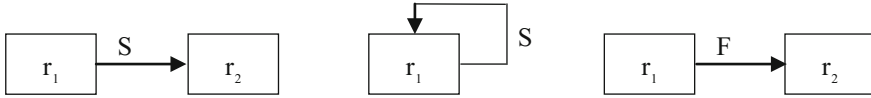


Fig. 1 Sequence (left), iteration (center) and selection (right), graphically represented using Liew’s [6] ‘directive’. Rule application, represented as a rectangle, may be successful or not. Dependent on success or failure, the next rule to be applied is indicated by an arrow marked ‘S’ or ‘F’, respectively. Such arrow is omitted if no subsequent rule is available

```

s ← s - t1(lhs1) + t1(rhs1)
end
    
```

Finally, selection specifies two (or more) alternative rules. These are attempted to be applied in order and as soon as one rule applies, the remaining rules are ignored (Fig. 1 (right)). Algorithmically, this can be written as follows:

```

if ∃ t1 ∈ T : t1(lhs1) ≤ s then
    s ← s - t1(lhs1) + t1(rhs1)
else if ∃ t2 ∈ T : t2(lhs2) ≤ s then
    s ← s - t2(lhs2) + t2(rhs2)
end
    
```

Backtracking

The sequential application of two rules may assume the successful application of both rules. Both Fig. 1 and the algorithm above only require rule r_1 to be successful. If rule r_2 subsequently fails, rule r_1 will remain applied. However, in many cases, we may want rule r_1 to apply only if r_2 subsequently applies as well, that is, rules r_1 and r_2 are considered as (part of) a sequence of rules that all apply or none at all. This is difficult to express algorithmically using the constructs above (if-then-else and do-while), as we would need to combine the two conditions, whether rules r_1 and r_2 apply, as well as the intermediate calculation of the result of rule r_1 , all within the condition of a single if-then structure before actually applying both rules. If the sequence contains more than two rules, the algorithmic expression will only be more complicated. However, such is simply an application of backtracking which can be achieved using a recursive algorithmic structure. More importantly, Liew’s [6] ‘directive’ does not support backtracking, as such, we must adopt a different way of expressing a *flow* (composite shape rule).

Instead, we adopt the notation from regular expressions. Regular expressions are patterns that are used to match strings by string searching algorithms. Regular expressions are composed of tokens that are combined in a prescribed order, with

some variation built into the expression, in order to match a goal string. Similarly, *flows* are composed of shape rules that are combined in a prescribed order, with some algorithmic variation, in order to produce a valid final shape. In both cases, partial matches may not lead to any final result, requiring backtracking to undo the partial match and attempt a different match. The main difference lies in the fact that in the case of regular expressions, the goal string is given and can guide the matching process. In the case of composite shape rules, the goal is to arrive at any valid final shape. As such, there is no guide but the algorithmic expression of rules, and a purely trial-and-error approach must be adopted.

Another difference lies in the vocabulary of terminals. In the case of regular expressions, these are any character as can be represented in a string. These characters are finite and ordered. As such, regular expressions generally use shortcuts such as ‘.’ to match any single character, ‘a-z’ to indicate any letter between ‘a’ and ‘z’, ‘a’ and ‘z’ included, and ‘[\wedge abc]’ to match any character other than ‘a’, ‘b’ or ‘c’. The latter may be useful when explicitly searching for combinations of ‘a’, ‘b’, and ‘c’, treating any other characters simply as separation characters, thereby ignoring the specific characters used for separation, or vice versa. In the case of *flows*, we also have a finite set of rules. However, they are usually unordered, or only partially ordered, and we tend to be only interested in a very limited subset of rules at any one time. As such, we require rules to be explicitly enumerated at all times, rather than be identified as a group or by exclusion.

A Notation from Regular Expressions

Let us revisit the algorithmic structures of sequence, selection and iteration. In a regular expression, a sequence of characters can be literally explicated as such, e.g., ‘abc’ matches the substring ‘abc’. In the case of rules, we use rule names to identify individual rules and separate these rule names with spaces. Note that rule names can be required to be identifiers, that is, any combination of letters, digits, or the underscore symbol (‘_’), excluding any spaces or other special characters that could be misinterpreted. Thus, ‘ $r_1 r_2$ ’ (or ‘ $r1 r2$ ’) is a representation for the sequence of two rules r_1 and r_2 .

In terms of selection, matching one from a series of alternative tokens, regular expressions commonly offer two variant notations. The first one we have touched upon before and uses square brackets to collect alternative tokens. We adopt the same notation for the selection and application of a single rule from a number of alternative rules, separating the rule names with spaces and enclosing them within square brackets. Thus, ‘[$r_1 r_2$]’ (or ‘[$r1 r2$]’) is a representation for the selection of one from two rules r_1 and r_2 . The second notation commonly used within regular expressions is to separate the tokens (or sequences or groups of tokens) with vertical bars. We omit this notation for *flows*. Instead, as is possible within regular expressions as well, we allow for the grouping of a sequence of rules within parentheses. Thus, ‘[$r_1 (r_2 r_3)$]’ (or ‘[$r1 (r2 r3)$]’) is a representation for the selection between rule r_1

and the sequence of rules r_2 and r_3 . A grouping of rules can be considered to specify a composite rule; both square brackets and parentheses group rules into composite rules. Much of the notation below applies equally to rules and composite rules; we will write (composite) rule when we want to ignore the distinction between a single rule or a group of rules. Otherwise, we will use the term *sub-flow* to denote a grouping of rules as distinct from a single rule.

In a regular expression, a quantifier after a character (or token) specifies how often the character (or token) is allowed to occur. We allow the same quantifiers to be used in the expression of *flows*. The following quantifiers are distinguished (Table 1):

- A question mark ('?') indicates at most one (zero or one) application of the preceding (composite) rule.
- An asterisk ('*') indicates any number (zero or more) of applications of the preceding (composite) rule.
- A plus sign ('+') indicates any number, excluding zero, (one or more) of applications of the preceding (composite) rule.
- Any positive number n within curly brackets ('{ n }') requires the preceding (composite) rule to be applied exactly n times.
- Any positive number n followed by a comma, within curly brackets ('{ n ,}') requires the preceding (composite) rule to be applied minimally n times.
- Any two positive numbers n and m , separated by a comma and within curly brackets ('{ n , m }') requires the preceding (composite) rule to be applied minimally n times, but no more than m times.

A Greedy Algorithmic Approach

In the case of iterating a (composite) rule, the quantifier only specifies how many times the (composite) rule may be applied. For example, in the case of the asterisk, the preceding (composite) rule may apply zero, one or more times. We may interpret this to mean as many times as possible, however, we may as well opt to not apply the (composite) rule at all. In a regular expression, how many times a token, when followed by an asterisk, is matched usually is dependent on the string that is being matched. For example, matching the regular expression 'a*b' to the string "aaab" will normally force the token 'a' to be matched three times such that the token 'b' can match the letter 'b'. Similarly, given a *flow* ' $r_1 * r_2$ ', there may be only one solution for the number of times rule r_1 must be applied in order for the application of rule r_2 to succeed.

However, often, there may be multiple solutions involving a different number of applications for rule r_1 leading to a successful application of rule r_2 . For example, in Stiny and Mitchell's [8] Palladian grammar, in stage 1 (grid definition), rule 5 allows for the extension of the grid by adding two additional grid columns, one on either side of the existing grid columns. When applying the grammar to generate the plan for the Villa Malcontenta, rule 5 is applied exactly once, yielding a grid that is five

Table 1 Overview of the regular expression-inspired notation for *flows* (and sub-*flows*)

Metacharacter	Description
~	A space separates two rules in a sequence . If either rule fails to apply, the entire sequence fails to apply
[...]	Square brackets enclose a selection of alternative rules or sub- <i>flows</i> . Alternatives are attempted to be applied in the order specified. As soon as one application succeeds, subsequent rules are skipped. If no alternative applies, backtracking will occur. A selection specifies a <i>sub-flow</i>
(...)	Parentheses can be used to group a sequence of rules or sub- <i>flows</i> . A group specifies a <i>sub-flow</i>
?	The preceding rule (or sub- <i>flow</i>) may apply zero or one time. No backtracking occurs in the case of rule application failure
*	If following a rule (or sub- <i>flow</i>), this rule (or sub- <i>flow</i>) may apply zero, one or more times. In a greedy approach, application will be repeated as many times as possible, until rule application fails If preceding a selection (enclosed in square brackets), the alternative rules or sub- <i>flows</i> in this selection are attempted to be applied in a random order instead of in the order specified
+	The preceding rule (or sub- <i>flow</i>) may apply one or more times. In a greedy approach, application will be repeated as many times as possible, until rule application fails. However, if rule application fails at the very first try then backtracking will occur
{ <i>n</i> }	The preceding rule (or sub- <i>flow</i>) may apply exactly <i>n</i> times. If application succeeds fewer times, these applications will be undone and backtracking will occur
{ <i>n</i> ,}	The preceding rule (or sub- <i>flow</i>) may apply <i>n</i> or more times. If application succeeds fewer than <i>n</i> times, these applications will be undone and backtracking will occur. Otherwise, in a greedy approach, application will be repeated as many times as possible, until rule application fails
{ <i>n</i> , <i>m</i> }	The preceding rule (or sub- <i>flow</i>) may apply any number of times between <i>n</i> and <i>m</i> . If application succeeds fewer than <i>n</i> times, these applications will be undone and backtracking will occur. Otherwise, in a greedy approach, application will be repeated as many times as possible, but at most <i>m</i> times

columns wide. While most of Palladio’s villa plans are based on a five by three grid, Palladio also presents a villa ground plan in Book 2 of his *Quattro Libri* [9] based on a three by three grid [10]. As such, we might choose to use the quantifier ‘?’ for rule 5, allowing rule 5 to be applied either zero or one time. Nevertheless, since both zero and one time are equally applicable in generating valid designs, how should the algorithm select between these two options?

In fact, the same issue applies to regular expressions, as in some cases there might seem to be multiple valid options at first. For example, matching the regular expression ‘a*.b’ to the string ‘aaab’ should force the token ‘a’ to be matched twice, with the token ‘.’ matched to the third letter ‘a’, such that the token ‘b’ can match the last letter ‘b’. However, the string matching algorithm may first attempt to match

the token ‘a’ three times, followed by the token ‘.’ (matching the letter ‘b’), before realizing that the regular expression cannot be matched in this way as there is no letter ‘b’ remaining. It may then backtrack, reduce the number of matches of the token ‘a’ by one, leading to a complete match. Alternatively, it can first try skipping the token ‘a’, backtrack and match the token ‘a’ once, and backtrack once again before matching the token ‘a’ twice to successfully lead to a complete match. The first approach is denoted greedy matching, the second one lazy matching. Greedy matching is usually the default behavior for regular expressions.

Similarly, we apply a greedy approach to the application of *flows*. In the discussion, we will briefly consider other approaches as well.

Selecting from Multiple Rules and Valid Applications

Other issues that must be dealt with, different from regular expressions, are the order of selection from among alternative (composite) rules and the order of selection from among multiple valid applications of a same rule.

In the case of selection, the alternative (composite) rules are necessarily specified in some order. If we are interested in all possible derivations, the order is of little concern, but if we are exploring only a few derivations, we might not want these derivations to attempt to apply the (composite) rules each time in the same order. Therefore, we consider two alternative orders. The first is the order of the (composite) rules in the selection group. This is the default. The second is a random re-ordering of the (composite) rules in the selection group for the purpose of (composite) rule selection and application. In order to achieve this random re-ordering, we precede the selection (enclosed in square brackets) by an asterisk (*). In order to ensure the asterisk is not interpreted as a quantifier relating to any preceding rule or sub-*flow*, there can be no space between the asterisk and the opening square bracket. For example, in the case of ‘*[r₁ r₂]’ (or ‘*[r₁ r₂]’), the two rules r_1 and r_2 will be attempted to be applied in any order. Obviously, as soon as one application succeeds, any other (composite) rules that have not been tried yet are skipped.

In the case of multiple valid applications of the same rule, one application must be selected. Any ordering of valid alternative applications will be the result of the matching algorithm and may not be obvious to the user. For this reason, we consider a random selection of a single application from among multiple valid applications of a same rule. Only in a very few cases may a fixed ordering be appropriate. For example, in the case of developing and testing a *flow*, when some debugging is necessary, it may be appropriate for the application of a rule to always yield the same result, so as to be able to compare the *flow* result before and after debugging. For this, we do not provide any notation, but include a parameter within the implementation that can be set to apply to any *flow* in its entirety.

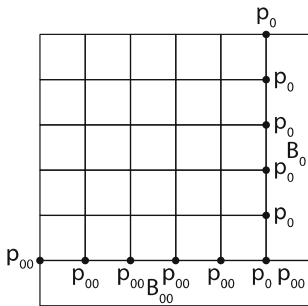
Considering the selection mechanisms above, note that *flows* are not exactly deterministic and generally include a probabilistic aspect.

Case Study: Layout Generation Problem

We consider an abstracted case study from the railway station design grammar previously mentioned. It concerns a layout generation problem for which we prefer a semi-automated solution. We consider a grid composed of 25 cells with two predefined, adjacent spaces (Fig. 2 above). We consider a set of activities ('B1' through 'B7') to occupy these cells. Each activity identifies how many times it should be made available and how much space must be provided (Table 2).

We adopt descriptions to encode all non-geometric information [11]. Static information is encoded in the descriptions activities and adjacencies. The results of the activity assignment process are encoded in the descriptions 'assigned' and 'unassignedCount'. In addition, we consider a few descriptions to contain temporary information. These are 'adjacenciesUse', 'adjacenciesNece', 'adjacenciesFound', 'assignment', 'surrounding' and 'candidates' [12]. Finally, each geometric data type, i.e., points, line segments and plane segments, has a description attribute, respectively, 'ptD', 'lnD' and 'plD'. Figure 2 presents the initial shape and descriptions; Table 3 shows the rules for the layout generation problem.

Figure 3 offers an abstract, graphical illustration of the *flow*, presenting the groups of rules and their roles in the *flow* and including intermediary results from the derivation. Figure 4 offers an expanded graphical illustration of the *flow* distinguishing individual rules, using Liew's [6] 'directive'.



activities: {('B₁', 1, 12, 2), ('B₂', 2, 5, 2), ('B₃', 1, 18, 5), ('B₄', 2, 8, 0), ('B₅', 1, 10, 3), ('B₆', 1, 4, 1), ('B₇', 1, 7, 1)}

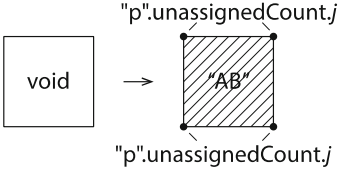

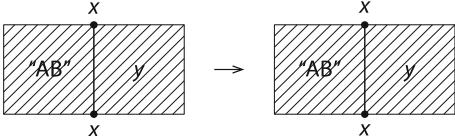
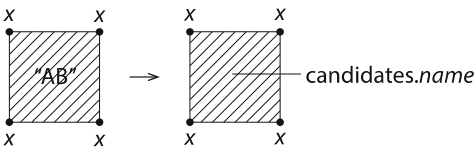
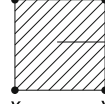
adjacencies: {('B₀', 'B₃'), ('B₀₀', 'B₁'), ('B₁', 'B₀₀'), ('B₁', 'B₇'), ('B₂', 'B₃'), ('B₂', 'B₅'), ('B₃', 'B₀'), ('B₃', 'B₂'), ('B₃', 'B₅'), ('B₃', 'B₆'), ('B₅', 'B₂'), ('B₅', 'B₃'), ('B₆', 'B₃'), ('B₇', 'B₁)}

assigned: {('p₀', 'B₀', 20, 1), ('p₀₀', 'B₀₀', 24, 1)}

unassignedCount: {9}

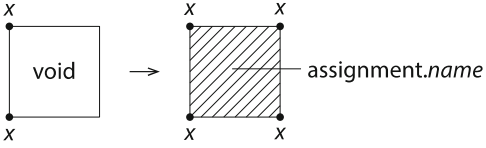
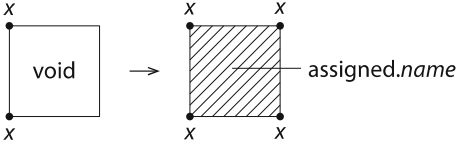
Fig. 2 The initial shape and descriptions for the layout generation problem: (above) the grid of 25 cells, with two predefined, adjacent spaces, denoted 'B0' and 'B00'; (below) the initial descriptions encompassing all information from Table 2

Table 3 The grammar rules addressing the layout generation problem

Rule	Shape component and description components
Ac-1	<p>Activate a grid cell</p>  <p style="text-align: center;">"p".unassignedCount.j</p> <p style="text-align: center;">void →  "AB"</p> <p style="text-align: center;">"p".unassignedCount.j</p> <p>unassignedCount: {j?>0} → {j-1}</p>
As-1	<p>Detect surrounding activities</p>  <p>assigned: { (index?=ptD.x, name?=plD.y, area, count) } → { (index, name, area, count) }</p> <p>surrounding: ∅ → { (assigned.index, assigned.name) }</p>
As-2a	<p>Find an activity with count greater than one that needs to be adjacent to one of the surrounding activities</p> <p>surrounding: { (index, name) } → { (index, name) }</p> <p>activities: { (name, count?>0, area, adj_count) } → { (name, count, area, adj_count) }</p> <p>adjacencies: { (name₁?=surrounding.name, name₂?= activities.name) } → { (name₁, name₂) }</p> <p>candidates: ∅ → { (activities.name, activities.area, activities.adj_count) }</p>
As2-b	<p>Find an arbitrary activity with count greater than one</p> <p>activities: { (name, count?>0, area, adj_count) } → { (name, count, area, adj_count) }</p> <p>candidates: ∅ → { (activities.name, activities.area, activities.adj_count) }</p>
As-3	<p>Assign the activity to the grid cell</p>  <p style="text-align: center;">x x x x</p> <p style="text-align: center;">"AB" →  candidates.name</p> <p style="text-align: center;">x x x x</p> <p>candidates: { (name, area, adj_count) } → ∅</p> <p>assignment: ∅ → { (plD.x, candidates.name, candidates.area - planeSeg2D.area, candidates.adj_count) }</p>

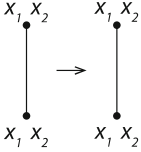
(continued)

Table 3 (continued)

Rule	Shape component and description components
As-4	Update information on satisfied adjacency relations surrounding: $\{(i, name)\} \rightarrow \emptyset$ assigned: $\{(i?=surrounding.i, name?=surrounding.name, area, count)\}$ $\rightarrow \{(i, name, area, adj_cnt-1)\}$ assignment: $\{(i, name, area, adj_cnt)\} \rightarrow \{(i, name, area, adj_cnt-1)\}$ adjacenciesUse: \emptyset $\rightarrow \{(assigned.i, assignment.i), (assignment.i, assigned.i)\}$
As-5	Clear the information collected on surrounding activities surrounding: $\{(index, name)\} \rightarrow \emptyset$
Ex-1	Extend the assigned space if the area constraint is not yet satisfied  assignment: $\{(index?=plD.x, name, area?>0, adj_count)\}$ $\rightarrow \{(index, name, area - planeSeg2D.area, adj_count)\}$
Ex-2	Finalize the assignment assignment: $\{(index, name, area, adj_count)\} \rightarrow \emptyset$ activities: $\{(name?= assignment.name, count, area, adj_count)\}$ $\rightarrow \{(name, count-1, area, adj_count)\}$ assigned: $\emptyset \rightarrow \{(assignment.index, assignment.name,$ $assignment.area, assignment.adj_count)\}$
Fi-1	Assign any unassigned grid cells with the activity of an adjacent cell  assigned: $\{(index?=plD.x, name, area?>0, adj_count)\}$ $\rightarrow \{(index, name, area - planeSeg2D.area, adj_count)\}$
Ch-1	Identify any adjacencies that failed to be satisfied adjacencies: $\{(name_1, name_2)\} \rightarrow \{(name_1, name_2)\}$ assigned: $\{(i_1, name_1?= adjacencies.name_1, area_1, adj_cnt_1, ?>0),$ $(i_2, name_2?= adjacencies.name_2, area_2, adj_cnt_2, ?>0)\}$ $\rightarrow \{(i_1, name_1, area_1, adj_cnt_1), (i_2, name_2, area_2, adj_cnt_2)\}$ adjacenciesNece: $\emptyset \rightarrow \{(assigned.i_1, assigned.i_2)\}$

(continued)

Table 3 (continued)

Rule	Shape component and description components
Ch-2	Check this information against adjacency relations marked as satisfied $\text{adjacenciesUse: } \{(i_r, i_c)\} \rightarrow \{(i_r, i_c)\}$ $\text{adjacenciesNece: } \{(i_r \neq \text{adjacenciesUse}.i_r, i_c \neq \text{adjacenciesUse}.i_c)\} \rightarrow \emptyset$
Ch-3	Check this information against adjacency relations satisfied in the grid  $\text{adjacenciesNece: } \{(i_r \neq \text{ptD}.x_1, i_c \neq \text{ptD}.x_2), (i_r \neq \text{ptD}.x_2, i_c \neq \text{ptD}.x_1)\} \rightarrow \emptyset$ $\text{adjacenciesFound: } \emptyset \rightarrow \{(\text{ptD}.x_1, \text{ptD}.x_2)\}$
Ch-4	Update the information on related activities $\text{adjacenciesFound: } \{(index_r, index_c)\} \rightarrow \emptyset$ $\text{assigned: } \{(index_r \neq \text{adjacenciesFound}.index_r, name_r, area_r, cnt_r),$ $\quad (index_c \neq \text{adjacenciesFound}.index_c, name_c, area_c, cnt_c)\}$ $\rightarrow \{(index_r, name_r, area_r, cnt_r - 1), (index_c, name_c, area_c, cnt_c - 1)\}$

Thick lines represent line segments, thin lines denote a boundary, and a hatched area represents a plane segment. Description parameters are italicized. The predicate ‘void’ indicates the area should be devoid of any geometry

There are 14 rules, collected into five groups, Activation (Ac), Assignment (As), Extension (Ex), Filling (Fi) and Checking (Ch). The Activation group contains a single rule (Ac-1), which randomly selects a grid cell and activates it by marking the grid as well as its four vertices (Fig. 3). As soon as a grid cell is activated, the derivation continues with rules from the Assignment group (Fig. 4). A first rule (As-1) detects an activity surrounding the cell and stores it in the surrounding description. This rule is repeatedly applied until all such activities have been identified (zero, one or more iterations). A second rule (As-2a) is then used to find an activity to be assigned (with count greater than one) that needs to be adjacent to one of these surrounding activities. If none is found, instead, rule As-2b finds any activity to be assigned (with count greater than one). Thus, As-2a and As-2b are alternative rules, with As-2a to be tried first. Then, rule As-3 assigns the activity found to the grid cell, using a transitional assignment description. It also initializes the as yet unsatisfied area as the difference between the activity’s minimum area and the grid cell’s area (identified as *planeSeg2D.area*) If this assignment satisfies any adjacency requirements, this information is updated in rule As-4. As such, rule As-4 is repeated zero, one or more times. As-4 will also clear the ‘surrounding’ description with respect to these adjacent activities. As the last rule in the Assignment group, rule As-5 clears the information collected on other surrounding activities (zero, one or more iterations). This derivation, so far, can be described by the *flow*

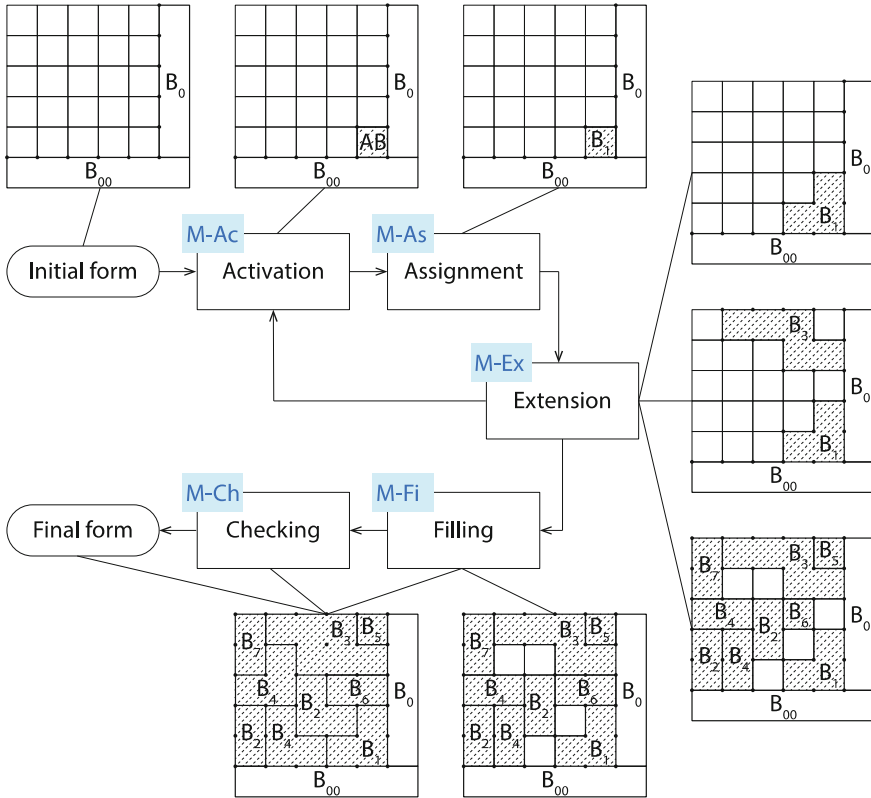


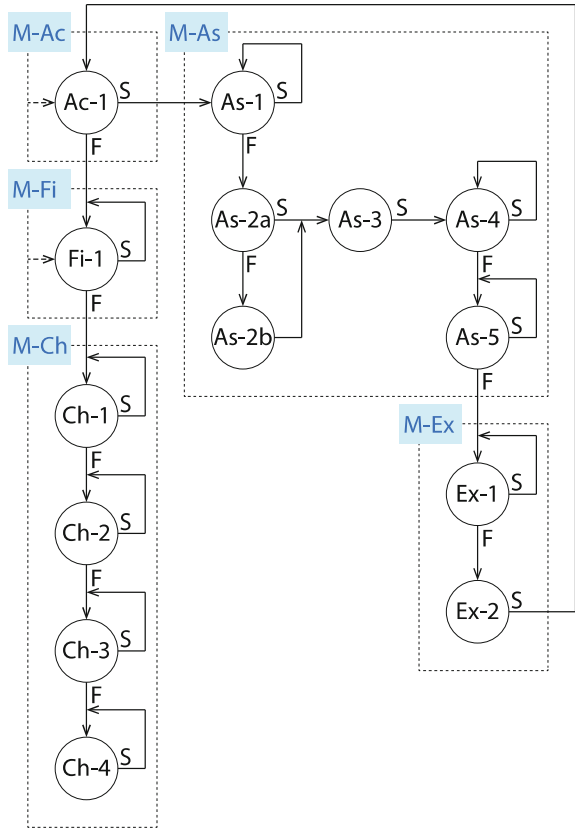
Fig. 3 A flowchart showing the groups of rules and their roles in the *flow*. Results from a derivation are also shown to illustrate the flowchart. Note that labels to points are omitted for the sake of clarity

$$Ac-1 \ As-1*[As-2a \ As-2b] \ As-3 \ As-4* \ As-5*$$

Before a new grid cell can be activated, two tasks remain relating to the current assignment. The first task is to fulfill on the area requirement, specifically, to reduce the area value in the ‘assignment’ description to zero (or negative) as it represents the as yet unsatisfied area. As long as this value is positive, rule Ex-1 of the Extension group finds an adjacent grid cell and assigns the current activity to it, while updating the area value (zero, one or more iterations). The second task is to finalize the assignment, moving the assignment information from the transitional ‘assignment’ description into the resulting ‘assigned’ description, while also updating the activity count (how many times the activity should go through the assignment process). Rule Ex-2 performs this in a single application. At this stage, the process can now be repeated with a new activation, leading to the following *flow*:

$$(Ac-1 \ As-1*[As-2a \ As-2b]As-3 \ As-4* \ As-5* \ Ex-1* \ Ex-2)*$$

Fig. 4 The complete *flow* graphically represented using Liew’s [6] ‘directive’. Note that this visualization omits backtracking



Although all activities should now have been assigned, this does not necessarily result in the grid being completely filled. The Filling group contains a single rule that is very similar to rule Ex-1 and takes a single cell and assigns any adjacent activity to it. At the same time, it updates the area value in the respective ‘assigned’ description. At the very end of the process, the (negative) area value will be a measure of the excessive space that has been assigned to the activity. After rule Ex-1 has been repeatedly applied until all grid cells have been assigned an activity, the derivation continues by updating all adjacency relationship information. It is entirely possible that due to the application of rule Ex-1, additional adjacency constraints have been satisfied. It is important to capture this information as it serves to evaluate the success of the particular solution resulting from this single derivation. The Checking group of rules ensures the correctness of this information. The first step (rule Ch-1) identifies which adjacency pairs failed to be satisfied, by checking the adjacency count value in the ‘assigned’ descriptions. However, it is possible that some adjacency count values are still positive even though these adjacency constraints are satisfied. Therefore, rule Ch-2 removes any ‘failed’ adjacencies if these have already been captured in

the ‘adjacenciesUse’ description. Subsequently, rule Ch-3 checks if an adjacency constraint that is still marked as unsatisfied is fulfilled in the drawing. Finally, the information of related activities is updated by rule Ch-4. Each of rules Ch-1, Ch-2, Ch-3 and Ch-4 iterates zero, one or more times. The final *flow* becomes:

$$(Ac-1 As-1*[As-2a As-2b]As-3 As-4* As-5* Ex-1* Ex-2)*$$

$$Fi-1* Ch-1* Ch-2* Ch-3* Ch-4*$$

Discussion and Conclusion

We have presented a concept of composite shape rules embedding algorithmic patterns for rule automation. We have denoted these composite shape rules *flows*, and adopted a notation from regular expressions. More than an algorithmic notation for automated rule selection, we consider *flows* as an alternative approach to rule specification. Next to rules of the form $r: lhs \rightarrow rhs$, we consider rules of the form $f: [r_1 (r_2 r_3)]$, allowing rules to be composed (or decomposed) at will.

The notation we adapt from regular expressions. We have adopted most of the operations to construct regular expressions as operations to construct *flows*, including quantification, grouping and the square bracket expression for selection. We have omitted a few metacharacters that are rather irrelevant to *flows*, such as the starting and ending position of the string, or matching any single character. We have also omitted the choice operator, as we have opted for the square bracket expression to be used instead. At the same time, we have extended the notation with an asterisk preceding a square bracket extension, allowing for a random selection of a single (composite) rule from the collection of (composite) rules identified as alternatives within the square bracket expression.

However, we have only briefly discussed the distinction between greedy and lazy matching and specified that *flows* apply greedy matching by default. In fact, regular expressions may also allow for a possessive matching, which is a form of greedy matching, but disallows backtracking. That is, in the example of matching the regular expression ‘a*.b’ to the string ‘aaab’, a possessive matching would fail as the algorithm will match the token ‘a’ three times, followed by the token ‘.’ (matching the letter ‘b’), and being unable to backtrack. Similarly, in rule application, sometimes it is appropriate to disable backtracking, locally. That is, a sub-*flow* may disable backtracking, such that when a rule application failure occurs in a sequence within the sub-*flow*, no backtracking occurs within the sub-*flow*, though the algorithm may backtrack to a point before the sub-*flow* and try alternatives from there on.

In addition, next to greedy, lazy and possessive matching, we also consider a probabilistic matching in an iteration. Let us assume that we allow a (composite) rule to be applied any number of times between some minimum and maximum number of times. Rather than trying to iterate the rule as many times as possible and allowed (greedy matching) or iterating the (composite) rule as few times as

allowed (lazy matching), the algorithm could randomly select any value between the minimum and maximum values as the specific number of times to try to iterate the (composite) rule. To clarify, in the example of the width of the grid for Palladio's villa plans, a greedy algorithm would only search for grids that are five columns wide, a lazy algorithm for grids that are three columns wide, while a probabilistic matching algorithm would randomly choose between a five-column wide grid and a three-column wide grid. Where the question mark is used to specify lazy matching and the plus sign is used to specify possessive matching in regular expressions and in *flows*, the asterisk is selected to specify probabilistic matching. This is in line with the choice for the asterisk as the preceding symbol to identify a random re-ordering within a selection.

Even though there are important differences between regular expressions and composite shape rules, most notably the fact that regular expressions work towards a target, while composite shape rules are intended to be used exploratory or enumerative, we must conclude that the notation for regular expressions fits composite shape rules remarkably well. The notation is also very compact, simplifying the specification of a *flow*. Although Liew's [6] graphical approach may seem much more readable, such visual readability would be entirely lost if we attempted to explicate the ability for backtracking.

The above case study demonstrates the applicability of shape grammars—using *flows*—to solve a design problem in an algorithm way, organizing rules into sequences, selections and iterations. While doing so may not be the primary value of a shape grammar, we argue it is rather unavoidable when designing with rules. Nevertheless, the case study only demonstrates the outcome of the *flow* development process and not the possible interaction of the user with the process. Obviously, the process of developing such a flow is not straightforward and involves trial and error. We envision to assist the designer in this process by providing him or her with a toolbox of simple patterns of rules or *flows*. For example, rule Ch-1 presents an example of requirement checking, iterating over the list of required adjacencies and creating a new description for every failed adjacency requirement. The number of descriptions created in this way serves as a measure for success or failure. In particular cases, this search process may need to be reversed and extended, and involve making a copy of existing descriptions before removing any description that actually meets the requirement. In the extended case study [12], we explore different deductive strategies as well, resulting in different *flows* (and a few different rules).

Acknowledgements This work received some funding support from Singapore MOE's AcRF start-up grant, WBS R-295-000-129-133. The second author also benefited from a China Scholarship Council grant. We want to thank Bui Do Phuong Tung and Bianchi Dy for their work on the SortalGI shape grammar interpreter and API.

References

1. Stiny G (1980) Introduction to shape and shape grammars. *Environ Plann B Plann Des* 7(3):43–351
2. Yue K, Krishnamurti R (2013) Tractable shape grammars. *Environ Plann B Plann Des* 40(4):576–594
3. Stiny G (2006) *Shape: talking about seeing and doing*. MIT, Cambridge, MA
4. Beirão JN, Duarte JP, Stouffs R (2009) Grammars of designs and grammars for designing—grammar based patterns for urban design. In: Tidafi T, Dorta T (eds) *Joining languages, cultures and visions*. Université de Montréal, Montreal
5. Knight TW (1999) Shape grammars: six types. *Environ Plann B Plann Des* 26(1):15–31
6. Liew H (2004) *SGML: a meta-language for shape grammar*. PhD thesis. MIT, Cambridge, MA
7. Grasl T, Economou A (2014) Towards controlled grammars. In: Thompson EM (ed) *Fusion*, 2nd vol. eCAADe, Brussels, pp 357–363
8. Stiny G, Mitchell WJ (1978) The Palladian grammar. *Environ Plann B Plann Des* 5(1):5–18
9. Palladio A (1965) *The four books of architecture*. Dover, New York (Reprinted from the 1738 translation by Isaac Ware of *I Quattro Libri dell' Architettura*)
10. Stiny G, Mitchell WJ (1978) Counting Palladian plans. *Environ Plann B Plann Des* 5(2):189–198
11. Stouffs R (2016) Description grammars: a general notation. *Environ Plann B Urban Anal Smart Cities* 45(1):106–123
12. Hou D, Stouffs R (2018) An algorithmic design grammar for problem solving. *Autom Constr* 94:417–437

Shape Grammars as a Probabilistic Model for Building Type Definition and Computation of Possible Instances: The Case Study of Ancient Greek and Roman Libraries



Myrsini Mamoli

This paper discusses a shape grammar for the reconstruction of archaeological building remains of ancient Greek and Roman libraries with metadata pointing to the evidence on which each rule is based. A frequency analysis graph analyzes the frequency with which each rule occurs in the grammatical computations of the known cases in the corpus and provides a probabilistic model for the definition of the building type, i.e., the mandatory characteristics, the optional, and the probability with which they appear.

Introduction

Uncertainty, multiple readings and variation are inherent in the interpretation and reconstruction of fragmentary archaeological remains. Generative grammars have been used in archaeology as a systematic tool of classification and reconstruction [1, 2]. More recently, shape grammars [3, 4] have been used as a computational methodology that encodes the design principles of a set of designs in visual rules, the recursive application of which is able to generate designs within the same language [5–8].

The contribution of this work is that it proposes the use of shape grammars for the analysis and definition of a whole building type. With the use of metadata next to each rule that references the instance in the corpus with available evidence in the archaeological record, it provides a probabilistic model that defines the type as a sum of mandatory and optional building components and also presents the probability with

M. Mamoli (✉)
Louisiana State University, Baton Rouge, USA
e-mail: mamoli.myrsini@gmail.com

M. Mamoli
Athens, Greece

which optional components are likely to appear. The more evidence a rule is based on, the more likely it is that this rule is important in the definition of the building type as a whole and the more likely it is that this rule will apply in the reconstruction of an instance in the type.

The Case Study of Ancient Libraries

This work is part of a larger project that looks into the architectural form of ancient Greek and Roman libraries, a building type that is not as clearly understood and defined as other building types such as the temple or the stoa due to the diversity and variability of scale, urban context, monumentality, and the use of component elements with which it appeared.

The scholarship on ancient libraries started in the end of the nineteenth century and the beginning of the twentieth century with the first discoveries of libraries in Pergamon, Ephesus, and Timgad. Since then, 17 buildings in the archaeological record have been identified with ancient libraries based on reference to the epigraphic and literary sources, which are at the core of the study of the building type of the ancient library.

Despite the variation, the underlying design characteristic of the type is the existence of a formal room, an *oikos*, in association with a stoa, a semiopen space as a threshold to the *oikos*. This threshold could be as elaborate as a whole building complex including additional support rooms for the functions of the library, *exedras*, a monumental entry and an interior courtyard with landscaped gardens, or as small as being shared as part of a building complex with different functions, for example, a forum. The *oikos*, functioning as the main hall of the library, was a formal room with statues, furnishings, and even an elaborate articulation of the interior space with niches, podium, interior columnar screens, and a focal point. Generally, the increase in scale and monumentality progressed with time, i.e., simpler forms appeared early on in the Hellenistic period, while more elaborate forms appeared in the Roman High Imperial period. However, both simple and elaborate forms appeared throughout the Roman period, so that efforts of classification of libraries, for example, Greek versus Roman libraries [9], imperial libraries in Rome versus provincial libraries [10, 11], wide or long rectangle libraries in the Roman Empire [9, 12], are undermined by the number of exceptions to the rule that they try to establish. An additional layer of difficulty in defining a building type for libraries is that characteristics attributed to libraries, for example, the niches and the podium can also be found in other buildings, such as stoas or temples [11]. All these make researchers very hesitant in identifying a building as a library and even when the identification is secure, they do not know how to reconstruct it.

A first effort to order and formalize the design of Roman libraries was by Makowiecka [12], who suggested that Roman libraries could fit into eight schemata. However, these schemata do not account for all Roman libraries, they mislead us to

believe that components that appear in one schema cannot be identified in another and limit our ability to understand the diversity in their design, and the variation with which a library can be reconstructed.

To overcome the conundrum of defining a concrete theory about the building type of the ancient library, a parametric shape grammar (Figs. 1, 2, and 3) was developed that summarizes and encodes in design rules the design principles that occur in the 17 cases of known ancient libraries [13, 14]. The design principles are known following traditional methods of archaeological research through bibliographical review of measurements and state of preservation drawings, primarily plans but also sections and elevations, where the archaeological remains are at a sufficient height. This evidence has been compiled in a database that informs the parameters of the rules with the valid range of values.

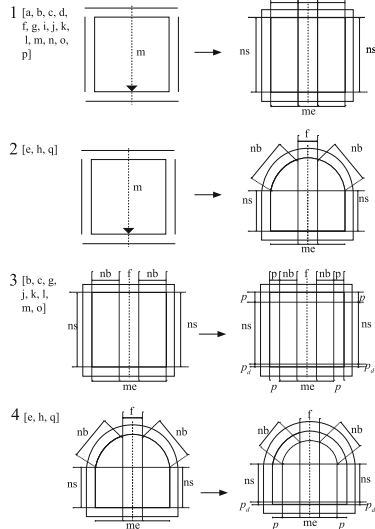
The grammar is designed in two dimensions in plan and consists of 91 design rules, subdivided in stages that roughly correspond to the generation process of a library: the layout of the main hall with its interior design: the podium, the niches, the focal point, the interior colonnade, and the entry (Fig. 1); and the general layout of the building or the building complex with the side rooms, the stoas with the exedras, the entrance, and the courtyard, if any (Figs. 2 and 3).

The grammar is able to generate libraries within the language that it defines, known and hypothetical: it can generate all 17 known libraries and also variations of these that shed light into multiple possibilities for their reconstruction and also the design intent of the patron and the architect. To demonstrate how the grammar works, the computation of a well-preserved library, the Library of Hadrian in Athens is given in Fig. 4. Starting with the underlying state of preservation plan of the library given in gray color, and the initial shape of the grammar, one applies the appropriate rules that can be embedded in the remains and generates the reconstructed plan of the library.

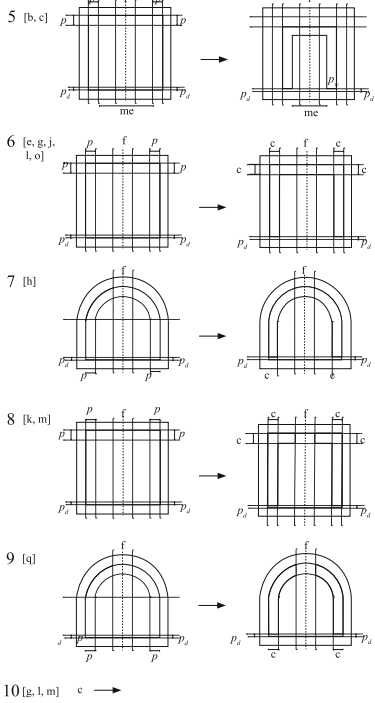
Also, the grammar can be used as a tool to evaluate whether a reconstruction of a building can be generated and therefore whether this building could be a library or not. Finally, the grammar can be used as a tool for the prediction of hypothetical but possible libraries that might be excavated one day.

These possibilities showcase a great variety; with 91 rules, some of which are mandatory, and some of which can apply more than once, the possibilities are calculated into hundreds. The question is how can we know which possible libraries are more important in the building type of the library and therefore more likely to capture the original design in a reconstruction? When we evaluate a building as a possible library, how can we know that it is more or less likely that it was a library or not? To answer these questions and to have guidance in defining a building type that showcases such variability, a systematic approach to evaluating the designs is needed. This work proposes the connection of the rules of the grammar with the archaeological record through the use of metadata, the analysis of which can provide a probabilistic model for the definition of building type.

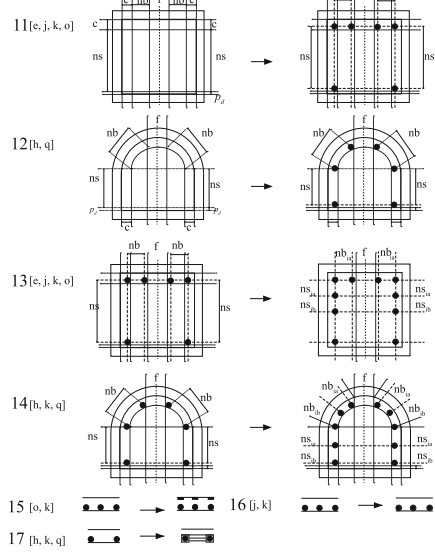
STAGE I Main Hall Layout



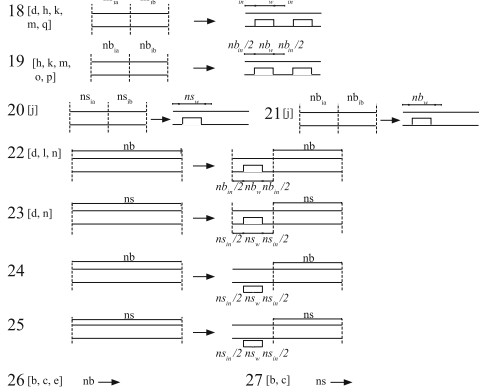
STAGE II Podium



STAGE III Interior Colonnade



STAGE IV Niches



STAGE V Focal Point

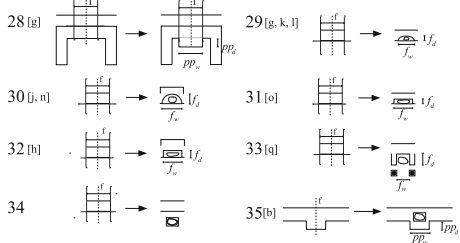
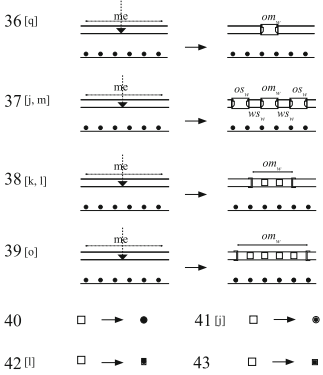
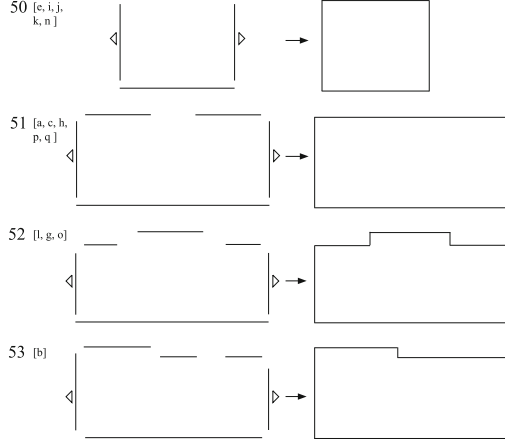


Fig. 1 Stages 1–5 that generate the main hall of the library

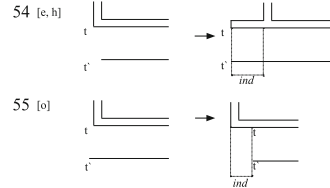
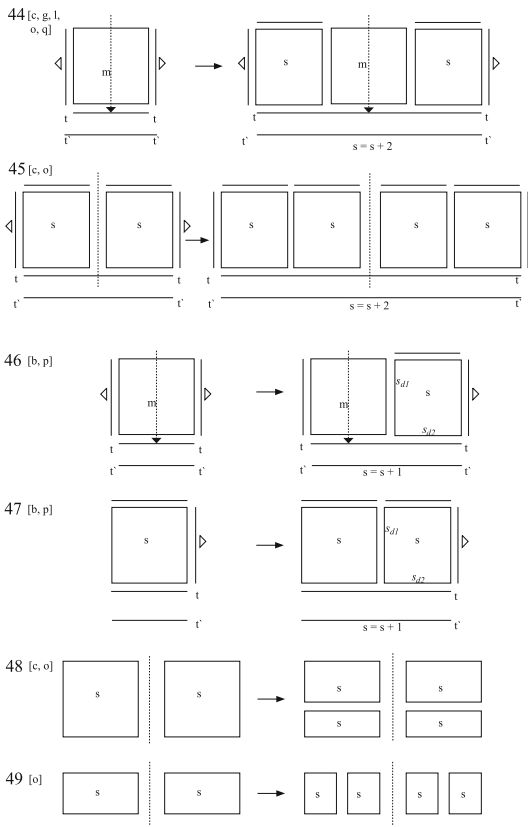
STAGE VI Entrance openings



STAGE VIII Exterior Walls



Stage VII Main Hall and Side Rooms



STAGE IX Thresholds, stoas and courtyards

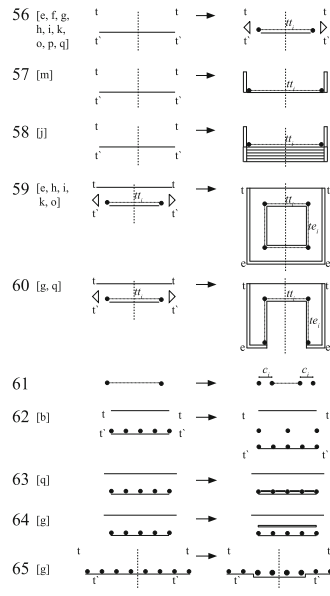
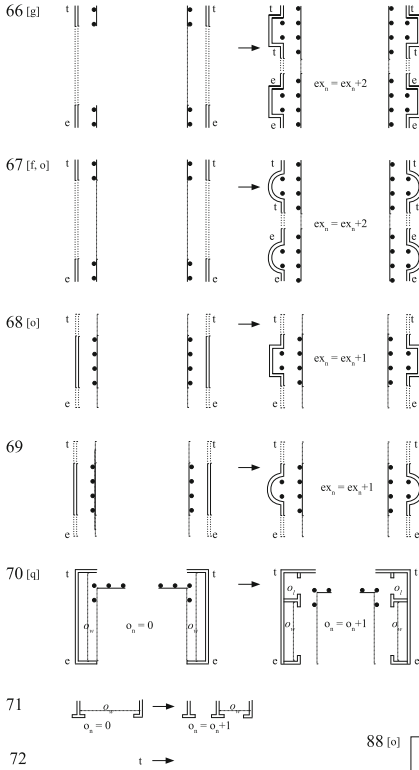
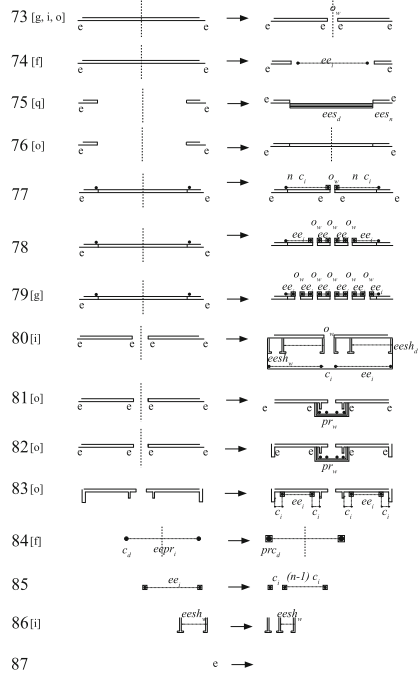


Fig. 2 Stages 6–9 that generate the entry of the main hall and the general layout of the library

STAGE X Exedras



STAGE XI Entry to complex



STAGE XII Functional characteristics of support spaces

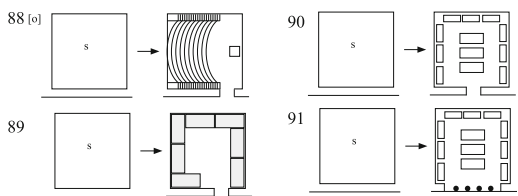


Fig. 3 Stages 10–12 that generate the exedras and the entry of the library complex and the interior design of the side rooms

Frequency Analysis for the Building Type Definition of the Library

Each rule in the grammar includes metadata, letters that point to the instances in the archaeological record that show evidence for it, as documented in the database. We make the assumption that the more instances a rule is based on, the more important that rule is in the definition of the building type. Also, the more important a rule is, the more likely it is that it will apply to many cases. Reversing this argument, in reconstructing the known cases in the corpus, and taking the strings of the rules that

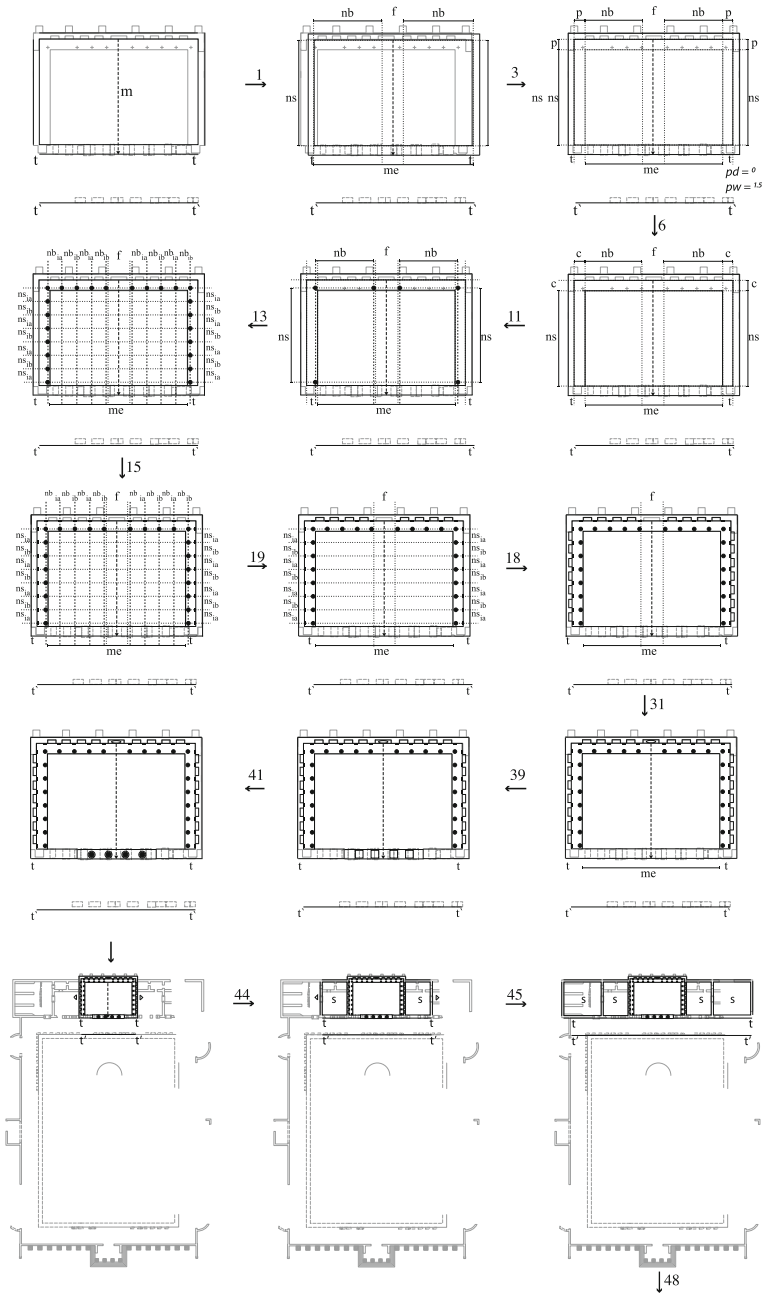


Fig. 4 Computation of the Library of Hadrian in Athens: the main hall of the library and the general layout of the library

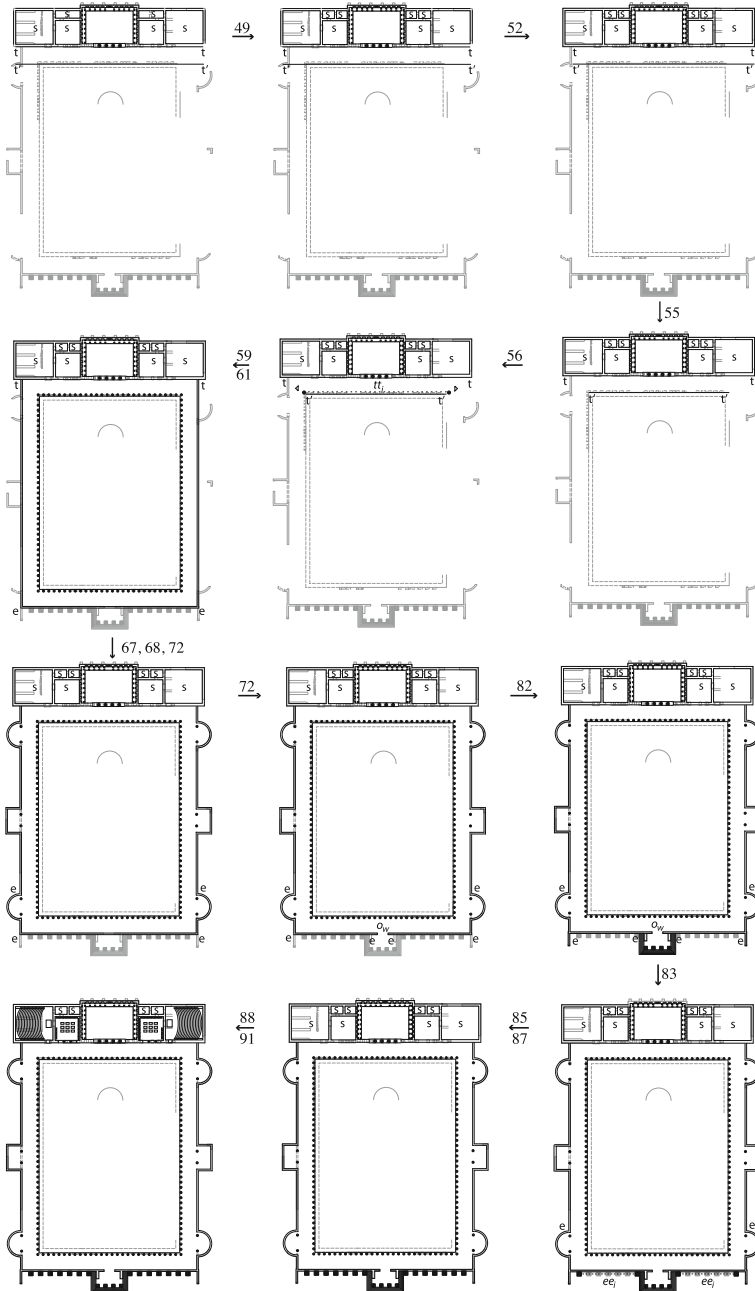


Fig. 4 (continued)

Table 1 The strings of rules used in the derivation of each case in the corpus of known libraries

	Known libraries	Strings of rules used in their derivations
a.	Library at the Serapeum, Alexandria, 300–250 BCE	1, 24, 24, 25, 25, 34, 38, 41, 46, 47, 51, 54, 46, 59, 61, 90
b.	Library of Pergamon, 200–175 BCE	1, 3, 5, 26, 27, 28, 35, 38, 41, 46, 47, 47, 53, 54, 56, 61, 61, 62, 90
c.	Academy of Plato, Athens, Hellenistic Period	1, 3, 5!, 26, 27, 28, 35, 38*, 41, 44, 45, 48, 51, 56, 59!, 61, 72, 73, 81*, 87, 90*
d.	Library at the Gymnasium of Rhodes, Hellenistic Period	1, 22, 22, 23, 23, 29, 38, 41, 50, 57, 61
e.	Augustan Palatine Library, Rome, 28 BCE	2!, 4!, 6!, 11, 13, 16*, 18*, 26, 34, 38*, 41*, 50, 54, 56, 59, 61
f.	Library in the Portico of Octavia, Rome, 23 BCE	1*, 3*, 8*, 11*, 13*, 16*, 18*, 19*, 33*, 39*, 44*, 45*, 52*, 56, 59, 61, 62, 67, 68, 72, 74, 84, 87, 89, 90*
g.	Library at the Templum Pacis, Rome, 75 CE	1, 3, 6, 10, 24*, 25*, 28, 30, 39, 43, 44, 45, 52, 56, 60, 61, 64, 65, 66, 72, 76, 79, 85, 87, 90*
h.	Domitianic Palatine Library, Rome, 80 CE	2, 4, 7, 12, 14, 17, 18, 19, 32, 39*, 41*, 50!, 54, 56, 59, 61
i.	Pantainos Library, Athens, 102 CE	1, 24*, 25*, 34, 38*, 50*, 54, 56, 59, 61, 73!, 80!, 86, 85
j.	Celsus Library, Ephesus, after 117 CE	1, 3, 6, 11, 13, 16, 20, 21, 30, 37, 52, 54, 58, 61, 61, 61
k.	Ulpian Library, Rome, 114–128 CE	1, 3, 8, 11, 13, 15, 16, 17, 18, 19, 33, 38, 41*, 50, 56, 61
l.	Neon Library, Sagalassos, after 120 CE	1, 3, 6, 10, 22, 23*, 29, 37, 50*, 57, 61
m.	Library of Nysa, 100–200 CE	1, 3, 8, 10, 18, 19, 32, 37, 52, 54, 57, 61
n.	Melitone Library, Pergamon, after 123 CE	1, 22, 23, 30, 37!, 52
o.	Hadrian's Library, Athens, 131 CE	1, 3, 6, 11, 13, 15, 19, 18, 31, 39, 41, 44, 45, 48, 49, 52, 55, 56, 59, 61, 67, 68, 72, 72, 82, 83, 85, 87, 88, 91
p.	Library in the Forum of Philippi, 100–200 CE	1, 24*, 25*, 34*, 38!, 41, 50, 54, 56, 61, 62
q.	Rogatinus Library, Timgad, 150–200 CE	2, 4, 9, 12, 14, 15, 17, 18, 19, 33, 36, 44, 52, 60, 61, 63, 70, 72, 75, 90

apply in each of them, one can have quantitative data about the architectural form of ancient libraries and draw conclusions about the type.

The corpus of known libraries consists of 17 cases. The strings of rules applied to generate the derivations of the 17 libraries are given in Table 1. When a rule is denoted with a “!” it means that the rule is approximately embedded in existing building remains, and when a rule is denoted with a “*” it means that the rule is applied in a conjectural manner, due to the lack of evidence in the building remains.

These strings of rules are analyzed in a frequency analysis graph to draw conclusions about the occurrence of the individual architectural components as part of the building type of the library, and the forms that these components take as generated by different rules within each stage. Only rules that are based on building remains are considered in this analysis, and thus the rules accompanied by an asterisk are omitted. The frequency analysis reflects both the occurrence of the rules in the derivations and the metadata of the rules in the grammar.

First, the frequency analysis of the stages that appear in the 17 derivations gives the mandatory and the optional architectural features of a library that correspond to the different stages in the grammar (Fig. 5), and second, the frequency analysis within each stage of rules shows the most probable forms that each architectural feature might have (Fig. 6).

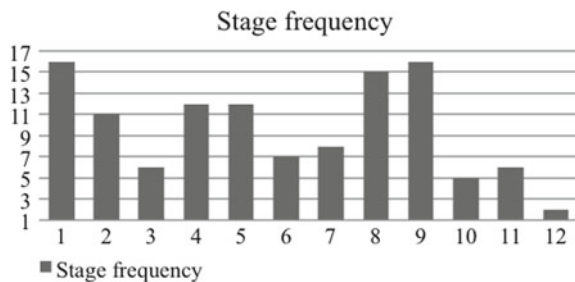
The most obvious result of the frequency analysis is that the most frequent stages are the stages that generate the main hall (stage 1, 16 occurrences) and the threshold (stage 9, 16 occurrences). The first stage occurs in all libraries except for the Library in the Porticus Octavia, for which there is no evidence, and the stage for the threshold occurs in all libraries except for the Melitine Library, which was a later addition to a preexisting complex, and thus, its design was restrained by the preexisting conditions. This shows the importance of the threshold, the stoa, or the peristyle, in the architectural form of a library. The second most frequent stage is the stage that generates the exterior walls of the library (stage 8, 15 occurrences).

Second, the frequency analysis shows that the stages that occur more often are the stages that generate the niches (stage 4, 12 occurrences) and the focal point (stage 5, 12 occurrences). This shows that the two architectural components most probable to be found in a library are the niches and the focal point.

Third, the frequency analysis shows that the next more frequent architectural component in a library is the podium (stage 2, 11 occurrences) and that the least frequent component of interior design of the main hall is the interior colonnade, which occurs almost half of the times (stage 3, 6 occurrences) that a podium occurs.

Moreover, the frequency analysis shows that the library includes side rooms in less than half of the times (stage 7, 8 occurrences). If we add to this number the cases of the Ulpian Library and the Domitianic Library in Rome that might have had identical duplicate halls, then the library consists of more than one room in

Fig. 5 Histogram showing the occurrence of stages in the derivations of the known libraries



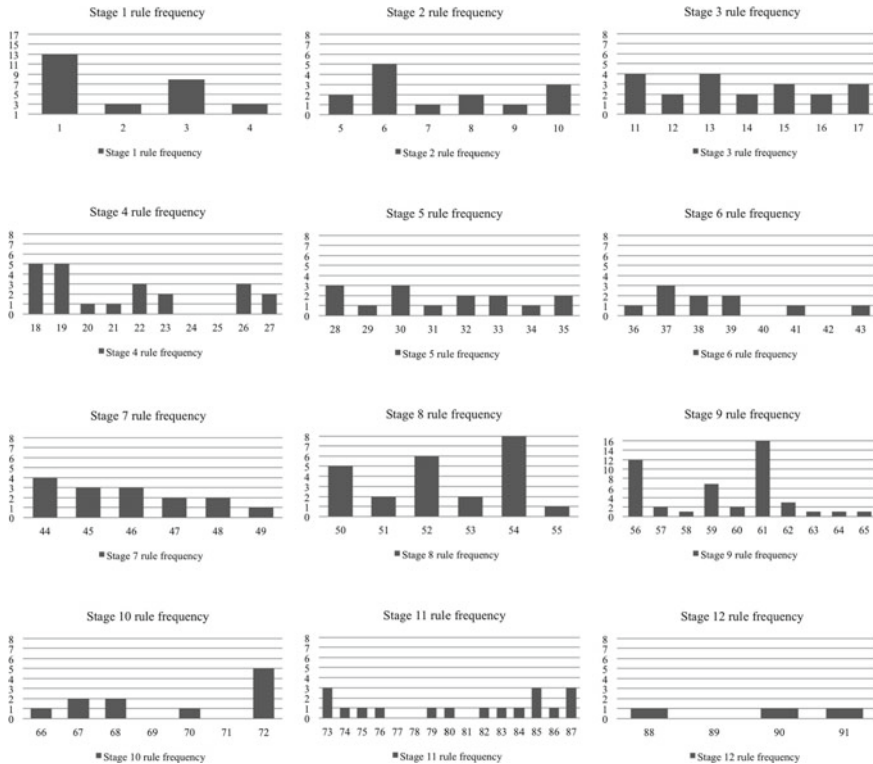


Fig. 6 Histograms showing the occurrence of rules in each stage in the derivations of the known libraries

only 10 cases. The least frequent features in a library are the exedras (stage 10, 5 occurrences) and the monumental propylon (stage 11, 6 occurrences). These facts emphasize the diversity of scale that occurs in the corpus of known libraries. Finally, the very limited occurrences in stages 6 and 12 show the limited building remains that keep conjectural the reconstruction of the entries and the interior of the side rooms.

More specific conclusions can be drawn by looking at the frequency of rules within each stage separately, as shown in Fig. 6.

The main hall of the library is more probably rectangular rather than apsidal (rules 1 vs. 2). If there is a podium, it is most likely that the podium is continuous (rule 6), along the three walls of the room, and that on it is set a colonnade (rule 10). If there is a colonnade, it is equally likely the columns to be supported on pedestals among which there are steps (rule 15) and to be directly placed on the podium (rule 17), which remains unmodified. Also, a library is most likely to have niches on the walls, which take advantage of the whole wall length (rules 18 and 19), and a focal point. There is a great variety in the type of the focal point, and all types appear equally

frequently, thus making the focal point the most flexible characteristic of the library. Finally, the main hall of the library is most likely to have a wide entrance with more than one opening (rules 37 and 39), which is equally likely to be articulated with openings among columns (rules 38 and 39) and door openings in the wall (rules 36 and 37).

In terms of the general layout of the library, it is most likely that the library constitutes a symmetric complex (rules 44 and 45). This is also reflected in the frequency of the rules that generate the exterior walls (rules 50 and 52). The frequency of rule 54 shows that the threshold of the library very often extends beyond the boundaries of the library, and this accounts for cases that the library is part of a larger complex, with or without extra rooms.

In stage 9, the frequency of rule 61 that generates columns reflects the importance of the threshold as expressed by the frequency of rule 9 as well. Rules 62–65 occur only sporadically, and this reflects that they are derived from stylistic characteristics of the colonnades, rather than from typological characteristics inherent to the building type of the library. The frequency of rule 59 emphasizes the importance of a peristyle, which occurs as many times as a single stoa and a U-shaped stoa combined (rules 57, 58, and 60). Regarding exedras and auxiliary oikoi along the stoas, semicircular exedras appear almost with the same frequency as rectangular. The rule that occurs mostly is the rule that erases the label *t* in order to stop the computation of exedras, which shows the limited occurrence of exedras in the corpus.

In relationship to entrances, the distribution of occurrences shows that there is no consistent way of making a monumental entry to the complex, and that most rules are derived by stylistic characteristics of building remains. Finally, the limited occurrences in stage 12 reflect the limited building remains that keep most reconstructions of side rooms conjectural.

All these conclusions are summarized in Table 2, where the different stages of the grammar are orders with the frequency they appear, from the mandatory to the optional components, and rules within each stage according to the frequency they appear in the corpus of known libraries and the probability they have in the generation of hypothetical libraries.

Finally, conclusions about the building type of the library, the more paradigmatic, and the more exceptional can be drawn by comparing the rules used for the derivation of each library to the frequency with which they appear in the whole corpus (Table 3). A set of libraries (j, k, l, m, o) generated by rules with higher frequencies (13, 8, 4, 5) can be identified. These libraries constitute libraries built in Hadrian's period, in Athens, Rome, and Asia Minor, and are the Celsus Library, the Ulpian Library, the Library of Nysa, the Neon Library, and Hadrian's Library in Athens. These libraries differ to each other in urban context, and scale (with and without peristyle), but have common underlying characteristic the well-articulated interior with niches and focal point. Therefore, we can identify a temporal aspect in the high frequency of the

Table 2 Stages ordered from the mandatory to the least frequent components and rules ordered from the highest to the lowest probability

Stage according to frequency	Rules according to frequency
1. Main Hall	(1, 3)>(2, 4)
9. Threshold	61>56>59>62>60>57>(58, 63, 64, 65)
8. Exterior Walls	54>52>50>(51, 53)>55
4. Niches	(18, 19)>(22, 26)>(23, 27)>(20, 21)
5. Focal Point	(28, 30)>(32, 33, 35)>(29, 31, 34)
2. Podium	6>10>(5, 8)>(7, 9)
3. Interior Colonnade	(15, 17)>(11, 13)>(12, 14, 16)
7. Side Rooms	44>(45, 46)>(47, 48)>49
10. Exedras	72>(67, 68)>(66, 70)
11. Entry to Complex	(73, 85, 87)>(74, 75, 76, 79, 80, 82, 83, 84, 86)
6. Entry to Main Hall	37>(38, 39)>(36, 41, 43)
12. Side Room Interior Design	(88, 90, 91)

particular rules that generate the well-articulated interior: they point to the second century CE. This conclusion can be used in the interpretation of other hypothetical libraries too and conclude that libraries in which these rules apply could have been built in the second century CE.

Finally, the table shows that Hadrian's Library (o) and the Templum Pacis (g) use many rules that appear only in these libraries, which emphasizes that these libraries are exceptional in scale and monumentality.

Computing Variation of Probable Hypothetical (and/or Unexcavated) Libraries

In computing ancient libraries, in theory, the number of possibilities is equal to the product of the different possibilities for each architectural element and is calculated into thousands. This probabilistic model for the definition of the building type of the ancient library makes us aware of the various degrees of probability with which variant plans of ancient libraries of variant scale, monumentality, and the use of component elements can be computed. On the one hand, it helps us look at the type as a whole without excluding components and forms less likely to appear, and on the other hand, it helps us focus on forms that are more likely to iterate.

Table 3 The rules used in the derivations of the known libraries (a–q) with their frequency in the whole corpus given in square brackets

a	1[13] 24[0]* 25[0]* 34[1]* 90[1]* 61[16] 59[7] 56[12] 54[8] 51[2] 47[2] 46[3] 41[1]* 38[2]*
b	1[13] 3[8] 5[2] 26[3] 27[2] 28[3] 35[2] 38[2]* 41[1]* 46[3] 47[2] 53[2] 54[8] 56[12] 61[16] 62[3]
c	1[13] 3[8] 5[2] 26[3] 27[2] 28[3] 35[2] 38[2]* 41[1]* 44[4] 45[3] 48[2] 51[2] 56[12] 59[7] 61[16] 72[5] 73[3] 81[0]* 90[1]*
d	1[13] 18[5] 22[3] 23[2] 29[1]* 38[2]* 41[1]* 50[5] 57[2]* 61[16]
e	2[3] 4[3] 6[5] 11[4] 13[4] 16[2]* 18[5]* 26[3] 34[1] 38[2]* 41[1]* 50[5] 54[8] 56[12] 59[7] 61[16]
f	1[13]* 3[8]* 8[2]* 11[4]* 13[4]* 16[2]* 18[5]* 19[5]* 33[2]* 39[2]* 44[4]* 45[3]* 50[5]* 52[6]* 54[8]* 56[12] 59[7] 61[16] 62[3] 67[2] 68[2] 72[5] 74[1] 84[1] 87[3] 89[0]* 90[1]*
g	1[13] 3[8] 6[5] 10[3] 23[2]* 24[0]* 25[0]* 26[0]* 28[3] 30[3] 39[2] 43[1] 44[4] 45[3] 52[6] 56[12] 60[2] 61[16] 64[1] 65[1] 66[1] 72[5] 76[1] 79[1] 85[3] 87[3] 90[1]*
h	2[3] 4[3] 7[1] 12[2] 14[2] 17[3] 18[5] 19[5] 32[2] 39[2]* 41[1]* 50[5] 54[8] 56[12] 59[7] 61[16]
i	1[13] 24[0]* 25[0]* 34[1]* 38[2]* 46[3] 53[2] 54[8] 56[12] 59[7] 61[16] 73[3] 80[1] 85[3] 86[1] 90[1]*
j	1[13] 3[8] 6[5] 11[4] 13[4] 16[2] 20[1] 21[1] 30[3] 37[3] 52[6] 54[8] 58[1] 61[16]
k	1[13] 3[8] 11[4] 13[4] 15[3] 16[2] 17[3] 18[5] 19[5] 33[2] 38[2] 50[5] 56[12] 61[16]
l	1[13] 3[8] 6[5] 10[3] 22[3] 23[2]* 29[1] 37[3]* 44[4]* 50[5]* 52[6]* 57[2] 61[16] 90[1]*
m	1[13] 3[8] 8[2] 10[3] 18[5] 19[5] 32[2] 37[3] 52[6] 54[8] 57[2] 61[16]
n	1[13] 22[3] 23[2] 30[3] 37[3] 50[5]* 52[6]
o	1[13] 3[8] 6[5] 11[4] 13[4] 15[3] 18[5]* 19[5] 31[1] 39[2] 41[1]* 44[4] 45[3] 48[2] 49[1] 52[6] 55[1] 56[12] 59[7] 61[16] 67[2] 68[2] 72[5] 73[3] 82[1] 83[1] 85[3] 87[3] 88[1] 91[1]
p	1[13] 24[0]* 25[0]* 34[1]* 38[2] 41[1] 46[3]* 47[2]* 50[5] 53[2]* 54[8] 56[12] 61[16] 62[3] 90[1]*
q	2[3] 4[3] 9[1] 12[2] 14[2] 15[3] 17[3] 18[5] 19[5] 33[2] 36[1] 44[4] 52[6] 56[12] 60[2] 61[16] 63[1] 70[1] 72[5] 75[1] 90[1]

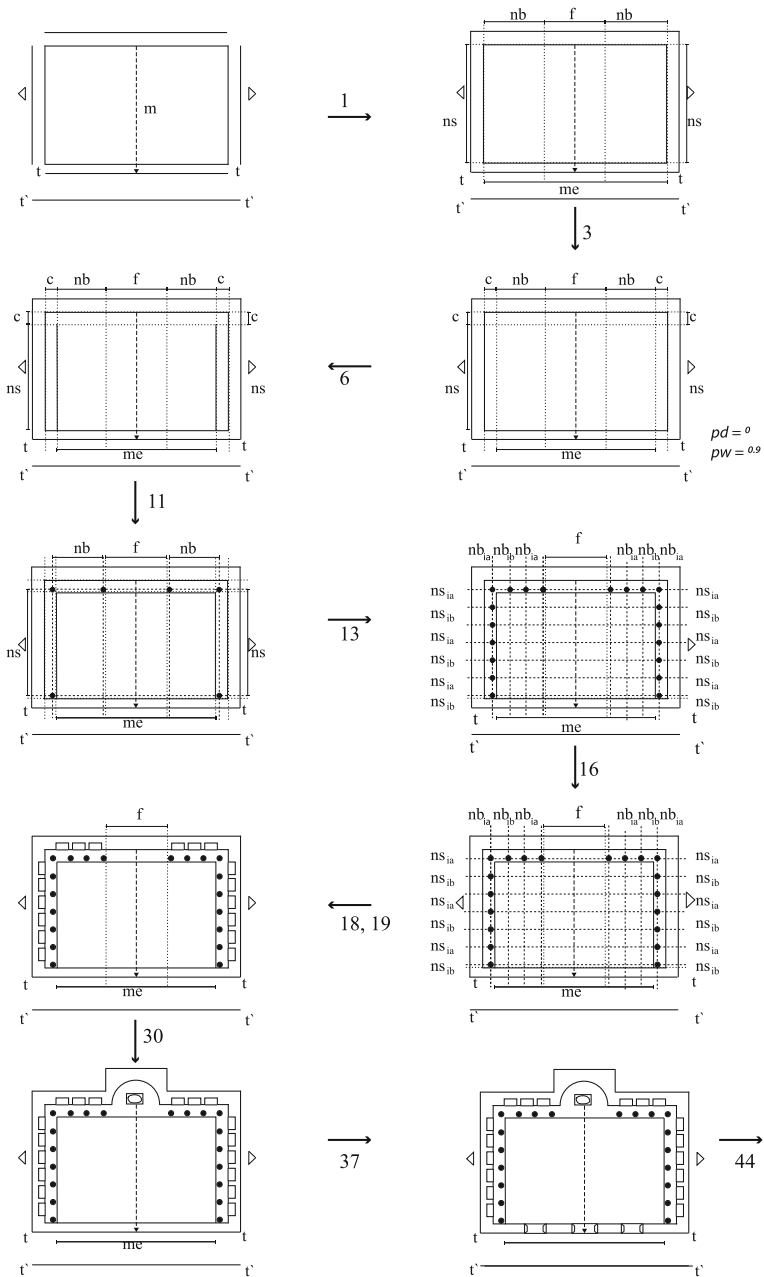


Fig. 7 Derivation of a hypothetical library with main hall, two side rooms, and peristyle with courtyard and propylon

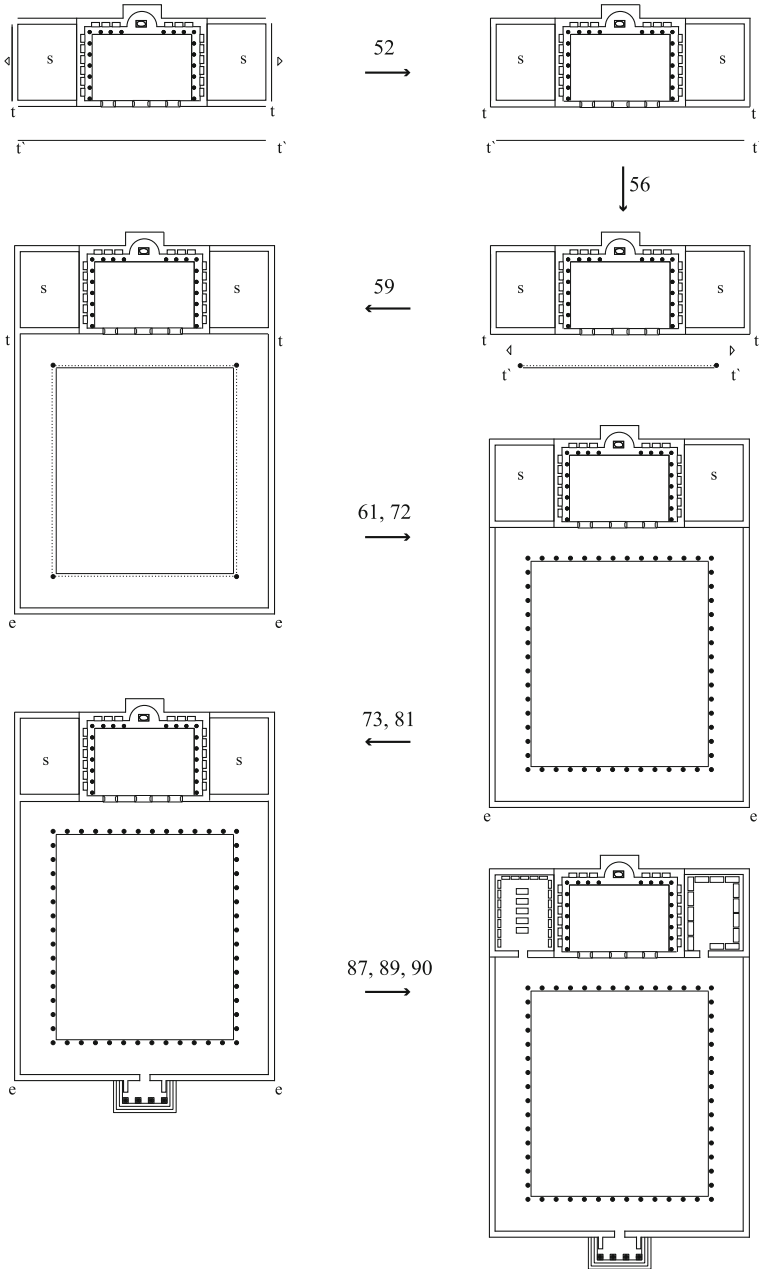


Fig. 7 (continued)

Using the frequency analysis as a guidance, and starting with the initial shape and applying the rules with the highest frequency, one can generate an exemplary hypothetical library (Fig. 7): a library with a rectangular main hall (rule 1), with a colonnaded peristyle as a threshold (rules 61, 56, and 59), exterior walls that project behind the main hall (rule 52), rectangular niches along the whole length of the back and sidewalls (rules 18 and 19), an enlarged apse with the projection of the wall in the back as a focal point, a continuous U-shaped podium on top of which steps a colonnade, an entry with three door openings (rule 37), one support room in each side (rule 44), no exedras (rule 72) and a tetrastyle propylon on the axis of the short side of the peristyle (rules 73 and 87). This particular library combines components from several libraries in the corpus and the probability that it existed is reinforced by the metadata of the rules, pointing to a great degree of occurrence in the corpus of known libraries.

The grammar and the probabilistic model suggest more variations of this library, others more likely and others less likely, by changing the rules that define the form of each of its component elements: any of the architectural elements of the main hall, the threshold and/or the general complex. Figure 8 shows different possible variations of the main hall only, by changing the rules of one component each time, the niches, the focal point, the podium, the colonnade, and the entry. The library main halls illustrated are arranged in rows, one for each variant component, from the more to the less probable according to the frequency analysis of the rules in the derivations of the known libraries. Whole classes of variations can be defined by combining one component with all the other and by extending this enumeration to variations at the scale of the general building complex with the stoas, the side rooms, the exedras, and the monumental entry.

Conclusions

In conclusion, the shape grammar solves the problem of the definition of a difficult building type that showcases great variability in scale, monumentality, and the use of component elements. It provides a probabilistic model that ranks the architectural components in mandatory and optional architectural features, and ranks the architectural forms, which these take, according to the frequency they appear in the known instances in the type, and therefore the probability to appear in other cases as well. This model addresses the diversity of scale, monumentality, and function of libraries, emphasizing the general principles rather than the most monumental features that might attract attention but are not mandatory components of a library, and gives a guide to the evaluation of possible libraries and the reconstruction of hypothetical libraries.

A complete enumeration of hypothetical possible libraries would be very useful in archaeological research as it can help give form to libraries known through ancient testimonia, and especially to libraries, for which extensive descriptions of their architectural form are given in dedicatory inscriptions or ancient authors but have not been

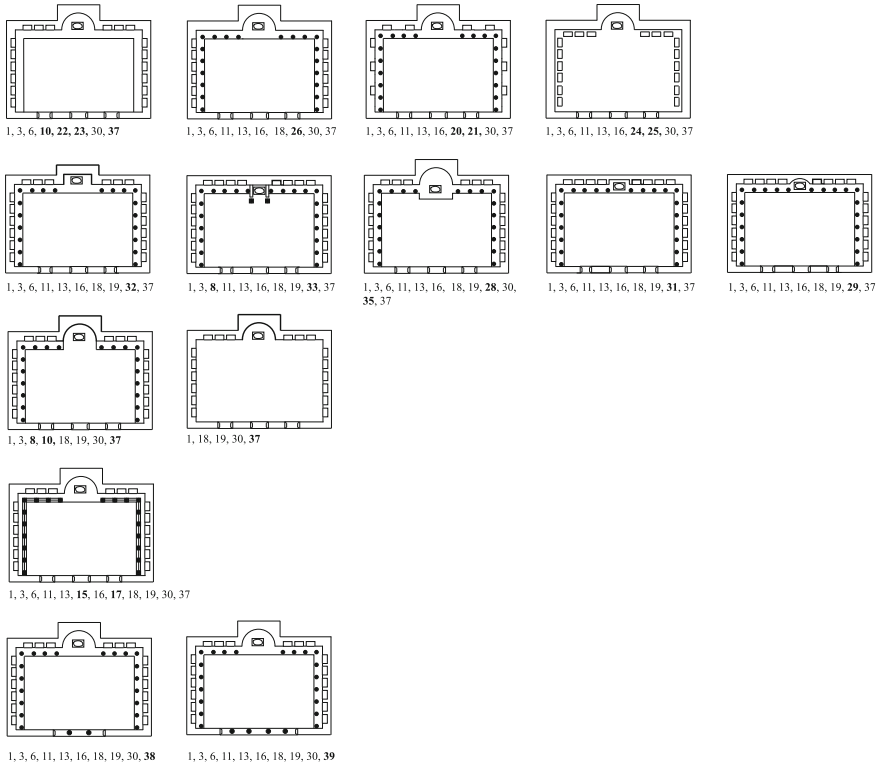


Fig. 8 Variations of rectangular wide library main halls with only one component changing each time

identified with building remains. Also, it can provide a catalogue of possible forms that can assist in identifying new libraries. Finally, it can help researchers look at the available evidence with fresh eyes and perhaps suggest different reconstructions.

References

1. Chippindale C (1992) Grammars of archaeological design. A generative and geometrical approach to the form of artifacts. Representations in archaeology. Indiana University Press, Bloomington, pp 251–276
2. Hodder I (1982) Symbolic and structural archaeology. New directions in archaeology. Cambridge University Press, Cambridge; New York
3. Stiny G (1980) Introduction to shape and shape grammars. Environ Plan 7(3):343–351
4. Stiny G, Gips J (1972) Shape grammars and the generative specification of painting and sculpture. Inf Process 7:1460–1465
5. Dylla K, Frischer B, Müller P, Ulmer A, Haegler S (2009) Rome Reborn 2.0: a case study of virtual city reconstruction using procedural modeling techniques. In: Proceedings of the 37th

- International Conference Computer Applications and Quantitative Methods in Archaeology (CAA), 22–26 March 2009, Williamsburg, VA, USA, pp 62–66
6. Knight T (1994) Transformations in design: a formal approach to stylistic change and innovation in the visual arts. Cambridge University Press, Cambridge, New York
 7. Müller P, Vereenoghe T, Wonka P, Paap I, Van Gool L (2006) Procedural 3D reconstruction of Puuc buildings in Xkipché. In: Proceedings of the 7th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST), 30 October–4 November 2006, Nicosia, Cyprus, pp 139–146
 8. Stiny G, Mitchell W (1978) The Palladian grammar. *Environ Plann B* 5(1):5–18
 9. Callmer C (1944) Antike Bibliotheken. *OpA* 3:145–193
 10. de Gregori (1937) *Biblioteche dell' Antichità*. *ABI* 11:9–24
 11. Johnson LL (1984) The Hellenistic and Roman library: studies pertaining to their architectural form. Classics, Brown University, Rhode Island
 12. Makowiecka E (1978) The origin and evolution of architectural form of Roman library, vol 1, *Studia antiqua*. Wydawa UW, Warsaw
 13. Mamoli M (2014) Towards a theory of reconstructing ancient libraries. PhD Dissertation, Georgia Institute of Technology, USA
 14. Mamoli M (2015) Library grammar: a shape grammar for the reconstruction of fragmentary ancient Greek and Roman libraries. In: Proceedings of the 33rd eCAADe Conference, Vienna, Austria, 16–18 September 2015. Vienna University of Technology, Vienna, pp 463–470

Grammars for Making Revisited





Djordje Krstic

The domain of shape grammars has been recently extended to include the field of making. This paper examines what makes abstract shapes look like concrete spatial entities (things) and what changes to the shape grammar formalism are needed to support calculations with things. Several new grammars capable of handling things are developed. Algebras supporting these are briefly addressed as well.

Introduction

In recent years, there has been a push to extend the domain of shape grammars. Traditionally, their usage was limited to the design field where they either served to analyze existing designs or produce original ones. Grammars are now extending into the field of making and manufacturing. Instead of computing with shapes, they are computing with things as in Knight and Stiny's seminal paper on making grammars [1].¹





It seems that shape grammars were always (to some extent) about things and making [ibid]. They model how things in making work. This is done as interplay of different representations [2] key among which are the pictorial ones, or shapes. However, shape grammars do not treat shapes as generic and abstract.

For example, if two identical squares are placed next to each other so that they have one side in common, or , and then one is removed, say the right square, what will remain? An obvious answer is  —the other square. This is a good

¹ See also other papers in Computational Making, Design Studies Special Issue, November 2015, Vol. 41 Part A.

D. Krstic (✉)
BioSig Technologies, Los Angeles, USA
e-mail: gkrstich@aol.com

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_26

answer typical CAD software will give and mathematicians would agree. It relies on a simple arithmetic. Each square has four lines as sides, so the two squares have eight lines, $4 + 4 = 8$ lines. If one square is removed, we have $8 - 4 = 4$ lines = 1 square remaining. However, if one takes a pencil and draws  and then takes an eraser and erases  what remains is , which certainly is not the other square. Here a different arithmetic is at play. Again, each square has four lines, but now two squares have only five lines , $4 + 4 = 5$ lines. When one square is erased we have $5 - 4 = 1$ line. This is an arithmetic in which collinear lines fuse if they touch or overlap creating longer lines. The original lines are lost in the process only to be replaced by a wealth of infinitely many lines embedded in the new lines. The eight lines of the original two squares are now reduced to five, however, the freedom to choose from infinitely many lines embedded in the five is gained in return, so that $4 + 4 = 5 = \infty$. This allows for parts of the two horizontal lines to be chosen and erased together with two of the three vertical lines leading to $5 - 4 = 1 = \infty$.

Shapes of shape grammars behave like the ones in the process of drawing. These are not abstract shapes residing in our computers, but physical things consisting of traces of graphite on paper, which are created by our hands using drafting tools such as T-squares, triangles, compasses, pencils, and erasers. One may implement shape grammars and run them on a computer. All kinds of sophisticated formal devices like graphs, closure algebras, quaternions, and differential geometry² may be used to achieve this, however, the shapes on the computer screen will—if done right—behave as the modest graphite traces on paper.

In short, grammars model the design process by simply capturing the properties of shapes as designers manipulate them. Grammars are about making, however, there are other things to be made besides drawings. There is nothing to prevent grammars in modeling some other process of making. The approach will stay the same. The shapes representing things and the way they are manipulated may be different. One should model the process of making by simply capturing the properties of things as makers manipulate them [1].

Before moving to grammars for things, we will review the shape grammars as they are used to model design processes.

Shape Grammars and Algebras for Design

A shape grammar is defined by its rules such as shape rule $a \rightarrow b$ (1), where a and b are shapes. A rule may be applied to some shape c to change it into (a different) shape c' . This process is repeated iteratively with the same or different rules until no rule from the grammar can be applied, or until some other predefined condition is met. The resulting shape is an element of the language defined by the grammar.

²Advances in Implemented Shape Grammars: Solutions and Applications AIEDAM Special Issue, Spring 2018, Vol. 32, No. 2.

A rule application is a two-step process consisting of a condition and a rule action. Rule $\mathbf{a} \rightarrow \mathbf{b}$ is applied to shape \mathbf{c} if there is a transformation \mathbf{t} —translation, rotation, scaling, reflection, or a combination thereof—such that transformed \mathbf{a} becomes a part of \mathbf{c} . That is, \mathbf{t} has to satisfy condition $\mathbf{t}(\mathbf{a}) \leq \mathbf{c}$ (2), after which a new shape \mathbf{c}' is created via a rule action described by $\mathbf{c}' = [\mathbf{c} - \mathbf{t}(\mathbf{a})] + \mathbf{t}(\mathbf{b})$ (3). An algebra for shapes provides the framework for computations (1), (2), and (3). Algebra U_{12} may handle our modest drawings on paper. It has shapes made of lines in a plane as well as Boolean operations of sum and difference—which model drawing and erasing, respectively—to combine them. It also has transformations to move the shapes around and supports combinations of transformations to create more complex ones. Sum and difference may be used to define additional operations two of which are frequently used in calculating with shapes. One is product \cdot defined as $\mathbf{u} \cdot \mathbf{v} = \mathbf{u} - (\mathbf{u} - \mathbf{v}) = \mathbf{v} - (\mathbf{v} - \mathbf{u})$ and the other is symmetric difference \oplus given by $\mathbf{u} \oplus \mathbf{v} = (\mathbf{u} - \mathbf{v}) + (\mathbf{v} - \mathbf{u})$, where \mathbf{u} and \mathbf{v} are shapes. Product $\mathbf{u} \cdot \mathbf{v}$ is the greatest shape shared by \mathbf{u} and \mathbf{v} , while $\mathbf{u} \oplus \mathbf{v}$ is the greatest shape consisting only of parts belonging either to \mathbf{u} or to \mathbf{v} , but not to both.³ Algebra U_{12} is generalized to U_{ij} , ($i, j=0, 1, 2, 3; i \leq j$), which manipulates shapes made of i -dimensional elements in j -dimensional space.

Shape grammars are seen as simple sets of rules defined in U_{ij} algebras and nothing else. This format evolved over the years with shape grammars being gradually distilled to capture the pure essence of shape behavior in the design process. Unnecessary formal devices have been removed or changed in the process without compromising the formal underpinning of shape grammars. It was the result of George Stiny's desire to keep grammars as intuitive as possible, but formally sound. Left behind were terminal and nonterminal shapes, shape vocabularies, and finally the initial shape all of which appeared in the early grammars. Although one still encounters papers with anachronous shape grammar treatment using vocabularies of shapes or even terminal and nonterminal shapes, these are rare. In contrast, the initial shape as a starting point of a derivation with a shape grammar is still preferred by many authors. We will opt here for a more recent approach where initial shape \mathbf{c} is replaced by initial rule $0 \rightarrow \mathbf{c}$ (1a) [3]. This has couple of advantages. An initial shape is a rigid entity of a fixed size and orientation defined at a fixed place in space, which constrains possible derivations with grammars. In contrast, the initial rule with its empty left-hand side may act under arbitrarily transformation \mathbf{t} to produce initial shape $\mathbf{t}(\mathbf{c})$ of desired size orientation and position, thus broadening the range of starting points for the grammar. Better yet, this approach is Stinyan in spirit as it models what designers do when they start their designs on whichever part of their drafting boards, screens, canvases, or studios they wish. It subsumes the white-canvas-start, with no shape given ahead, and the preexisting-shape-start, like in remodeling of the existing building. In the context of grammars for things, the initial rule supports moving from the design scale to the world scale. For example, transformation \mathbf{t} —under which the initial rule applies—should scale-up shape \mathbf{c} 48

³Formally: $x \leq \mathbf{u} + \mathbf{v} \Rightarrow x \leq \mathbf{u} \mid x \leq \mathbf{v}$; $x \leq \mathbf{u} - \mathbf{v} \Rightarrow x \leq \mathbf{u} \ \& \ \neg(x \leq \mathbf{v})$; $x \leq \mathbf{u} \cdot \mathbf{v} \Rightarrow x \leq \mathbf{u} \ \& \ x \leq \mathbf{v}$ and $x \leq \mathbf{u} \oplus \mathbf{v} \Rightarrow [x \leq \mathbf{u} \ \& \ \neg(x \leq \mathbf{v})] \mid [x \leq \mathbf{v} \ \& \ \neg(x \leq \mathbf{u})]$, where $\&$, \mid , \neg , and \Rightarrow stand for and, or, negation, and implication, respectively.














times to go from the standard architectural scale $\frac{1}{4}'' = 1' - 0''$ to the world one. In contrast, c should be scaled-down if what was designed is a computer chip.

Towards Grammars for Things⁴





A rule application described by (1) through (3) works well with shapes and it is no surprise that it has been in use for nearly half a century. Almost all published shape grammars are based on these. Notable exceptions were structure grammars [4], earlier mentioned collision protecting grammars and more recently grammars in which rules are defined by transformations and not shapes [3]. However, in the context of grammars for things, these may not be adequate and need to be changed. Some other restrictions may also apply for shapes to behave like things.


Shapes that Are Things


If shapes of U_{ij} are to be close approximations of things, then some otherwise usual calculations with shape grammars may prove to be problematic.









For example, let shape  defined in U_{12} —representing a small crane hook—be generated by a shape grammar. Also, let the last step in the derivation be drilling of the hook’s eye into hook blank . This is done via an action of rule  \rightarrow , which proceeds in accordance with formulas (1) through (3). Starts with finding transformation t to satisfy (2) followed by computing shape $c - t(a)$ and summing it with $t(b)$ to complete (3). Shapes , , , and  are c , $t(a)$, $c - t(a)$, and $t(b)$, respectively. This works well with shapes, however, not so well when they represent other types of things. Shape  represents a real hook by tracing the hook’s outline as does  for the hook blank. Unfortunately, shapes , , and , appearing in the computation above, are not shape outlines and cannot


⁴Term thing is used here in its dictionary meaning: concrete spatial object. Knight and Stiny [1] are more precise by distinguishing things from stuff—materials the things are made of. Nevertheless, grammars defined here will work with Knight and Stiny’s things. In that, they are not different than Knight and Stiny’s grammars for things.

represent things. The same is true for shapes  and , which define the rule. If we wish for realism in rule applications—and realism should be a feature of a true grammar of things—then all the shapes in computations should be representations of things. There is another problem with the realism of our representation. Rule  \rightarrow , which facilitates drilling a hole in the hook blank is based on a spatial relation in which $a < b$. This defies expectation for a thing with a hole to have less material than the same thing without it. The rule adds to the shape instead of subtracting from it.



One may mitigate the above problems by defining the hook as a 3D shape in U_{33} , or . It is well known that in the framework of a diagonal algebra—i.e., an algebra

U_{ii} where shape elements and the space they occupy are of the same dimension—any rule applies in infinitely many ways [5: 214]. We, thus, include boundary lines to provide registration for transformations and restrict the applications of rules. Now the hook is represented by , which is a shape with a solid and a linear component defined in a compound algebra. The algebra is the noncommutative sum [3] of U_{33}

and U_{13} . The rule is now  \rightarrow  and shapes , ,  and  are c , $t(a)$, $c - t(a)$, and $t(b)$, respectively. All the above are 3D shapes that may represent real things. In addition to this  $>$ ,

which is what should be expected given that shape b (right) is the same as shape a (left) only with  removed. A careful reader may have noticed that only the U_{33} component of the algebra behaves desirably while the U_{13} component, if taken alone, has the same problems we encountered earlier. This may not be of concern because (the well behaved) U_{33} component represents things while the other one is there only to streamline the rule applications. The answer may not satisfy the fastidious among us thus the following remedy.

We may define the hook and do the calculations in a framework of a UB_3 algebra [6]. The latter is a subalgebra of $U_{33} + U_{23}$ in which solid shapes of U_{33} are paired with their boundaries (surface shapes) in U_{23} and restricted to only one operation: symmetric difference \oplus . The latter has a nice property of preserving boundaries of shapes. That is if x and y are shapes and b is an operator that takes a shape to its boundary then $b(x) \oplus b(y) = b(x \oplus y)$ holds [6, 7]. The only caveat here is that UB_3 has only symmetric difference operation while rule action (3) calls for the operations of difference and sum. Luckily, symmetric difference acts as a difference with comparable shapes and as a sum with the discrete ones. Condition (2) assures that c and $t(a)$ are comparable and we may add condition $[c - t(a)] \cdot t(b) = 0$ (4) to render $c - t(a)$ and $t(b)$ discrete. Under conditions (2) and (4), rule action (3) becomes equivalent to $c' = [c \oplus t(a)] \oplus t(b)$ (5). The first occurrence of \oplus in (5) acts as a difference and the second as sum. Because it is impossible to distinguish, when represented on paper, between a solid shape and its 3D bounding surface, the hook

example is—without the loss of generality—redone in the framework of a UB_2 algebra. The shapes representing things are defined as planar segments in U_{22} and their boundaries as line shapes in U_{12} . The hook is now  , the rule  →

 and the intermediate shapes are  . Note that both

the shapes defined in U_{22} and their boundaries in U_{12} may represent things. There is an additional benefit to keeping track of shape boundaries. Because they are a dimension lesser than the shapes they bound, boundaries are simpler representations. For example, in geometric modeling boundary representation or B-rep is a widely used method of representing 3D shapes by their 2D boundaries. The calculation of physical or engineering properties of things could be simplified by considering the boundaries of their representative shapes instead of the shapes themselves. Many such properties could be expressed via volume integrals over shapes. The latter could be simplified by taking the sums of simpler surface integrals over the boundaries of the shapes—in accordance with a well-known method of direct integration.

Collision Protecting and Biconditional Grammars

One may point to a couple of problems with the last example.

First, condition (2) may not hold for shape boundaries. That is, $t(a) \leq c$ does not guarantee that $b(t(a)) \leq b(c)$ —which only works if $t(a) = c$. When $t(a) \leq c$ holds condition $b(t(a)) \cdot b(c) \neq 0$ (2a) is necessary—but not sufficient—in order to have a proper registration.

Second, we introduced condition (4) just to be able to use \oplus in place of $+$, which may be an arbitrary and unnecessary constraint. Indeed, in standard shape grammars, drawing over $c - t(a)$ when adding $t(b)$, which (4) prevents, does not affect the result. It only amounts to a redundant addition of shape $[c - t(a)] \cdot t(b)$. However, with shapes representing things, the harmless redundancy becomes a dangerous collision. This calls for a new type of grammars, like the collision protecting ones [6].

In collision protecting grammars, the original condition (2) is augmented by condition (4), to assure that what is added does not collide with what is there. The latter grammar may be defined in U_{ij} algebra and is the only type of grammar that can work in UB_i algebra to carry on the parallel generation of shapes and their boundaries.





Collision protecting grammars could be generalized. Condition (4) is equivalent to $c \cdot t(b - a) = 0$ (4a),⁵ which could be generalized to $c \cdot t(a') = 0$ (6). Shape a' is specified with each rule and appears on the left side of the rule together with a , or




⁵Based on Boolean identity $(u - v) \cdot w = u \cdot (v - w)$, validity of which is shown by $x \leq (u - v) \cdot w \Rightarrow x \leq (u - v) \cdot w \Rightarrow x \leq u \cdot w \ \& \ x \leq w \Rightarrow x \leq u \ \& \ x \leq w - v \Rightarrow x \leq u \cdot (w - v)$ so that $(u - v) \cdot w \leq u \cdot (w - v)$; $x \leq u \cdot (w - v) \Rightarrow x \leq u \ \& \ x \leq (w - v) \Rightarrow x \leq u \ \& \ x \leq w \ \& \ \neg(x \leq v) \Rightarrow x \leq u - v \ \& \ x \leq w \Rightarrow x \leq (u - v) \cdot w$ so that $u \cdot (w - v) \leq (u - v) \cdot w$ and finally $(u - v) \cdot w = u \cdot (w - v)$.

$a, a' \rightarrow b$ (7). Shapes a and a' are the respective inclusive and exclusive precedents of the rule while b is its consequent. Note that $a \cdot a' = 0$ has to hold because of conditions (2) and (6). A grammar with rule (7) applying under conditions (2) and (6) is a biconditional grammar [ibid]. The latter rule/application is modeled after that of structure grammars [4]. Biconditional grammars subsume both standard and collision protecting grammars. For $a' = 0$, the rule is a standard one while for $a' \geq b - a$ it is collision protecting. If at least one rule of a grammar has a nonempty a' , the grammar is biconditional. In contrast, all the rules must be collision protecting for the grammar to be a collision protecting one.

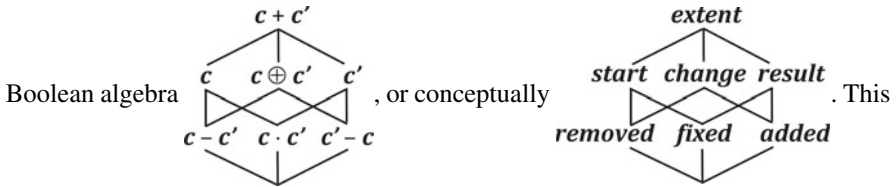
Shape Flip-Flopping

There is yet another problem related to the way rules are applied. Because condition (2) serves to establish transformation t , shape a must be sufficiently big to allow for a meaningful matching of $t(a)$ and c . A big a may take out more than needed when—in accordance with (3)— $t(a)$ is subtracted from c . This needs to be recovered, in the next step, when $t(b)$ is added, which may render b big as well. As a result, shape $t(a \cdot b)$ gets removed first and then put back during a rule action. This shape flip-flopping may not be practical for a maker, which would render shapes $t(a)$, $c - t(a)$, and $t(b)$ unsuitable for constructing (or making) c' .

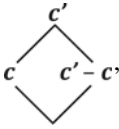
For example,  do not have much relevance for making the hook. It is highly unlikely that, instead of simply drilling a hole, someone would cut off  from  and in its place weld , yet that is exactly what the rule does.

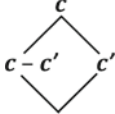

In contrast, shapes , , and  are relevant for making the hook. The first is c , the hook's blank, the second is $c - c'$, the shape removed by drilling to form the hook's eye, while the third is the finished hook, or c' .

This renders c and c' as shapes of the main interest from a maker's point of view. The two are, respectively, the starting and resulting shapes of the rule action, but they go further than that in showing what the rule does. Shapes $c' - c$ and $c - c'$ account, respectively, for what has been added and what has been removed. Their sum $(c' - c) + (c - c') = c \oplus c'$ captures the scope of the change while shape $c \cdot c'$ is what remains unchanged—a fixed point of the rule action. Finally, shape $c + c'$ subsumes all the shapes involved establishing the extent of the rule action. The shapes above form



is the argument lattice for all of the (Boolean) computations with c and c' as arguments [8]. The latter establishes the upper bound for such computations and incorporates all the shapes involved including arguments, results, and intermediate shapes. An argument lattice exposes the structures these shapes acquired in computations. It does it via argument decompositions, which are prime ideals of the argument lattice generated by the relevant shapes [ibid]. The relation between c and c' also points to the type of rule used. If $c < c'$, it is an addition rule [9: 49], characterized by $a < b$. Similarly, if $c' < c$ then $b < a$, which characterizes the subtraction rules [ibid.]. Even $c = c'$, which is equivalent to $a = b$ where no change in design occurs, points to a certain rule type. Not surprisingly, it is a useless rule [10], which, surprisingly, turns

to be useful in structuring c —by recognizing $t(a)$ as its part. Lattices 

, and  describe addition, subtraction, and useless rules, respectively.

Some New Grammars

Designers want to have options to be able to try different things and explore multiple paths. This does not cost much—just strokes of pencil on paper. Standard rule $a \rightarrow b$ is well suited for that. It has 32-element argument lattice (Fig. 1) generated by shapes c , $t(a)$, and $t(b)$, which provides a wealth of shapes and spatial relations to choose from. In contrast, engineers among makers want to have a straightforward, nonambiguous, and economical way of making things. There is no space for trial and error as making is expensive. Standard rules allow for shape flip-flopping, which is not economical and may be ill suited for making—as it was in our hook example.

There should be no overlaps between what is removed, what is added or what stays put. Consequently, rules for making should not be related to big 32-element argument lattices, but to much smaller 8-element lattices generated by *start* and *result* shapes or c and c' . Such a rule is *removed* \rightarrow *added*, or $c - c' \rightarrow c' - c$, which when expressed in terms of c , $t(a)$, and $t(b)$ becomes $t(a - b) \rightarrow t(b) - c$. Note that the 8-element lattice related to this new rule is embedded in the 32-element lattice of the standard rule (Fig. 1, gray). Unfortunately, the rule contains variables c and t that depend on the instance of the rule application. An exercise in shape (Boolean) arithmetic may fix that. Because $t(a) \leq c$ —in accordance with (2)—the rule’s right-

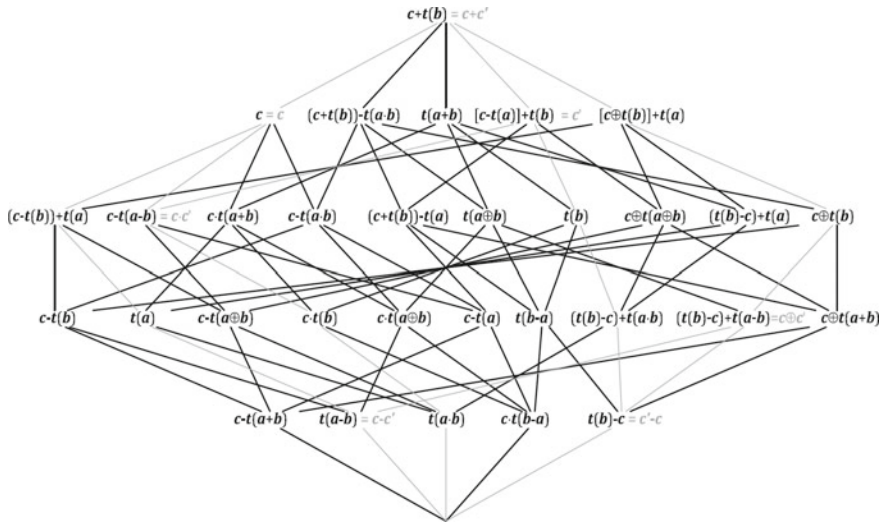


Fig. 1 Standard rule argument lattice generated by shapes c , $t(a)$, and $t(b)$ (black and gray) with the embedded new rule argument lattice (gray)

hand side $t(b) - c$ is a part of $t(b - a)$ so it would not change if multiplied by the latter, or $t(b) - c = t(b - a) - [c \cdot t(b - a)]$. With collision protecting grammars, the expression in the angular brackets is equal to 0—in accordance with (4a)—leading to rule $t(a - b) \rightarrow t(b - a)$. Now only one variable t remains, however, tools, such as the rules equivalence, are needed to remove it.

Definition 1 Two rules are equivalent if whenever applied to identical shapes the resulting shapes are identical [3].

Proposition 1 Identical rules are equivalent.

Proposition 2 A rule is equivalent to its transformed copy [ibid]⁶

Consequently, if both sides of rule $t(a - b) \rightarrow t(b - a)$ are acted upon by transformation t^{-1} an equivalent rule $a - b \rightarrow b - a$ arises. This is summarized in the following lemma.

Lemma 1 Rules $a \rightarrow b$ and $a - b \rightarrow u$, where $b - a \leq u \leq b$, are equivalent when applied under the same transformations. The latter rule prevents shape flip-flopping.

The two rules are equivalent only with respect to (3). Conditions $t(a) \leq c$ and $t(a - b) \leq c$ are not equivalent, in general, and may yield different sets of allowable transformations.

⁶Indeed, rules $a \rightarrow b$ and $g(a) \rightarrow g(b)$, where g is a transformation, are equivalent. If the former applies to shape c under transformation t such that $t(a) \leq c$ then the latter applies under tg^{-1} . That is, $tg^{-1}(g(a)) = t(a) \leq c$ and $[c - tg^{-1}(g(a))] + tg^{-1}(g(b)) = (c - t(a)) + t(b) = c'$.

The lemma holds for case $u = b - a$ as shown above. This is the most restrictive case, as it prevents not only shape flip-flopping but also the redundant addition of shapes—though only for collision protecting grammars. With standard grammars if shape $[c - t(a)] \cdot t(b)$ exists, it gets added redundantly. Bigger shapes could also be added without running a risk of shape flip-flopping. The upper bound is $u = b$, in which case shape $t(a \cdot b) + [c - t(a)] \cdot t(b) = c \cdot t(b)$ is added redundantly.

We may take the other way around: start with rule $a \rightarrow b$ and find out which relation between a and b prevents shape flip-flopping. This is handled by a straightforward corollary of the lemma above.

Corollary *Rule $a \rightarrow b$, where $a \cdot b = 0$, prevents shape flip-flopping.*

Unfortunately, condition $a \cdot b = 0$ may constrain a in such a way that it impedes its usefulness for determining t .

For example, let a rule just glue a part to an object while leaving it otherwise intact. Because nothing needs to be removed and a and b do not overlap, a ends up being empty. Empty a allows for t to be any transformation. Consequently, shape $t(b)$ is glued randomly, which is not what the rule is supposed to do.










We may use a only for determining t and have a different shape b' for removing what the rule needs to remove. Because shape $t(b')$ must be recognized as a part of c , before it is removed, we set $b' \leq a$ (8). Condition $a \cdot b = 0$, which took care of shape flip-flopping, is replaced by $b' \cdot b = 0$ (9) to go with rule action $c' = [c - t(b')] + t(b)$ (10). The new rule format is $a \rightarrow b, b'$ (11), which feels like biconditional grammar format $a, a' \rightarrow b$. There are other similarities between the two types of grammars. We may recall that inclusive and exclusive rule precedents a and a' are related to shape c via $t(a) \leq c$ and $c \cdot t(a') = 0$ —the familiar conditions (2) and (6) instrumental in determining t . Shapes b and b' are similarly related to shape c' . That is, $t(b) \leq c'$ (12) and $c' \cdot t(b') = 0$ (13)—because of (10) and (9). Conditions (12) and (13) render shapes b and b' as inclusive and exclusive consequents of the rule, respectively. For lack of a better name, the new grammar is called biconsequential. This approach has an intuitive appeal. A maker may observe (sense, grab, hold, handle) some part $t(a)$ of c , but ends up removing (drilling, cutting off, filing out) a smaller part $t(b') \leq t(a)$.

We have two new grammar formats developed to solve the problems standard shape grammars have when adopted for things. Biconditional grammars prevent redundant shape additions, which in the context of things are no less than collisions, while biconsequential grammars take care of shape flip-flopping. A grammar capable of calculating with things could take advantage of properties of both biconditional and biconsequential grammars. Such is a quad (conditional) grammar with rules of the form $a, a' \rightarrow b, b'$ (14) acting in accordance with (10) under a transformation determined by conditions (2) and (6). Shapes a and a' are inclusive and exclusive precedents of the rule while b and b' are its inclusive and exclusive consequents. The latter shapes satisfy (9), (12) and (13), while b' also satisfies (8).

Quad grammar's addition, subtraction, and useless rule formats are $a, a' \rightarrow b, 0$ (15), $a, a' \rightarrow 0, b'$ (16) and $a, a' \rightarrow 0, 0$ (17), respectively. The respective rule actions are $c' = c + t(b)$ (18), $c' = c - t(b')$ (19) and $c' = c$ (20), in accordance with (10).

One can easily tell the rule type—via (15), (16) and (17)—and what to expect from it—via (18), (19) and (20). In contrast, with standard shape grammars, one needs to know the relation between a and b to be able to tell the type and do some calculations to find out what to expect. A quad grammar useless rule is truly perceptual. It just recognizes that—under some $t - a$ is a part of c and a' is not but does nothing about that. The latter is clear from rule action (20), which advises that c should be left alone. In contrast, a useless rule of a standard shape grammar is heavy on action: $t(a)$ is recognized as a part of c but is then removed only to be added back, or $c' = [c - t(a)] + t(a)$.

The collision-protecting variety of quad grammars prevents collision between $t(b)$ and $c - t(b')$, or $t(b) \cdot [c - t(b')] = 0$. Because of (9), b and b' do not collide and it is sufficient to prevent $t(b)$ and c from colliding, or $t(b) \cdot c = 0$. The latter is true whenever condition (6) is satisfied for $a' \geq b$. To maximize the freedom in determining t , we keep a' as small as possible leading to $a' = b$ and rule $a, b \rightarrow b, b'$ (21). The additive, subtractive, and useless rules are $a, b \rightarrow b, 0$ (22), $a, 0 \rightarrow 0, b'$ (23) and $a, 0 \rightarrow 0, 0$ (24), respectively. Note that because with subtractive and useless rules, nothing is added there is nothing to collide with, so condition (6) should be omitted.

Our hook example could be recast to accommodate the new quad grammar format. Because the original rule  \rightarrow  is subtractive, which results in shape  being removed while nothing is added, the quad grammar rule should be ,  \rightarrow ,  in accordance with (22). It acts on  under transformation t determined by $t(\img alt="A hook shape with a hole on the left side." data-bbox="538 458 588 488"/>) \leq \img alt="A hook shape with a hole on the right side." data-bbox="538 488 588 538$, condition (2). The action proceeds in accordance with (19), $\img alt="A hook shape with a hole on the right side." data-bbox="588 488 638 538} - t(\img alt="A hook shape with a hole on the left side." data-bbox="598 498 648 528}) = \img alt="A hook shape with a hole on the right side." data-bbox="608 488 658 538} - \img alt="A hook shape with a hole on the left side." data-bbox="608 498 658 528} = \img alt="A hook shape with a hole on the right side." data-bbox="608 488 658 538}$. The rule now does what it should: drills a hole and nothing else. Note that in formulas above,  denotes the empty shape. All other shapes are represented pictorially, and the empty shape should be no exception.

Equivalence of Grammars

Despite the similarities between biconditional and biconsequential grammars, there is a notable difference in how they relate to the standard shape grammars. Biconditional grammars, as shown earlier, subsume standard shape grammars. That is, if each rule $a, a' \rightarrow b$ of a biconditional grammar has $a' = 0$, the grammar is a standard one. In contrast, a biconsequential shape grammar could at most be equivalent to a standard

one. The equivalence and related strong equivalence of grammars is handled by the following:

Definition 2 Two equivalent rules are strongly equivalent if their actions defining the equivalence have identical argument lattices.

For example, rules $a \rightarrow b$ and $g(a) \rightarrow g(b)$, where g is a transformation, are equivalent in accordance with Definition 2 and Proposition 2. The rules act in accordance with (3) so that their argument lattices are generated by $\{c, t(a), t(b)\}$ and $\{c, tg^{-1}(g(a)), tg^{-1}(g(b))\} = \{c, t(a), t(b)\}$, respectively. This renders the lattices equal and the rules strongly equivalent.

Definition 3 Two grammars are equivalent/strongly equivalent if there is a bijective mapping between the rules of the grammars such that a rule of one grammar is mapped to an equivalent/a strongly equivalent rule of the other grammar.

Note that the relations of rule and grammar equivalence/strong equivalence are reflexive symmetric and transitive⁷ and that all strongly equivalent rules/grammars are also equivalent, but not vice versa.

Proposition 2 *Equivalent/strongly equivalent grammars generate the same language.*

Proposition 3 *Identical grammars are equivalent/strongly equivalent.*

These are consequences of definitions 1, 2, and 3.

Proposition 4 *A quad grammar with all of the rules of form $a, 0 \rightarrow b, b'$ is strongly equivalent to a biconsequential grammar if for each rule $a \rightarrow b, b'$ of the latter, there is a rule $a, 0 \rightarrow b, b'$ of the former and vice versa.*

The proposition defines a bijection between the rules of the two grammars as prescribed by definition 3. Because $a' = 0$ both rules act on c under the same transformation t —determined by a —and because rule action is (10) for both, they both generate the same shape $c' = [c - t(b')] + t(b)$, which makes the rules equivalent. Additionally, argument lattices of both actions are generated by the same set $\{c, t(b'), t(b)\}$ rendering them equal and the rules strongly equivalent, which completes the proof.

Proposition 5 *A biconsequential grammar is equivalent to a standard grammar if for each rule $a \rightarrow b, b'$ of the former there is a rule $a \rightarrow b + (a - b')$ of the latter and for each rule $a \rightarrow b$ of the latter there is a rule $a \rightarrow u, a - b$, where $b - a \leq u \leq b$, of the former.*

⁷ $(x, x) \in R, (x, y) \in R \Rightarrow (y, x) \in R, (x, y) \in R \ \& \ (y, z) \in R \Rightarrow (x, z) \in R$, respectively, where R is a relation and x, y , and z are elements of the set on which R is defined.

The proposition already defines a bijection between the rules of the two grammars so that only the rule equivalence remains to be shown to satisfy definition 3. Because all four rules have a as the left-hand side, they apply under the same transformation t determined by $t(a) \leq c$. Rule $a \rightarrow b, b'$ acts in accordance with (10) to produce $c' = [c - t(b')] + t(b)$. Because $b' \leq a$ one may write $c - t(b') = [c - t(a)] + t(a - b')$ so that $c' = [c - t(a)] + t(a - b') + t(b) = [c - t(a)] + t(b + (a - b'))$, which is an action of rule $a \rightarrow b + (a - b')$ in accordance with (3). Rules $a \rightarrow b, b'$ and $a \rightarrow b + (a - b')$ are equivalent as they produce the same shape c' when applied to c . Rule $a \rightarrow u, a - b$ acts in accordance with (10) to produce $c' = [c - t(a - b)] + t(u)$. This reassembles equation (3) for standard grammar rule $a - b \rightarrow u$ rendering the two rules equivalent. The latter rule is also equivalent to $a \rightarrow b$ in accordance with lemma 1, so that rules $a \rightarrow b$ and $a \rightarrow u, a - b$, are equivalent as well. This completes the proof.

Corollary *Proposition 5 is valid for a quad grammar with all of the rules of form $a, 0 \rightarrow b, b'$ in the place of the biconsequential grammar.*







The proof follows from propositions 4 and 5 and transitivity of the equivalence relation.

Note that grammars from propositions 4 and 5 and the corollary are equivalent, but they need not be strongly equivalent.


For example, $a \rightarrow b$ and $a \rightarrow u, a - b$, which are equivalent for $b - a \leq u \leq b$ —as shown above—are not strongly equivalent. The actions defining their equivalence are (3) and (10) with argument lattices $Ar_{(3)}$ and $Ar_{(10)}$ generated by $\{c, t(a), t(b)\}$ and $\{c, t(a - b), t(u)\}$, respectively. Consequently $Ar_{(3)} \neq Ar_{(10)}$.⁸

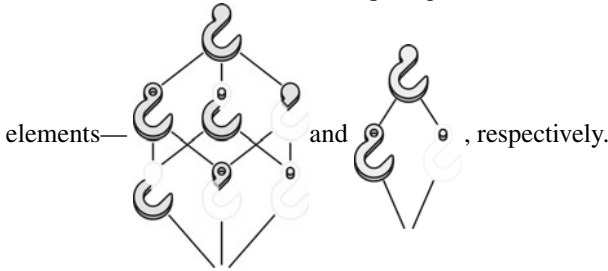
Two equivalent grammars produce the identical shapes, and so do two strongly equivalent grammars, however, the latter grammars divide these shapes in the same way while the former may fail to do so. Equivalent shapes generated by two strongly equivalent grammars have identical argument decompositions, which may not be the case if the grammars were only equivalent.

This is important when dealing with things in place of shapes. If a result of a making process, which is modeled by a shape grammar, is a thing featuring a set of parts, then a different process modeled by a grammar strongly equivalent to the original one can produce the same thing with the same parts. In contrast, with equivalent grammars, both the parts and process may change.

For example, hook  may be generated by a standard grammar rule  \rightarrow  acting on . It can also be generated by actions of rule  \rightarrow  and quad

⁸For $b - a < u < b$, no combination of $c, t(a - b)$ and $t(u)$ can produce $t(a)$ and $t(b)$, and no combination of $c, t(a)$ and $t(b)$ can produce $t(u)$ so that $Ar_{(3)} \neq Ar_{(10)}$. For $u = b - a$ and $u = b$, no combination of $c, t(a - b)$ and $t(b - a)$ can produce $t(a)$ and $t(b)$, no combination of $c, t(a - b)$ and $t(b)$ can produce $t(a)$, and combining $t(a)$ and $t(b)$ can produce $t(a - b)$ and $t(b - a)$ so that $Ar_{(10)} \subset Ar_{(3)}$.

grammar rule . The first two rules are strongly equivalent and are both equivalent to the third one. The actions of the first two rules recognize the same shape parts and have identical argument lattices. In contrast, the third rule action recognizes different parts and has a different argument lattice. Because all of the rules are subtractive, the standard 32-element rule action argument lattice is reduced to 8 element, while the quad grammar 8-element lattice now has only 4



Transformation Rules

Transformation rules, or rules that are transformations, may play important role in making. Parts are not always produced by adding and subtracting as in 3D printing and milling. Some other operation that transforms the part like bending, twisting, folding, and knotting may be used as well. These could be handled in the framework of standard shape grammars via rules $a \rightarrow g(a)$ [11], where g is a transformation and shape a is replaced by its transformed version $g(a)$. This works well in U_{12} when modeling a process of drawing. A square traced on a piece of paper can only be moved—without moving the paper itself—by erasing and redrawing, in accordance with the rule above. In contrast, a 3D object can be moved directly. A grammar defined in U_{33} could make a use of rules that do not replace shapes but transform them instead. Such a rule has the familiar format $a_t \rightarrow g(a)$ (25), except for the subscript t , leading the arrow, which distinguishes the rule from the standard replacement one. The rule applies under condition $t(a) = c$ (26) and rule action $c' = g(t(a)) = gt(a)$ (27). Similar rules have been developed in [3] except that shape a was a variable taking the value of c at each rule application. It was sufficient to know transformation g to define the rule, or simply put: g is the rule. It applies to shape c to create $c' = c + g(c)$.

Transformation rules adhere to the broader definition of replacement rule given by Knight and Stiny “The replacement operation (\rightarrow) is a general operation which subsumes all kinds of doing and sensing, whether simultaneous or independent” [1: 15].

Towards Algebras for Things

Algebras for shapes U_{ij} , which underpin standard shape grammars, may work with things that behave like shapes, or do so under certain conditions.

For example, things created by the additive process of 3D printing as well as the ones created by the subtractive process of milling are in this category. The former process is based on the sequential fusion of melted material droplets, which lends itself to the U_{33} addition operation while the latter involves gradual removal of material by a rotating tool, which is akin to the subtraction operation. Shape $a = \sum_{i=1,\dots,n} t_i(\mathbf{p})$, models a 3D printed object with \mathbf{p} being a droplet, t_i a transformation describing an incremental movement of the printer's nozzle, and n the total number of droplets making a . In contrast, shape $c = \mathbf{b} - \sum_{i=1,\dots,m} t_i(\mathbf{e})$, represents a milled part with \mathbf{b} a block of material, \mathbf{e} the envelope of the rotating tool when the rig is stationary, t_i a transformation describing an incremental movement of the rig, and m the total number of such movements to make c .

Another making process that may be modeled with a standard algebra for shapes is MIT Self-Assembly Lab's Rapid Liquid Printing. It uses a robotic arm with an extruder tool attachment to create lines of silicon rubber—or some other extrudable material. The extruder nozzle moves while submerged in liquid gel, which serves to support the extrusion. This is in effect a line drawing in 3D and can be modeled by U_{13} algebra, which manipulates lines in space. By changing the speed of the extruder nozzle, one can regulate the thickness of the extruded lines, which can be modeled via W_{13} algebra—a version of U_{13} for weighted lines [12].

Similarly, laser cutting can be modeled using standard algebras for shapes. Laser cuts are defined in U_{12} algebra—for lines in a plane—while the objects delineated by the cuts are defined in U_{22} algebra—for planar segments in plane. Object $\mathbf{o} = \mathbf{b}^{-1}(\mathbf{c})$, where \mathbf{b} is a boundary operator and \mathbf{c} is a linear shape tracing the laser cuts. Note that the inverse of \mathbf{b} may fail to be a function, but in this case, it must be one, or the laser cuts would not result in an object. That is, \mathbf{c} must be devised so that $\mathbf{c} = \mathbf{b}(\mathbf{o})$, which is always true for objects defined in UB_2 algebra.

The processes above may produce parts that could be assembled into products. Unfortunately, the assembly process cannot be modeled by the standard algebras for shapes that were used for parts. Adding two parts in U_{33} may fuse them, which would create a single part instead of an assembly.

Algebras for Parts Assembly

The addition operation of U_{33} is a Boolean one, which fuses together shapes if they overlap or their boundaries do so. This does not work for the assembly of parts where parts need to remain separated. Luckily, shape grammar theory is rich enough to provide tools for overcoming this obstacle. We will look at two possible ways of constructing algebra for assembly of parts.

Algebra of Labeled Shapes

Algebras V_{ij} of labeled shapes [12], or shapes with labeled geometric elements, are well suited for handling shape assembly. Two geometric elements—of such a shape—that overlap, or their boundaries overlap, fuse if they are embedded in the same geometric element and have the same labels. They stay unchanged otherwise. A shape of V_{33} may represent a part that enters the assembly if all its geometric elements are labeled the same way. That is, if the whole shape appears to be labeled by a single label. The collection of such shapes may represent a kit of parts if each shape in the collection has a different label and shapes do not overlap. The sum of the shapes in the collection is an assembled object. The following notation may help with adding the details.

With each label l , we associate a function with the same name such that $l(\mathbf{a}) \in V_{ij}$, where $\mathbf{a} \in U_{ij}$, consists of geometric elements of \mathbf{a} labeled l . Conversely, if $\mathbf{a} \in V_{ij}$, then $l^{-1}(\mathbf{a}) \in U_{ij}$ consists of geometric elements of \mathbf{a} , which were labeled l . For $\mathbf{a} \in U_{ij}$, $l^{-1}(l(\mathbf{a})) = \mathbf{a}$, and for $\mathbf{a} \in V_{ij}$, $l(l^{-1}(\mathbf{a})) \leq \mathbf{a}$. Note that Stiny uses \mathbf{a}^l for $l(\mathbf{a})$ and \mathbf{a}_l for $l^{-1}(\mathbf{a})$ [5: 218].

If $\{l_1, l_2, \dots, l_n\}$ is a set of labels and $\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n\}$ is a set of shapes such that $\mathbf{a}_i, \mathbf{a}_j \in U_{33}$ and $\mathbf{a}_i \cdot \mathbf{a}_j = 0$ for every $i \neq j \in \{1, 2, \dots, n\}$ (28), then $\{l_1(\mathbf{a}_1), l_2(\mathbf{a}_2), \dots, l_n(\mathbf{a}_n)\}$ is a kit of parts and $\mathbf{c} = \sum_{i=1, \dots, n} l_i(\mathbf{a}_i) \in V_{33}$ is the assembled object. One may use $\mathbf{a}_i = l_i^{-1}(\mathbf{c})$, for $i = 1, 2, \dots, n$, to disassemble \mathbf{c} . We may add transformations $\{t_1, t_2, \dots, t_n\}$ to move parts to the right positions for assembly. Set $\{t_1(l_1(\mathbf{a}_1)), t_2(l_2(\mathbf{a}_2)), \dots, t_n(l_n(\mathbf{a}_n))\}$ such that $t_i(\mathbf{a}_i) \cdot t_j(\mathbf{a}_j) = 0$ for every $i \neq j \in \{1, 2, \dots, n\}$ (29), is the kit of parts, positioned for assembly, $\mathbf{c} = \sum_{i=1, \dots, n} t_i(l_i(\mathbf{a}_i))$ is the assembled object and $\mathbf{a}_i = l_i^{-1}(t_i^{-1}(\mathbf{c}))$, for $i = 1, 2, \dots, n$, when disassembled.

Algebra V_{33} is well suited to deal with shape assemblies, however, it may equally well handle the making of the parts for assembling. In this respect, V_{33} is not different than U_{33} . Indeed, by extending the notation above to sets of shapes one gets, for any label l , $l^{-1}(V_{33}) = U_{33}$ and $l(U_{33}) \subseteq V_{33}$. That is, for every label l there is a subalgebra of V_{33} isomorphic to U_{33} . Like U_{33} , V_{33} will not prevent shape collisions when making parts. It also does not enforce condition (28) so that the part collisions are possible during assembling. It is up to grammars to take care of that. Collision protecting grammars are an obvious choice there.

Algebra of Shape Decompositions

Shape decompositions [2, 13, 14] seen as finite sets of shapes, or more precisely, discrete shape decompositions with pairwise discrete elements [14], are good representations of part assemblies. There is a variety of algebras for shape decompositions [ibid], but the one that uses set operations [2, 3] is the right one for this purpose. Algebra D_{ij} manipulates shape decompositions with set operations of union \cup as the sum, intersection \cap as the product and difference \setminus as the difference. It partially

orders them via subset relation \subseteq . Unlike V_{33} , D_{33} does not support the making of the parts. Algebra U_{33} is still needed for this. Once the parts are generated, they are moved to their positions for assembly, “packed” into singleton sets—one part per set—to become decompositions and summed. The resulting decomposition is the assembled object, which is a set of pairwise discrete shapes representing parts. That is, if $a_1, a_2, \dots, a_n \in U_{33}$ are shapes representing parts and $t_1, t_2, \dots, t_n \in U_{33}$ are transformations to move them to the right positions for assembly, then $t_1(a_1), t_2(a_2), \dots, t_n(a_n)$, satisfying (29), are the positioned parts, $\{t_1(a_1)\}, \{t_2(a_2)\}, \dots, \{t_n(a_n)\} \in D_{33}$ are the parts as decompositions, and $c = \cup(\{t_1(a_1)\}, \{t_2(a_2)\}, \dots, \{t_n(a_n)\}) = \{t_1(a_1), t_2(a_2), \dots, t_n(a_n)\}$ is the assembled object.

Conclusion

We conclude our grammars for making re-visit by reinforcing Knight and Stiny’s [1] view that the principles on which shape grammars and underlying algebras for shapes are based are powerful enough to handle things and making. Grammars for making were required not only to produce the right result—made object—but also to reflect the making process. To this end, one may look into designing the appropriate rules—as Knight does in [15]—and also into changing the formalism—which we pursued here. The standard replacement rules are altered as well as the way they apply, which results in new types of grammars, quad grammars being the most advanced ones. Quad grammar rule format $a, a' \rightarrow b, b'$ allows for more elaborate sensing and doing than the standard one. On the sensing side, we have $t(a)$ and $t(a')$. The first allows seeing, grabbing, and positioning, while the latter helps in avoiding obstacles and collisions. Shapes $t(b)$ and $t(b')$ add and remove stuff, respectively, and are on the doing side. We introduced transformation rules, which work differently than the standard ones. They just move things around but may also bend or fold them. Some work has been put in defining the equivalence of rules and grammars although very little of that has been used. The scopes of algebras and grammars defined here are different. The latter could handle many types of things while the former are constrained to things produced by machining, laser cutting or 3D printing and assembled into functional objects. Even so, we expected to put more work into algebras. It was clear that Boolean processes like 3D printing and milling could be handled by U_{33} but anticipated a need for something new to handle assembling. However, existing algebras— V_{33} and D_{33} —proved to be up to that challenge.

There are many making processes out there waiting for grammar treatment. Some may require elaborate changes to grammar and rule formats as well as the creation of new algebras. Nevertheless, the approach will stay the same. One should model the process of making by simply capturing the properties of things as makers manipulate them [1].

References

1. Knight TW, Stiny G (2015) Making grammars: from computing with shapes to computing with things. *Des Stud* 41:8–28
2. Stiny G (1990) What is a design. *Environ Plann B Plann Des* 17:97–103
3. Krstic D (2014) Algebras of shapes revisited. In: Gero JS (ed) *DCC'12*. Springer Science + Business Media B. V, pp 361–376
4. Carlson C, Woodbury R, McKelvey R (1991) An introduction to structure and structure grammars. *Environ Plann B Plann Des* 18:417–426
5. Stiny G (2006) *Shape: talking about seeing and doing*. The MIT Press, Cambridge, MA
6. Krstic D (2001) Algebras and grammars for shapes and their boundaries. *Environ Plann B Plann Des* 17:97–103
7. Earl CF (1997) Shape boundaries. *Environ Plann B Plann Des* 17:669–687
8. Krstic D (2017) From shape computations to shape decompositions. In: Gero JS (ed) *DCC'16*. Springer Science + Business Media B. V, pp 361–376
9. Knight TW (1994) *Transformations in design: a formal approach to stylistic change and innovation in visual arts*. Cambridge University Press, Cambridge
10. Stiny G (1996) Useless rules. *Environ Plann B Plann & Des* 23:235–237
11. Stiny G (2012) What rule(s) should I use? *Nexus Network J* 13:15–47
12. Stiny G (1992) Weights. *Environ Plann B Plann Des* 19:413–430
13. Stiny G (1994) Shape rules: closure, continuity, and emergence. *Environ Plann B Plann Des* 21:S49–S78
14. Krstic D (2005) Shape decompositions and their algebras. *AIEDAM* 19:261–276
15. Knight TW (2017) Craft, performance, and grammars. In: 2nd International workshop on cultural DNA

Part VII
Design Processes

Rule-Based Systems in Adaptation Processes: A Methodological Framework for the Adaptation of Office Buildings into Housing



Camilla Guerritore and José P. Duarte

This paper proposes that conversion of redundant office buildings into housing is a viable strategy to respond to both structural vacancy and high demand for new dwellings at an affordable cost, which are widespread problems in the main urbanized contexts.

By using shape grammars, it is possible to encode the principles and rules of the proposed transformation strategy in a general methodology that generates multiple scenarios for the reuse of office buildings as housing. Such an approach can thus play a challenging role in the design phase, allowing a faster and more informed decision-making process.

In particular, it will be discussed the translation of data input into conditions to be embedded in the grammar and some examples for encoding housing requirements and building constraints into the rules of the grammar. Finally, some directions for identifying compositional principles to guide the derivation process and generate the adapted floor plans will be illustrated.

Adaptation: Domain and Barriers in the Actual Practice

Focusing on the built environment and promoting investigations on feasible, short-term rehabilitation strategies constitutes a concrete response to the warning launched by the Inter Governmental Panel on Climate Change [1], Fourth Assessment Report (2007), which states that “over the whole building stock, the largest portion of carbon savings by 2030 is in retrofitting existing buildings and replacing energy using equipment due to the slow turnover of the stock”. In the light of this, it is absolutely crucial to deal with the reduction of mankind’s environmental impact, and this means

C. Guerritore (✉) · J. P. Duarte
Politecnico di Milano, Italy and Pennsylvania State University, State College, USA
e-mail: camilla.guerritore@polimi.it

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_27

also dealing with the adaptation and reuse of existing buildings, with the aim to meet the needs for building-related GGE reductions globally needed.

An important step in this direction is to improve rehabilitation processes of existing, obsolete building stocks.

Within the built environment, commercial buildings play quite an important role, particularly considering the crescent structural vacancy rate of this building type.

This paper, that is part of a wider research [2], focuses on redundant office buildings and their possible reuse as housing, developing a methodological framework for building adaptation. In particular, it is proposed an “across-use adaptation strategy” [3], that is, the reuse of existing buildings that still have a potential lifespan, extending their lifecycle by adapting them to a new function. In addition, “the economic crisis burst in 2008 has produced both a dramatic intensification of affordable rental apartments request and a further reduction of the already small availability of investments to expand the social housing stock” [4]. Transforming obsolete vacant office building into housing could therefore also respond to the high demand for affordable dwellings.

The methodology developed relies on rule-based systems and in particular on the contribution of shape grammars for the definition of the adaptation process. By using a shape grammar [5], it is in fact possible to encode the principles and rules of the proposed transformation strategy into a general methodology that can generate multiple scenarios for the reuse of office buildings. The possibility of quick simulation of different scenarios of conversions for an existing building can offer decision makers a key instrument for identifying the “adaptive capacity” of the building and formulate specific policies for supporting and facilitating the adoption of rehabilitation strategies.

The opportunity of understanding the implications of the multiple scenarios since the very early stages of the design process could in fact induce a more extensive evaluation of refurbishment scenarios rather than simply choose demolition and reconstruction strategies that are economically and environmentally heavier.

The Urban Context of Milan: From a Specific Case to a Broader Application

The urban region of Milan provides a sample-field for the research, because of the massive presence in it of obsolete office buildings whose use needs to be redefined. Through a survey on vacant and abandoned offices that has been conducted in this area [6], it has been identified a building stock classified according to parameters of localization and typological, morphological, constructive, and performative characteristics. Starting from real cases, it has been verified the possibility of classifying them into generic types, described through parametric models, abstract representations of the entire categories. In this way, it is feasible to propose refurbishment actions applicable to all the buildings with the same features in terms of typological

schemas, type and position of the core, total depth of the building and of its bays, type of structure, and so forth.

Figure 1 shows the three building types that are more largely diffused in the region, providing the corresponding simplified schemas of their internal organization.

The building type considered for the development of the grammar methodology is a rectangular-shaped building with a central core and a perimetral structural system (“block and tower type”) particularly widespread in the region of Milan and that urgently requires the development of specific rehabilitation strategies.

To achieve a representative model for an entire category of building types with the same characteristics, the first step is to reduce the floor plan of the case building to its abstract schemata (Fig. 2). The following step is to parameterize the building schemata that serve as initial shape in the following application of the grammar.

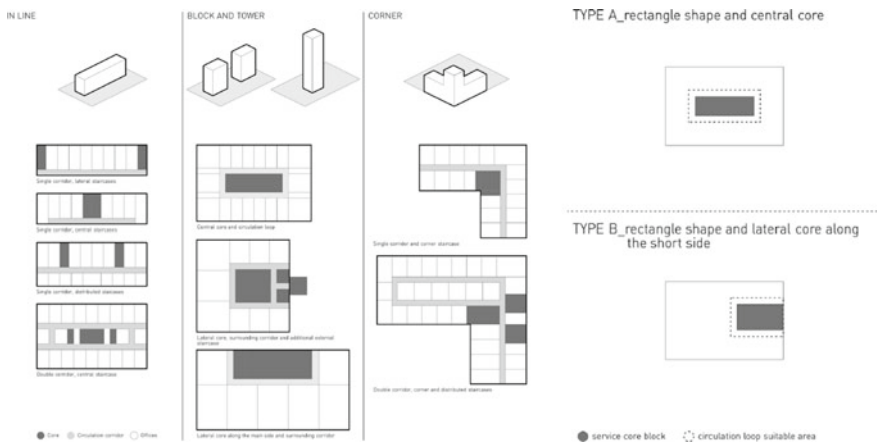


Fig. 1 Some of the building types emerging from the survey, highlighting the floor plan schemata

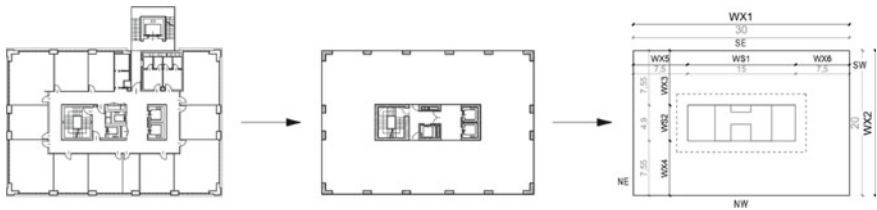


Fig. 2 From the original building to a parameterized schema of its building type

A Grammar-Based Approach to the Conversion of Office Buildings into Housing

The two main questions that guided the development of the grammar-based methodology were: (1) how to use shape grammars in refurbishment projects for existing obsolete building types; and (2) how to infer the grammar with adaptation principles and context-based rules that could be applied not only to the single case but also to a wider sample of buildings—within the same building type located in different urban contexts, thus subject to different constraints and conditions.

The proposed rehabilitation methodology aims to define a spectrum of the feasible scenarios of conversion of an “office building type” into a “housing building type”, taking the following variables into account: the functional programme, the use of volumetric addition for the redesign of the building, the future users and/or developers.

The Office into Housing Adaptation (OHA) grammar, dwells on previous experiences on design grammars for the customization of mass housing [7] and on transformation grammars for housing rehabilitation [8], which stem from Knight’s transformations in design concept [9] and it can offer a valid support to traditional approaches to design and redesign. In particular, the advantage of a grammar-based strategy is twofold: first, adaptation rules and principles can be codified in a more transparent and usable way; second, the grammar can generate multiple designs, thus broadening the range of possible solutions. This second aspect is of great interest in qualitative terms rather than quantitative: in the predesign stage of a reuse process the main goal consists of evaluating the feasibility of different alternatives in order to gauge (i) which design solution is more appropriate for a particular building in a specific context with a predetermined future user/developer; and (ii) which functional programme suits the refurbishment objectives better, also in economic terms.

The algorithmic structure for design encoded in the grammar can, therefore, provide a set of design instructions; these prescribe general principles to be observed in an adaptation process. The principles, in turn, reflect: qualitative aspects (contextual and building qualities); design principles (compositional, functional); sustainable principles (such as solar orientation, demolition strategies); technical principles (structural, energetic) and constraints (regulatory and constructional).

When the design actions, set for a parameterized building type and encoded in the transformation rules, are applied to a specific case, they produce as an effect the customization of the transformation process relative to the specific context, and to its features and constraints.

The methodology for the adaptation of office buildings into housing has been developed as a shape grammar and then converted into a parametric design tool for the ease of a future computer implementation. The automatized execution of the grammar can in fact constitute a powerful design tool that promotes a design-oriented process without limiting the range of admissible solutions to a given formal prefiguration and, at the same time, making evident the spatial relations underlying

a design choice and, possibly, taking into account implications in terms of costs and other aspects such as energetic behaviour.

The power of a grammar-based methodology, with respect to other computational approaches with similar purposes, consists not only in evaluating and testing a number of possible alternative functions in a given form, but in its visual capacity of encoding in a logical system design principles responding to specific strategies. These may provide us with the understanding and description of the existing building as well as express, in the generative rules of grammars, the related knowledge that can be then applied to other buildings of the same type. Finally, the formulated methodology of adaptation can later be translated into a programming language as a practical tool for the exploration of design alternatives.

General Framework for the OHA Grammar

The Office into Housing Adaptation grammar will be organized in two parts: the first one, based on *user inputs*, takes data on the initial building and its context and states the desired brief and design strategy; the second one, based on *generated outputs*, formulates a functional programme responding to the precedent inputs, adapted to the specific building, and generates matching solutions.

More specifically, the *user input process* comprises two phases: (1) creation of general knowledge on the initial building, its urban context and the usable additional surfaces (*Data Collection*); and (2) the *Problem setting* (that comprises *Brief Statement* and *Design Strategy*), through the expression of the desired functional brief and of the design guidelines for the definition of an “ideal transformation pattern” for the rehabilitation.

The generated outputs comprise three phases: (3) *Formulation*, whose objective is the translation of the wished brief into a quantitative programme responding to the real features of the building; (4) *Generation*, that is the part of the design process in which the universe of solutions matching the given inputs is generated; and (5) *Evaluation* to appraise the generated solutions.

At the current stage of the research, it has been developed only part of the *Generation* phase.

Structuring the Grammar

The structure of the OHA adaptation grammar is organized in different levels (Fig. 3), subdivided in stages and steps, that correspond to the various phases of the design process, often independent from one another.

The different levels are described as follows:

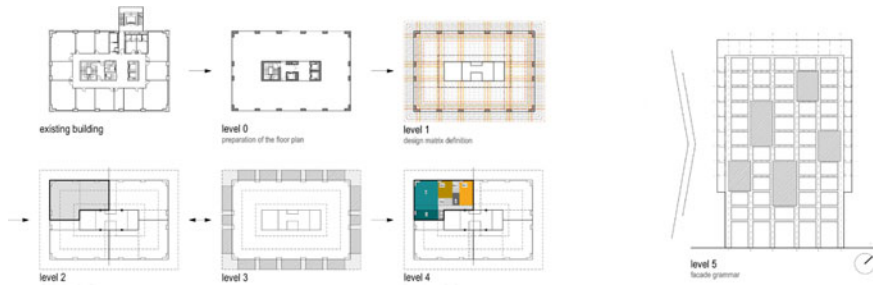


Fig. 3 General structure of the OHA grammar organized into six levels

- (i) preparation of the floor plan (level 0): deals with the definition of a representational system that identifies the existing building components and the characteristics of its context, in order to give a recognizable meaning and a system of coordinates to the constitutive elements of the initial shape;
- (ii) design matrix definition (level 1): to guide the design process according to compositional principles amenable for designers, constraining alignments of walls and assignments of functions to a design grid;
- (iii) division into dwellings (level 2): is meant to explore the capability of the floor plan to accommodate a specific or variable functional programme. Figure 4 shows the stages and steps of this level in detail;
- (iv) volumetric additions (level 3): regulates the spatial configurations deriving from the allocation of the additional surfaces (both new inside spaces and new exterior areas);
- (v) internal organization of the floor plan (level 4): will be used only as means of verification of the appropriateness of adapted floor plan layout (division into dwellings) the deriving by the application of level 2;
- (vi) façade grammar (level 5): it is possible to generate the corresponding facades and three-dimensional model, thereby allowing one to understand the volumetric implications produced by the different refurbishment strategies.

The research, focused on the development of the level 2, the division into dwellings, because the grammar final aim is to define the adaptability at the building scale, exploring the possible arrangements of the floor plan at the variation of the functional programme and of the volumetric additions to be used. The grammar can in fact be a useful instrument for the prefiguration of different transformation scenarios in terms of the spatial capacity of an existing building to accommodate different housing programmes.

Stage	Action	Step
1	Customization of the Design Matrix	1.1: Customizing the Design Matrix to the contextual conditions 1.2: Customizing the Design Matrix to the chosen design strategies 1.3: Change from Stage 1 to Stage 2
2	Circulation Schema and access to the dwellings	2.1: Circulation loop definition (Cl) (if strategy 1) 2.2: Direct access from the core landings (if strategy 2 - to be executed after Stage 4) 2.3: Create transition hallway (if strategy 2 - to be executed after Stage 4) 2.4: Change from Stage 2 to Stage 3
3	Allocation of the dwellings in the programme (D)	3.1: Allocating corner dwellings (if strategy 1 or 2) 3.2: Allocating adjacent dwellings (if strategy 1 or 2) 3.3: Connecting the dwelling with the additions allocated in Level 3 (to be executed after the allocation of each dwelling) 3.4: Re-adapt dwelling dimension after the allocation of the additions 3.5: Change from Stage 3 to Stage 4
4	Assignment of the unused areas	4.1: Assign a function to the remaining surfaces 4.2: Connect the uncompliant 'ua' with the allocated volumetric addition 4.3: Connect the uncompliant 'ua' with already allocated dwelling types 4.4: Re-adapt dwellings dimension after addition of the 'ua' 4.5: Change from Stage 4 to Stage 5
5	Refining of the floorplan layout	5.1: Defining the used corridor (if strategy 1) 5.2: Connecting 'uca' with the allocated dwellings 5.3: Adjusting the dwellings dimension (if strategy 2 - after the execution of step 2.3)
6	Terminates	

Fig. 4 Stages and steps of the level 2 of the OHA grammar

Elements of the Grammar and a Rule in Detail

The adaptation grammar combines different representations devices. These include: (1) lines, which indicate walls, combined on a plane to generate the floor plan; (2) areas, defined by planar surfaces to represent the dwellings areas; (3) labels, which express the attributes of the shapes and sub-shapes; (4) graphs, made of points and lines, which represent the topological relations between different components, and show the adjacency requirements; (5) weights that indicate the properties of the shapes (for example structural features) and identify the acceptable areas for the allocation of specific functions.

The geometrical representation of labelled shapes together with graphs is used in the left and right side of the rules in order to deal with the complex constraints and conditions to be respected in the process of derivation.

The OHA grammar shows how an adapted floor plan can be constructed by applying transformation rules recursively in a step-by-step sequence.

The generation of “appropriate” transformations involves fulfilling of requirements defined a priori in the adaptation methodology. For this reason, rules application needs to be restricted in definite ways for example by associating labels with shapes, thus restricting the kinds of designs produced.

The proposed set of shape rules include different types of instructions according to the transformation to be performed. There are rules for: allocating, dissecting, connecting, extending and reducing rectangles; rules for assigning and modifying functions and rules for allocating position weights.

Figure 5 shows the main simplified shape rules that are recursively applied in the execution of Level 2 of Division into Dwellings.

A complete set of rules specific to the Office into Housing Adaptation grammar is still to be developed.

Figure 6 shows the schematic partial generation of a floor plan that matches a specific programme in accordance with the grammar rules. The derivation applies

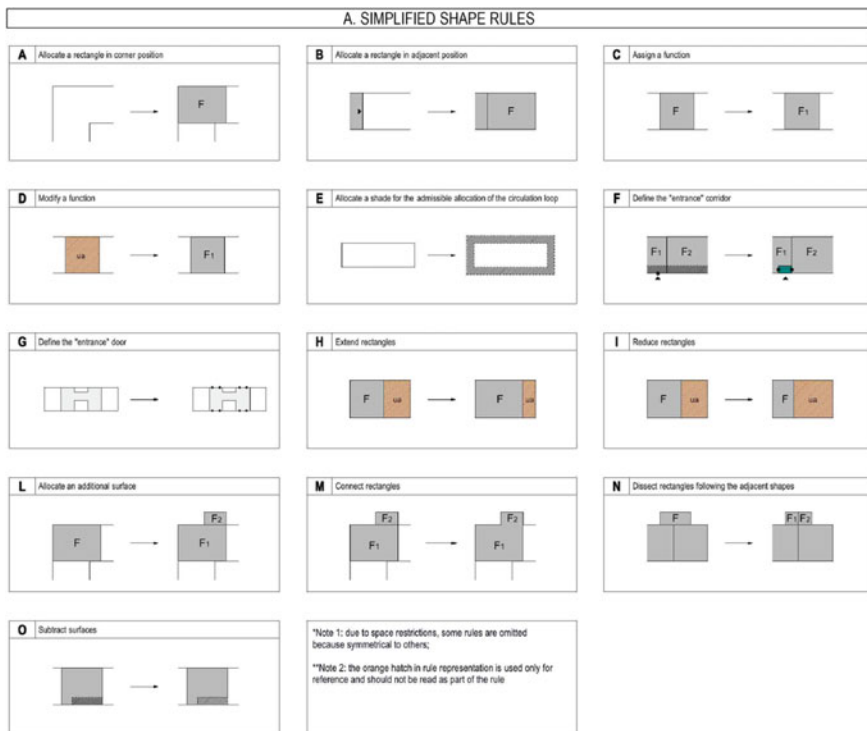


Fig. 5 Simplified shape rules for allocating, dissecting, connecting, extending, reducing rectangles and rules for assigning and modifying functions

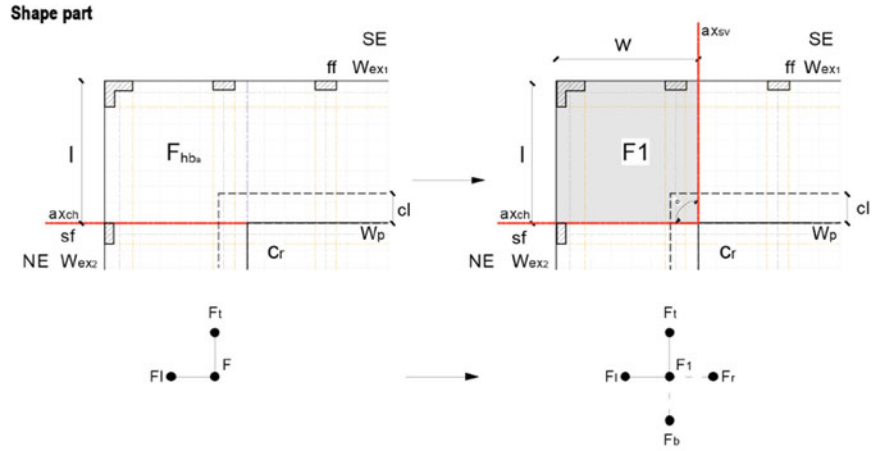


Fig. 6 Simplified partial derivation of a design for a formulated programme

by satisfying the imposed conditions in order to ensure that each dwelling (allocated shape) has a correct definition.

In order to facilitate the understanding of the logic behind the development and structuring of the grammar rules, it is shown the application of a rule in detail.

Each rule is organised in four parts: a shape part (S); a label part (L); a conditional part, which imposes restrictions on functional, dimensional and regulatory aspects; a description part.



Conditional part

$$D \supseteq d(t_n) \wedge D' \not\supseteq d(t_n)$$

Dimensions:

$$Ad(t_n) = (w \cdot l) \wedge 35 \text{ sqm} \leq Ad(t_n) \leq 120 \text{ sqm}$$

$l = \perp$ distance between the W_p and W_{ex1}

$$w \geq 4.5 \text{ m}$$

$$e \in \{90^\circ\}$$

Functions:

$$F_t \wedge F_l \in \{W_{ex2}\}$$

$$F \in \{h_{ba}\} \wedge [\text{adjacent to } (ax_{ch})]$$

$$F_1 \in \{d(t_n)\} \wedge [\text{adjacent to } (ax_v) \wedge (ax_{ch})]$$

$$F_r \wedge F_b \in \{h_{ba}\}$$

Description part

$$R_i \quad \langle D_n: W_{exn}, \{h_{ba}\}, F; D'; E \rangle \dashv \dashv \langle D_n: W_{exn}, \{h_{ba}\}, F_b; d(t_n); D' + \{d(t_n)\}; E - \{h_{ba}\}, E + \{d(t_n)\} \rangle$$

Fig. 7 Example of detailed rule for the allocation of a dwelling type in the corner position: shape part, condition part and abbreviated description part are shown

In Fig. 7, the rule defines the allocation of a dwelling in the corner position and aims at embedding the instructions for the definition of any type of dwelling in the programme [10].

The following section of the paper illustrates how to translate contextual data, such as solar orientation, contextual location, position of the building within the lot, views types and so forth, into conditions that must be respected in the application of the grammar; it also shows how to encode housing requirements, such as dimensional, functional-typological, health and hygiene constraints into rules. Lastly, compositional guiding principles for the division of the floor plan into dwellings are described.

Translating Contextual Data, Housing Requirements, and Building Conditions into Rules

Contextual Aspects

Adaptation processes need to take into account in an accurate way both contextual and physical characteristics of a specific building on which to intervene, which play a key role in the success of the rehabilitation process. The adaptation grammar has to include as preliminary knowledge: (1) boundary conditions; (2) urbanistic regulations (the research is embedded in the Italian context and considers national and local regulations), (3) building features considering functional, constructive and typological characteristics, (4) functional housing requirements following national and local regulations—containing data on the functional and spatial requirements for housing and dwelling organization.

These conditions, which reflect both regulatory aspects and qualitative features, have an important impact on the allocation of the different functions (the dwelling types) in the floor plan, orienting and restricting their positioning.

Four main parameters that affect the assignment of the dwellings were identified: (i) solar orientation; (ii) contextual location of the different elevation of the building; (iii) position of the building in its lot; and (iv) view types and view quality.

Solar orientation (N, NW, W, SW, S, SE, E, NE) of the building is a regulatory parameter of both the allocation of the volumetric additions and the dwelling types. For example, the allocation of dwelling types will follow the following principles (Fig. 8):

- *In line dwellings*, with a single facing, only if the dwelling surface is $\leq 60 \text{ m}^2$ and the single facing is not located on $\pm 30^\circ$ at the North side. According to these restrictions, this group comprises dwelling types T_1 and T_2 *standard*;
- *Double facing* and *Corner dwellings* can be allocated following any possible couple of coordinates for the solar orientation and their prioritization will take place according to other variables such as *contextual location* and *view types*.

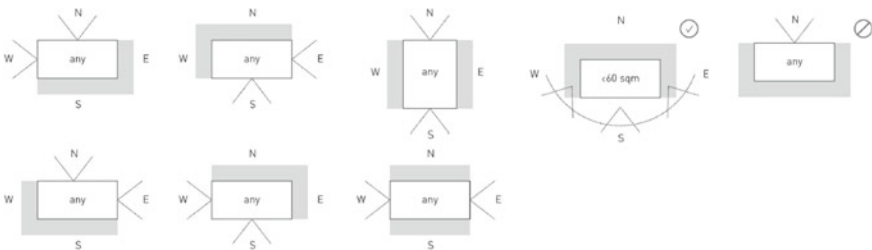


Fig. 8 Solar orientation

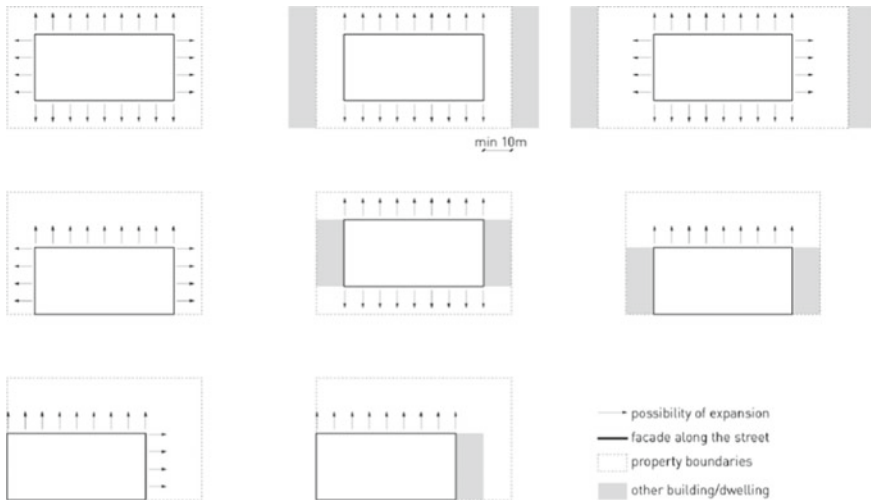


Fig. 9 Position of the building in its lot, highlighting possible directions for expanding the existing building according to the contextual location

Contextual location (front façade, back façade, side façade, side wall) characterizes the different elevations of the initial building according to its collocation with respect to the urban context.

Connoting the front, back or side of the building will provide information for locating dwelling types and organizing the internal subdivision in day, night and service zones in the following phases of derivation.

Position of the building in its lot gives information on the position in plan of the building within its lot perimeter and with respect to the surrounding buildings. Because it regulates the maximum admissible depth of the volumetric additions, this is a key condition in the definition of the admissible external boundaries of the transformed building: it combines data on the distance between the building footprint (including balconies) and the lot limits, the distance to be respected from the surrounding buildings and eventual specific urban restrictions (Fig. 9).

Views types (street view, backyard view) and *view qualities* (good, sufficient, mediocre) are qualitative aspects of the building faces, and are additional parameters for guiding positioning choices of the dwellings types and their internal organization.

Housing Requirements and Building Constraints

Once the data from the context are organized as a system of general conditions on rule application, the following step is encoding in the grammar the information concerning the housing programme. This information involves data on functional and spatial requirements for housing and dwelling organization, to be verified while allocating

a new dwelling. For this purpose, explicit design knowledge can be represented as sets of constraints on dimensional, functional, technical, as well as aesthetic and constructional aspects. To accomplish this task, prescriptions deriving from local and national regulations regarding housing standards have been considered.

Figure 10 shows dimensional and functional conditions related to the different dwelling types included in the grammar, specifying the dwelling capacity (area ranges, number and type of rooms for a given number of users following the regulations) and different room types, showing the room capacity (with, area ranges, etc.) and the adjacency conditions to be respected. The specific constraints developed at the current stage of this research are outlined as follows:

- *Dimensional constraints*: related to areas and linear dimensions for the dwellings and the internal subdivision into rooms. The definition of the actual dwelling types has been established on the basis of the analysis of competition on social housing in the municipality of Milan in the last ten years. Furthermore, it has been identified for each dwelling type a dimensional range for the minimum and maximum admissible surface.
This, together with the proportional ratio between width and length, defines a minimum width of the side of the dwelling adjacent to any exterior walls and a maximum length in order to guarantee adequate lighting and ventilation of the dwellings.
- *Functional-topological constraints*, concerning where to allocate dwellings and the relation between the different zones that compose them—day, night and service areas.
- *Health and hygiene constraints*, regarding issues related to natural lighting and ventilation.
- *Spatial constraints*: concern the positioning of the new functions with respect to some preexisting elements of the floor plan (such as load-bearing structure or shafts that will be maintained in the adaptation process). This constraint has a prioritization role in the positioning of eventual volumetric additions, especially in the case of the ones that use the existing structural system, and in guiding the alignments of the new functions.

The above conditions and requirements and their combined deployment provide the grammar with the contextual qualities and the regulatory aspects that restrict the allocation of the different functions on the initial floor plan. In Fig. 11, it is summarized the translation of these properties into general conditions on rule application in the grammar, that comprises the system of conditions on the relation between contextual aspects and the functional programme to be tested in the initial building.

Dwellings types: dimensional and functional conditions	
Dwelling types	
T ₁ : studio flat	
	$D \supset d(t_1)$ $d(t_1) \in \{t_1, 1_p, 1_r, 1_d\}$ $Ad(t_1) = (w-l) \wedge 35 \text{ sqm} \leq Ad(t_1) \leq 45 \text{ sqm}$ $w \geq 4,5 \text{ m}$ $4,5 \text{ m} \leq l \leq 6,5 \text{ m}$ $F \in \{d(t_1)\}$ $W_{ex} = 2 \vee 1$ $F_l \vee F_b \vee F_r \vee F \in \{w_{ex}\}$ $F_l \wedge F_b \wedge F_r \wedge F \in \{d(t_1), W_{ex}, W_{ex}\}$ $d(t_1) \supset \{li, di, ki, 1, be, 1 \vee be, 2, ba, 1, sr, c\}$
T ₂ : 1-bedroom apartment	
	$D \supset d(t_2)$ $d(t_2) \in \{t_2, 1_p, 1_r, 1_d\}$ $Ad(t_2) = (w-l) \wedge 50 \text{ sqm} \leq Ad(t_2) \leq 70 \text{ sqm}$ $w \geq 4,5 \text{ m}$ $4,5 \text{ m} \leq l \leq 6,5 \text{ m}$ $F \in \{d(t_2)\}$ $W_{ex} = 2$ $F_l \wedge F_b \vee F_l \wedge F_r \vee F_l \wedge F \vee F_b \wedge F_r \vee F_b \wedge F \vee F_r \wedge F \in \{w_{ex}\}$ $F_l \wedge F_b \wedge F_r \wedge F \in \{d(t_2), W_{ex}, W_{ex}\}$ $d(t_2) \supset \{li, di, ki, be, 1, be, 2, ba, 1, ba, 2, sr, c\}$
T ₃ : 2-bedroom apartment	
	$D \supset d(t_3)$ $d(t_3) \in \{t_3, 1_p, 1_r, 1_d\}$ $Ad(t_3) = (w-l) \wedge 70 \text{ sqm} \leq Ad(t_3) \leq 90 \text{ sqm}$ $w \geq 4,5 \text{ m}$ $4,5 \text{ m} \leq l \leq 6,5 \text{ m}$ $F \in \{d(t_3)\}$ $W_{ex} = 2$ $F_l \wedge F_b \vee F_l \wedge F_r \vee F_l \wedge F \vee F_b \wedge F_r \vee F_b \wedge F \vee F_r \wedge F \in \{w_{ex}\}$ $F_l \wedge F_b \wedge F_r \wedge F \in \{d(t_3), W_{ex}, W_{ex}\}$ $d(t_3) \supset \{li, di, ki, st, 2, ba, 2, ba, 1, ba, 2, sr, ha, c\}$
T ₄ : 3-bedroom apartment	
	$D \supset d(t_4)$ $d(t_4) \in \{t_4, 1_p, 1_r, 1_d\}$ $Ad(t_4) = (w-l) \wedge 90 \text{ sqm} \leq Ad(t_4) \leq 120 \text{ sqm}$ $w \geq 4,5 \text{ m}$ $4,5 \text{ m} \leq l \leq 6,5 \text{ m}$ $F \in \{d(t_4)\}$ $W_{ex} = 2$ $F_l \wedge F_b \vee F_l \wedge F_r \vee F_l \wedge F \vee F_b \wedge F_r \vee F_b \wedge F \vee F_r \wedge F \in \{w_{ex}\}$ $F_l \wedge F_b \wedge F_r \wedge F \in \{d(t_4), W_{ex}, W_{ex}\}$ $d(t_4) \supset \{li, di, ki, 3, be, 2, st, ba, 1, cl, ba, 1, 2, ba, 2, sr, la, ha, c\}$

◀**Fig. 10** Representation of the dimensional and functional conditions according to the different dwelling types. Legend of the used labels: Wex = exterior wall; Wcl = circulation wall; F = function; Ff = front side function; Fb = bottom side function; Fr = right side function; Fl = left side function; d(tn)=dwelling types; li.di = living/dining; k = kitchen; k.l = kitchenette; st = study room; rc = recreational room; be.2 = double bedroom; be.1 = single bedroom; ba.1 = main bathroom; ba.2 = secondary bathroom; sr = storage; la = laundry; c = corridor; ha = hall; cl = walk-in closet

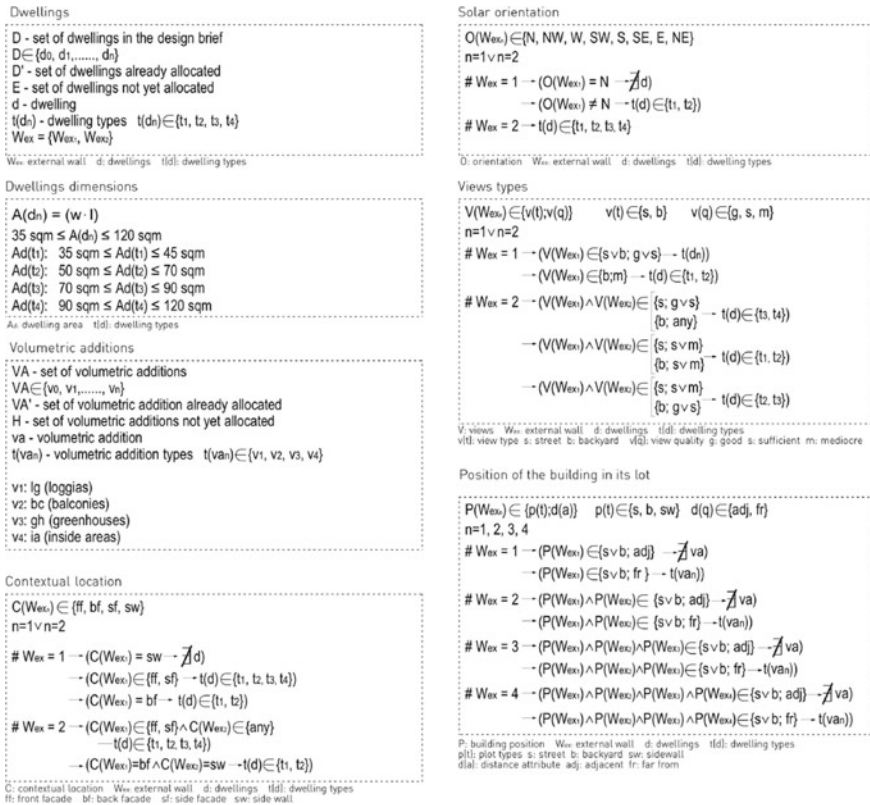


Fig. 11 Translation of contextual data and housing requirements into general conditions of the grammar rules

Defining Compositional Principles

Design Matrix Definition and Guiding Design Principles

Once contextual and building characteristics and constraints were defined and encoded in the rules of the grammar, it was necessary to define compositional principles that could guide the generation of the floor plan starting from a given functional

programme to be allocated within a specific building. These principles have been encoded in a “design matrix”, a system of axis and boundaries that represents both the features of the building initial type and the admissible boundaries for the different design solutions.

Apart from reflecting the characteristics of the initial building, the design matrix defines the metrics for the composition of the new inside and outside spaces; it also defines modules and sub-modules for the organization of the space, and ensures that this results in a rational organization of the floor plan, to be subsequently subdivided internally.

The process of generation of solutions can in fact work as a process of “populating” the design matrix: it introduces and manipulates 2D rectangular shapes—the dwelling types and the eventual volumetric additions. This can be regulated by constraining the position of the design components on the grid of the matrix.

Rules for connecting and extending rectangles, as well as for assigning and changing the functions associated with them, had to be defined to control the generation of the adapted floor plan.

By means of a dedicated set of rules for customizing the design matrix, the external boundaries will be modified to accomplish with the contextual regulations of the site, thus yielding a preliminary definition of the admissible perimeter of the project.

The allocation of the different functions will then proceed by constraining their dimensional parameters to the matrix lines, snapping whenever defining a shape its endpoints to the grid lines, to locate and proportion the new functions.

Subsequently, some principles for the population of the design matrix, following different compositional strategies, were identified (Fig. 12).

In particular, there were identified four main strategies: (1) *contiguity*, that consists in the allocation of the first dwelling and all the other dwellings in the programme are subsequently allocated adjacent to the one previously allocated; (2) *reference lines*, based on the preliminary definition of hierarchical elements on the existing floor plan; (3) *symmetry*, that constrains the distribution of dwellings within the building boundaries in a symmetrical way by respecting the symmetry axis; and (4) *prioritization of the corner position*, concerns the allocation of the dwellings included in the functional programme in descendant order from the bigger to the smaller dwelling types, first by occupying the corner positions of the building and only then the inner positions with a single facing (Fig. 13).

Conclusions and Future Developments

Developing a grammar-based approach for rehabilitation processes represents the possibility of systematizing a methodological framework for adaptation of existing buildings to new functions into a codified system.

At this stage of the work, it has been outlined a first draft of the grammar and there have been identified contextual data, housing requirements and design principles to be encoded in the adaptation grammar for converting office buildings into housing.

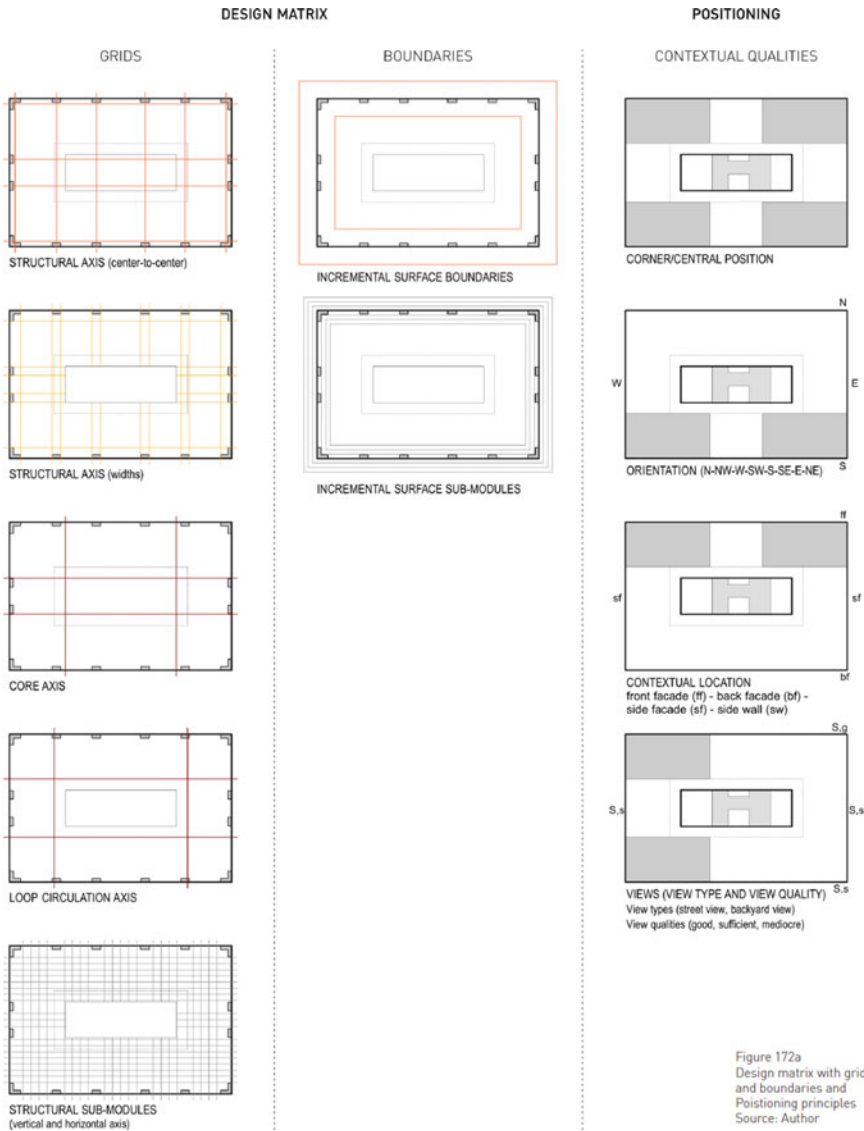


Figure 172a
 Design matrix with grids and boundaries and Positioning principles
 Source: Author

Fig. 12 Design matrix with grids, boundaries and positioning principles

With such a system of contextual conditions, functional requirements, regulations and design principles, it is possible to define a priori the principles and rules for allocating new functions on the initial floor plan, considering both quantitative and qualitative variables. These principles and rules constitute a systematic methodological process that incorporates all the valid transformation rules for a given building.



Fig. 13 Compositional principles of contiguity, reference lines, symmetry, and corner prioritization

Additional parameters may consider also the three-dimensional space, such as height or volume. These can be successively added as new constraints can be imposed on added parameters. Furthermore, a more extensive grammar could be obtained comprising, among others, aspects related to constructional requirements, fire regulations, energy efficiency, and allocation of technical installation. To pursue this goal, a considerably more complex set transformation rules, in terms of shapes and conditions, will have to be developed.

Future work will need to focus on the full development, refinement and validation of the grammar.

The computer implementation is the last objective, crucial to achieve the most important promises of the proposed grammar-based approach, namely the fast generation of design alternatives adequate to specific design contexts.

Acknowledgements This paper is part of a Ph.D. research developed at Department of Architecture, Built Environment and Construction Engineering of the Politecnico di Milano. The grammar part was developed at the DCG FA/Ulisboa.

References

1. Intergovernmental Panel on Climate Change (IPCC) (2007) Fourth assessment report [online]. Available at: www.ipcc.ch/publications_and_data/ar4/wg3/en/ch6-ens6-es.html. Accessed on Feb 2017
2. Guerritore C (2017) RE(ho)USING. A methodological framework for the adaptation of office buildings into housing. Doctoral thesis, Politecnico di Milano (ABC Department), Italy
3. Wilkinson SJ, Remøy HT, Langston C (2014) Sustainable building adaptation: innovations in decision making. Wiley-Blackwell, Hoboken
4. Antonini E, Gaspari J, Olivieri G (2011) Densificare per migliorare: strategie di riqualificazione del parco italiano di edilizia abitativa sociale (Densifying to upgrading: strategies for improving the social housing built stock in Italy), *Techne* no. 04. University Press, Firenze, pp 306–314. Available at www.fupress.com/techne, Accessed on Feb 2017
5. Stiny G (1980) Introduction to shape and shape grammars. *Environ Plann B* 7:343–351
6. Guerritore C (2017) ‘The sample in the urban region of Milan’ part III in RE(ho)USING. A methodological framework for the adaptation of office buildings into housing. Doctoral thesis, Politecnico di Milano (ABC Department), Italy
7. Duarte JP (2001) Customizing mass housing: a discursive grammar for Siza’s Malagueira houses. Doctoral thesis, Massachusetts Institute of Technology, Cambridge, E.U.A
8. Eloy S (2012) A Transformation grammar-based methodology for housing rehabilitation: meeting contemporary functional and ICT requirements. Thesis submitted for the Degree of Doctor of Architecture, IST, TU Lisbon
9. Knight TW (1994) Transformations in design. A formal approach to stylistic change and innovation in the visual arts. Cambridge University Press
10. Guerritore C, Duarte JP (2016) Manifold façades. A grammar-based approach for the adaptation of office buildings into housing. In: Aulikki H, Osterlund T, Markkanen P (eds) Complexity and simplicity: proceedings of the 34th international eCAADe conference, Oulu, Finland, 24–26 Aug 2016, vol 2. eCAADe (Education and Research in Computer Aided Architectural Design in Europe) and Oulu School of Architecture, University of Oulu, Brussels, pp 189–198

Using Argumentative, Semantic Grammar for Capture of Design Rationale



Raymond McCall

Efforts to use design rationale (DR) to improve design have been frustrated by difficulties in capturing such rationale in a format structured by a DR schema, such as PHI, QOC, or DRL. These difficulties disappear when rationale is captured as unstructured transcripts of communication among collaborating designers, but the lack of structure in such transcripts severely hinders retrieval. This problem can be solved by parsing transcripts of collaborative design using an argumentative, semantic grammar to produce PHI-structured DR. The ASGARD (Argumentative, Semantic Grammar for Analysis of Rationale for Design) software uses this technique to extract structured DR from transcripts of collaborative architectural design. Preliminary tests were made to see if ASGARD could successfully model and parse three collaborative design transcripts. Results of these tests suggest that this approach has promise for automating DR capture.

Introduction

Design rationale (DR), i.e., records of the reasoning that designers use in devising artifacts, can aid design in two ways. The first is by enabling the authors of the rationale to recall and critique their own reasoning; the second is by enabling other people to understand the reasoning underlying design. Because of the latter, DR can support participatory design [1], collaborative design [2], maintenance of designed artifacts [3], and learning from past design projects [4]. Since the 1970s, devising means of capturing and using design rationale has been the central topic of the field that has come to be called *design rationale research* [5].

R. McCall (✉)
University of Colorado, Boulder, USA
e-mail: mccall@colorado.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_28

519

Unfortunately, the efforts of DR researchers to aid design have been frustrated by profound difficulties in capturing design rationale [6–8]. More specifically, the difficulties have been associated with attempts to record DR in the sort of structured format required for effective retrieval of the rationale when it is needed [6].

Perhaps the most important question facing the field of design rationale research is whether the DR capture problem can be solved. Without the ability to readily capture rationale in a retrievable form, there will be little use for the ongoing research on rationale management and retrieval. There appears to be broad consensus that the conventional approach of recording rationale in a form structured by a rationale schema has simply not worked well enough [6–8]. The question is whether there is some alternative approach that eliminates or at least mitigates the capture problem.

It has been proposed that the difficulties of capture could be mitigated by first capturing the verbal communication amongst collaborating designers as simple transcripts and then having people incrementally structure them over time [9]. This proposal has the advantage that due to the rapid progress in voice recognition technology and the ubiquity of recording devices such as phones and computers, transcripts of design sessions are likely to become easy and inexpensive to make [10, 11]. Its disadvantage is that the structuring requires a great deal of human skill, effort, and time.

This paper describes a new and fundamentally better approach to structuring transcripts of collaborative design discussions. Specifically, it describes ASGARD (Argumentative Semantic Grammar for Analysis of the Rationale for Design), an approach for automated analysis, and capture of structured DR. The ASGARD software inputs a transcript of a collaborative design discussion and outputs structured rationale.

The ASGARD software first identifies the types of rationale elements that sentences in a transcript represent, and then links the sentences together to form structured DR. ASGARD identifies rationale elements using a parser that implements an argumentative, semantic grammar based on an augmented version of the PHI (Procedural Hierarchy of Issues) DR schema [12], a variant of IBIS. ASGARD then uses its linker to connect the sentences using an augmented version of the PHI linking schema.

This paper begins by briefly summarizing the history of DR research and the notorious DR capture problem. It then explains why rationale must be structured for effective retrieval. Following this, it outlines a strategy for attacking this problem by extracting structured DR from transcripts of discussions among collaborating designers. It then explains the ASGARD approach for automatically detecting, extracting, and structuring design rationale. It next describes a preliminary test of ASGARD's capabilities by using it to extract structured DR from three transcripts of collaborative design discussions by student designers. The paper lists the results of this preliminary test and discusses their significance. It then discusses the potential value of ASGARD for DR capture. Finally, the paper describes future work that is needed to realize this potential.

Design Rationale Research and the Capture Problem

Design Rationale Approaches and Software

Horst Rittel felt that design problems are what he called *wicked problems* [13, 14]. To deal with such problems, he devised Issue-Based Information System (IBIS), as a schema for documenting design rationale [15]. IBIS models design discourse as a graph structure where there are four types of nodes that Rittel labeled *issues*, *positions*, *arguments*, and *decisions*. *Issues* are design decision tasks represented as questions. *Positions* are proposed answers to those questions. *Arguments* describe pros and cons of the positions, and *decisions* indicate which positions are ultimately accepted for the various issues. These nodes are linked together in various ways. Arguments link to the positions they discuss. Positions link to the issues they respond to. And issues link to other issues that they help to answer.

Rittel's work pioneered the field of *design rationale research* [16]. For forty-plus years, DR researchers—this author included—have created various DR schemas and software for capture and use of DR. The schemas include PHI [13], QOC (Questions, Option, and Criteria) [17], DRL (Decision Representation Language) [18], and RATSpeak [3]. The software includes PROTOCOL [19], MIKROPLIS [20], gIBIS [21], PHIDIAS [22], SEURAT [3], DRIM [23], DReD [24], and Compendium [25].

The Capture Problem and the Need to Structure Rationale

Since Rittel's work on IBIS, the standard approach to DR capture has been to record rationale in a structured format dictated by a DR schema. While there have been some conspicuous successes with this approach, overall the results fall far short of the expectations of DR researchers. Despite all of the work done on rationale schemas and software, the capture of substantial amounts of structured DR remains difficult and rare.

The standard approach to DR capture has failed to harvest adequate amounts of rationale [7, 8, 26, 27] because designers and others have generally been unable or unwilling to record rationale in structured form [9, 26, 27]. The reason for this reluctance appears to be that structuring DR is difficult and time-consuming, while providing designers with no immediate payoff. In addition, it interrupts and thus disrupts the design process [9, 27].

Analysis of the Problem

Since structuring DR impedes its capture, it is crucial to ask whether we really need rationale to be structured. The answer, unfortunately, is “yes”. Without structure,

DR cannot be effectively retrieved. Retrieval of DR requires not only content-based indexing but also associative indexing—i.e., indexing by links to other rationale. Structuring DR creates these links.

For example, if collaborating designers are dealing with a decision task—called an *issue* in PHI—they should be able to retrieve all the rationale useful for making that decision. They would need to retrieve the proposed decision alternatives—called *positions* in PHI—for the issue. To evaluate these positions, they would need to retrieve the discussion of their pros and cons—called *arguments* in PHI. To know whether the positions are accepted or rejected, they would need to retrieve the decisions on these positions—called *decisions* in PHI. Such retrieval can only happen if arguments and decisions are linked to the positions they evaluate and positions are linked to the issues they respond to. Without structuring, DR has no such links.

In summary, the central dilemma is that structuring rationale during capture hinders capture of DR, but the structure is required for retrieval of DR.

Solution Approach

One approach to solving the capture problem is to capture DR in an unstructured fashion and then structure it later. But if designers do not want to structure their own DR, how does it get structured? An approach that has worked in some of the successful DR capture projects has been to have DR experts extract and structure the rationale from unstructured transcripts of design discussion. But this approach has not sufficed to make DR capture widely used in design projects.

The idea proposed by Shipman and McCall [9] was to create unstructured transcripts of collaborative design and then subsequently structure the rationale using a computer-supported human process of gradual, incremental formalization over a relatively long time, perhaps weeks or months. The alternative suggested here is to have the computer automatically extract structured rationale from such transcripts using an argumentative semantic grammar. If successful, this approach would make it possible to structure rationale rapidly without requiring designers to do it.

A few attempts have been made to capture DR automatically. Myers, Zumel, and Garcia [8] captured rationale in the form of actions that designers performed with CAD systems, but they did not record reasons for these actions. This corresponds to recording positions in PHI. Mix, Jensen, and Ryskamp recorded design actions together with “inferences” of the designers [28]. This appears to correspond roughly to the recording of positions and arguments in PHI. Unfortunately, the authors did not explain how they structure the rationale. Finally, Schneider has shown that DR can be captured as a side effect of using a highly structured design method [26]. Of course, the method might itself require extra effort by designers. This paper proposes a method for capturing DR without any such extra effort.

The ASGARD Approach to Analysis and Capture of Rationale

The idea for ASGARD originated when a colleague asked for a list of rules for categorizing rationale according to PHI. The author wrote a list of such rules and found that they took the form of rules of replacement—in effect, a grammar. But this grammar was unlike the grammars of linguists, for it made no use of syntactic categories, such as *NOUN-PHRASE*. Research revealed that it was a *semantic grammar* [29], which uses semantic categories, such as *FIRST-FLOOR*. Unlike other semantic grammars, however, the PHI-based grammar uses categories describing argumentative structures in sentences, such as *CONDITION*. The PHI rules constituted an argumentative, semantic grammar—apparently the first of its kind.

The initial grammar was not intended to be comprehensive. Later it became clear that a more complete set of rules might be used to detect, categorize, and link the DR resulting from informal discourse among collaborating designers, thus providing an automated approach to extracting DR from transcripts of design communication—here called ASGARD.

A central goal of this project is eventually to use the rationale ASGARD extracts from design transcripts as input to PHIDIAS—i.e., the PHI-based Design Intelligence Augmentation System—which is designed to manage large collections of PHI rationale and deliver it to designers when they design [22]. Ultimately, however, ASGARD could supply DR to any software system that can manage PHI-based rationale.

The ASGARD Parser

The ASGARD parser uses simple depth-first, recursive traversal to parse its context-free grammar. It does no optimization, such as dynamic programming. The parser's one unconventional feature is the ability to skip words in the sentences parsed. This is needed in part because natural language DR is “noisy”, in the sense of containing words that are not relevant as rationale. This is especially true for transcripts of spoken rationale, as is shown by the following statement from one of our transcripts of collaborative design: “I think like maybe in this section over here we’ll have like all year-round plants.” ASGARD skips the two “like” words in the sentence, since they play no role in determining that the sentence is a position.

The other reason for skipping words is that even when the words represent relevant rationale, many sentences can be correctly categorized by type of rationale element—such issue, position, argument, and so forth—without having to look at all the words in the sentence.

The ASGARD Grammar

A context-free grammar is defined here in the conventional fashion as consisting of four elements: a start symbol, a terminal vocabulary, a nonterminal vocabulary and a set of rules of replacement [30]. The start symbol represents the sentence as a whole. The terminal vocabulary contains words that are actually used by people in sentences. The nonterminal vocabulary contains terms that are not used in ordinary speech or writing, but which categorize parts of sentences.

The ASGARD software currently uses grammatical rules of replacement represented in simple (non-extended) BNF form. The rules indicate alternative ways in which nonterminal terms (or *nonterminals*) can be replaced by strings containing terminal terms (or *terminals*) and/or other nonterminal terms. Because the nonterminal terms of the ASGARD grammar are not the familiar syntactic terms of conventional grammar, the nonterminals are written in the form of long strings that attempt to convey the meanings that they represent. The hope here is that this “self-documenting code” strategy will make the rules easier to understand.

BNF representations of grammars typically write the nonterminal terms in uppercase letters and write the terminal terms in lowercase. For the domains of building design and software design, however, it is useful to allow uppercase letters in terminals on occasion—for example, to represent the terminal words HTML (Hypertext Markup Language) and HVAC (Heating, Ventilation and Air Conditioning). As a consequence, in addition to writing nonterminals in uppercase, ASGARD represents them with an initial @ symbol to distinguish them from uppercase terminals.

Positions are statements that propose some sort of design feature, i.e., the creation of some object, property, relationship, or functionality in the artifact being designed. Here are some very simple examples of positions drawn from a transcript of collaborative design of a lunar habitat:

1. The crew quarters will have to be bigger.
2. This area might require some partitions.
3. Crew members need to be able to fold up the table.

To parse these position sentences, ASGARD could use the BNF form rule shown below which designates three alternative replacements for the nonterminal term “@POSITION”.

```
@POSITION ::=
```

```
  @SPECIFIED-DESIGN-OBJECT @MUST-HAVE
  @NEW-ATTRIBUTE-VALUE |
```

```
  @SPECIFIED-AREA @COULD-NEED
  @QUANTITY-OF @DESIGN-OBJECTS |
```

```
  @USERS @MUST-BE-ABLE-TO @MANIPULATE
  @SPECIFIED-DESIGN-OBJECT
```

Each of the three replacements shown above can parse one of the three sentences shown above it. Thus, the first sentence would be parsed using the first replace-

ment, which contains nonterminals that correspond to different parts of the sentence. “The crew quarters” correspond to the “@SPECIFIED-DESIGN-OBJECT” in the first replacement. The “will have to be” part of that sentence corresponds to the “@MUST-HAVE” nonterminal. Finally, “bigger” corresponds to “@NEW-ATTRIBUTE-VALUE”.

It should be noted that it is only by means of rules of replacement that ASGARD has any knowledge of what the nonterminal categories “mean”. For example, ASGARD only knows what the term @POSITION means by searching the list of replacement strings that the rule contains. The rules of replacement in an ASGARD grammar are often large and complex for the simple reason that they contain large lists of possible replacements for each nonterminal. The example shown above of a rule for replacing @POSITION is in fact far too simplistic to be of use in a practical context.

Example of an ASGARD Parse of an Issue

One of the project goals was to see to what extent, if any, the informal design rationale found in collaborative design discussions contains the elements and links used in the PHI approach to IBIS. The idea was to test the validity and completeness of that schema against “rationale in the wild”.

To understand what ASGARD is and how it works, it is useful to look at examples of how it parses an issue, a position, and an argument. These are by no means the only types of rationale elements needed to parse transcripts of collaborative design, but the examples provide a sketch of how ASGARD identifies rationale elements.

In IBIS and PHI, a design task is represented as the process of answering a design question, called an *issue*. Here is a simple issue from the transcript of collaborative design of a Mars habitat: “What if there was an emergency?” Listed in Fig. 1 is the ASGARD parse tree for this question. Note that this tree is shown in outline format rather in the more traditional tree-graph format. Outline format is more convenient when using long names for nonterminals.

Since ASGARD only aims to identify and categorize DR sentences, the top-level node in its parses is always @RATIONALE-ELEMENT—shown in Fig. 1. Indented beneath this is the type of DR element detected by ASGARD, in this case, @ISSUE. ASGARD detects whether a sentence contains rationale and, if so, what type of rationale-element by seeing whether it can parse the sentence. If the software fails to parse a sentence, it assumes that that sentence contains no rationale. Of course, this strategy only works if ASGARD can parse all the sentences that it encounters that do contain DR—currently an aspirational goal.

Note that the lowest level nonterminals are variables that can have values that are concatenations of multiple words—for example, the @YOU-HAVE nonterminal has a value of “there was”. Taken out of context it might seem that @YOU-HAVE is unrelated in meaning to “there is,” but *within the context of the given higher level phrases*, these two things are semantically equivalent. In effect, “there is” is one way

Fig. 1 An example of an ASGARD parse of an issue

```

SENTENCE TO PARSE:
'What if there was an emergency?'

PARSE TREE:
@RATIONALE-ELEMENT
  @ISSUE
    @WH-DIRECT-QUESTION
      @WHAT-IF
        What
        if
      @SOMETHING-HAPPENED
        @SOMETHING-BAD-HAPPENED
          @YOU-HAVE
            there
            was
          @A
            an
          @BAD-EVENT
            @EMERGENCY
              emergency

```

of expressing the concept “@YOU-HAVE” within the context of the higher level phrases.

Again, note that the labels for nonterminals are meant to make the meaning of different levels of the parse tree more self-explanatory. Thus, for example, level 2 of the above parse tree reads as “@WHAT-IF @SOMETHING-HAPPENED”. This is in effect a more general account of what the sentence means than the actual sentence, “What if there was an emergency?” We could also read the sentence as a mix of levels, for example, “@WHAT-IF” @ YOU-HAVE @A @BAD-EVENT”. This is more specific than “@WHAT-IF @SOMETHING-HAPPENED” but more general than “What if there was an emergency?” The point is that, wherever possible, the intent was that the sentence should be meaningful at each level of specificity/generality in the hierarchy of nonterminals.

Example of an ASGARD Parse of a Position

In ASGARD, a position is defined as a proposal for the creation of some feature of the artifact being designed. An example of a position from a transcript of the collaborative design of a Lunar habitat is as follows: “We’re going to have to provide a space for another workstation”. Figure 2 shows the parse of this sentence.

The second line of the parse shows that ASGARD has correctly identified this as a position. It should be noted that the structure of this position contains an introductory phrase, “We’re going to have to,” that corresponds to “@PERSON-MUST”, which makes it highly likely that this sentence is a position. Following this is a

SENTENCE TO PARSE:**'We're going to have to provide a space for another workstation.'****PARSE TREE:****@RATIONALE-ELEMENT****@POSITION****@PERSON(S)-MUST**

We're

going

to

have

to

@DESIGNER-ACTION**@CREATE**

provide

@A-PLACE-FOR

a

space

for

@SPECIFIED-DESIGN-OBJECT**@SPECIFIER-SINGULAR**

another

@DESIGN-OBJECT

workstation

Fig. 2 An example of an ASGARD parse of a position

@DESIGNER-ACTION indicating some action that the designers are supposed to take to modify the design of the artifact so that it has a certain feature.

Example of an ASGARD Parse of an Argument

Figure 3 shows the ASGARD parse of an argument from the transcript of the lunar habitat design. Once again, we see that this sentence contains an introductory phrase that suggests what type of rationale element it is. Such indicators are common in arguments. In this case, the indicator is **@INTRO-TO-ARG-AGAINST**, that is, an introduction to an argument against something, typically against a previously proposed position or argument.

The ASGARD Linker

The linking of sentences is based on three things. One is a PHI DR schema that has been extended to deal with the additional rationale elements that were found in the

Fig. 3 An example of an ASGARD parse of an argument

SENTENCE TO PARSE:
'The problem is these workstations are very public'

PARSE TREE:
@RATIONALE-ELEMENT
@ARGUMENT
@INTRO-TO-ARG-AGAINST
 The
 problem
 is
@OBJECT(S)-HAS/HAVE-PROPERTY
@SPECIFIED-OBJECT(S)
@DEICTIC-SPECIFIER
 these
@DESIGN-OBJECT(S)
 workstations
@IS/ARE
 are
@DEGREE-OF-PROPERTY
@DEGREE-OF
 very
@PROPERTY
 public

three transcripts of collaborative design. A DR schema is a graph that shows which rationale elements can be linked together as well as which links can be used to connect these elements. The schema used was developed as part of the study described below that attempted to use ASGARD to extract structured design rationale from transcripts of collaborative design sessions.

Also used in constructing the larger structures of DR are indications of whether a sentence is stated as a direct response to a statement recently preceding it. One such indication is the type of rationale element involved. Arguments are most frequently given in response to the most recently proposed position. Judgments and agree/disagree statement (both of which are defined below in the description of the study) are typically stated in response to the sentence immediately preceding them.

Finally, a connected graph of linked rationale elements often deals with a named topic, such as a type of design object, a floor in a building, a type of building use—and so forth. Commonalities between sentences are typically indicated by use of the same word or related words in those sentences. ASGARD linker detects these from inspection of the parses. These can be used to modify the order of display of the rationale so that similar discussions are adjacent in the display of the rationale structure.

Test and Demonstration of the ASGARD Approach

The Goals and Approach for Testing ASGARD

Fully evaluating ASGARD would require determining whether it can successfully extract DR from design transcripts that were not used to create the ASGARD grammar. Unfortunately, it is far too early in the history of the project for such testing. What needs to be done first is to test whether an ASGARD grammar can be created that can extract structured DR from actual transcripts of collaborative design. Passing this test would constitute a weak proof-of-concept that showed the plausibility of ASGARD.

Getting computers to understanding everyday natural language is a notoriously error-prone task. The idea motivating the preliminary testing of ASGARD was that while it could not prove that the approach would work on a large scale, it might very well prove that the approach would not work even on a small scale. In other words, the testing might falsify the hypothesis underlying the ASGARD approach.

An additional goal of the testing was to find out to what extent the elements of the PHI DR schema are found in the informal discussions of collaborating designers. A further goal was to discover what elements and relationships might be missing from PHI and other DR schemas.

The Tests of ASGARD

The testing involved an attempt to build a single ASGARD grammar that could extract structured DR from the transcripts of three collaborative design sessions, each transcript being from a different group of students at the University of Colorado, Boulder. The students were all seniors in the Environmental Design Program, and the transcripts were made in courses where they studied the nature of design processes in various ways, including by making and analyzing records of their own efforts at collaborative design. None of the students whose transcripts were used had any knowledge of design rationale at the time they recorded their design work. They also had no knowledge of how the transcripts would be analyzed.

The collaborative design projects that the three groups of students undertook were as follows: One was the project for design of a lunar habitat undertaken by Group 1, a pair of seniors in the Environment Design Program. This was a semester-long project undertaken in 1998 and the transcript used here is from a one-hour session of that project. That session involved detailed design of a work area in the habitat and the furniture for that area. A video recording was made by the students of this session. This author made a transcript of the conversation in that recording.

The second project was a collaborative design effort undertaken in 2014 by Group 2, a pair of seniors in environmental design. But in the session that the transcript covers, only one of the students is actually designing and the other is merely listening

to and talking to that person about the design. The project they discuss is the redesign of the Temple Grandin School for autistic children in Boulder, Colorado. The session recorded on video dealt with the design of an outdoor play area next to the school. One of the students made a transcript of the conversation in the video of that project.

The project by Group 3 was the design of a Mars habitat for the first astronauts exploring Mars. This project was undertaken nearly 15 years ago as a test of the usability of a commercial software package for collaborative design at a distance. The software was the Groove system, which provided basic functionality for a shared drawing board and text-based communication. Group 3, a team of three seniors in environmental design, undertook this as their semester project for a class. The session used here dealt with the overall layout of that habitat. The students were at three separate locations. The transcript of that session was produced automatically by the Groove software from the text-based communication among the students.

Results of the Tests

Constructing an ASGARD grammar for extracting structured rationale from the above-described student efforts at collaborative design revealed that most of the IBIS-PHI rationale element types were also found in all three transcripts. The one type of PHI rationale element missing in all three transcripts was @DECISION. Also found in the transcripts were other types of rationale elements not found in any published rationale schema—including such things as @AGREE/DISAGREE statements and @JUDGMENTS. Examples of the former include the following:

- I agree
- Of course
- Fine with me
- I doubt it
- No way
- I disagree.

@AGREE/DISAGREE statements are similar to the decisions in PHI, which are statements that reject or accept positions. But in the transcripts, @AGREE/DISAGREE statements appear as individual opinions and are seldom explicitly ratified or even discussed by the group.

@JUDGMENTS are special types of arguments that indicate approval or disapproval of positions but do not give any reasons for these judgments. These are, again, very common in the transcripts. Examples include the following:

- Good idea
- Seems logical to me
- I feel the height is OK
- I think the height is problematic
- Having it on the third floor seems best

- I don't like it.

The ASGARD parser handles judgments differently from full arguments, and it seemed useful to exploit this difference to point out where reasons are and are not being given for opinions. So, @JUDGMENTS were not counted as @ARGUMENTS in the ASGARD parses.

Another discovery was that in the transcripts, positions were very often stated as questions. In PHI, any question automatically counts as being an issue. If some of the positions are parsed as issues and others are parsed as positions, it is hard to get a sense of what all the positions are. This problem was typically handled in PHI by simply choosing one of these two forms as the only form in which positions were to be stated. But this strategy simply will not work with the parsing of the transcripts. A hybrid rationale element called the @POSITION-ISSUE handled this case. This separates out the positions stated as questions from the other issues.

Two other types of issues were found in the transcripts that were not accounted for in PHI. One was that of the *process issue*, defined as a question about what issues the group should deal with. The typical response to a process issue is one or more proposed issues.

Arguments were also found to be more varied in the transcripts than is suggested by PHI. One example is conditional arguments, that is, arguments whose assertions depend on some stated condition being true. There are also occasionally some complex compound arguments. Perhaps the biggest surprise was that descriptions of how users might use features of a design were used as arguments for those features. PHI had not anticipated this type of argument, which requires very different parsing rules.

Table 1 shows the types of sentences that ASGARD detected in the transcripts from the three groups of collaborating students. Figure 4 shows typical output from the ASGARD Linker. Here one page of rationale is shown from the group that designed the Mars habitat.

Discussion of the Results of the Test

The results shown in Table 1 indicate what ASGARD found. All of these things were correctly categorized by ASGARD. No rationale sentences were missed and no non-rationale sentences were incorrectly categorized as rationale elements. These results suggest strongly that informal design discourse is highly structured. The structure extends both to the internal organization of the individual sentences and to the relationships (links) between sentences.

Table 1 Types of elements found by ASGARD in the three transcripts

Sentence information	Group 1 transcript	Group 2 transcript	Group 3 transcript
All sentences	79	91	86
Non-DR sentences	22	13	21
DR sentences	57	78	65
Issues	12	12	11
Process issues	0	0	1
Proposed issues	0	0	1
Positions	15	46	12
Conditional positions	2	0	0
Position issues	0	0	1
Arguments	12	14	6
Conditional arguments	1	0	2
Compound arguments	0	4	0
Multi-sentence arguments	0	0	1
User behavior arguments	1	2	0
Judgments	10	0	21
Agree/disagree statements	2	0	6
Decisions	0	0	0
Other DR elements	2	0	3

Conclusions and Future Work

This paper described ASGARD (Argumentative, Semantic Grammar for Analysis of Rationale for Design), a computer-based technique for automatically capturing and structuring design rationale (DR). The goal of the ASGARD project is to overcome the notorious DR capture problem, which has become a serious obstacle to the use of DR to improve design. The paper argued that the capture problem is the consequences of designers' reluctance to record their rationale in the structured form dictated by a rationale schema. ASGARD avoids this problem, first, by capturing DR in unstructured form from transcripts of collaborative design and, second, by using an argumentative semantic grammar to structure the captured DR.

A preliminary test of plausibility of ASGARD was conducted by determining whether a single grammar could be constructed that was capable of extracting structured DR from three transcripts of collaborative design by students. ASGARD was in fact successful in automatically capturing all DR in the transcripts while producing no false positives.

POSITION-ISSUE

So moving to 3 stories?

AGREE/DISAGREE

We thought so

JUDGMENT

Seems logical to me

ISSUE

Then I was wondering how high it's gonna be

POSITION

1000 ft

AGREE/DISAGREE

Probably not

ARGUMENT

Yeah, it might get higher than we want

JUDGMENT

Well I feel that the height is OK

CONDITIONAL-ARGUMENT

Especially if we get rid of 1 of the pods

AGREE/DISAGREE+JUDGMENT

Me too, I think we can do it

CONDITIONAL-ARGUMENT

Yeah, as long as height is not gonna be a big issue here

JUDGMENT

It'll be OK

PROCESS-ISSUE

So what's for today then?

PROPOSED-ISSUE

Make better floor plan drawings?

JUDGMENT

K sounds good

POSITION-ISSUE

On the first floor ... do you think the lab and sample facility should be separate

ARGUMENT

I feel like these will be fairly extensive centers

POSITION-ISSUE

Do u think they should be moved (lab and sample)

ARGUMENT

Well I am not sure how big these facilities will be

ARGUMENT

Yeah ... I think that's our biggest prob here...I just have no idea how big these facilities should be

Fig. 4 One page of the structured rationale produced by the ASGARD Linker from the group who designed a Mars habitat

The Future of ASGARD: Work Needed and Potential of the Approach

The preliminary testing described above only suggests the plausibility of the ASGARD approach. Strong evidence for this approach will require demonstration of its ability to parse many transcripts not used to construct its grammar. This will require major efforts to acquire such transcripts and to build a large-scale ASGARD grammar to parse them.

If the ASGARD project ultimately succeeds in proving the viability of its approach, it may well be possible to extract and structure most of the rationale in design projects that is in text or spoken form. This would require no extra effort from designers. Many years of experience with rationale management systems—including PROTOCOL, MIKROPLIS, and PHIDIAS—suggest that it is like to be easy and inexpensive to store, manage, and retrieve this rationale.

Much work remains to be done before it can be known whether the capture of large amounts of DR is possible. If it is possible, we will need ways of determining the relevance and priority of the rationale and how to retrieve what is needed when it is needed. This, however, is not a problem for a rationale capture system, like ASGARD. It is a problem for a rationale management system, like PHIDIAS.

References

1. Selvin AM, Buckingham Shum SJ, Aakhus M (2010) The practice level in participatory design rationale: studying practitioner moves and choices. *Hum Technol* 6(1):71–105
2. Fischer G, Grudin J, Lemke A, McCall R, Ostwald J, Reeves B, Shipman F (1992) Supporting indirect collaborative design with integrated knowledge-based design environments. *Hum Comput Interact*, 7(3):281–314, L. Erlbaum Associated, Hillsdale, NJ
3. Burge JE, Brown DC (2006) Rationale-based support for software maintenance. In: Dutoit AH, McCall R, Mistrik I, Paech B (eds) *Rationale management in software engineering*. Springer, Heidelberg, pp 273–296
4. Hordijk WTB, Wieringa RJ (2006) Reusable rationale blocks: improving quality and efficiency of design choices. In: Dutoit AH, McCall R, Mistrik I, Paech B (eds) *Rationale management in software engineering*. Springer, London, pp 353–371
5. Moran TP, Carroll JM (eds) (1996) *Design rationale: concepts, techniques, and use*. L. Erlbaum Associates, Mahwah, NJ
6. Dutoit AH, McCall R, Mistrik I, Paech B (eds) (2007) *Rationale management in software engineering*. Springer, New York NY
7. Horner J, Atwood ME (2006) Effective design rationale: understanding the barriers. In: Dutoit AH, McCall R, Mistrik I, Paech B (eds) *Rationale management in software engineering*. Springer, New York NY, pp 73–90
8. Myers K, Zumel N, Garcia P Automated capture of rationale for the detailed design process. In: *Proceedings of the 11th national conference on innovative applications of artificial intelligence*, AAAI Press, CA, pp. 876–883
9. Shipman FM, McCall R (1997) Integrating different perspectives on design rationale: supporting the emergence of design rationale from design communication. *Artif Intell Eng Des Anal Manuf* 11:141–154

10. Yu D, Deng L (2015) Automatic speech recognition: a deep learning approach. Springer, London
11. Watanabe S, Delcroix M, Metze F, Hershey JR (2017) New era for robust speech recognition: exploiting deep learning. Springer, Cham, Switzerland
12. McCall R (1991) PHI: a conceptual foundation for design hypermedia. *Des Stud* 12(1):30–41 (Elsevier)
13. Rittel HWJ (1972) On the planning crisis: systems analysis of the ‘first and second generations’. *Bedriftskonomen* 8:390–396
14. Rittel HWJ, Webber M (1973) Dilemmas in a general theory of planning rittel. *Policy Sci* 4:155–169
15. Kunz W, Rittel HWJ (2010) Issue as elements of information systems. In: Protzen J-P, Harris DJ (eds) *The universe of design: Horst Rittel’s theories of design and planning*. Routledge, New York NY, pp 181–186
16. Moran TP, Carroll JM (eds) (1996) *Design rationale: concepts, techniques, and use*. Lawrence Erlbaum Associates, Mahwah, NJ
17. MacLean A, Young RM, Bellotti VME, Moran TP (1996) Questions, options and criteria: elements of design space analysis. *Design rationale: concepts, techniques and use*. Lawrence Erlbaum Associates, Mahwah, NJ, pp 53–105
18. Lee J (1991) Extending the Potts and Bruns model for recording design rationale. In: *Proceedings of the thirteenth international conference on software engineering*, ACM, New York NY, pp. 114–125
19. McCall R (1979) *On the structure and use of systems in design*. Dissertation, University of California, Berkeley, 1978, University Microfilms
20. McCall R (1989) MIKROPLIS: a hypertext system for design. *Des Stud* 10(4):228–238
21. Conklin J, Begeman M (1988) gIBIS: a hypertext tool for exploratory policy discussion. *ACM Trans Off Inf Syst* 4:303–331
22. McCall R, Bennett P, d’Oronzio P, Ostwald J, Shipman F, Wallace N (1990) PHIDIAS: integrating CAD graphics into dynamic hypertext. In: Rizk A, Streitz N, Andre J (eds) *Hypertext: concepts, systems and applications (Proceedings of the 1990 European Conference on Hypertext: ECHT ‘90)*. Cambridge University Press, Cambridge, UK, 152–165
23. Pena-Mora F, Vadhavkar S (1997) Augmenting design patterns with design rationale. *Artif Intell Eng Des Anal Manufac* 11:93–108 (DRIM)
24. Bracewell RH, Wallace KM (2003) A tool for capturing design rationale. In: *International conference on engineering design ICED 03, Stockholm, 19–21 Aug 2003 (DReD)*
25. Buckingham Shum SJ, Selvin AM, Sierhuis M, Conklin J, Haley CB, Nuseibeh B (2006) Hypermedia support for argumentation-based rationale: 15 years on from gIBIS and QOC. In: Dutoit AH, McCall R, Mistrik PB (eds) *Rationale management in software engineering*. Springer, Heidelberg, pp 111–132
26. Schneider K (2006) Rationale as by-product. In: Dutoit AH, McCall R, Mistrik I, Paech B (eds) *Rationale management in software engineering*. Springer, Heidelberg, pp 91–109
27. McCall R. (2013) *Critical Conversations: Feedback as a Stimulus to Creativity in Software Design*. In: Carroll J. (eds) *Creativity and Rationale*. Human–Computer Interaction Series, vol 20. Springer, London
28. Mix K, Jensen CG, Ryskamp J (2010) Automated design rationale capture within the CAX environment. *Comput Aided Des Appl* 7(3):361–375
29. Jurafsky D, Martin J (2009) *Speech and language processing*. Pearson Education, Upper Saddle River, NJ
30. Sivanandam SN, Janaki Meena M (2009) *Theory of computation*. IK International Publishing House, New Delhi, pp 235–236

Identifying Design Rationale Using Ant Colony Optimization



Miriam Lester and Janet E. Burge

Design rationale (DR), the reasons behind decisions made during design, can provide valuable insights into the decision-making process. This is especially valuable in software development, where systems are frequently repaired and extended over their lifetime. DR is often not explicitly recorded as such, but can occur in other artifacts of the development process. We would like to be able to extract this information and have been investigating using text mining to classify and extract rationale. One of the challenges is determining which document features are the most useful in building models. Some features may introduce noise and reduce classification accuracy. In this work, we explore using Ant Colony Optimization (ACO) to obtain optimal feature sets. Our results show that, in this particular study, features obtained using ACO produce better results than classifying with all features and that our classifiers are comparable in accuracy to human annotators.

Introduction

There are many reasons why developing software is difficult. Technical challenges, changing requirements, multiple stakeholders, and increasing complexity are some of them. Another is the often long life span of software systems, which are frequently developed by many programmers over many years and, if successful, spend more time in maintenance than in development. Keeping software working while adding new functionality is very challenging and is hampered by usually not having easy access to records of the many design decisions made during development and the

M. Lester
Wesleyan University, Middletown, USA

J. E. Burge (✉)
Colorado College, Colorado Springs, USA
e-mail: jburge@coloradocollege.edu

reasoning behind them. This information, known as design rationale (DR) is a way to capture the institutional knowledge that goes into a software product.

DR is usually not explicitly captured but may still be present in many types of documents created during the development process. The goal of this research is to examine methods for automatically detecting and extracting this information. Our goal is to use text mining to identify rationale from existing documentation so that it can be presented to software developers and maintainers. One of the challenges in text mining is to determine which text features create the best classification model. Our earlier work [1] experimented using Genetic Algorithms (GAs) [2]. In the work described in this paper, we turned to another biologically inspired algorithm—Ant Colony Optimization (ACO) [3].

In this paper, we discuss the following. In Section “[Motivation and Challenges](#)” we give a brief introduction to the motivation for and issues behind using Ant Colony Optimization for extracting rationale. Section “[Related Research](#)” presents related work. In Section “[Using Meta-Heuristic Algorithms for Feature Subset Selection](#)”, we describe the meta-heuristic text mining pipeline developed for this work. In Section “[Ant Colony Optimization for Rationale Identification](#)”, we described how we used Ant Colony Optimization as our meta-heuristic algorithm for feature selection within that pipeline. In Section “[Experimental Results](#)”, we describe our experimental results. We then end with our summary and conclusions in Section “[Conclusions and Future Work](#)”.

Motivation and Challenges

Our goal for this work was to use machine learning techniques to identify rationale in text documents. One of the challenges in text mining is that there are potentially thousands of features that can be used in this analysis [4]. The documents used in this research were annotated with 723 different feature categories (where a category could refer to parts of speech—verbs, adverbs, etc., or elements from an augmentation ontology). Some of these features may be noisy, irrelevant, or redundant and may end up being detrimental to the classifier [5]. This is why it is necessary to preform feature selection, or feature reduction, to determine which features are most relevant to the classification problem.

Combinatorial optimization problems like feature selection are NP-Hard (no known polynomial time solution) and one option is to turn to heuristic algorithms. Algorithms that guide the heuristic search, called metaheuristics, can be aid in finding near-optimal solutions. Ant Colony Optimization is an example of a meta-heuristic that can be used to explore the search space.

Ant Colony Optimization (ACO) is an approach inspired by the way that ants find their way to a food source. Foraging ants find the shortest path to a food source by laying down pheromone trails [6]. Ants communicate indirectly with each other by following pheromones laid by other ants. Successful paths end up with a stronger

pheromone trail. Ant Colony Optimization [3] models this approach in order to solve the graph theory problem known as the “minimum cost path problem”.

ACO works by mapping a combinatorial optimization problem to a problem that is characterized by a set of n components (C) arranged in a completely connected graph where a solution is a walk through the graph. Solutions are constructed by artificial ants where ants are only allowed to add a component to the current solution if the resulting solution is feasible [7]. Each component $c_i \in C$ has an associated pheromone trail τ_i and a heuristic value η_i . Pheromones provide a global, long-term memory of good solutions found throughout the search procedure, and are updated by each ant. The algorithm allows pheromone trails to evaporate over time to avoid convergence to suboptimal local extrema. Moreover, the heuristic value is defined by a source other than the ants (like the cost of adding a component to the solution).

Probabilistic transition rules are used to move the ants from component to component, where each rule uses a combination of the pheromone trails, heuristic values, and problem constraints. After a solution has been constructed, the pheromones of the components along the path are updated based on the solution quality. While an individual ant is able to find a solution to the problem, good solutions can only be found by interactions between the ants [3].

ACOs are often used to solve ordering problems like the classic traveling salesperson problem [8] where the “ants” are looking for the most efficient way to visit all the cities. Our problem is a different type of problem—a subset problem [9] where we are looking for a subset of the elements in the graph rather than a tour that visits each node. Unlike in ordering problems, the order of the nodes visited is not important. The pheromone trails are not associated with the connections between components but are instead associated with the components themselves.

In this work, our goal was to use an ACO to determine optimal feature sets for identifying rationale in text documents. We were interested in evaluating which parameters were best for selecting an optimal feature set with the ACO algorithm, if optimized feature sets outperformed using all document features, and how generalizable the feature set would be to different datasets.

Related Research

The following subsections describe related research in two areas: rationale extraction and Ant Colony Optimization for feature selection.

Rationale Extraction

There are many potential sources of design rationale in software projects, ranging from informal documents such as e-mail messages and meeting minutes, semi-structured documents such as bug reports, and formal documentation such as require-

ments specifications. The amount of rationale per document will vary depending on the documents purpose and the thoroughness of its author. Prior work has focused on extracting rationale from a number of different data sources. These papers do not necessarily use the same definition of rationale as we do and their data sources are quite different from ours both in content and (where available) the ratio of non-rationale to rationale.

Palau and Moens [10] tackled a problem similar to ours—identifying arguments in legal texts. They achieved an accuracy of 73% on a corpus of sentences where 50% of the sentences contained arguments and 80% on a corpus of legal texts. Their approach used n -grams and keywords along with linguistic features found in the text (verb tense, modal auxiliaries, and adverbs).

The goal of Toeska Rationale Extraction (TReX) [11] was to extract “knowledge units,” only some of which can be considered rationale. TReX is based on the General Architecture for Text Engineering (GATE) [12]. TReX works using Information Extraction NLP techniques that use manually created extraction rules to detect properties defined in architecture and rationale. They used these rules to extract knowledge units from 26 pages of architecture documents. Their aggregated extraction results had an F-1 value of 0.50.

Liang et al. [13], like in our work, were specifically interested in design rationale. They used a three-tiered model to look for artifacts, issues, and design solutions. A modified PageRank [14] algorithm was used to identify artifacts by looking for frequently appearing words. They also used issue summarization using issue language patterns as part of manifold ranking. Then, the third tier used reason language patterns to find reason sentences which were paired with solution sentences to create reason–solution pairs. They obtained F-measures of 0.185 artefact identification, 0.520 for issue summarization, and 0.562 for reason–solution extraction.

Our initial research in rationale extraction [15] investigated using ontologies to provide feature sets and compared a large number of different classifiers to extract design rationale from Chrome bug reports. The best F-1 measure achieved was 0.597 for binary classification (rationale/not rationale). When we used linguistic features and n -grams rather than the ontologies the classification accuracy was improved to F-1 measures of 0.676 for binary rationale and 0.569 for the argumentation subset (only looking at argumentation and excluding the answers, questions, and procedures that appeared in boilerplate text and were easier to extract) [16]. The two papers used different datasets, with the [16] set following a more rigorous annotation process. The feature sets used in those experiments were identified by what the researchers felt would provide the most promising results. We then decided to see if we could improve upon them by using genetic algorithms to optimize the feature sets [1] and if we could get better results with an alternative dataset that was less sparse than the Chrome bug reports. We also split our dataset into test and training data. For the Chrome bug reports and the argumentation subset we achieved F-1 measures of 0.576 for training (using 10-fold cross-validation) and 0.432 for testing. The alternative dataset, interview transcripts from the Studying Professional Software Designers Project (SPSD), we achieved F-1 measures of 0.652 for training (also using 10-fold cross-validation) and 0.354 for the test. This dataset was smaller than the Chrome bug

reports (2310 vs. 19,521 sentences) but had a higher density of rationale (52.4% vs. 10.9%). Those experiments did not use as many features as we did in the experiments described in this paper because they used the Waikato Environment for Knowledge Analysis (WEKA) [17], which limited the number of features that could be used.

Ant Colony Optimization for Feature Selection

Feature selection for classification problems is an interesting use of ACO and others have found success when compared to more traditional methods. Our work is based on that of Al-Ani [18], who used ACO to select feature sets to use in speech segment and image texture classification. He compared performance to using Genetic Algorithms [2]. The ACO algorithm had an accuracy of 0.842 compared to the GA's 0.835 accuracy. He experimented with different numbers of features and found that the ACO had similar success with using 20 features versus the full set of features. Unfortunately, he presented his results in classification accuracy rather than F-1 measure so we cannot compare our results directly. We used his variation of Rank-Based Ant System [7] in our implementation of using the ACO for feature subset selection.

Aghdam et al. [5] also used the ACO algorithm for feature set selection with the goal of reducing the search space dimensionality when categorizing text. They compared their ACO-based approach to using Genetic Algorithms, Information Gain (IG), and Chi-squared Statistic using the Reuters-2157 benchmark dataset (a collection of Reuters newswire documents). They calculated a Macro-F1 score (an F-1 measure for multi-class classification where all classes are equally weighted) and achieved Macro-F1 scores of 0.784 with ACO, 0.763 with GA, 0.709 with CHI, and 0.698 with IG.

Saraç and Özel [19] used ACO to assist with feature selection for web page classification. They created a feature space by looking at pairs of "tagged terms" (e.g., <url><term>). In their experiments, they used tagged terms on five different datasets and found that the F-measures were better using the ACO for feature selection rather than not using feature selection at all. Overall the datasets they had an average F-measure of 0.952 with ACO, and an average F-measure of 0.684 without feature selection. They implemented their ACO using an ant feature subset construction method that chooses all the features in groups rather than one by one to reduce unnecessary computation, and we used this approach in our ACO implementation to improve efficiency.

Using Meta-Heuristic Algorithms for Feature Subset Selection

The ACO Algorithm is one type of meta-heuristic for feature selection. One of our goals was to provide a way to experiment with different heuristics and compare their results. A first step in doing this was to create a processing pipeline to evaluate alternative Feature Subset Selection algorithms. This pipeline is shown in Fig. 1.

The process starts using a sentences.csv file that contains all the features extracted from each sentence along with the rationale annotation. The features include the Penn Treebank parts of speech tags [20], ontology terms identified in our earlier research

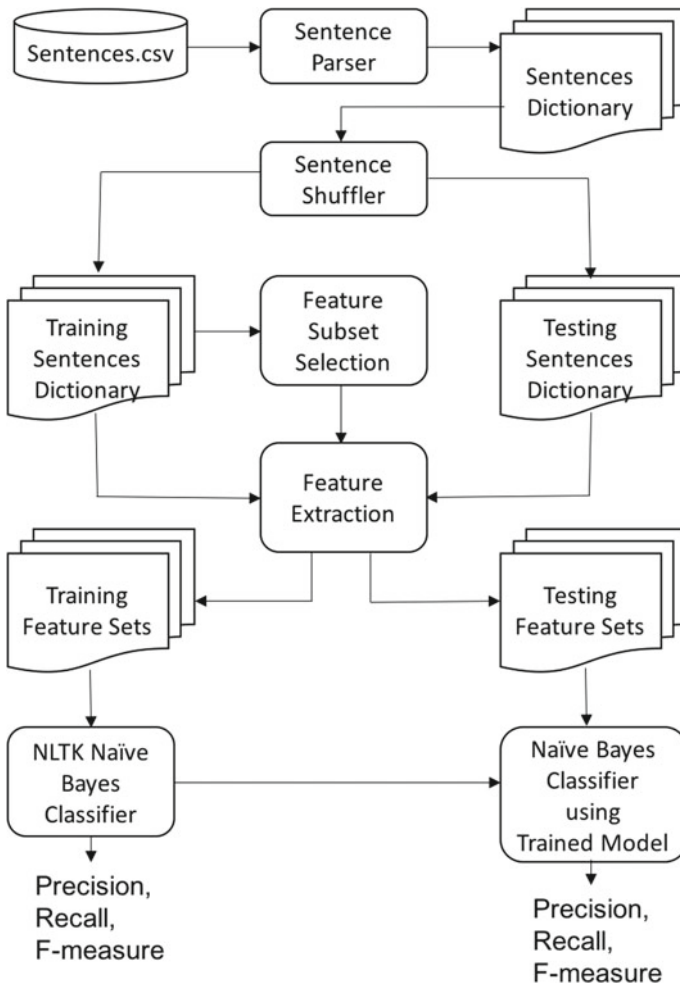


Fig. 1 Meta-heuristic pipeline for classifier building

[15], and feature combinations from neighboring sentences, since identifying a sentence as rationale may require more context than a single sentence [16]. N -grams from 1 to 5 g were also captured.

The sentence parsing takes the input data from the sentences.csv file and creates a Python dictionary structure that allows easy mapping from feature type to the feature instances in each sentence. The software then uses the dictionary to create random training and test sentence dictionaries where 70% of the sentences are used for training and the remaining 30% for testing.

The training sentences are then inputted into the Feature Subset Selection meta-heuristic algorithm (for this paper, this was the ACO), and are used to determine the optimal feature set. This feature set is then used to extract training and test feature sets. The training feature set is used to create the classification model, which is then evaluated using the test feature sets.

Ant Colony Optimization for Rationale Identification

Several components need to be defined in order to implement an ACO [5, 20–22]:

- (1) *Graph Representation of the Problem*: A graph that represents the search space of the problem with nodes and edges between them [20]. It must be possible to represent a problem solution using this graph. For the feature selection problem, the nodes are features and traversing an edge means selecting a feature. The graph is “fully connected” where it is possible to move from any node (feature) to any other node. In our implementation, we chose feature categories (parts of speech such as verbs or nouns, ontology terms, etc.).
- (2) *Feasible Solution Construction Constraint*: Constraints that ensure that only feasible solutions are built [23]. For our problem, our stopping criteria is the selection of a prespecified number of nodes (features) having been chosen. In our implementation, we chose twenty feature categories as the number of nodes that comprised a complete solution.
- (3) *Heuristic Desirability*: A way to measure the “goodness” [22] when adding a new component to a partially constructed solution. This is used to determine which features should be added to a solution. In our system, we use the Local Importance (LI) as the heuristic measure. This is the highest Chi-squared statistic of all the terms that fall into a featured category within our text corpus. This value is static and is calculated before the ACO runs. We use the Chi-squared statistic implementation provided by the Python NLTK library [24].
- (4) *Pheromone Update Rule*: A way to update the pheromone levels—both increasing levels when successful solutions are found and an evaporation rule to decrease levels over time [22]. For subset problems, the pheromone value is associated with a feature and not with the edge between two features. We use the standard method found in [22] in order to select the k best ants and to update their paths. The pheromone level for each node f_i in the graph (feature) is rep-

represented by T_i and the new level is represented by T'_i . For each feature f_i , the new pheromone level for an ant in the set of k best ants is

$$\Delta T_i = \frac{\text{fitness}(\text{ant}) - \text{worst}_{\text{fitness}}}{\max_{j=1:k}(\text{fitness}(j) - \text{worst}_{\text{fitness}})}$$

$T'_i = (1 - \rho) * T_i + \Delta T_i$, where ρ is the evaporation rate.

In our implementation, we used $k = 20$ and used the 20 best ants to reinforce their success in subsequent iterations.

The fitness function uses a Naïve Bayes classifier to evaluate the ant's subset of features. We use a wrapper method provided by Python NLTK that takes as input all the terms from the categories in the feature set for an ant and then outputs the F-measure of classifying that set, calculated using 10-fold cross-validation. We tested several other classifiers provided by the Python NLTK as well as the Sklearn [25] machine learning libraries (NLTK Decision Tree, Sklearn Bernoulli Naïve Bayes, Sklearn SVM SVC, Sklearn SVM Linear SVC, Sklearn MaxEntropy GIS, and Sklearn MaxEntropy IIS). The Python NLTK Naïve Bayes classifier gave the best balance between performance and efficiency.

- (5) *Probabilistic Transition Rule*: A rule that calculates the probability of an ant next moving to a new node in the graph [23]. Our implementation, again based on [18], selects p features that maximize the updated selection measurement (USM):

$$USM_i^{S_j} = \begin{cases} \frac{(\mathcal{T}_i)^\eta (LI_i^{S_j})^\kappa}{\sum_{g \notin S_i} (\mathcal{T}_g)^\eta (LI_g^{S_j})^\kappa} & \text{if } i \notin S_j \\ 0 & \text{Otherwise} \end{cases}$$

USM of feature i with respect to S_j (the subset of ant j):

- \mathcal{T}_i pheromone level of feature f_i
- LI_i local importance of feature f_i
- η relative weight of pheromone intensity
- κ relative weight of local importance.

Figure 2 shows the pseudocode of the ACO-based feature subset selection algorithm used in our implementation (based on [18]). We chose 100 as the number of ants (NA), 40 as the number of iterations (I), 1 as the initial pheromone level (T_i), 0.1 as the pheromone evaporation rate ρ , 20 as the number of best ants used to update the pheromone levels (K), and 2 and 1 as the relative weights of pheromone intensity (η) and local importance (κ), respectively. When selecting each ant's features for the next iteration, we chose three-fourths randomly from the featured sets of the 20 best ants and the remaining one-fourth to maximize the update selection rule (USM).

Figure 3 shows a flow-chart of the algorithm in action.

```

Initialize:
  I = # iterations
  NA = # ants
  M = feature subset size
  Ti = initial pheromone level
  ρ = evaporation rate
  K = # best ants' subsets used to update pheromone
  P = # remaining features to be selected each iteration
  USM (Updated Selection Measurement) parameters
    η = relative weight of pheromone trails
    κ = relative weight of local importance
  Subsets = For each ant:
    randomly assign Sant = {f1, f2 ..., fM} ⊂ {all features}

For iteration in range(I):
  For each ant: FSant = Generate feature set from Sant and training sentences
  For each ant In Parallel: calculate Fitness(FSant)
  Sort subsets by highest fitness
  Update best subset found so far if necessary
  For each of K best ants:
    Update pheromone trails of paths they constructed
  K-Set = {union of all features in K best ants' subsets}
  For each ant:
    randomly assign m-p feats from K-Set: Sant = {f1, f2, ..., fm-p} ⊆ K-Set
  For each ant:
    Choose p best features that maximize USM(feature, Sant)
    Sant = Sant ∪ {fm-p+1, ..., fm}

Return Best Subset

```

Fig. 2 ACO pseudocode

Experimental Results

Input Dataset

We used two datasets for this experiment. The first was a set of 200 Chrome bug reports extracted randomly from a subset of the data provided for the Mining Software Repositories 2011 mining challenge (<http://2011.msrconf.org/msr-challenge.html>). We chose to work with bug reports because in addition to the description of the problem that required solving they also provided discussion data on alternative responses to fix the problem. We created our test and training data by having two researchers annotate each bug report and a third researcher adjudicate. The bug report

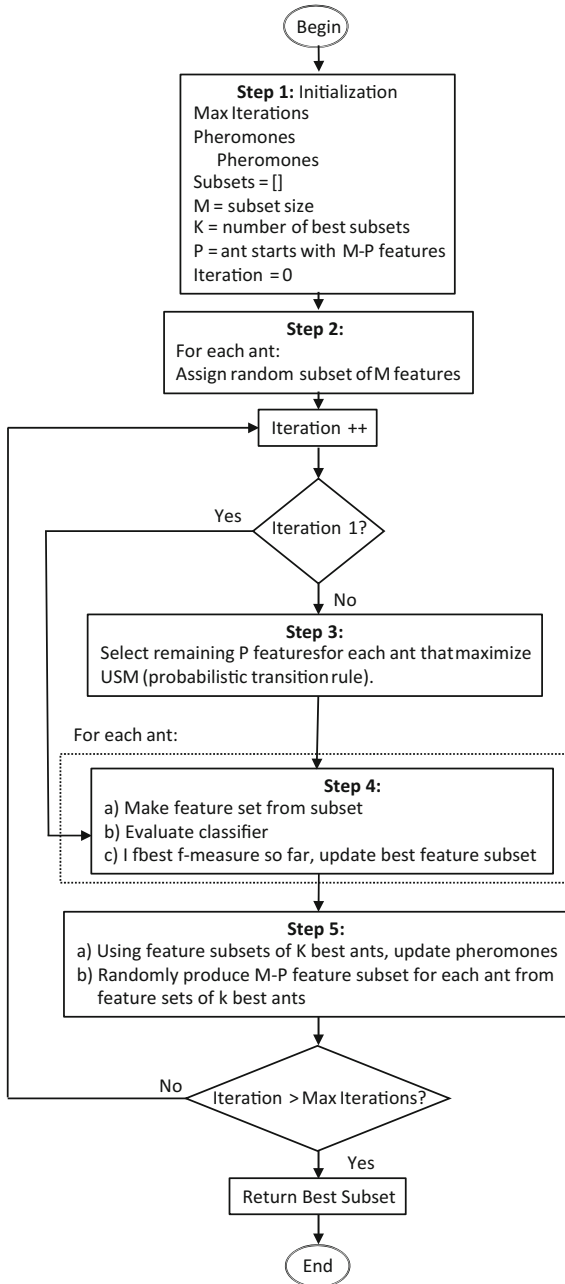


Fig. 3 ACO flow-chart

data was a sparse dataset, with 10.9% of the sentences containing rationale. Additional information on how bug reports were selected and annotated is provided by Rogers et al. [16].

We used a second set of data to test our algorithms on a less-sparse dataset. We used design interview transcripts from the Studying Professional Software Designers (SPSD) project. 52.4% of the sentences in this data contained rationale, making it a more balanced dataset than the bug reports. Details on the annotation process can be found in Rogers et al. [1].

In both datasets, we annotated text according to the following categories [16]:

- *Requirements*—statements that indicated software was required to do something;
- *Decisions*—statements that identified an issue to be resolved;
- *Alternatives*—alternative solutions for resolving the issue described by a decision;
- *Arguments*—reasons for or against an alternative;
- *Assumptions*—arguments where there appeared to be some uncertainty;
- *Questions*—questions that needed to be answered as part of the solution process;
- *Procedures*—description of actions needed to gain information required resolve the issue or answer a question;
- *Answers*—answers to questions asked as part of the decision-making process.

A sentence can be classified as more than one type of rationale since parts of a single sentence may have been tagged as different elements or types might overlap completely (for example—an alternative that has selected may then require a decision to be made on how it should be implemented).

Results

After the experiments were run, the results were collected and then evaluated. Here, we present the results on the training set (computed using 10-fold cross-validation), and the test set. For comparison, we include the results if the model was created using all the features rather than a subset optimized by the ACO and the inter-annotator agreement. The inter-annotator agreement is computed by comparing the annotations created by the two researchers that annotated each bug report. This was done using GATE with the most lenient setting—if any part of an annotation overlapped it was considered a match. The inter-annotator agreement indicates how difficult the rationale identification task is for humans.

Chrome Bug Reports

Each Chrome Bug Report experiment took about 1.3 h to complete. We ran five trials of each experiment. The results reported here are the best values, presented along with their standard deviation. We present the results along with the results for using

Table 1 Bug report, binary rationale

	Precision	Recall	F-measure	Std. deviation
Training	0.708	0.943	0.809	0.001
Validation	0.688	0.953	0.799	0.003
All features	0.593	0.984	0.740	
Inter-annotator agreement			0.752	

Table 2 Bug report, argumentation subset (alternative, argument, assumption, decision, requirement)

	Precision	Recall	F-measure	Std. deviation
Training	0.478	0.849	0.610	0.002
Validation	0.507	0.843	0.633	0.007
All features	0.339	0.986	0.504	
Inter-annotator agreement			0.524	

Table 3 Bug report, decisions

	Precision	Recall	F-measure	Std. deviation
Training	0.331	0.496	0.394	0.009
Validation	0.335	0.484	0.396	0.021
All features	0.077	0.994	0.142	
Inter-annotator agreement			0.262	

Table 4 Bug report, alternatives

	Precision	Recall	F-measure	Std. deviation
Training	0.224	0.559	0.318	0.001
Validation	0.246	0.563	0.342	0.023
All features	0.078	0.984	0.145	
Inter-annotator agreement			0.332	

all the features to build the classification model and the inner annotator agreement (Tables 1, 2, 3, 4 and 5).

In most cases, the optimized feature results were better than the Inter-Annotator agreement (the exception being alternatives where the Training results were 0.318 vs. 0.332). The optimized feature results were better than the results if all features were used.

Table 5 Bug report, arguments-all (requirement, argument, assumption)

	Precision	Recall	F-measure	Std. deviation
Training	0.348	0.730	0.470	0.002
Validation	0.334	0.672	0.446	0.016
All features	0.186	0.988	0.314	
Inter-annotator agreement			0.261	

Table 6 SPSD data, binary rationale

	Precision	Recall	F-measure	Std. deviation
Training	0.627	0.934	0.750	0.004
Validation	0.626	0.926	0.7467	0.007
All features	0.664	0.754	0.706	
Inter-annotator agreement			0.622	

Table 7 SPSD data, argumentation subset (alternative, argument, assumption, decision, requirement)

	Precision	Recall	F-measure	Std. deviation
Training	0.618	0.918	0.736	0.021
Validation	0.612	0.915	0.733	0.032
All features	0.615	0.846	0.713	
Inter-annotator agreement			0.606	

SPSD Data

Each SPSD data experiment took about 0.4 h to complete. We ran 30 trials of each experiment. The results reported here are the best values, presented along with their standard deviation. We present the results along with the results for using all the features to build the classification model and the inter-annotator agreement.

In all cases, the optimized feature results were better than the Inter-Annotator agreement. The optimized feature results were mostly better than the results if all features were used (the exception being alternatives where the training results were 0.594 vs. 0.595) (Tables 6, 7, 8, 9 and 10).

Generalizability to Other Datasets

We investigated how generalizable these results were by comparing how a feature set optimized for one dataset performs when used to classify the other dataset. Tables 11,

Table 8 SPSD data, decisions

	Precision	Recall	F-measure	Std. deviation
Training	0.292	0.490	0.350	0.015
Validation	0.281	0.439	0.342	0.043
All features	0.113	0.895	0.201	
Inter-annotator agreement			0.262	

Table 9 SPSD data, alternatives

	Precision	Recall	F-measure	Std. deviation
Training	0.496	0.745	0.594	0.003
Validation	0.544	0.809	0.651	0.016
All features	0.441	0.914	0.595	
Inter-annotator agreement			0.391	

Table 10 SPSD data, arguments-all (requirement, argument, assumption)

	Precision	Recall	F-measure	Std. deviation
Training	0.284	0.478	0.349	0.012
Validation	0.313	0.554	0.400	0.035
All features	0.133	0.969	0.234	
Inter-annotator agreement			0.205	

Table 11 Result comparison for binary rationale

Dataset		BR optimized	SPSD optimized	Difference
Bug	Training	0.809	0.759	-0.131
Report	Validation	0.799	0.744	-0.055
SPSD	Training	0.651	0.750	-0.099
	Validation	0.676	0.747	-0.071

12, 13, 14 and 15 present the result, along with the difference between the difference in F-1 measure between the dataset the features were optimized for and the other dataset.

Using features optimized for a different dataset decreased performance in all cases. For datasets, the difference was largest for the SPSD data. For data types, the difference was largest for Decisions, where performance drops ranged from 0.249 to 0.336, followed by alternatives where differences were between 0.146 and 0.29555.

Table 12 Result comparison for the argumentation subset

Dataset		BR optimized	SPSD optimized	Difference
Bug	Training	0.610	0.544	-0.132
Report	Validation	0.633	0.572	-0.061
SPSD	Training	0.658	0.736	-0.078
	Validation	0.670	0.733	-0.063

Table 13 Result comparison for decisions

Dataset		BR optimized	SPSD optimized	Difference
Bug	Training	0.394	0.145	-0.249
Report	Validation	0.396	0.147	-0.249
SPSD	Training	0.014	0.350	-0.336
	Validation	0.034	0.342	-0.308

Table 14 Result comparison for alternatives

Dataset		BR optimized	SPSD optimized	Difference
Bug	Training	0.318	0.172	-0.146
Report	Validation	0.342	0.185	-0.157
SPSD	Training	0.381	0.594	-0.213
	Validation	0.3555	0.651	-0.29555

Table 15 Result comparison for arguments-all

Dataset		BR optimized	SPSD optimized	Difference
Bug	Training	0.470	0.356	-0.114
Report	Validation	0.446	0.370	-0.076
SPSD	Training	0.194	0.349	-0.115
	Validation	0.200	0.400	-0.200

Conclusions and Future Work

The experiments in this paper were designed to answer the following questions:

1. **What parameters are best for selecting the optimal feature set with the ant colony optimization algorithm?** Some of our ACO parameters were set to be consistent with our earlier work using Genetic Algorithms (for future comparison). These were the number of ants (100), feature subset length (20), and top k ants ($k = 20$). Others were set through experimentation. Early experiments converged to poor solutions quite early when ants were getting stuck in local optima. After experimental trial and error, pheromone evaporation rate was set to 0.1. The probabilistic transition rule parameters for relative weight of pheromone intensity

and local importance were set, respectively, as 2 and 1—giving more influence to pheromone intensity for feature set construction. To preserve high-quality solutions, $\frac{3}{4}$ of features were chosen from k best ants of the previous generation, and one-fourth of features were chosen to maximize the update selection measurement rule.

2. **How does the ant colony optimization selected feature set compare to results using all the features for classification?** For the bug report data, the optimized feature sets were better than using all the document features, with the largest differences being for decisions and alternatives (the most fine-grained of the classification targets). The same was true for the SPSD data except in the case of alternatives where using all features gave a training result of 0.595 and the optimized feature set gave a result of 0.594.
3. **How do optimized feature sets generalize to other document corpuses?** Using features optimized for one document corpus to classify another decreased performance. These differences were small when using SPSD features on bug report data, except for decisions, and larger when using bug report features to classify the SPSD data.

Classifying text as rationale is a challenging problem even for humans, as shown by inter-annotator agreement values. Using ant colony optimization for feature selection improved classification over using all the features and suggests that there are likely to be noisy redundant features that hurt classification. The models trained using the optimization feature set always improved upon the inter-annotator agreement results, suggesting that while not perfect the automatic classification may be better than classification done by humans.

We are currently comparing the ACO with a new implementation of the genetic algorithm we used in our earlier work to see which technique is most effective. We would also like to experiment with using classifiers other than Naïve Bayes, which was chosen because of its speed.

We would also like to experiment with modifying the fitness function. We have been using the F-1 measure because it combines precision and recall. What we do not know is if it would be more useful to only identify some of the rationale, but accurately, or if it is better to make sure we have more of it, but with false positives. We have developed a tool that allows humans to mark rationale in text—we would like to import our classification results into that tool and study how difficult it is to update classifications and structure the rationale when the tool gives a head start rather than starting with plain text documents.

Acknowledgements We would like to thank Miami University graduate and undergraduate students James Gung, Michelle Flowers, John Malloy, Tanmay Mathur, Yechen Qiao, and Benjamin Rogers for their work in annotating the data used in these experiments and Wesleyan student Connor Justice for his assistance in creating the sentences.csv file. Miriam Lester was supported by a fellowship from the American Association of University Women (AAUW). The design sessions that produced the SPSD data were funded by the National Science Foundation (NSF) (award CCF-0845840). We would like to thank the workshop organizers, André van der Hoek, Marian Petre, and Alex Baker for granting access to the transcripts. The data annotation work was supported by NSF CAREER Award CCF-0844638 (Burge). Any opinions, findings, and conclusions or recom-

mentations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

References

1. Rogers B, Justice C, Mathur T, Burge JE (2016) Generalizability of document features for identifying rationale. In: Gero J (ed) *Design, computing, and cognition*, pp. 633–651
2. Holland JH (1975) *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, Cambridge, MA
3. Dorigo M (1992) *Optimization, learning and natural algorithms*. Ph. D. Thesis, Politecnico di Milano, Italy
4. Saraç E, Özel SA (2014) An ant colony optimization based feature selection for web page classification. *The Scientific World Journal*
5. Aghdam MH, Ghasem-Aghaee N, Basiri ME (2009) Text feature selection using ant colony optimization. *Expert Syst Appl* 36(3):6843–6853
6. Deneubourg JL, Aron S, Goss S, Pasteels JM (1990) The self-organizing exploratory pattern of the argentine ant. *J Insect Behav* 3(2):159–168
7. Dorigo M, Stützle T (2004) *Ant colony optimization*. MIT Press, Cambridge, MA, p Xi, 305
8. Stützle T, Dorigo M (1999) ACO algorithms for the traveling salesman problem. In: *Evolutionary algorithms in engineering and computer science*, pp. 163–183
9. Leguizamón G, Michalewicz Z (1999) A new version of ant system for subset problems. In: *Evolutionary Computation*
10. Palau M, Moens MF (2009) Argumentation mining: the detection, classification and structure of arguments in text. In: *Proceedings of the 12th International Conference on Artificial Intelligence and Law (ICAIL '09)*, pp 98–107
11. López C, Codocedo V, Astudillo H, Cysneiros LM (2012) Bridging the gap between software architecture rationale formalisms and actual architecture documents: an ontology-driven approach. *Sci Comput Program* 77(1):66–80
12. Cunningham H, Maynard D, Bontcheva K, Tablan V (2002) GATE: a framework and graphical development environment for robust NLP tools and applications. In: *Proceedings of the 40th anniversary meeting of the association for computational linguistics (ACL'02)*, Philadelphia, July 2002
13. Liang Y, Liu Y, Kwong C, Lee W (2012) Learning the ‘Whys’: discovering design rationale using text mining – an algorithm perspective. *Comput Aided Des* 44(10):916–930
14. Brin S, Page L (1998) The anatomy of a large-scale hypertextual Web search engine. *Comput Netw ISDN Syst* 30:107–117
15. Rogers B, Gung J, Qiao Y, Burge JE (2012) Exploring techniques for rationale extraction from existing documents. In: *Proceedings of the international conference on software engineering*, IEEE Press, pp 1313–1316
16. Rogers B, Qiao Y, Gung J, Mathur T, Burge JE (2014) Using text mining to extract rationale from existing documentation. In: Gero J (ed) *Design, computing, and cognition*, pp 457–474
17. Hall M, Frank E, Holmes G, Pfahringer B, Reutmann P, Witten IH (2009) The WEKA data mining software: an update. *SIGKDD Explor* 11(1):10–18
18. Al-Ani A (2005) Feature subset selection using ant colony optimization. *Int J Comput Intell* 2(1):53–58
19. Saraç E, Özel SA (2009) An ant colony optimization based feature selection for web page classification. *The Scientific World Journal*
20. Marcus M, Marcinkiewicz M, Santorini B (1993) Building a large annotated corpus of English: the penn treebank. *Comput Linguist* 19(2):313–330
21. Dorigo M, Maniezzo V, Coloni A (1996) Ant system: optimization by a colony of cooperating agents. *IEEE Trans Syst Man Cybern Part B (Cybern)* 26(1):29–41

22. Kanan HR, Faez K, Hosseinzadeh M (2007) Face recognition system using ant colony optimization-based selected features. In: Computational intelligence in security and defense applications
23. Basiri ME, Nemati S (2009) A novel hybrid ACO-GA algorithm for text feature selection. In: Evolutionary computation, IEEE congress on. IEEE
24. Bird S, Klein E, Loper E (2009) Natural language processing with Python—analyzing text with the natural language toolkit. O'Reilly Media
25. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. *J. Mach Learn Res* 12:2825–2830

Biased Decision-Making in Realistic Extra-Procedural Nuclear Control Room Scenarios



Emil Andersen, Igor Kozine and Anja Maier

In normal operations and emergency situations, operators of nuclear control rooms rely on procedures to guide their decision-making. However, in emergency situations, these procedures may be insufficient in guiding operators. Little is known about the decision-making strategies that operators employ in these extra-procedural situations. To address this, a realistic simulation study was conducted with five crews of active, licensed nuclear operators to see the behavioural patterns that occur when procedures are not sufficient. This paper, a reanalysis of a previously collected dataset, investigates how the design and existence of procedures influence, and possibly bias, decision-making. Evidence is found that operators were affected by confirmation bias, and that mismatches between their home power plant and the simulated power plant made them commit errors due to misapplied expertise. Furthermore, this effect was amplified by the existence and design of the procedures used. Avenues for debiasing through design are discussed.

Introduction

Studies of operators in nuclear control rooms, airplane cockpits and medical decision-making have led to a greater understanding of decision-making in high-stakes complex environments over the last several decades. To deal with the complex requirements of these environments and strengthen performance, researchers in decision-making and design have worked hand in hand to improve the interfaces, environments and tools of the specialists that operate these fields (e.g. [1–5]). An important early development was the move towards the use of written procedures and checklists. Usually, in the form of physical paper copies of varying length, procedures and

E. Andersen (✉) · I. Kozine · A. Maier
Technical University of Denmark, Kongens Lyngby, Denmark
e-mail: emand@dtu.dk

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_30

checklists are written documents that specify conditions for their use, followed by a list of diagnosis and action steps. However, previous studies have found that for the critical situations in nuclear control rooms, the majority of real-life operating events included non-typical conditions [6, 7]. In such situations, where the predicted situation in the procedure does not match the observed situation, procedures may become inefficient or lack proper guidance [8].

In the design field, an increasing amount of studies have sought to uncover patterns in human behaviour in order to guide designers in their efforts, e.g. on topics ranging from basic perceptual functioning to aesthetic product preferences and other critical factors for consumer decisions (e.g. [9–17]). Through evaluations of real and/or stylized products and product attributes, these studies have started to shape our understanding of how humans in general and in relation to specific groups in particular perceive and interact with various design characteristics. Furthermore, studies of designers have found biases in their decision-making such as design fixation [18], for a recent review see [19], the preference effect [20], strategies for how these effects can be mitigated and how these strategies interact with expertise [21]. Outside of the design field, developments over the last several decades have led to an increased understanding of decision-making strategies and biases in general [22, 23]. However, little is known about how these decision-making insights apply in the practical situation of a control room emergency, and whether the decision-making biases are reduced or amplified by the existence of the designed objects such as procedures and checklists—particularly in the non-typical situations that characterise real-life emergencies.

To address this, the present paper reanalyses data collected for a project involving two realistic pressurised water reactor scenarios conducted at the Halden Man Machine Laboratory (HAMMLAB) simulator in 2014. The scenarios were designed such that multiple complications would lead to situations where crews had to perform autonomous extra-procedural actions to achieve optimal performance. The results of the original study were documented by Massaiu and Holmgren [24]. They investigated how operators perceive discrepancies between their own plans and the procedure, how crews compromised between needing to act fast and to follow procedures, and how the crew size and composition affected diagnosis and decision-making. They found that crews, with some exceptions, prioritised strict adherence to procedures and that crew size and composition did not influence performance. The scenarios were described in detail by Massaiu and Holmgren [25] to allow for future reanalysis, such as this paper.

Adding to this former work, the present paper aims to show biases and heuristics that may have caused divergences in behaviour amongst the crews. The purpose of this study then is to create an exploratory platform to show how biases may influence expert decision-making in critical situations, as well as to begin shaping our knowledge of how these biases may arise from the designed objects that these operators interact with during their work.

Background

In this paper, we focus on two biases that have been related to expert decision-making: The first is the bias that occurs when expertise is transferred to a similar but different situation, thus causing misapplications of one's expertise. The second is confirmation bias, which is the tendency to overly prioritise and seek for information that benefits existing views. Both biases have been shown to impact decision-making of experts in many diverse fields, such as medicine, engineering and law [26].

In this section, these biases are described, and we outline which behavioural patterns should be observed if the nuclear control room operators were affected by them.

Expertise

What Is an Expert?

In this paper, we use as the basis for our definition of expertise the one given by Simon [27]: *'The situation has provided a cue: This cue has given the expert access to information stored in memory, and the information provides the answer. Intuition is nothing more and nothing less than recognition'*. From this viewpoint, an expert is one who has been exposed to a high variety of situations and has learned the correct response, which allows him to swiftly recall and apply it in future situations. Furthermore, we extend our definition of expertise based on the arguments by Kahneman and Klein [22]. Drawing on a study by Shanteu [28], they argue that expertise will only form if (a) the context of training provides valid cues for learning, meaning cues that reflect real patterns in the context, and (b) if the context of training is sufficiently regular to allow for learning of patterns. Without these aspects, they argue, it is not possible to learn whether your behaviour is resulting in good or bad results, and thus expertise cannot be achieved. The nuclear power plant is a vastly complex system, with a myriad of technical details that needs to be acquired over several years an operator is certified. However, from the criteria of Kahneman and Klein [22], the nuclear power plant control room is a valid context for acquiring expertise, as the relation between inputs and outputs of actions is both consistent and readily available of observation for the operators. Given the extensive training required for certification, as well as the substantial experience of all the participants in the study, a high level of expertise should thus be expected.

Misapplied Expertise

Given this expertise, it is expected that the operators employ highly refined heuristics (shortcuts for decision-making) that allow them to make (near-) optimal decisions

for the context they have been trained in, with lower effort [23, 29, 30]. However, these strategies can decrease performance if applied to other contexts, where they are not adequate. For the present simulation study, this may be the case. First, the crews were trained at a power plant in a different country than the simulated power plant. Second, not all parts of the simulation perfectly matched what would be observed in the reference power plant. The operators could perform suboptimally due to lack of plant-specific knowledge or due to expectations of and/or reliance on signals that do not come due to plant differences. Furthermore, due to these differences, the operators need to adjust their behaviour to reflect a lower level of expertise than what they have for their home plant, taking the more explorative mindset of a novice. However, previous research has shown that experts, when put in a similar but not identical situation to what they have expertise for, tend to act as if their expertise also applies to the novel situation [22]. This is caused by a false belief that there is a perfect transfer of skill between the two situations. While the operators receive training in operating the simulation power plant prior to the simulation scenarios, there may nevertheless be deviations between the two power plants that will cause operators to use heuristics that are inappropriate for the specific context.

If the nuclear power plant operators are affected by the bias of misapplied expertise, we expect that one, they will double-check their decisions in an insufficient manner, as they would not need to do this if they were highly trained, and two, they will deviate in ways that turn out wrong due to lacking plant-specific knowledge.

Confirmation Bias

Confirmation bias is the non-conscious tendency to seek for and to give higher value to information that confirms our existing views and, conversely, to ignore and deprioritise information that goes against our existing views. Confirmation bias is thus an overarching term that covers the tendency to strongly persist in existing beliefs, as a result of biased evaluation of information and in search for information [26].

Belief Persistence

The first aspect of confirmation bias is belief persistence, which is the term for a collection of tendencies that cause early beliefs to be very resistive to change: First, the tendency to persist in early hypotheses for no reason than them being the first adopted hypotheses [31]; and second, the tendency to be more likely to question information that contradicts their existing belief, while being less likely to question information that confirms their pre-existing belief [32, 33]. Third, the tendency to be likely to explain away events as random, etc., if they conflict with their existing beliefs, thus discrediting the events rather revisiting the belief [34].

If operators are susceptible to belief persistence, we expect that operators will persist in their early hypotheses if they do not contradict the operating procedures (regardless of whether or not the procedures are correct for optimal decision-making at the time).

Biased Search for Information

The second aspect of confirmation bias is the tendency to only seek information that confirms one's existing view, or to only seek for information that would only exist if the existing view was correct. Conversely, it is the tendency to avoid information that would disconfirm one's view and/or not to seek for information that would exist if an alternate view was correct [35]. This tendency thus allows one to never disconfirm one's view through never exposing oneself to situations that threaten the viewpoint. Furthermore, given that one only samples information that supports the view, confidence in the view increases [36].

Similarly, we expect that operators will perform confirmatory search by looking at power plant locations that will show problems only if their hypothesis was true and will tend not to search for disconfirming information through, e.g. alternate sources such as field operators.

Case Study

The data that form the basis for this paper are the decisions of nuclear control room operators in two realistic simulation scenarios conducted in the HAMMLAB simulator in 2014. Two scenarios of realistic emergency situations in a pressurised water reactor were run by five crews of 3–5 crew members. The size of the crews and the exact scenario durations are shown in Table 1. In nuclear operations, operators rely on emergency operating procedures to solve emergencies. The operators have knowledge of a vast array of 'entering conditions' for various procedures, and will 'enter' a given procedure in response to these conditions. Once entered, the procedures will guide the operators through identifying and alleviating problematic symptoms. Operators are at no point required to know the cause of the observed problems, only to follow the procedures that instruct how to respond to these symptoms.

The two scenarios are unique with respect to the cause of the problem. However, in both scenarios, emergency operating procedures are entered in response to the reactor 'tripping' (this term refers to neutron absorbing control rods being inserted into the core, thus stopping the chain reaction). While tripping the reactor stops further power from being produced, the power plant is not safe until problems such as leaks causing spread of radiation are solved, and the plant is cooled and depressurized. Until safe shutdown is achieved, adverse effects such as release of radioactive material to the atmosphere, or, in the worst case, core meltdown, are still possible.

Table 1 Crew size and duration of scenarios. Note that shorter duration does not necessarily indicate better performance

Crew	1	2	3	4	5
Size	5	5	3	4	3
Scenario 1 duration	02:06:21	01:54:47	02:50:00	01:52:51 ^a	02:10:37
Scenario 2 duration	01:26:21	01:14:47	02:10:00	01:12:51 ^a	01:30:37

^aScenario was stopped before the crew had completed the final goal

Both scenarios were designed such that the following operating procedures were not sufficient for safe and effective shutdown. Operators were thus required to perform autonomous actions to avoid adverse effects. The scenarios are described in detail below. Overall, the complex problems of the scenarios caused several problems for all crews in both scenarios, albeit to varying degrees for the various crews, as will be elaborated below.

Scenario 1

The first scenario involved multiple leaks on the piping system that connects the core with the plant's three steam generators. The crew is given the cover story that construction is ongoing nearby, which, shortly after start, is cited as the cause for a blast that gives vibrations to the plant, including the control room. In the simulation, this blast results in immediate release of radioactive material in all steam generators, followed shortly (12 min after start) by a small leak in a tube connected to Steam Generator #2, and subsequently (20 min after start) a rupture in a tube in Steam Generator #3. The leak in Steam Generator #3 will increase in size two times, first at 25 min and then at 40 min after start, with the latter being equivalent of a complete tube rupture. If the crew has not manually tripped the reactor at 40 min, the automatic tripping system will do so shortly after the complete tube rupture.

The challenge for the crew is ensured cooldown while avoiding using the two damaged steam generators' relief valves, as this would result in release of radioactive material to the atmosphere. To do so, the two damaged steam generators should be isolated. This task is complicated by the fact that it is not clear from following the procedures and the information displayed whether Steam Generator 2 is causing problems, as it is obscured by the effects of the rupture in Steam Generator 3. Operators must thus actively look for additional information to successfully handle the task (Fig. 1).

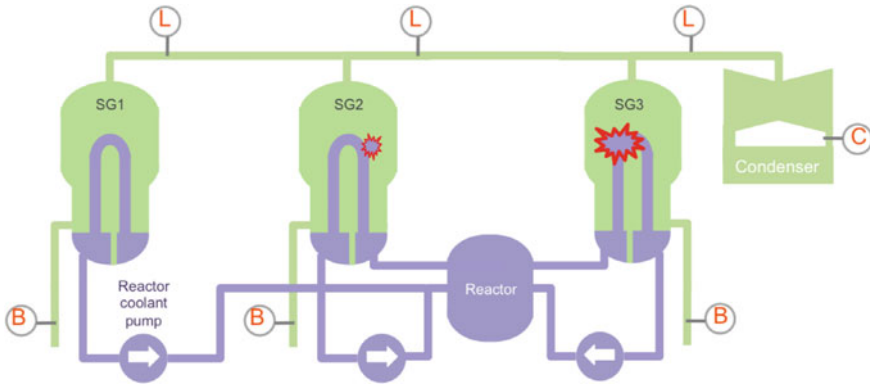


Fig. 1 Diagram of Scenario 1, courtesy of Massaiu and Holmgren [24]

Scenario 2

The second scenario involves an irreversible loss of coolant following leaks to the reactor coolant system, which results in water spilling on the floor of the auxiliary building. The scenario begins with a distracting task in the form of a pump trip. This will occupy the operators at the start of the scenario. The first major complication happens when two valves start leaking in the residual heat removal system (one at start, the other after 8 min). At around 11 min from start, a pipe in the residual heat removal system of the auxiliary building will break, resulting in reactor coolant fluid spilling on the floor. Finally, a smaller leak will occur in the reactor coolant pump thermal barrier, which will complicate the detection of the primary leak. The loss of pressure will cause an automatic trip of the reactor if it is not initiated manually (Fig. 2).

The challenge for the crew is to ensure safe and effective cooldown while reducing the effects of the leaks. This task is complicated by the fact that the procedure for this type of event aims at identifying the main leak and to isolate it, but in the present scenario, the procedure's directions are insufficient. Furthermore, although the operators do not know that the leaks are not isolable, it is considered optimal performance to discover the location of the leaks and to try isolation actions, beyond the procedures' guidance.

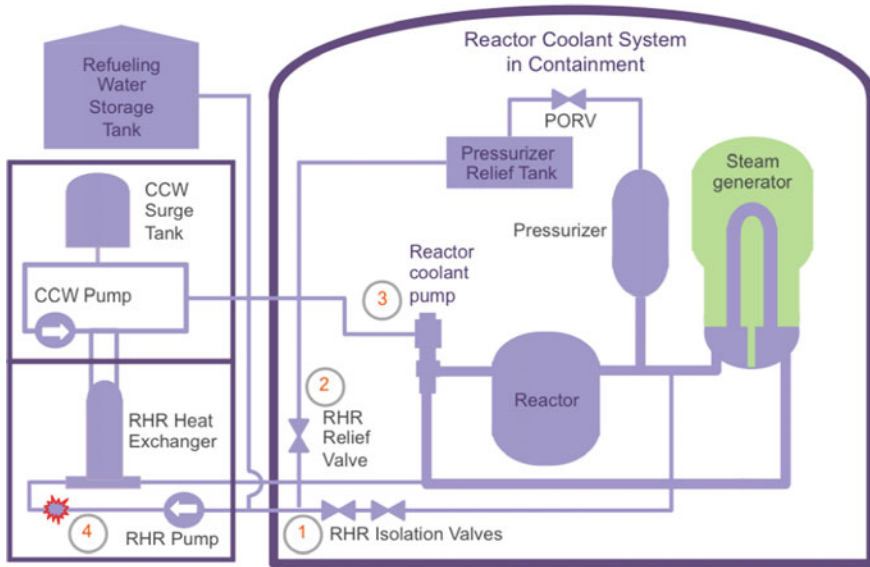


Fig. 2 Diagram of Scenario 2, courtesy of Massaiu and Holmgren [24]

Methods

Dataset

Five crews of certified operators from three nuclear power plants were recruited. Two crews had three members, two crews had four members and one crew had five members. All crews participated in both scenarios.

The study took place at the Halden Man Machine Laboratory (HAMMLAB) at the Institute for Energy Technology, Halden, Norway, in a realistic simulation setup that mimics a Swedish Pressurised Water Reactor (PWR). Audio and video materials were recorded during the scenarios, which serve as the raw dataset for this analysis. The audio material included all conversation between operators, sounds played in the environment, conversations between the operators and the experimenters (who, at various times, roleplayed as field operators) and conversations between experimenters. The video material consisted of four streams of serially played still shots of the operators and recordings of the displays used by the operators (including mouse movements on these). The raw dataset was processed by a former shift supervisor from the simulated plant, who has many years of experience as an operator and thus has the required skills to evaluate performance, together with a human factors specialist from the Halden Reactor Project. These evaluations and additional comments were noted down in detail in a spreadsheet, along with the timestamp and a brief description of the various events that the process expert had deemed signif-

icant. The contents of the comments included, but were not limited to: examples of good behaviours, errors and deviations from protocols. To measure performance, each team was evaluated by the process expert using a 5-point rating scale, which gives a score of one to five on task critical operations, and with five being optimal performance. For the purpose of this study, the crew performance scores for each scenario were averaged as an indicator of overall performance.

As this study reanalyses previous data, the dataset for this paper includes the aforementioned raw data and processed spreadsheet, as well as a detailed report with further processed descriptions of the events [25]. The study was approved by the Ethical Council of the Institute of Energy Technology and complied with their ethical guidelines.

Analysis Procedure

To perform the analysis for this report, the first author first read the report by Massaiu and Holmgren [25], which gives a detailed description of the scenarios for each crew. This was to better familiarise himself with the terminology and to look for possible connections to theory. Second, he analysed the detailed spreadsheets, marking events that were of particular interest for further processing. Third and final, he reinspected the video and audio material to find examples of specific dialogue exchanges, and to listen for cues, such as tonality, formality of word usage, volume in the speech, overall to get a better understanding of the marked events. In order to ensure that the context was properly understood, the procedure for reinspecting an event was to start the recording approximately one minute before the timestamp of the event and to end approximately one minute after the event had ended. The purpose of the reinspection was to ensure that the writing of the process expert (which did not contain comments about tonality, etc.) was not misunderstood. This was especially important when the process expert had written ad verbatim quotes from the operator dialogue.

Following the reinspection of the data, the first author noted all examples of good behaviour (as commented by the process expert) in the aforementioned spreadsheet with a green mark and all examples of errors and deviations with a red mark. Following this, these behaviour examples were coded into categories and the frequency of each category was counted to give an overview of the performances.

Based on these steps, behavioural patterns of the highest, medium and lowest scoring crews as defined by the crew performance ratings were compared.

Results

In this section, we show the behavioural findings from the simulation scenarios. The results are divided by scenario, and for each scenario we describe the behavioural patterns of the top, medium and lowest scoring crew(s). The differences between

Table 2 Crew size, duration of scenario and performance scores for Scenario 1

Scenario 1 performance scores					
Crew	1	2	3	4	5
Size	5	5	3	4	3
Alarm handling	5	4	4	3	5
Identification and isolation	3	4	3	3	4,5
Cooldown	3	5	4	4	5
Depressurization	4	1	4	3	4,5
Stop safety injection	n/a	5	5	2	5
Pressure balance	n/a	3	2,5	1	4
Average	3.8	3.7	3.8	2.7	4.7
Duration	02:06:21	01:54:47	02:50:00	01:52:51 ^a	02:10:37

^aScenario was stopped before the crew had completed the final goal

these behavioural patterns and the extent to which they are caused by the use of procedures and decision-making biases will be discussed in Section ‘[Discussion](#)’.

Scenario 1

In Scenario 1, the major point of divergence in team performance lied in identifying that both Steam Generator #2 and #3 suffered from structural damage. Two of five crews detected that there were problems with both steam generators, whereas the remaining three crews proceeded as if only one steam generator was leaking at any given time. However, all teams were challenged in this detection, as it adhered to procedure and was plausible that any effects observed from Steam Generator #2 could have been caused by ‘shine’ (which is the term for radioactive measures spilling over from a larger nearby rupture). The crew performance scores for Scenario 1 are found in Table 2. Problems were observed for all teams, and radioactive material was released to the atmosphere by all crews (crew 5 was the only crew who did so intentionally). As was previously reported by Massaiu and Holmgren [24], crew size did not predict performance and the teams with a Shift Technical Advisor did not outperform teams without a person performing this role.

In the following, we characterise the differences between the top, medium and lowest scoring crews, by describing their behavioural patterns. Differences in these behavioural patterns will be discussed in concordance with our hypotheses in Section ‘[Discussion](#)’.

Factors Causing High Performance

The highest scoring crew was crew 5, with an average performance score of 4.7. Four aspects were observed that may have caused this performance: First, the process expert observed few technical errors and/or instances of suboptimal execution. Second, the crew was very active in looking for alternate sources of information and in testing multiple hypotheses. For example, crew 5 was the only crew to autonomously ask for information about Steam Generator #2 integrity, follow up to ensure that they received the information and perform actions to isolating Steam Generator #2. The focus on looking for alternate sources and on investigating alternate hypotheses was also visible in the language used in strategy meetings, where crew members would use utterances such as *'ruptured steam generator or generators'* or *'we can't tell WHICH steam generator'* to describe the problem. Third, the Unit Supervisor did not read aloud the notes in the procedures and was, along with the rest of the crew, very proactive in planning future actions. Specifically, the crew frequently called for status updates, wherein ideas were discussed and shared and plans were laid for future actions. Fourth, the Unit Supervisor in crew 5 would read all procedure steps aloud before any actions were taken, whereas in other crews, the Unit Supervisor would read each procedural one step at a time as they were completed.

Factors Causing Medium Performance

The medium scoring crews were crew 1, 2 and 3, with average performance scores of 3.8, 3.7 and 3.8. Of these, one crew, crew 3, identified that both Steam Generator #2 and #3 had suffered structural damage. However, all three crews considered, to varying degrees, whether Steam Generator #2 had leaked as well. The different strategies for investigating these considerations were as follows: Crew 3 called a 'Field Operator' (roleplayed by the control room), but without opening sampling valves (thus not allowing for sampling), and did not call the field operator back after the sampling valves had been opened. After some deliberation in the experimenters' gallery, the 'Field Operator' decided to call the crew and share the information that both Steam Generator #2 and Steam Generator #3 showed radiation, after which the crew promptly performed actions towards isolating Steam Generator #2 as well. However, the crew used Steam Generator #2 for cooldown. Crew 1 and 2 discussed the possibility of a leak in Steam Generator #2, opened sampling valves and sent a 'Field Operator' to collect samplings. However, both crews did not follow up with the Field Operator and abandoned the hypothesis that Steam Generator #2 could be leaking as well. Compared to crew 5, the three middle scoring crews used singular terms about the steam generator problem early on, such as *'the ruptured steam generator'* (crew 1). Notably, some medium scoring crews originally believed the measures in Steam Generator #3 were due to shine (when it had first leaked) and then changed their hypothesis to it being the Steam Generator #2 measures that were caused by shine. These crews thus essentially did not change the hypothesis that it was only one steam generator that was faulty—they simply changed their mind about which one it

was. Finally, the medium scoring crews tended to search for information that backed up (rather than falsified) their views, as is reflected in language such as *'request chemistry back that up with local sample'* (crew1).

Factors Causing Low Performance

The lowest scoring crew was crew 4, with an average score of 2.7. Two aspects were observed that may have caused this performance: First, the process expert observed technical errors in executing the procedures. The effects of these errors caused ripple effects that made the scenario more and more complex, eventually resulting in the scenario being ended before depressurization was achieved. Second, we observed examples of inefficient communication between crew members. For example, the crew's Reactor Operator and Shift Technical Advisor suggested several times to perform steps to test whether Steam Generator #2 was also leaking. However, the Unit Supervisor was of a different belief, and thus did not translate the recommendations into actions towards isolation. As a result, the crew quickly abandoned the possibility that two steam generators were damaged and instead focused on Steam Generator #3.

Scenario 2

In Scenario 2, the major point of divergence was the degree to which teams chose to invest resources into identifying the leak locations by going outside of the procedures. These differences are detailed below. Overall, the impact of choosing to use resources to identify and isolate the leaks varied: Teams hit procedural goals at comparable speeds, with crew 3 being slightly faster. Using resources for non-procedural operations thus did not slow down the progression through procedures.

Factors Causing High Performance

The highest scoring crew was crew 2 with an average performance score of 4.0. Two aspects were observed that may have caused this performance: First, the crew was one of two crews (along with crew 5) that invested heavily into finding the location of the leaks. The crew decided early on that they needed information from external sourced information. Therefore, they communicated frequently with 'Field Operators' (roleplayed by the Control Room) throughout the scenario. Based on these communications, they were able to identify both leaks and to perform actions towards isolating them. Second, the crew's Unit Supervisor chose not to read notes in the procedures aloud. In addition to these performance measures, the crew was furthermore the only crew to execute on restoring water to the Refuelling Water Storage Tank.

Factors Causing Medium Performance

The medium scoring crew was crew 1, with an average performance score of 3.3. One aspect was observed that may have caused this performance: Compared to two of the lower scoring crews, crew 4 discussed and performed some preliminary actions towards identifying and isolating the leak. However, rather than spending resources on investigating further the exact locations, they tried to deduce the locations from secondary information instead of testing their hypotheses with alternative sources of information, whereas crews 2 and 5 relied on dialogue with 'Field Operators'. Based on this information gathering, crew 1 was able to achieve some degree of identification of the location of the leaks and to perform some isolating actions.

Factors Causing Low Performance

The lowest scoring crews were crew 3, 4 and 5, with average performance scores of 2.3, 2.0 and 2.5, respectively. Different aspects were observed for each team that may have caused this lower performance: For crew 3 and 4, the lower score was caused by the fact that they did not attempt to identify and isolate the leaks. For crew 5, their lower score was caused by suboptimal performance during attempts at isolating the leak in the reactor coolant system and during cooldown.

Crew 3 and 4 had preliminary suspicions and discussions about the possibility of a leak. Crew 3 decided early on that there was only one minor leak and to not spend resources on communicating with a field operator regarding alternative information—they only communicated with the Field Operator for practical tasks, such as energising valves. Crew 4 discussed calling an FO to investigate further, but decided not to do so as they believed there to be too many possible candidates for the leak location.

A common factor for the lower scoring crews was that they did not show appropriate patience in watching the effects of the procedure actions. As a result, the crews entered subsequent procedures based on misleading information about whether or not the preceding procedure had been effective. Crew 5 attempted to compensate for this by rerunning procedures while simultaneously entering another procedure. This increased the workload on the crew, which may in turn have caused the lower performance with regard to cooling.

As a final factor, crew 5 was, as in Scenario 1, very active in calling for strategy briefs and in collaborative planning of future steps (Table 3).

Discussion

This study investigated the decision-making and performance of five crews of nuclear control room operators in two realistic simulated scenarios. The scenarios involved non-typical situations, which were caused by multiple failures in the nuclear power

Table 3 Crew size, duration of scenario and performance scores for Scenario 2

Scenario 2 performance scores					
Crew	1	2	3	4	5
Size	5	5	3	4	3
Attempt to identify?	Limited	Yes	No	No	Yes
Identification of leak in RCS	3	4	1,5	1,5	2
Identification of leak in RHR	3	4	1,5	1,5	4,5
Cooldown	4	4	4	3	1
Average	3.3	4.0	2.3	2.0	2.5
Duration	01:26:21	01:14:47	02:10:00	01:12:51 ^a	01:30:37

^aScenario was stopped before the crew had completed the final goal

plant system and could only be detected in full through deviation from procedures, such as autonomous requests or search for additional information. We found that some crews strictly adhered to procedures despite this leading to suboptimal performance. Furthermore, we found that crews did generally not persist in the mindset of testing multiple hypotheses once procedures had been entered. This was seen in the reformulation and/or further commitment to the hypothesis that only a single steam generator was leaking, which was consistent with the procedure in Scenario 1 and a rationalisation/explaining away of reasons to go outside of procedures in Scenario 2. Crucially, these behavioural patterns were observed despite the fact that, generally, teams who also pursued actions outside of procedures were not slower or less accurate. Furthermore, we found examples of both types of bias behaviour. In the following, we elaborate our findings with respect to each bias and suggest implications for design.

Expertise

Despite crews receiving extensive training in the simulated power plant, we observed several possible indicators of biased behaviour due to misapplied expertise: First, in Scenario 2, the decision of crew 1, 3 and 4 to not allocate additional resources towards detecting the leaks could have been an optimal strategy at their home plant. Their expertise may have thus guided them not to continue, as conserving resources would lead to greater success. Second, in Scenario 2, crew 1 chose not to collect additional information from outside sources as they believed they could reach sufficient data from secondary calculations in the control room, and crew 4 decided not to pursue identification because they believed there to be too many possible causes. These behavioural patterns are consistent with the tendency not to seek for additional information due to expertise, which, in this case, caused suboptimal performance.

Confirmation Bias

Early Hypotheses

Despite not being required to do so by the procedures, we observed that all crews created hypotheses about the cause of the problems early on in both scenarios. Furthermore, in both scenarios, we found that for the majority of crews, the first hypotheses persisted through the entire scenario. Deviations were seen in Scenario 1, where crew 5 continuously explored multiple hypotheses throughout the scenarios (thus never committing to just one hypothesis) and where crew 3 committed to a hypothesis but reconsidered when salient outside information from a field operator caused them to reconsider. In particular, as evidenced by the performance of crew 4 in Scenario 1, it seemed that the early beliefs of the Unit Supervisor were especially important for choice of strategy. These observations are thus consistent with the notion that operators were affected by confirmation bias in the form of commitment to early hypotheses.

Confirmatory Search

As expected, we found several examples of confirmatory search. In Scenario 1, all but one crew had adopted the hypothesis, which was consistent with the procedures, that only one steam generator was damaged. To test this hypothesis in a non-confirmatory manner, crews would need to look for information about the integrity of the other steam generators and see whether there was damage in multiple locations. Furthermore, given that there was a considerable delay between Steam Generator #2 and #3 leaking, the hypothesis that only a single steam generator was damaged was true for an extended period of time. To avoid confirmatory search, crews would thus have to continuously look for changes in information that caused the previously true hypothesis to be false. We observed that only one crew, crew 5, employed a strategy that allowed them to continuously search for alternate sources of information, while another crew, crew 3, was given alternate information by the field operators, which prompted them to adopt the true hypothesis that two leaks had occurred. For the remaining crews, two sources of information were used to confirm the original hypothesis, which caused the teams not to search for additional information. First, after the rupture in Steam Generator #3 had reached its maximum, the severity of its effects was much larger. Consequently, crews could readily explain radiation measures from Steam Generator #1 and Steam Generator #2 as 'shine' effects which is a common occurrence and thus a theoretically valid data point to confirm that there was only damage to Steam Generator #3. Second, due to the relative small size of the Steam Generator #2 leak, it was difficult to detect the effects of the leak due to the presence of the large rupture in Steam Generator #3. In fact, looking at the instruments in the nuclear control room, the pressure level was stable in Steam Generator #2 for extended periods of time. This could be interpreted as a valid data point for

confirming that only Steam Generator #3 was damaged. These factors thus served as salient factors for relying only on confirmatory search.

Implications for Design

Our results suggest that the presence of and design of the procedures may have been conducive to increased risk of being influenced by these biases. In Scenario 1, we found that all crews made initial efforts to obtain local samples through communications with a field operator, but only one crew abandoned these efforts after emergency procedures were entered, as the procedures did provide a follow-up. In Scenario 2, several crews specifically chose not to pursue any additional actions towards diagnosing the problem, as the procedures did not require doing so. These results thus suggest that the fact that procedures did not require diagnosis of the problem, nor encouraged operators to look for alternate sources of information, dissuaded crews from exploring alternate hypotheses.

While our findings are exploratory, rather than confirmed in a deductive way, previous studies in design research have shown that interaction with written materials can cause a decrease in idea generation [18, 19, 37], which has been explicitly linked to confirmation bias in designers [38]. Furthermore, research has shown that confirmation bias influences how pilots plan decisions in adverse weather conditions [39] and how military analysts prioritise information [40]. Therefore, the design of procedures and checklists that prevent (or minimise) these biases has great potential for improving performance and, thus, safety in these fields. Research has been conducted that shows some success with regard to minimisation of design fixation [21] and debiasing decision-makers through design [39, 40], as well as reducing confirmation bias in designers [38]. In particular, the use of counterfactuals, meaning examples that are opposite to the observed events or encourage thinking of the opposite of the present view, has been successful in combating confirmation bias [40, 41]. Our findings showed that successful teams employed counterfactual checks as part of their strategy, which resulted in fewer examples of biased behaviour. Therefore, we suggest that procedures created to explicitly include counterfactual checks may be successful in increasing the performance of nuclear control room operators and other operators in emergency environments through decreased confirmation bias. Another avenue for decreasing confirmation bias could be the construction of decision matrices, which have been successful in reducing confirmation bias in designers [38]. However, further research is needed to validate these proposals for operators in emergencies.

Conclusion

This paper investigated biased decision-making in realistic extra-procedural nuclear control room scenarios. It focused on two biases that have been related to expert decision-making: The first is the bias that occurs when expertise is transferred to a similar but different situation, thus causing misapplications of one's expertise. The second is confirmation bias, which is the tendency to overly prioritise and seek for information that benefits existing views. This paper presented a reanalysis of data from two realistic simulated accidents at a pressurised water reactor, which were conducted with five nuclear control room operating crews. The scenarios required operators to perform autonomous actions outside of procedures to achieve optimal performance. We investigated whether the misapplication of expertise and confirmation bias caused suboptimal performance in the applied case of a nuclear control room emergency. While the study presented is exploratory, and the findings have thus not been validated in a deductive manner, findings in this paper provide evidence that both biases could explain differences in performance. Furthermore, we found that the use of procedures may have increased the effect of confirmation bias. We concluded by discussing implications and opportunities for the design of procedures.

Acknowledgements We thank our research partners from the Halden HAMMLAB at the Institute for Energy Technology (IFE) in Norway for their continued support, in particular, Andreas Bye, Lars Holmgren, Salvatore Massaiu, Espen Nystad and Stine Strand. The work reported in this paper is part-funded by the OECD Halden Reactor Project.

References

1. Burian BK (2006) Design guidance for emergency and abnormal checklists in aviation. In: Proceedings of the Human Factors and Ergonomics Society 50th Annual Meeting, vol 50, pp 1–6
2. Burian BK (2005) Do you smell smoke? Issues in the design and content of checklists for smoke, fire, and fumes. In: Emergency and abnormal situations symposium
3. Lau N et al (2008) Ecological interface design in the nuclear domain: an application to the secondary subsystems of a boiling water reactor plant simulator. *IEEE Trans Nucl Sci* 55(6):3579–3596
4. Weyer U, Braseth AO, Eikås M, Hurlen L, Kristiansen P, Kvalem J (2010) Safety presentation in large screen displays - a new approach. In: SPE intelligent energy conference and exhibition
5. Braseth AO, Øritsland TA (2013) Visualizing complex processes on large screen displays: design principles based on the information rich design concept. *Displays* 34(3):215–222
6. Wreathall J, Reason J (1993) Latent failures and human performance in significant operating events. JWC Co. Inc
7. U.S. Nuclear Regulatory Commission (2002) Review of findings for human performance contribution to risk in operating events (NUREG/CR-6753, INEEL/EXT-01-01166)
8. Massaiu S (2010) Critical features of emergency procedures: empirical insights from simulations of nuclear power plant operation. *Training*, pp 277–284
9. Du P, MacDonald EF (2015) Products' shared visual features do not cancel in consumer decisions. *J Mech Des* 137(7):71409

10. Du P, MacDonald EF (2013) Eye-tracking data predicts importance of product features and saliency of size change, vol 5. In: 25th Int Conf Des Theory Methodol ASME 2013 Power Transm Gearing Conf, vol 136, no 8, p V005T06A024
11. Goucher-Lambert K, Cagan J (2014) The impact of sustainability on consumer preference judgments (DETC2014-34739). In: ASME 2014 Int Des Eng Tech Conf Comput Inf Eng Conf, vol 137, no c, pp 1–11, Aug 2014
12. Goucher-Lambert K, Moss J, Cagan J (2017) Inside the mind: using neuroimaging to understand moral product preference judgments involving sustainability. *J Mech Des* 139(4):41103
13. Faerber SJ, Carbon CC (2013) Jump on the innovator's train: cognitive principles for creating appreciation in innovative product designs. *Res Eng Des* 24(3):313–319
14. Reid TN, MacDonald EF, Du P (2013) Impact of product design representation on customer judgment. *J Mech Des* 135(9):91008
15. Wei ST, Ou LC, Luo MR, Hutchings JB (2014) Package design: colour harmony and consumer expectations. *Int J Des* 8(1):109–126
16. Perez Mata M, Ahmed-Kristensen S, Brockhoff PB, Yanagisawa H (2017) Investigating the influence of product perception and geometric features. *Res Eng Des* 28(3):357–379
17. Andersen E, Maier A (2017) The attentional capture of colour in visual interface design: a controlled-environment study. In: Proceedings of the international conference on engineering design, ICED, vol 8, no DS87-8
18. Jansson DG, Smith SM (1991) Design fixation. *Des Stud* 12(1):3–11
19. Vasconcelos LA, Crilly N (2016) Inspiration and fixation: questions, methods, findings, and challenges. *Des Stud* 42:1–32
20. Nikander JB, Liikkanen LA, Laakso M (2014) The preference effect in design concept evaluation. *Des Stud* 35(5):473–499
21. Viswanathan VK, Linsey JS (2013) Design fixation and its mitigation: a study on the role of expertise. *J Mech Des* 135(5):51008
22. Kahneman D, Klein G (2009) Conditions for intuitive expertise: a failure to disagree. *Am Psychol* 64(6):515–526
23. Gigerenzer G, Gaissmaier W (2011) Heuristic decision making. *Annu Rev Psychol* 62(1):451–482
24. Massaiu S, Holmgren L (2014) Diagnosis and decision-making with emergency operating procedures in non-typical conditions: a HAMMLAB study with U.S. Operators. In: HWR-1121. Halden, Norw. OECD Halden React. Proj.
25. Massaiu S, Holmgren L (2016) A HAMMLAB HRA data collection with U.S. operators. HWR-1123. In: HWR-1123. Halden, Norw. OECD Halden React. Proj. OECD Halden Reactor Project, Halden, Norway
26. Nickerson RS (1998) Confirmation bias: a ubiquitous phenomenon in many guises. *Rev Gen Psychol* 2(2):175–220
27. Simon HA (1992) What is an explanation of behavior? *Psychol Sci* 3(3):150–161
28. Shanteau J (1992) Competence in experts: the role of task characteristics. *Organ Behav Hum Decis Process* 53(2):252–266
29. Shanteau J (1992) How much information does an expert use? is it relevant? *Acta Psychol (Amst)* 81(1):75–86
30. Dreyfus SE (2004) The five-stage model of adult skill acquisition. *Bull Sci Technol Soc* 24(3):177–181
31. Bruner JS, Potter MC (1964) Interference in visual recognition. *Science* 144(3617):424–425
32. Ross L, Anderson CA (1982) Shortcomings in the attribution process: on the origins and maintenance of erroneous social assessments. *Judgm Under Uncertain Heuristics Biases* 1977:129–153
33. Anderson CA, Lepper MR, Ross L (1980) Perseverance of social theories: the role of explanation in the persistence of discredited information. *J Pers Soc Psychol* 39(6):1037–1049
34. Baron J (1995) Myside bias in thinking about abortion. *Think Reason* 1(3):221–235
35. Wason PC (1960) On the failure to eliminate hypotheses in a conceptual task. *Q J Exp Psychol* 12(3):129–140

36. Griffin D, Tversky A (1992) The weighing of evidence and the determinants of confidence. *Cogn Psychol* 24(3):411–435
37. Vasconcelos LA, Cardoso CC, Sääksjärvi M, Chen C-C, Crilly N (2017) Inspiration and fixation: the influences of example designs and system properties in idea generation. *J Mech Des* 139(3):31101
38. Hallihan GM, Cheong H, Shu LH (2012) Confirmation and cognitive bias in design cognition, In: Volume 7: 9th international conference on design education; 24th international conference on design theory and methodology, p 913
39. Walmsley S, Gilbey A (2017) Debiasing visual pilots' weather-related decision making. *Appl Ergon* 65:200–208
40. Cook MB, Smallman HS (2008) Human factors of the confirmation bias in intelligence analysis: decision support from graphical evidence landscapes. *Hum Factors J Hum Factors Ergon Soc* 50(5):745–754
41. Galinsky AD, Moskowitz GB (2000) Counterfactuals as behavioral primes: priming the simulation heuristic and consideration of alternatives. *J Exp Soc Psychol* 36(4):384–409

Part VIII
Design Modelling

Modeling Collaboration in Parameter Design Using Multiagent Learning



Daniel Hulse, Kagan Tumer, Chris Hoyle and Irem Tumer

This paper presents a model of collaboration in multidisciplinary engineering based on multiagent learning. Complex engineered systems are often designed through the collaboration of many designers or experts. A variety of frameworks have been presented and put in practice to help manage this collaboration, with good results; however, there have been few attempts to create an underlying model of collaboration. Previous attempts to model collaboration in design have produced results contrary to the orthodoxy that collaboration is helpful, showing that the best design team maximizes autonomy. This is most likely because these previous models were focused upon computational synthesis as opposed to distributed design. The model presented in this work uses computational designers acting on a multidisciplinary design model with cross-disciplinary constraints to better represent design in collaborative scenarios in which designers are delegated different interacting components. Collaboration and autonomy are then modeled by having these computational designers propose changes to the design together or apart. Results on this model show collaboration leading to better design outcomes, due to the ability of designers to propose joint changes to the design. This shows that collaboration can benefit design processes not just by reducing lag but also by enabling designers to explore the design space in a way which leads to higher value solutions.

Introduction

Engineers rarely design alone. Engineered systems such as aircraft [1], automobiles [2], space systems [3], power plants [4], and a host of others are designed by teams with members having expertise in the subjects relevant to what they design. As a

D. Hulse · K. Tumer · C. Hoyle (✉) · I. Tumer
Oregon State University, Corvallis, USA
e-mail: chris.hoyle@oregonstate.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_31

result, engineers must collaborate with others to design a system which works not just at the part or discipline-level, but as a whole. In practice, the designers of interfacing subsystems communicate about how they will interact in a number of ways including design reviews, face-to-face communication, phone conferences, and text chat [5].

Managing this multidisciplinary collaboration is at the core of complex engineered systems design frameworks. Systems engineering focuses on the practice of developing the system architecture, defining interfaces between subsystems, and negotiating trade-offs between subsystems, effectively delegating the task of coordinating the design of components to the systems engineer(s) [6]. Concurrent engineering and integrated product development have been adopted as management frameworks for approaching the increasing complexity of engineered systems since the 1950s and 1960s, promising a reduction in development cycles [7]. Concurrent engineering achieves this by involving all functional areas in the design process concurrently, rather than sequentially designing and revising the product based on the expertise of each functional group [8]. A special case of concurrent engineering is integrated concurrent engineering, which manages the cross-functional design activities by radically co-locating engineers to perform the high-impact design activities requiring collaboration, decreasing communication lag, and resulting in a faster process [3, 9].

While these frameworks have been introduced and adopted out of practical necessity and have been shown empirically to work well, there is not yet an underlying general theory of how teams work—most studies focus on the success of a methodology in a given context [10]. Nevertheless, interaction and overlap of design activities have been shown across studies to have a positive effect [10]. Principles have been proposed based on their ability to help the design process overcome the bounded rationality of individual designers [11]. Additionally, it has been shown that coupled design features increase iterations in design processes [12]. Finally, team performance on a given task has been shown to be highly sensitive to coordination, especially when individual team members utilize specialized knowledge [13].

Some work has been done to study distributed parameter design. It has been found that, for individuals acting on coupled tasks, completion time increases geometrically with the number of variables [14]. This was confirmed in [15], which also studied the effect of distributing design problems throughout teams, showing that collaboration took up a large percent of the total team effort which increased with team size. While this paper went on to identify this increased effort as a cost to increasing the number of designers, it did not identify whether that collaboration was necessary or beneficial—only that it occurred. More recent work has shown that collaboration and synchrony between designers correlates with better design outcomes in design teams [16]. Modeling designer behaviors using Markov processes have been presented as a path for future research [17].

Modeling engineering teams with Markov-based multiagent models has also been used to study how teams react to design changes [18], the effect of designer search behaviors [19], and the effect of design problems on optimal team structure [20]. While these models have been constructed to be psychologically accurate [19, 21], it is unclear that they apply to the technical aspects of distributed parameter design. This is because the problems are quite different: in team-based computational syn-

thesis, agents can make any change to a design at any time, which in turn changes the graph of the design, while in parameter design, agents can only change the local value of certain variables, which only changes the values of other parameters—not the overall structure. As a result, this research contends that the papers studying optimal team structure find that the best team structures maximize designer autonomy [20]—contradicting research in distributed parameter design [16] and team performance literature [13] and implicitly putting some of the justification for concurrent engineering [3, 9, 10] in doubt—because they do not represent the technical aspects of distributed parameter design well.

Aims

The aim of this paper is to provide a computational model for design teams that represents distributed parameter design, with a focus on representing the interaction between multidisciplinary design problems and designer behaviors. This model will then be used to study how designer independence and collaboration affect design outcomes. The hypothesis presented in this paper is that collaboration between designers in parameter design yields better design outcomes compared to teams in which there is no collaboration, and that this is a quality of systems with interdisciplinary design constraints. That is, this work contends that when two or more designers are given individual parts of a problem to act on, they must explore joint design choices to navigate their interacting design constraints to reach a truly optimal design outcome.

Significance

Collaborative engineering frameworks have been widely adopted in industry, however, with few underlying models of design, it is difficult to parse why these frameworks have positive effects. This research is significant because it provides an underlying model for why one of the most important aspects of these frameworks—collaboration—is beneficial in design settings. Having this underlying model can, in turn, inform the adaptation of design frameworks or generation of new design frameworks to different design problems based on the principles at play.

Modeling Distributed Design with Agents

Modeling distributed parameter design requires two things: a problem scenario that encompasses the technical aspects of distributed parameter design and a system capable of modeling different team behaviors. In this paper, the technical aspects of distributed parameter design are represented with a multidisciplinary model, while

the team behaviors are represented using a distributed multiagent learning system. These are discussed below.

Modeling the Multidisciplinary Problem Domain

In distributed parameter design, different parameters in a design problem are delegated to different designers. The values of these parameters may be communicated with other team members or not, depending on the type of communication protocol. Interestingly, this is nearly the same as a multidisciplinary analysis or multidisciplinary design optimization problem, a problem in which multiple models have information dependencies and cross-disciplinary requirements [22]. While various multidisciplinary design optimization frameworks provide methods of coordinating disciplinary optimizations through communication protocols in a distributed archi-

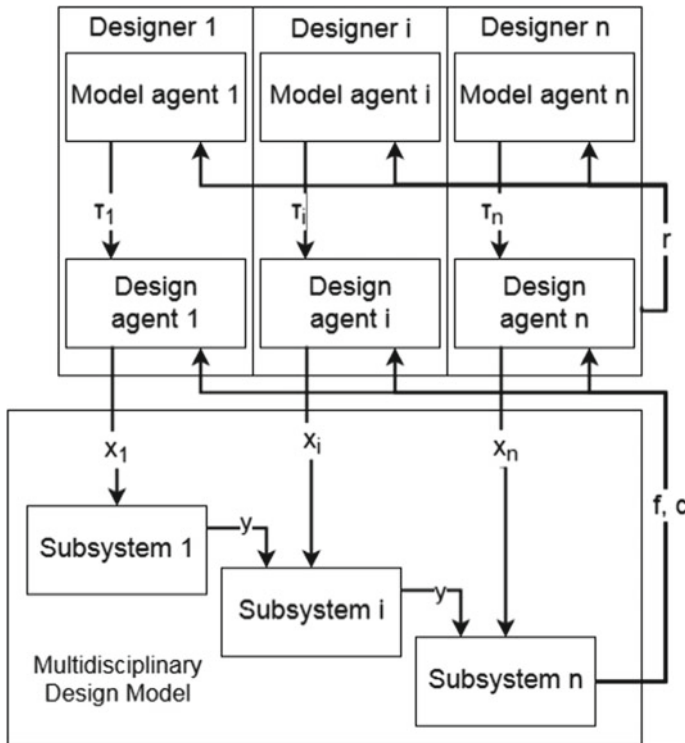


Fig. 1 Overview of the model presented in this work. Designers, each composed of a model agent and a design agent, are delegated each parameter in a multidisciplinary model. The design agent selects a parameter value x_i based on its objective and feasibility f, c , while the model agent determines the level of exploration τ_i based on the increase of design agent knowledge r

Table 1 Design variables in model

Component	Design choice	Parameter range/Options
Motor	Motor	9
Battery	Cell	6
	Cells in series	7
	Cells in parallel	4
Propeller	Airfoil	7
	Diameter	0.02–0.2 m
	Angle	0–45°
	Twist	(unitless)
	Chord	0.005–0.02 m
	Taper	0–1 (unitless)
Rod	Material	4
	Thickness	0.0009–0.06 m
	Width	0.0065–0.0380 m
	Height	0.0065–0.0380 m
ESC	ESC	6
Landing skid	Material	4
	Leg angle	20–60°
	Diameter	0.0065–0.0380 m
	Thickness	0.0009–0.006 m
Mission	Climb velocity	0.1–30 m/s
	Steady flight angle	0.1–45°

ecture, representing designer collaboration in this paper is done in a monolithic all-in-one model for simplicity. That is, to separate the aspects of communication (sharing information about parameter values) from collaboration (designing together in a coordinated way), it is assumed in this work, that all designers act on the same model which happens to calculate the interdisciplinary dependencies and resulting performance automatically as a function of the parameters chosen at any point in time. This is consistent with a modern computational team environment in which designers can share and change model parameters and aspects of a design instantly.

When a design is delegated across members of a team, designers are often delegated different components to design. However, these components often have interfacing constraints or requirements which must be fulfilled to have a working design. To model this situation, a quadrotor design problem outlined in [23] is used which represents a few of the heterogenous challenges that are presented to designers in a real system. The multidisciplinary problem has several different components (the battery, motors, design rods, landing gear, etc.) with both discrete (e.g., battery configuration) and continuous (e.g., thicknesses, lengths) parameters and cross-disciplinary constraints which must be satisfied for a feasible solution (e.g., the motor must pro-

Table 2 Constraints in model

Component	Constraint	Explanation
System	$c_1 = \text{failure} = 0$	Model must not return an error
	$c_2 = 1 - \frac{r_{\text{hov}} + \text{tol}}{T_{\text{req}}} \leq 0$	System must produce enough thrust
Motor	$c_3 = \frac{I_{\text{oper}}}{I_{\text{mmax}}} - 1 \leq 0$	Current drawn must not exceed max
	$c_4 = \frac{P_{\text{oper}}}{P_{\text{mmax}}} - 1 \leq 0$	Power drawn must not exceed rating
Battery	$c_5 = \frac{I_{\text{oper}}}{I_{\text{bmax}}} - 1 \leq 0$	Current drawn must not exceed max
	$c_6 = \frac{P_{\text{oper}}}{P_{\text{bmax}}} - 1 \leq 0$	Power drawn must not exceed rating
Propeller	$c_7 = \frac{\sigma_p}{\sigma_{\text{pmax}}} - 1 \leq 0$	Must not exceed maximum stress
Rod	$c_8 = 1 - \frac{f_{\text{sysnx}}}{3 * f_f} \leq 0$	Natural frequency in the x over three times the motor frequency
	$c_9 = 1 - \frac{f_{\text{sysny}}}{3 * f_f} \leq 0$	Natural frequency in the y over three times the motor frequency
	$c_{10} = \delta_x - 0.01L_r \leq 0$	Must not deflect one percent length
	$c_{11} = \frac{\sigma_x}{\sigma_{x,\text{max}}} - 1 \leq 0$	Must not exceed max stress in the x
	$c_{12} = \frac{\sigma_y}{\sigma_{y,\text{max}}} - 1 \leq 0$	Must not exceed max stress in the y
ESC	$c_{13} = 1 - \frac{V_b}{V_{e\text{min}}} \leq 0$	Must support input voltage
	$c_{14} = \frac{V_b}{V_{e\text{max}}} - 1 \leq 0$	Must support input voltage
	$c_{15} = \frac{I_{\text{oper}}}{I_{e\text{max}}} - 1 \leq 0$	Current must not exceed ESC max
Landing skid	$c_{16} = \frac{F_S}{F_{S\text{max}}} - 1 \leq 0$	Must absorb force in a minor fall

vide enough thrust to carry the rest of the craft). The performance of the craft is judged by its performance attributes in a theoretical mission, with the various objectives folded into a single metric of utility based on the performance of that mission

While a full description of this model is out of the scope of this paper, Tables 1 and 2 show the design variables of the problem as well as resulting constraints. Of note is that this model allows designers to change parameters which result in differing performances, but in constraint values based on the general consistency of parts. For this paper's purposes, this model provides an abstraction of a complex design scenario characterized by the cross-disciplinary requirements and interactions which are typically used as justification for a collaborative design process (Fig. 1).

Modeling Design with Multiple Learning Agents

In this research, designers are delegated the design of different parameters in the model and are hierarchically composed of two learning agents. The hierarchical agent structure represents a high-level and low-level decision that must be made by a designer: the high-level decision to explore new regions of the design space or leverage known design information and the low-level decision of which resulting specific parameter values to pick. Each designer is accordingly represented as two learning agents:

- a **model agent**, which determines the level of exploration or exploitation used by the design agent based on new knowledge gained by the design agents and
- a **design agent**, which determines the parameter value to use in a design based on previous objective and constraint values.

Both the model agent and design agent use the basic framework of learning from the environment to find the best actions which have been used in the field of multiagent learning to optimize distributed tasks such as air traffic management [24] and multi-robot coordination [25]. While the model agent uses a traditional reinforcement learning approach covered in [26], the design agent is based on our previous work in [27] covering a specialized learning heuristic for general optimization using multiple learning agents.

Representing the designer as a combination of the design agent and model agent processes captures a few heuristic processes which real designers in multidisciplinary design teams use, such as:

- creating new designs based on changing the current best design, as is done in the embodiment design tasks represented in distributed parameter design in which a concept has been selected but the set of parameters has not been chosen [15]. This is encoded as the best value each design agent has, which is always the most probable action;
- seeking out different values for parameter values based on “surprise,” or the increase in value over a prior, similar to how multidisciplinary research teams reshape themselves based on new information about interdisciplinary research teams [28]. This is encoded as the learning of the model agent, which encourages exploration that results in new design agent knowledge.

Furthermore, the distributed aspects of team-based design are represented in this model by delegating each computational designer the design of a parameter, enabling collaboration and autonomy to be modeled by allowing the designers to choose values all at once and keeping the values of each of the other agents’ parameters constant while each agent chooses its parameter value, respectively. The processes used by the agents which make up each designer are as follows, starting with the design agent for clarity:

Design Agent Parameter Selection

A design agent, following the representation introduced in our previous work [27], selects parameter values to be used in the design and receives rewards based on the resulting objective and constraint values when that parameter value is used with those picked by the other design agents. The design agent for each parameter iteratively follows the following process:

- It stochastically chooses a value for its assigned parameter value based on its stored relative merit.
- An objective and feasibility value is calculated based on the parameter values picked by it and the other design agents.
- If this merit of the design is better than the currently stored merit of the parameter value, the stored merit is updated to the new value.

This process follows the basic framework of action selection, rewards, and learning used in the field of multiagent learning [26]. However, in this case, actions are the design parameter values available to the design agent, rewards are the resulting objective function and feasibility values, and learning is accomplished via a custom heuristic. Parameter values are selected by each design agent by first normalizing the stored values and then using softmax action selection. Since both constraint and objective values are stored, these are combined using a penalty (as in [29], but left constant) for action selection using

$$M(x_j) = f_s(x_j) - \sigma c_s(x_j)$$

where $M(x_j)$ is the merit of parameter value x_j , $f_s(x_j)$ is the stored objective value, σ is a penalty factor, and $c_s(x_j)$ is the stored feasibility value given by the sum of the squares of the constraints. This merit is then normalized with the softmax normalization per [29] to accommodate the scale of functions using

$$M_n(x_j) = \frac{1}{1 + e^{-\frac{M(x_j) - \mu_x}{\sigma_x}}}$$

where $M_n(x_j)$ is the normalized merit, and μ_x and σ_x are the mean and standard deviation of the merit function $M(x_j)$. Then, the parameter value is selected for the design agent stochastically based on the probabilities in a softmax selection procedure with probabilities given by

$$p(x_j) = \frac{e^{M_n \frac{(x_j)}{\tau}}}{\sum_{j=1}^m e^{M_n \frac{(x_j)}{\tau}}}$$

where $p(x_j)$ is the probability of choosing parameter value (x_j) , τ is a temperature parameter determining the level of exploration, and $M_n(x_j)$ is the previously normalized merit.

Design Agent Learning

After each parameter value has been selected by each design agent, the reward (in this case, the objective and constraint values) is then calculated using the model. This reward is learned using a specialized heuristic found in earlier work in [27] to allow for learning agents to act as a general optimization process. This heuristic is discussed and justified briefly here, and then described as it is implemented in the multiagent model.

Reinforcement learning is a common approach for agents to adapt to a dynamic, stochastic environment [26] given by the heuristic

$$V(a) \leftarrow V(a) + \alpha(r - V(a))$$

where a is an agent's action, V is the value of that action, r is the reward, and α is the learning rate. Unfortunately, when used alone, this approach can suffer from coordination difficulties since each agent is taking what is effectively a running average of the merit of each of its actions without considering the actions of other agents, a problem which has been overcome on specific domains using more advanced types of reinforcement learning and specialized reward structures [24, 25]. The heuristic used here, on the other hand, exploits a property of optimization problems not present commonly in multiagent systems—a lack of stochasticity in rewards—to provide a learned value of design merit which is accurate within the feasible design space. This learning heuristic is given by

$$V(a) \leftarrow \max(V(a), r)$$

where a is an agent's action, V is the value of that action, and r is the reward.

To understand how this heuristic overcomes the problems inherent in using reinforcement learning on a collaborative optimization problem, consider the simple problem shown on the left side of Fig. 2. In this problem, there are two designers, each with a choice of two components: A and B . Designs with the same letter ($A-A$ or $B-B$) for each component are compatible with each other, yielding a reward of 10 and 15, respectively, while designs with different letters ($A-B$ or $B-A$) for each component are incompatible, yielding extremely low rewards of -4000 and -5000 , respectively. When this problem is traversed as shown, the values for each variable using reinforcement learning are shown on the right side of Fig. 2. As can be seen, the “best” set of actions identified by both agents is not the actual best set of actions, but one of the incompatible designs. This is because the running average taken by reinforcement learning is as influenced by the rewards from poor, incompatible designs as it is from good ones.

Using the heuristic used here, on the other hand, the agents can identify the best design, as shown in Fig. 3. As can be seen, the heuristic accurately represents the feasible design space to the agents, with both compatible designs producing learned values which reflect the true reward. Additionally, while reinforcement learning would

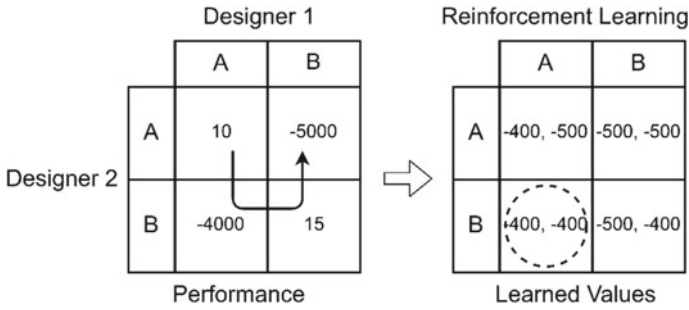


Fig. 2 A simple design problem representing challenges to learning in distributed design: if the agents traverse the space as shown, their learned values are likely to be adversely affected by incompatible, meaningless designs rather than compatible ones

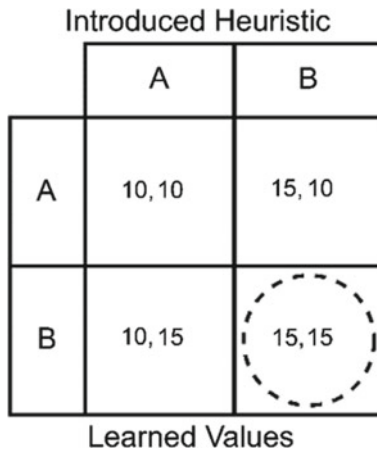


Fig. 3 Learned values of the agents using the custom heuristic used in this work. Using this heuristic, the agents are able to identify the set of actions with the highest reward

change the values based on how much they are exploited, this heuristic keeps a consistent record of design merit for good designs which only changes if a better design is found.

When used in the algorithm presented in this work, this learning heuristic is used both for the objective and constraint values $f_s(x_j)$ and $c_s(x_j)$ such that the feasibility value must be better than or equal to the previous to ensure the feasibility takes precedence over performance. This is shown (along with the calculation of meta-agent rewards) in Algorithm 1.

<p>if $c < c_s(x_i)$ then</p> <p style="padding-left: 20px;">$c_s(x_i) \leftarrow c$</p> <p style="padding-left: 20px;">$f_s(x_i) \leftarrow f$</p> <p style="padding-left: 20px;">$r_{ci}(x_i) = c - c_s(x_i)$</p> <p style="padding-left: 20px;">$r_{fi}(x_i) = \max(f - f_s(x_i), 0)$</p> <p>else if $c = c_s(x_i)$ and $f < f_s(x_i)$ then</p> <p style="padding-left: 20px;">$c_s(x_i) \leftarrow c_s(x_i)$</p> <p style="padding-left: 20px;">$f_s(x_i) \leftarrow f$</p>	<p style="padding-left: 20px;">$r_{ci}(x_i) = 0$</p> <p style="padding-left: 20px;">$r_{fi}(x_i) = \max(f - f_s(x_i), 0)$</p> <p>else</p> <p style="padding-left: 20px;">$c_s(x_i) \leftarrow c_s(x_i)$</p> <p style="padding-left: 20px;">$f_s(x_i) \leftarrow f_s(x_i)$</p> <p style="padding-left: 20px;">$r_{ci}(x_i) = 0$</p> <p style="padding-left: 20px;">$r_{fi}(x_i) = 0$</p>
---	--

Alg 1. Control logic is used to direct design agent learning and model agent rewards. $f_s(x_j)$ and $c_s(x_j)$ are the stored objective and constraint values, f and c are the found objective and constraint values, and $r_{fi}(x_j)$ and $r_{ci}(x_j)$ are the rewards produced by design agent i based on the increase of objective and constraints.

Model Agent Process

A model agent controls the level of exploration τ used in the action selection process of its corresponding design agent and is rewarded based on the improvement of the merit table of all design agents. Unlike the custom design agent process, the model agent action selection and learning process follows a standard ε -greedy table selection and reinforcement learning approach, as explained in [26]. The reward r for each model agent is calculated using

$$r = \sum_{i=1}^n r_{fi}(x_i) + \sigma r_{ci}(x_i)$$

where $r_{fi}(x_j)$ and $r_{ci}(x_j)$ are the rewards produced by design agent i and σ is the penalty factor. Each model agent then learns this increase in merit using reinforcement learning, updating the value of each temperature τ_k using:

$$V(\tau_k) \leftarrow V(\tau_k) + \alpha(r - V(\tau_k))$$

where $V(\tau_k)$ is the stored value, α is a learning rate, and r is the reward. Finally, they each select these temperatures using ε -greedy action selection, choosing the best-valued temperature with probability $1 - \varepsilon$ and choosing a random temperature with probability ε . Reinforcement learning is chosen here for the value of different actions taken by the model agents: while a certain temperature may be good for the agent to pursue at the beginning of the process, when less is known about the design

space, it may not be as good later, when it may be desirable to exploit known good parameter values. Discovering new good parameter values as a result of exploration may mean that the agent should continue to explore at that level until good values are no longer found, when it should change strategies.

Modeling Collaborative Behavior

Collaboration and autonomy are then modeled in this framework by changing the synchrony with which the computational designers interact with the model. In a totally independent design behavior, these designers asynchronously interact with the model—submitting changes to the current best design one-at-a-time. In a totally collaborative design behavior, these designers all submit new parameter values at the same time. Additionally, a common independent design scenario in which the design of several parameters is decomposed between disciplines, is simulated by allowing the designers controlling parameters associated with each component to submit changes at the same time, but asynchronously from the designers selecting variables in the other components. To illustrate how this collaboration happens in the model, the example problem shown in Fig. 2 is again considered. If the designers start at design $A-B$, in collaborative design, both designers may both propose new values for their design variables at each iteration. In asynchronous design, on the other hand, designer 1 can only choose design A or B while designer 2's design is held constant at B . If a designer finds a better design, as with collaborative design, that design is then made the current best design used by the other designers. That is, if designer 1 chooses design B from $A-B$, the new baseline design used by both designers will be $B-B$.

Summary

To summarize, distributed parameter design is modeled by considering an optimization problem with readily definable disciplines or components. The design parameters in each component are delegated to individual computational designers, which iteratively and stochastically choose the value of that design parameter based on the previously stored merit of that parameter value. Collaboration is then modeled by allowing the designers to choose new values simultaneously, while independence is modeled by keeping the rest of the design fixed while designers pick the parameter values of either a single parameter (for totally independent design) or multiple parameters associated with a component (for design in which individual components are independent).

Results

The following results show how the different levels of collaboration and independence perform in an optimization setting to judge how these behaviors influence distributed parameter design. As discussed in the method section, these behaviors are represented as how the designers submit changes to the model: either fully synchronously (collaborative design), with component design asynchrony (where only individual components are designed together), or with full asynchrony (where every design parameter is given to an individual person). These results are shown in Figs. 4 and 5 over 20,000 iterations of the method over 10 different runs, with the average trend over time shown in Fig. 4 and the distribution of final results shown in Fig. 5 and Table 4. The initial values chosen for various parameters in the method are shown in Table 3.

As can be seen in Figs. 4 and 5, collaboration has a significant effect on the ability of the designers both to design quickly and reach good design outcomes. While

Table 3 Test parameters used in results

ϵ	Probability the meta-agent chooses randomly	0.05
$[\tau_1, \tau_2 \dots \tau_n]$	The temperatures for the meta-agents to choose	[0.5, 0.1, 0.05, 0.01, 0.005, 0]
σ	Constraint scale factor	35,000
$f_{S_{init}}$	Initial stored objective values	-10,000
$c_{S_{init}}$	Initial stored constraint values	-10,000

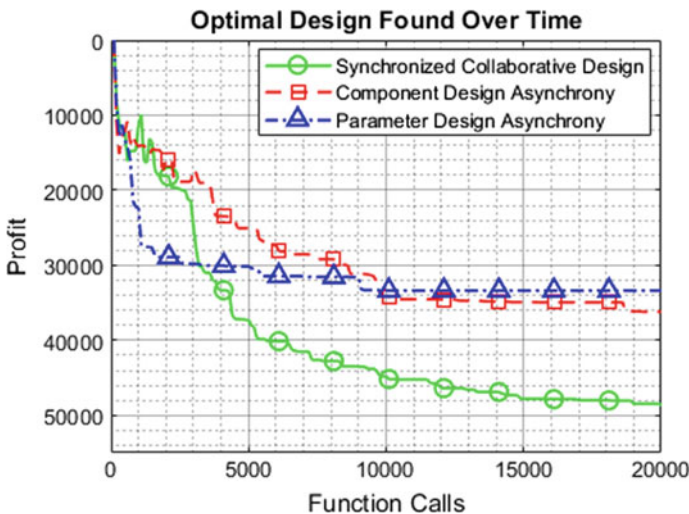


Fig. 4 Optimal design found over time using the multiagent method with differing levels of collaboration averaged over ten runs

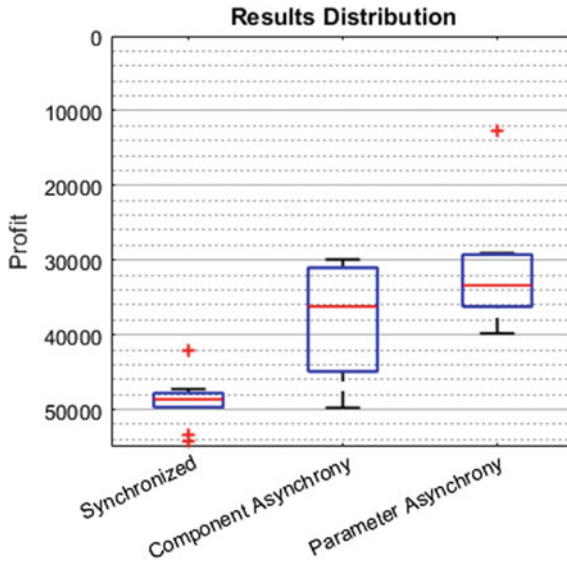


Fig. 5 Final results distribution of multiagent model with differing levels of collaboration over ten runs

Table 4 Summary of results from multiagent design model

	Collaborative	Variable asynchrony	Component asynchrony
Mean	48,641	33,375	36,208
Standard dev.	3360.3	7630.2	7248.5

all design strategies found feasible designs, synchronized collaborative design on average converged to much higher value designs than the two different types of independent design strategies. Looking at the trend over time of each strategy, the reason for this is quite apparent: while collaborative design steadily improves the performance of the design, both independent design strategies reach “floors” where the value of the design either does not improve or improves very slowly. Interestingly, the trend for parameter asynchrony and component asynchrony is slightly different. While parameter asynchrony reaches a relatively “good” design quicker than component design asynchrony (and is in fact more efficient than collaborative design over the first 4000 iterations), it demonstrates a more visible “floor” behavior. Component design asynchrony, on the other hand, always performs more slowly than totally collaborative design, but more continuously improves the design over the full number of iterations than parameter asynchrony, allowing it to reach lower minimums.

The reason for this “floor” behavior in asynchronous design can be inferred by reconsidering the example problem in Fig. 2 for asynchronous and collaborative design cases. If the designers start at design A–A, the only options available for them, given they design asynchronously, are designs A–B and B–A—the incompatible

designs. As a result, they will not be able to find the true best design $B-B$ because they cannot asynchronously choose the required set of joint actions. While this does not make it impossible for asynchronous design agents beginning in a random location to reach the global minimum, it does make it less likely. This is because, while these agents may randomly find the global minimum based upon selected starting point, they are far more prone to end up in process minimums. These process minimums will necessarily cause asynchronous designers in the process outlined here to reach “floors” which cannot be proceeded from, as shown in Fig. 3.

These results show the value of collaboration in distributed parameter design and are consistent with the hypothesis that collaboration between designers in parameter design yields better design outcomes in systems with interdisciplinary constraints. Furthermore, they highlight a reason that collaboration is useful in design settings: by allowing joint changes to be made to components which interact. This model of distributed design shows that when designers design collaboratively, they can better navigate the interdisciplinary constraint boundaries when changes of the parameters of one subsystem complement changes in another subsystem. This is because when designers do not design collaboratively, they are more prone to reaching process minimums which lock them into suboptimal designs.

Additionally, these results fall in line with design research studying integrated concurrent engineering which suggest that collaboration leads to a better design process. While these papers typically focus on decreases in time over a traditional process [3, 9], caused by there being less communication lag [30], these processes have additionally led to lower cost designs [3]. The results in this paper provide a basis for those better design outcomes not attributed to lag—that, when collaborating, designers are better able to make joint changes to the design which allow them to better navigate constraint boundaries. While previous work has highlighted the costs of collaboration [15] in taking an increased amount of the time of designers relative to the actual “design” work, this result shows the corresponding potential value that allowing collaboration has in helping designers seek higher value solutions.

Conclusions

This paper presents a model of distributed parameter design comprising a multiagent learning-based optimizer acting on a multidisciplinary design model. The goal of this model was to capture the technical aspects of collaborating or designing independently in complex systems design. Results show that collaboration produces better design compared to two forms of independent design. As discussed, this is because of the ability of collaborative design activities to allow designers to better navigate constraint boundaries between subsystems which interact or share some requirement. This shows the value of collaboration in complex systems design not just towards generating solutions faster, but in producing higher value designs.

While this model can show the value of collaboration in design due to design synchrony, future work will need to be done to better represent all other activities

having to do with collaboration in design. For example, in this model, communication is represented by using in a single connected design model, while, in practice, there are different ways of exchanging data which may cause lag, as identified in [15] and [30]. Additionally, while this model re designers of components as a multiagent system presented in earlier work, future work should be done to encode designer behaviors in this system, as in [21]. Finally, the problem definition (a well-defined optimization problem) neglects many of the challenges design teams might face in the various stages of design, such as understanding the problem, producing a concept, or performing the work of filling out every necessary detail of the design. Further work will be needed to model these activities to show where collaboration is most needed. Nevertheless, the model presented here shows good insights on the domain presented.

Acknowledgements This research is supported by the National Science Foundation award number CMMI-1363411. Any opinions or findings of this work are the responsibility of the authors and do not necessarily reflect the views of the sponsors or collaborators.

References

1. Bond AH, Ricci RJ (1992) Cooperation in aircraft design. *Res Eng Des* 4:115–130. <https://doi.org/10.1007/bf01580149>
2. Sobek DK, Ward AC, Liker JK (1999) Toyota's principles of set-based concurrent engineering. *Sloan Manag Rev* 40(2):67
3. Smith J (1998) Concurrent Engineering in the Jet Propulsion Laboratory Project Design Center. SAE Technical Paper 981869. Society of Automotive Engineers. <https://doi.org/10.4271/981869>
4. Boy GA, Jani G, Manera A, Memmott M, Petrovic B, Rayad Y et al (2016) Improving collaborative work and project management in a nuclear power plant design team: a human-centered design approach. *Ann Nucl Energy* 94:555–565. <https://doi.org/10.1016/j.anucene.2015.12.039>
5. Ostergaard KJ, Wetmore WR III, Divekar A, Vitali H, Summers JD (2005) An experimental methodology for investigating communication in collaborative design review meetings. *Co-Des* 1(3):169–185. <https://doi.org/10.1080/15710880500298520>
6. Shishko R, Aster R (1995) NASA systems engineering handbook. NASA Special Publication, 6105
7. Winner RI, Pennell JP, Bertrand HE, Slusarczuk MM (1988) The role of concurrent engineering in weapons system acquisition (No. IDA-R-338). Institute for Defense Analyses, Alexandria VA
8. Portioli-Staudacher A, Van Landeghem H, Mappelli M, Redaelli CE (2003) Implementation of concurrent engineering: a survey in Italy and Belgium. *Robot Comput Integr Manufact* 19(3):225–238
9. Mark G (2002) Extreme collaboration. *Communications of the ACM* 45(6) (pp. 89–93). Association for Computing Machinery, New York. <https://doi.org/10.1145/508448.508453>
10. Gerwin D, Barrowman NJ (2002) An evaluation of research on integrated product development. *Manag Sci* 48(7):938–953
11. Yassine A, Braha D (2003) Complex concurrent engineering and the design structure matrix method. *Concurrent Eng* 11(3):165–176
12. Smith RP, Eppinger SD (1997) Identifying controlling features of engineering design iteration. *Manag Sci* 43(3):276–293

13. Reagans R, Miron-Spektor E, Argote L (2016) Knowledge utilization, coordination, and team performance. *Organ Sci* 27(5):1108–1124
14. Hirschi N, Frey D (2002) Cognition and complexity: an experiment on the effect of coupling in parameter design. *Res Eng Des* 13(3):123–131
15. Grogan PT, de Weck OL (2016) Collaboration and complexity: an experiment on the effect of multi-actor coupled design. *Res Eng Des* 27(3):221–235
16. Grogan PT, de Weck OL (Apr 2016) Collaborative design in the sustainable infrastructure planning game. In: Proceedings of the 49th annual simulation symposium (p. 4). Society for Computer Simulation International
17. Alelyani T, Yang Y, Grogan PT (Aug 2017) Understanding designers behavior in parameter design activities. In: ASME 2017 International design engineering technical conferences and computers and information in engineering conference (pp. V007T06A030-V007T06A030). American Society of Mechanical Engineers
18. McComb C, Cagan J, Kotovsky K (2015) Rolling with the punches: an examination of team performance in a design task subject to drastic changes. *Des Stud* 36:99–121
19. McComb C, Cagan J, Kotovsky K (2015) Lifting the veil: drawing insights about design teams from a cognitively-inspired computational model. *Des Stud* 40:119–142
20. McComb C, Cagan J, Kotovsky K (2017) Optimizing design teams based on problem properties: computational team simulations and an applied empirical test. *J Mech Des* 139(4):041101
21. McComb C, Cagan J, Kotovsky K (2017) Utilizing Markov chains to understand operation sequencing in design tasks. In: Design computing and cognition'16, pp 401–418. Springer, Cham
22. Martins JR, Lambe AB (2013) Multidisciplinary design optimization: a survey of architectures. *AIAA J*
23. Hulse D, Gigous B (2017) Appendix A: Quadrotor design model. From: <https://github.com/hulsed/QuadrotorModel/blob/master/Quadrotor%20Design%20Model.pdf>
24. Agogino AK, Tumer K (2012) A multiagent approach to managing air traffic flow. *Auton Agent Multi-Agent Syst* 24(1):1–25
25. Yliniemi L, Agogino AK, Tumer K (2014) Multirobot coordination for space exploration. *AI Mag* 35(4):61–74
26. Tuyles K, Tumer K (2013) Multiagent learning. In: Weiss G (ed), *Multiagent systems: a modern approach to distributed artificial intelligence* (2nd ed, pp 423–484). MIT press
27. Hulse D, Gigous B, Tumer K, Hoyle C, Tumer IY (2017) Towards a distributed multiagent learning-based design optimization method. In: ASME 2017 International design engineering technical conferences and computers and information in engineering conference, pp V02AT03A008–V02AT03A008. American Society of Mechanical Engineers
28. Ben-Menahem SM, Von Krogh G, Erden Z, Schneider A (2016) Coordinating knowledge creation in multidisciplinary teams: evidence from early-stage drug discovery. *Acad Manag J* 59(4):1308–1338
29. Fiacco AV, McCormick GP (1966) Extensions of SUMT for nonlinear programming: equality constraints and extrapolation. *Manag Sci* 12(11):816–828
30. Chachere J, Kunz J, Levitt R (2009) The role of reduced latency in integrated concurrent engineering. CIFE Working Paper #WP116

Exploring the Effect of Experience on Team Behavior: A Computational Approach



Marija Majda Perišić, Mario Štorga and John S. Gero

The paper presents the results of research aimed at contributing to a better understanding of the effect of team experience and learning on the performance of a design team. An agent-based model of the design team was developed, and computational simulations were utilized to study how agent's knowledge changes by its use and what are the effects of such changes on the team behavior. Particularly, hypotheses stating that as team members work together, they become more efficient and explore less have been studied. Computational experiments demonstrated the positive impact of learning and prior experience on team efficiency, and indicated that team experience has a strong influence on the breadth of solutions examined by a team in a constructive task. However, results also suggest that increased knowledge grounding could have detrimental effects on a team's performance when teams are faced with tasks which do not fall into the team's expertise.

Introduction

Learning, as a process of acquiring knowledge and skills obtained from participating in events and activities [1], is a fundamental process for achieving progress in any field of human actions. By developing and reusing well-grounded beliefs, the humans

M. M. Perišić (✉) · M. Štorga
University of Zagreb, Zagreb, Croatia
e-mail: mperisic@fsb.hr

M. Štorga
Luleå University of Technology, Luleå, Sweden

J. S. Gero
University of North Carolina at Charlotte, Charlotte, USA

J. S. Gero
George Mason University, Fairfax, USA

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_32

can solve problems more efficiently, as well as develop generalizations which could be used in novel situations. Cognitive scientists, however, emphasize that learning and the resulting knowledge necessarily depend on the situation [2]. Situated cognition views learning as a constructive process where each new experience is shaped by previous experiences and understanding of the current situation, while new experiences, in turn, reshape the understanding of previous experiences. Particularly, the theory of situated learning states that learning cannot be separated from the social context in which it occurs, therefore emphasizing the role of social interactions in the formation of new knowledge.

Designers, while working in teams, are necessarily influencing each other's learning processes. It has been recognized in the literature that designers are affected by their prior experience and learned patterns [3]. How the experience of each team member impacts the performance of the design team and how does the team behavior change as team members work together are questions which are still not well addressed.

To contribute to a better understanding of the effect of team experience and learning on the performance of a design team, this work presents the results of a set of computational experiments performed utilizing an agent-based framework. These experiments aimed to explore the effect of knowledge grounding and experience on the efficiency of a stable design team by studying how the time needed to find a solution, as well as team's learning and solution-sharing behavior change over time. The goal of this paper is to show that, by modeling and implementing agents whose behavior is based on cognitive theories, computational simulations can increase the understanding of the behavior of the individuals and teams in constructive tasks, and produce results which match propositions based on cognitive theories.

The remainder of the paper is structured as follows: first, a short overview of related work in the fields of cognitive behavior of individuals, learning in teams and effect of team experience and tenure on design team performance is presented, as well as a short review of computational simulators used in the studies of team behavior in engineering design. Next, the aims of this work are explicated in the form of hypotheses derived from the literature review. The methods used to test the hypotheses are described in Section "[Design of the Experiments](#)" where the computational framework, measures, and case studies are presented in detail. In Section "[Simulation Results and Hypothesis Testing](#)", results of the experiments are given and are further analyzed and discussed in Section "[Discussion](#)". Finally, the paper concludes with a discussion of the limitations of the study and an outline of possible future work.

Related Work

Cognitive Behavior of Designer

When developing a model of situated reasoning and learning in design, several authors [1, 4] have built on a descriptive model proposed by Gero and Fujii [5]. Maher and Gero [6] distinguish three modes of reasoning: reflexive, reactive, and reflective. Reflexive reasoning is defined as producing a reaction without overt reasoning, a reflex. Reactive reasoning includes perceiving a situation and choosing actions which will lead toward the desired goal. In addition to perceiving the situation, reflective reasoning comprises hypothesizing on possible outcomes and determining desired states, therefore enabling the concepts to change, consequently changing the goals and experiences. As knowledge is being successfully used, it becomes more grounded and requires less cognitive effort to process, i.e., it becomes reflexive [1]. This theory is complementary to the Kahneman's notion of System 1 and System 2 thinking, presented in his book *Thinking, Fast, and Slow* [7]. Kahneman posits that there are two modes of thinking: System 1, which is fast, automatic, reflexive, and subconscious and System 2, which is effortful, conscious and based on logic. Further, he offers several pieces of evidence of environmental impact on the thought process of an individual, particularly emphasizing the anchoring, a cognitive bias directed toward first-seen proposals. By building on these theories, one can describe and model phenomena such as knowledge grounding and create experiments which have the potential to further enrich our understanding of some of the known design behaviors such as fixation or formation of design patterns [8].

Design Team Experience and Its Effect on Team Learning and Performance

In a detailed meta-analytic review of the determinants of team performance in new product development, Sivasubramaniam et al. [9] listed team tenure, internal and external communication, team ability, functional diversity, group cohesiveness, goal clarity, and leadership as critical factors influencing team performance. Team ability was defined as having the knowledge and experience needed to deal with complex design tasks and was hypothesized to have a positive impact on team performance. Strong support for this hypothesis has been found, thus confirming and extending the findings from a study by Carbonell and Rodriguez [10], which posited that team experience is positively related to speed to market. Sivasubramaniam et al. further found strong support for a hypothesis that team tenure is positively associated with speed to market.

Similar findings were presented by Akgun and Lynn [11] in their study of the consequences of team stability (i.e., tenure) on the team performance. Akgun and Lynn [11] posited that team tenure impacts team learning and team experience, as

stability promotes learning about team tasks and about the team itself. By creating a tacit task-related knowledge of “know how” and “know why,” and team related knowledge of “who knows what”, team members can better coordinate their efforts, and consequently achieve higher speed to market. These studies indicate that experienced teams create products in a timely manner, which is likely influenced by the similarity of team members’ mental models and preferences developed through their experience which allow them to reach consensus faster.

However, several studies [12, 13] emphasize a curvilinear relationship between design team tenure and performance. Researchers [11, 13] state that in turbulent environments, team mental models can become obsolete and introducing new members can bring beneficial new knowledge and promote success. Similarly, Choi and Thompson [14] stress that, due to the tenure’s impact on an increase in homogeneity of members’ mental models, stable teams may struggle to produce creative results in innovative tasks. These authors presented a study which shows that a number of non-redundant ideas are significantly higher in groups which suffered membership change. Interestingly, findings presented by Gibson and Gibbs [15] indicate that team innovativeness is lower in less stable teams. Possible explanations for such contradictory findings were proposed by Hirst [16], who suggests that timing of turnover matters, while Badke-Schaub et al. [17] stress the importance of balance between knowledge overlap and distributed knowledge. Nevertheless, more research is needed to understand positive and negative impacts of team member’s experience, in relation to team environment regarding the nature of team task, organizational and market factors, and temporal factors such as the part of the product life cycle the team is working on.

Computational Studies of Team Behavior in Design

Several computational models used to study different aspects of team behavior in product design have been developed. For example, McComb et al. [18] developed an agent-based model used to study the problem-solving behavior of design teams. In their model, the authors have defined and implemented agents based on research on the behavior of designers: agents share a common goal, they learn, interact at irregular intervals, have a bias in favor of their solutions, and focus on promising alternatives. In their work, an agent’s search for a solution is implemented as a simulated annealing process.

Ambler [19] presented a model where each design concept is represented as a distinct point in Kauffman’s NK model. Agents roam such space and perceive their relative fitness and current team situation, and can decide on their next steps. In addition to exploring neighboring locations, agents can provide jumping-off locations for future newcomers and can jump to previously found distant concepts. Agents communicate their relative fitness through collaborative linkages.

Models which focused on specific types of team learning and team expertise formation are found in Singh et al. [20] and Gero and Kannengiesser [21]. Singh

et al. [20] focused on learning about “who knows what” in design teams. They explored how transactive memory is formed in flat, distributed and functional teams and further studied its impact on activity coordination and team effectiveness. Gero and Kannengiesser [21] studied the formation of team expertise in temporary design teams where agents are frequently required to adapt to new team formation.

Team exploration of solution space has been studied by Sosa and Gero [22], who developed an idea-agent-social context framework within which agents combine known geometric shapes to create as many diverse forms as they can. This model has been used to simulate ideation task, but no cognitive mechanisms of ideation were implemented in their agents.

Aims and Hypotheses

Building on theoretical studies of the cognitive behavior of individuals and research findings on team learning, the current work aims to create a computational model of a design team and utilize it to gain experimentally based insights into how experience gained by working together impacts the behavior of the team. More specifically, this work is aimed at testing two hypotheses:

As team members collaborate, they learn about each other and from each other. In other words, interactions between team members enable the creation, modification, and reinforcement of a team mental model [23]. Shared mental models, in turn, reduce the coordination cost and support reaching consensus. Experience gained and the development of shared mental models result in higher efficiency of the team, in terms of less time required to find a solution. Therefore, the first hypothesis is formulated as:

H1: As team members work together over time, they become more efficient.

As team members work together, similar knowledge gets reused and becomes more grounded. As a result, teams develop a preference for particular solutions which proved to be suitable in previous projects and reuse them at the expense of other solutions. Therefore, the second hypothesis is formulated as:

H2: As team member work together over time, they explore less.

Model

An agent-based model of the design team is conceptualized, implemented and utilized to test these two hypotheses. Each team member is represented as an agent whose architecture is derived from previously published work [24].

Agent's Cognitive Behavior

The main driver of an agent's behavior is its mental model which is represented as a directed network. To contextualize the agents and tasks, the FBS ontology was selected [3]. Building on those two approaches, nodes in the agent's mental model represent design task's decomposition and abstraction to the function, behavior or structure domain that the agent is familiar with. The agent's experience is presented as weighted, directed links between nodes. The agent's mental model (i.e., agent's knowledge network) consists of nodes of three types: function, behavior and structure nodes, which are connected by directed links whose weight indicates the perceived association between function and behavior, and behavior and structures. There are no links between nodes of the same type, and structure nodes can be reached from function nodes only through the behavior nodes [3].

Following Kahneman's theory of cognitive behavior and theory presented in [7], the agents have two ways of reasoning. Reflexive reasoning (System 1) is implemented as spreading activation through the agent's mental model. The activation which spreads from an activated node to an adjacent node is proportional to the weight of the link connecting them, thus indicating the strength of their association. To model System 2 thinking agents had to be equipped with additional reasoning mechanisms. Thus, instead of modeling structure nodes as simple nodes, each structure node is associated with a randomly created network. Behavior nodes are associated with selected network measures: diameter, a number of clusters, betweenness, closeness, and degree centralities. By modeling behavior and structure nodes in such a manner, the natural association between structure nodes (i.e., networks) and behavior nodes (i.e., network measures) is created. The representation of the derived agent's mental model based on the FBS ontology and dynamic networks is presented in Fig. 1.

When a structure domain node's activation exceeds the analysis threshold, the agent analyzes the node (i.e., node's associated network) and creates, or further grounds, relationships to behavior domain nodes as determined by calculating the associated network's properties. As links increase in weight, they may exceed the reflexive threshold and agent will process them without extended reasoning, i.e., reflexively. This results in the activations passing through a node and enabling the activation to reach in a single step nodes which are two or more links apart. If a link is not being used, its weight is decremented. That is, if the knowledge is not used, agents slowly forget.

Agents can also expand their knowledge space by creating new structure nodes: if several structure nodes are activated, an agent can combine the networks associated with the activated structure nodes, thus producing a new network which is then named as a new structure node. Therefore, agents can create new solutions which display a novel combination of network properties, i.e., they can create structure nodes which present previously unseen combination of behaviors. Over time, these processes change the agent's mental model, Fig. 2.

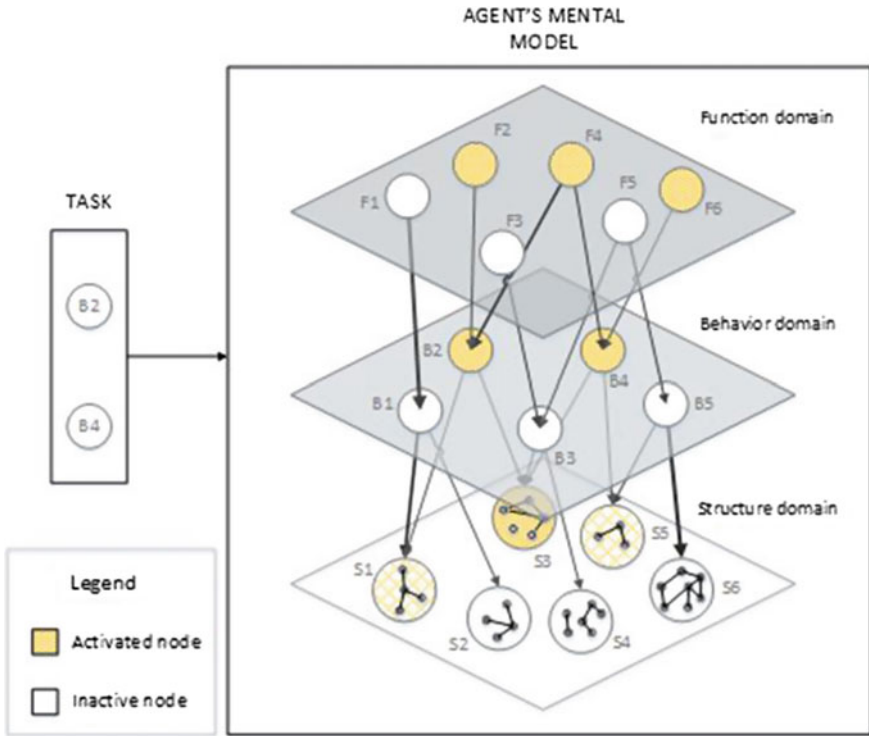


Fig. 1 The representation of agent's mental model

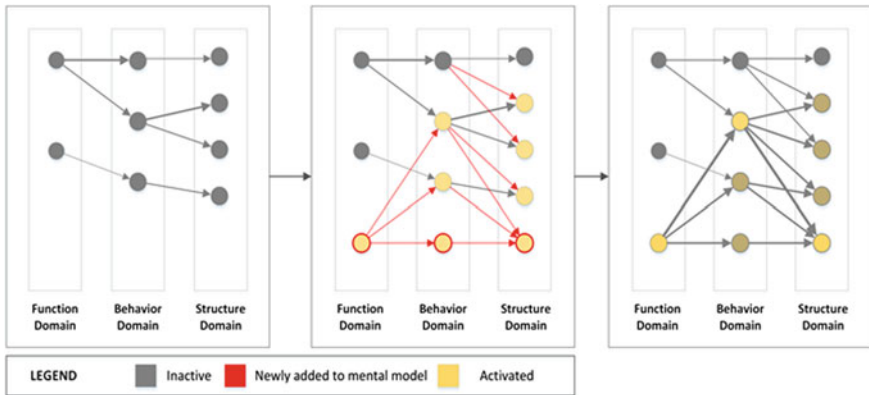


Fig. 2 The change of agent's mental model over time

Task Processing

A design task is given to the team of agents in the form of a list of behaviors that a structure node is required to meet to be considered as a valid solution for the task. As the task is received, an activation is generated. Since an activation can be passed to required behavior nodes only through function domain nodes, the agents also learn which function nodes are relevant (i.e., connected to desired behavior) and whether to activate them when a specific task is introduced. If a function domain node is sufficiently activated, or no progress has been made and activation accumulates in the task (i.e., it is not passed to function nodes), an agent can analyze the task or a function and determine whether a function node should be activated for a given task.

Through spreading activation and analysis, the agents search for the structure domain nodes which produce the required behaviors. Once a structure node is sufficiently activated, its associated network is evaluated against required behaviors, and if the structure node (more precisely, its associated network) is deemed as satisfying it will be proposed to others as the final solution. If, however, the structure domain node does not meet the required behavior, an agent can direct its actions by sending an activation to the function nodes which are connected to the unmet requirement.

To implement the previously presented theoretical background, the agent's learning is affected by interactions between team members. An "idea" consists either of a structure domain node whose activation level exceeds the sharing threshold, or of a knowledge link (a relation between a task and a function node, function and behavior node, or behavior and structure node) whose weight is sufficient and the respective nodes' activation exceeds the activation threshold. As agents encounter either structure domain nodes, or knowledge links worth sharing, they can propose them to others. Other agent's chain of actions is interrupted not just by additional activations sent to nodes the proposing agent has activated, but also by the creation of novel links and increase in weight of existing links which are being activated.

If a solution is proposed, every team member evaluates it against its mental model and produces a score. Each score is weighted with the factor indicating the agent's reputation (i.e., average trust in the agent). If agents rate the solution as sufficient, the simulation is over. If, however, the solution is rejected, agents continue their search until a new solution is found or time runs out.

Additional Agent's Behavioral Mechanisms

To account for the other elements of the agent's architecture presented in [24], the implemented agent's cognitive mechanisms are biased by its:

1. *Personality factors*, which influence the acceptance of novel ideas (based on agreeableness and openness to novel experiences factors), and sharing behavior (based on extraversion factor);

2. *Trust* it holds for others, which influences the impact other agent's proposals will have on the agent's mental model;
3. *Frustration (affect)*, which represents the agent's affective state and guides agent's behavior in the following manner: as time passes and no solution has been found, or sufficient time passes with no progress, the agent's frustration increases and causes the weaker links to temporarily increase in weight, therefore enabling a wider search of the knowledge space. In the extreme, frustration causes all of the knowledge link weights to be regarded as similar. If a team experiences progress, i.e., solutions better than the previous maximum has been found, the frustration drops, and the agent focuses on the part of the knowledge where improvement has been found.

Description of a Simulation Run

Agents and a task are initialized at the commencement of a simulation run. Agents' knowledge networks are created. However, they are neither exhaustive (i.e., none of the agents is familiar with all of the tasks, links, or function, behavior and structure domain nodes known to others), nor completely accurate (i.e., several incorrect links between behavior and structure domain nodes are introduced in an agent's model). Additionally, trust between agents is computed based on their initial knowledge related to the task. Trust others hold for an agent is proportional to the extent the agent is familiar with the task (i.e., with related functions and behaviors).

At the simulation start, each agent is presented with the task, which generates an activation, thus triggering a spreading activation mechanism. Through the cognitive mechanisms described above, agents search for a structure domain node which satisfies the requirements. If, either an existing or newly generated, structure domain node is deemed as suitable, it is stored in the agent's working memory and can be shared with others. Otherwise, if two nodes are both active and connected with a link of sufficient weight, the knowledge link is stored in the agent's working memory, and an agent can share it. An agent's working memory is constrained and can store a limited number of different links, and structure nodes. At any time step, an element stored in agent's working memory can be shared, but the content of working memory changes as additional structure domain nodes and knowledge links are activated. If in any time step two or more agents decide to share a structure domain node or a link with others, the dominant one is chosen randomly. As previously described, each agent is affected by what is shared as new activations are sent either to proposed structure domain nodes or to nodes activated when a link is shared.

The simulation is over either when agents agree on the solution or the time limit is exceeded, where time is treated as a scarce resource.

Design of the Experiments

To test the two hypotheses presented, the agent-based model has been run with two types of task sequences, where A and B are two separate tasks:

- T1. “ $AB_1 B_2 \dots B_n A$ ”, where $n = 1, 3, 5, \text{ or } 10$, and B_m is not equal to A for any m . Any two B_i and B_j can, but do not have to resemble each other, i.e., share some of the requirements.
- T2. “ A ” versus “ BA ” versus “ $BB (\dots) BA$ ”, where B has been repeated 3, 5, or 10 times. Task B is related to task A as they require at least one common function, but are not the same, i.e., at least one function is different.

In the first type of task sequences (T1), tasks B_i have been chosen randomly, and may or may not have some functions or required behaviors in common with A or with each other. Cases range from simple tasks where multiple solutions for the tasks are possible and known to the team, to cases where no solution is known for any given task. The experiment with the second task type (T2) is performed to test whether relevant, yet narrow experience on the related task impacts the performance on the task A . Following the hypotheses H1 and H2, it is expected that in both cases, where the team gained the diverse experience (T1) as well as restricted, but relevant experience (T2), will result in fewer simulation steps and less exploration of the solution space needed to find the solution.

To evaluate each simulation run, the following measurements have been collected:

- M1. Solution score as determined by the team (subjective, as determined by the agents).
- M2. Number of steps required to reach the solution, or max number of steps if the session is over and no solutions are found.
- M3. Number of new structure nodes created, i.e., number of distinct networks obtained through combining networks associated with existing structure domain nodes (overall, throughout the session).
- M4. Number of new links learnt (overall) during the session.
- M5. Number of distinct structure nodes proposed during the session (either new or existing).
- M6. Number of distinct links communicated during the session.
- M7. Actual (i.e., objective) score, calculated as a percentage of required behaviors that a given solution satisfies.

Measures M1 and M2 provide the basis for whether teams become more efficient measured by the number of steps needed to find a solution and change in a score determined by the team, i.e., they are related to hypothesis H1. Steps needed (M2) and the score as determined by the team (M6) are related as teams terminate their search if the solution whose score is determined to be high enough is found. The number of new structures created (M3), number of new links learned (M4), number of distinct structures proposed (M5), and number of links shared (M6), provide the basis for whether a team explores less, i.e., they provide the basis for insights

related to the hypothesis H2. Following the hypotheses, it is expected that scores for measures M3–M6 will be lower as more tasks have been performed by the team. Similarly, it is expected that the number of steps will decrease with team experience and that a score as determined by the team will be higher to show that a team reaches consensus quicker. The last measure, the objective (i.e., actual) score of the proposed solution (M7) measures a team’s effectiveness (i.e., an objective success), rather than efficiency (i.e., meeting the schedule). It is provided to gain better insights into positive and negative aspects of team experience in team performance.

The model is implemented in the multi-agent simulation environment MASON. Simulations were run 50 times for each of the task types (T1 and T2). The maximum number of steps ($M2_{\max}$) was set to 300, the number of agents within a team was set to three, and there were 30 initial structure domain nodes (i.e., distinct networks). A network associated with a single structure domain node consists of 4–10 nodes, with links randomly distributed between them. Nodes in the networks associated with structure domain nodes (i.e., nodes from “structure node networks”) were chosen from a set of 50 distinct nodes. The number of functions was set to 15, while the number of behavior domain nodes varied from 10 to 50.

Simulation Results and Hypothesis Testing

The average values for each of the measures M1–M7 in task sequences of types T1 and T2 are listed in Tables 1 and 2 respectively. The data for the first task sequence type (T1) consists of the details on team performances on the first task *A* and on the last task *A* (i.e., second-time task *A* was performed). The data for the second task sequence type (T2) consists of the details on team’s performance of task *A* if no tasks *B* were performed before task *A*, and on the team performance on task *A* after multiple performances of task *B*.

To test hypotheses H1 and H2, significance testing were carried out. The simulation results were tested for normality and were shown not to be normally distributed. As a consequence, the Wilcoxon’s signed-ranked test was utilized on the simulation results. More detailed hypotheses formulated and tested on the task sequences of the type T1 were as follows:

1. $dM1_{\text{First}} < dM1_{\text{Second}}$
2. $dM2_{\text{First}} > dM2_{\text{Second}}$
3. $dM3_{\text{First}} > dM3_{\text{Second}}$
4. $dM4_{\text{First}} > dM4_{\text{Second}}$
5. $dM5_{\text{First}} > dM5_{\text{Second}}$
6. $dM6_{\text{First}} > dM6_{\text{Second}}$
7. $dM7_{\text{First}} < dM7_{\text{Second}}$

where dMi_{First} and dMi_{Second} for $i = 1, \dots, 7$, represent the distributions of the collected data by measuring M1–M7 on the first and the second task *A*, respectively.

Table 1 Mean values and standard deviations (in brackets) for the results from T1

Task type	M1: Score rated by agents	M2: Steps	M3: New structures	M4: New links	M5: Proposed structures	M6: Communicated links	M7: Score
\underline{AB}_1A	0.6732 (0.26)	217.8 (103.6)	72.9 (63.4)	514.1 (371.1)	27.2 (20.3)	61.8 (44.7)	0.7553 (0.27)
$AB_1\underline{A}$	0.7246 (0.24)	165.6 (140.3)	21.9 (39.5)	154.9 (224.6)	20.8 (21.7)	45.3 (45.7)	0.773 (0.26)
$\underline{AB}_{1,2,3}A$	0.5371 (0.29)	267.3 (121.7)	82.1 (54.4)	576.5 (306.2)	27.8 (15.9)	63.5 (35.2)	0.6150 (0.33)
$AB_{1,2,3}\underline{A}$	0.6224 (0.30)	202.9 (121.7)	21.8 (29.5)	151.8 (176.3)	20.3 (17.8)	45.0 (39.3)	0.6547 (0.34)
$\underline{AB}_{1,\dots,5}A$	0.6156 (0.25)	254.9 (80.8)	95.7 (56.3)	661.6 (324.5)	33.4 (18.0)	77.5 (39.5)	0.7173 (0.28)
$AB_{1,\dots,5}\underline{A}$	0.6979 (0.29)	178.6 (118.7)	11.7 (21.6)	90.5 (133.1)	16.9 (19.1)	36.1 (39.3)	0.6927 (0.3)
$\underline{AB}_{1,\dots,10}A$	0.5542 (0.29)	215.8 (114.7)	71.2 (49.5)	495.6 (314.5)	26.9 (18.3)	61.8 (41.5)	0.6483 (0.31)
$AB_{1,\dots,10}\underline{A}$	0.6685 (0.30)	208.9 (104.5)	16.1 (24.9)	94.7 (132.9)	14.1 (14.5)	34.3 (32.2)	0.6294 (0.31)

Table 2 Mean values and standard deviations (in brackets) for the results from T2

Task type	M1: Score rated by agents	M2: Steps	M3: New structures	M4: New links	M5: Proposed structures	M6: Communicated links	M7: Score
\underline{A}	0.6520 (0.24)	252.3 (78.5)	93.7 (60.5)	642.4 (348.7)	29.2 (17.0)	69.0 (36.2)	0.7362 (0.26)
\underline{BA}	0.6881 (0.23)	233.6 (94.9)	61.7 (54.8)	419.1 (308.0)	27.0 (18.0)	62.9 (37.3)	0.7438 (0.25)
$B^3\underline{A}$	0.6951 (0.26)	221.1 (102.0)	33.7 (43.4)	245.5 (257.4)	23.2 (18.8)	51.1 (38.2)	0.7472 (0.24)
$B^5\underline{A}$	0.7555 (0.24)	210.3 (101.6)	28.7 (41.1)	209.9 (238.4)	16.6 (14.8)	40.9 (32.8)	0.7310 (0.25)
$B^{10}\underline{A}$	0.6772 (0.29)	220.6 (100.5)	16.3 (27.8)	129.5 (158.3)	12.1 (13.0)	30.9 (25.6)	0.6459 (0.30)

Similar hypotheses were formulated for the case where the performance of a single task A was compared to the performance of task A after multiple completions of task B (T2). The results are presented in Tables 3 and 4. Further, to gain deeper insights in how the performance of multiple tasks B impacts the performance of task A , the data for task type T2 were separated into cases where (objective) solutions for the task A were found by the agents while performing task B , and where no solution for task A was found while performing task B . The results for listed cases are presented in Tables 5 and 6.

Table 3 *p*-values for T1

Task type	M1: Score rated by agents	M2: Steps	M3: New structures	M4: New links	M5: Proposed structures	M6: Communicated links	M7: Score
<i>AB</i> _{1A}	0.0158*	0.0000*	0.0000*	0.0000*	0.0001*	0.0000*	0.0981
<i>AB</i> _{1,2,3A}	0.0003*	0.0001*	0.0000*	0.0000*	0.0001*	0.0000*	0.0960
<i>AB</i> _{1,...,5A}	0.0034*	0.0000*	0.0000*	0.0000*	0.0000*	0.0000*	0.8281
<i>AB</i> _{1,...,10A}	0.0031*	0.3773	0.0000*	0.0000*	0.0005*	0.0003*	0.7940

**p* < 0.05

Table 4 *p*-values for T2

Task type	M1: Score rated by agents	M2: Steps	M3: New structures	M4: New links	M5: Proposed structures	M6: Communicated links	M7: Score
<i>BA</i>	0.1025	0.0359*	0.0000*	0.0000*	0.2181	0.0927	0.2753
<i>B</i> ³ <i>A</i>	0.0033*	0.0036*	0.0000*	0.0000*	0.0029*	0.0001*	0.3659
<i>B</i> ⁵ <i>A</i>	0.0003*	0.0006*	0.0000*	0.0000*	0.0000*	0.0000*	0.6281
<i>B</i> ¹⁰ <i>A</i>	0.0593	0.0072*	0.0000*	0.0000*	0.0000*	0.0000*	0.9867

**p* < 0.05

Table 5 *p*-values for cases of T2 where solution for the task *A* were found while performing task *B*

Task type	M1: Score rated by agents	M2: Steps	M3: New structures	M4: New links	M5: Proposed structures	M6: Communicated links	M7: Score
<i>BA</i>	0.1007	0.0567	0.0020*	0.0005*	0.2110	0.1749	0.2853
<i>B</i> ³ <i>A</i>	0.2858	0.0106*	0.0000*	0.0000*	0.0247*	0.0024*	0.4684
<i>B</i> ⁵ <i>A</i>	0.0118*	0.0085*	0.0000*	0.0000*	0.0000*	0.0000*	0.7653
<i>B</i> ¹⁰ <i>A</i>	0.7112	0.3785	0.0000*	0.0000*	0.0000*	0.0000*	0.9918

**p* < 0.05

Table 6 *p*-values for cases of T2 where no solutions for the task *A* were found while performing task *B*

Task type	M1: Score rated by agents	M2: Steps	M3: New structures	M4: New links	M5: Proposed structures	M6: Communicated links	M7: Score
<i>BA</i>	0.3306	0.0899	0.0004*	0.0002*	0.4004	0.1637	0.3932
<i>B</i> ³ <i>A</i>	0.0154*	0.0544	0.0000*	0.0000*	0.0449*	0.0090*	0.2168
<i>B</i> ⁵ <i>A</i>	0.0051*	0.0136*	0.0000*	0.0000*	0.0017*	0.0009*	0.2476
<i>B</i> ¹⁰ <i>A</i>	0.0013*	0.0025*	0.0000*	0.0000*	0.0000*	0.0000*	0.4146

**p* < 0.05

Discussion

Analysis of Performance on the Task Sequences of the Type T1

As can be seen from the results presented in Tables 1 and 3, agents have rated solutions generated for task A significantly higher in the second turn comparing to the score proposed in the first turn, irrespective of the number of tasks performed in between turns. Furthermore, considerably fewer new structure domain nodes and links were learned, and significantly fewer structure domain nodes and links were shared. In other words, when they were faced with a familiar task, agents converged to the solutions faster, without the need to create novel structure domain nodes. Further, over the runs between two tasks A, more links between structure domain nodes and behavior domain nodes have been created. Some links to the proposed solutions became more grounded, i.e., agents became surer about the proposed structure's behavior (i.e., associated network's properties), resulting in agents rating their solutions as better. However, when ten tasks were performed between two tasks A, the number of steps needed to find (and agree on) the solution was not significantly lower than in the first turn. Inspection of data revealed that this effect was caused by forgetting. Namely, while some of the links get grounded, the links which are not used in tasks between two tasks A decrease in weight, consequently resulting in more steps needed for the agents to activate relevant nodes and links.

It is interesting to note that a team's objective performance did not significantly increase. While the null hypothesis can be rejected at a significance level of 0.01 for the cases where there is one or three tasks performed in between tasks A, when five and ten tasks were performed in between, the null hypothesis cannot be rejected. From a closer inspection of the data, it was found that such behavior in some cases results from the acceptance of the structure domain nodes as the final solution, although they were not adequate solutions for the given task (i.e., solutions did not meet all of the requirements). In other words, while performing other tasks, the faulty links in agents' mental model became grounded, thus causing the agents to accept inadequate structure nodes and rate their performance highly. Such grounding originated from either two agents sharing the same wrong link and reassuring its correctness through communication, or if one trusted agent posited a wrong link, causing others to learn it.

It is possible that this situation could be avoided if agents were presented with an objective evaluation of solutions after each task. However, in a closed system, agents developed faulty mental models. This serves to show that, apart from sharedness of the mental modes among team members, the accuracy of such models is important. As stated by Badke-Schaub et al. [17] "*All members of a team can share some identical knowledge, but all of them might be wrong.*"

Analysis of Performance on the Task Sequences of the Type T2

In the second type of task sequences, task *B* was chosen to have an overlap with the task *A*, in terms of having some function and behavior domain nodes in common. The overall results show a similar trend as seen in the case of the first task sequence type. However, it can be noticed that the score and number of structure domain nodes and links shared is not significantly different when a single task was performed prior to performing task *A*. The number of steps, number of new structure domain nodes created and number of new links learnt were all significantly lower when task *A* was performed after a related task. This implies that, although agents learned links related to task *A* and created several structure domain nodes while performing the task *B*, the knowledge relevant for task *A* neither became grounded, nor has it significantly decreased in weight. Therefore, the number of links and solutions proposed did not significantly differ from the case where agents performed only task *A*. In all other cases, agents reach the conclusion faster in terms of steps needed, and accordingly, number of newly created structures, links learned, and solutions and links shared. However, the score as determined by the team was not significantly better when ten tasks *B* were performed prior to task *A*. Further, while the objective score was not significantly lower for any of the cases, for the case where there are ten tasks *B* performed, the hypothesis that an actual shift of the score is toward negative can be accepted at significance level of 0.01. Close inspection of the data showed that when multiple performances of task *B* occurred, the weights of the links relevant for the task *A* were decreasing, while links important for the task *B* gained weight, causing agents to more often chose solutions appropriate for the task *B* rather than for the task *A*.

To further inspect whether the overlap between task *A* and task *B* impacts the results for the second task sequence type, two datasets were extracted: cases where suitable (objective) solutions for a task *A* were found by the agents while performing task *B* and cases where no real solutions for task *A* were found while performing any of the tasks *B*. The first contains instances where a solution for the task *A* was found in almost every run, and the second contains data on the simulations where no solutions for the task *A* were found in any of the runs.

It is expected that, in the case where no solution for task *A* has been found, objective solution scores do not vary significantly between cases. When no solution for task *A* has been found, the score as rated by agents is significantly higher after several runs of task *B* indicating that some partial solutions have been found during the performance of task *B*. Further, in several cases, wrong links caused the agents to finish their search, thus resulting in lower number of steps required. However, as expected, the objective score for the solutions found did not differ significantly between any of the cases.

Regarding the instances where solutions for task *A* were found while performing task *B*, the simulations where ten tasks *B* were performed are the most interesting. For such simulations, the hypothesis that an actual shift of the score is toward the negative was found supported at a significance level of 0.05. In other words, per-

forming task *B* a large number of times decreases the probability of finding a correct solution for the task *A*, even though several possible solutions have been encountered throughout completing the tasks. This example demonstrates the detrimental effects overspecialization of teams can have on the performance of the tasks which fall outside of the team's expertise area. On the individual level, team member's overspecialization has been demonstrated to have a detrimental effect on team performance as it related to de-skilling of individuals and the reduction in the common ground between team members [25, 26]. Research in [25] and [26] deals with the case where team members' expertise areas have *too little* in common. The simulated case demonstrates the other side of the spectrum: the case where team member's expertise areas have *too much* in common, i.e., the case of overspecialization on the team level. After multiple performances of a single task *B*, agents developed and grounded a shared knowledge related to the task *B* which enabled them to reach the solution efficiently, but the rest of their mental models remained unused and declined in weight which resulted in team's incapacity to find a solution when presented with a novel task. Therefore, as concluded in [17], there is a need for balance in shared and distributed knowledge within the team.

Conclusion

Team performance is necessarily influenced by knowledge held by its members. However, the relationships between members' mental models, team interactions, and team performance are still not adequately understood. This paper contributes to knowledge about the effect team experience has on team performance by utilizing computational simulations to demonstrate the increase in team efficacy as team members collaborate on multiple tasks. Further, by modeling designers as cognitively rich agents, the model enabled some insights into the cognitive behaviors of designers. The process of knowledge grounding and its effect on the team's convergence to the solution was studied, and findings match the existing cognitive theories, thus displaying the capability of agent-based models to aid research on design teams. However, additional studies are needed to further enhance understanding of coevolution of team members' mental models and the mutual impact of tasks performance and team experience. New simulation runs are needed to fully verify the model and to study how variability across factors influences results. Future studies will be aimed at running the model in different scenarios, therefore providing additional insights into each of the modeled aspects. Further, additional experiments will provide deeper insights into the scale independence of results (regarding task and team size) and will focus on detailed studies of the influence of various factors on the team performance. For example, questions such as how do the results vary in relation to the level of overlap between tasks performed, how does the diversity (among members) and scope of agent's initial knowledge influence the overall performance, how does cohesiveness of the group affect the results and how does the rate of forgetting affect performance. Finally, what is the effect of task uncertainty, goal clarity, and envi-

ronmental turbulence on team processes and behavior should be further explored. It is important to note that these experiments dealt with stable teams (i.e., with clear boundaries and with no change in membership) which are rare in product development domain. Therefore, future research will concentrate on expanding these experiments by including multiple teams working on different projects, and by simulating team members leaving, entering, or returning to a team. Finally, the limitations in the task representation should be tackled. Particularly, product development teams are often faced with the requirement of novelty and the currently implemented agents have limited capability in dealing with such a requirement. Future research will be aimed at providing a richer task and team representation and enabling more realistic simulations.

The results from the experiments reported here suggest there is a detrimental influence of knowledge inaccuracies, forgetting and overspecialization on team performance. Additional experiments should further explore how much of the knowledge overlap is beneficial for the team, as well as what knowledge should be shared [17]. The current model could serve as a valuable experimentation tool in such studies as it enables simulating multiple different scenarios, thus producing large amounts of data which can be further studied by deep learning algorithms and potentially reveal additional patterns of design behavior.

Acknowledgements This paper reports on work funded by the Ministry of Science, Education and Sports of the Republic of Croatia, and Croatian Science Foundation MInMED project (www.minmed.org). This research is supported by a grant from the National Science Foundation to the third author under Grant CMMI-1400466. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Gero JS, Peng W (2009) Understanding behaviours of a constructive memory agent: a markov chain analysis. *Knowl-Based Syst* 22:610–621
2. Light B, Butterworth G (2016) *Context and cognition: ways of learning and knowing*. Routledge, London
3. Gero JS, Kannengiesser U (2004) The situated function-behavior-structure framework. *Des Stud* 25(4):373–391
4. Kannengiesser U, Gero JS (2012) A process framework of affordances in design. *Des Issues* 28:50–62
5. Gero JS, Fujii H (2000) A computational framework for concept formation in a situated design agent. *Knowl-Based Syst* 13(6):361–368
6. Maher ML, Gero JS (2002) Agent models of 3D virtual worlds. In: *ACADIA 2002: thresholds*, pp 127–138. California State Polytechnic University, Pomona
7. Kahneman D (2011) *Thinking, fast and slow*. Farrar, Straus and Giroux, NY
8. Gero JS (2017) Whither design theory and methods? Back to the future: the next 50 years. *ANZAScA*, pp 563–572
9. Sivasubramaniam N, Liebowitz SJ, Lackman CL (2012) Determinants of new product development team performance: a meta-analytic review. *J Prod Innov Manag* 29:803–820

10. Carbonell P, Rodriguez AI (2006) Designing teams for speedy product development: the moderating effect of technological complexity. *J Bus Res* 59:225–232
11. Akgun AE, Lynn GS (2002) Antecedents and consequences of learning in new product development teams. *J Eng Technol Manage* 5:57–72
12. Gruenfeld DH, Fan ET (1999) What newcomers see and what oldtimers say: discontinuities in knowledge exchange. In: *Shared cognition in organizations: the management of knowledge*, pp 245–268. Lawrence Erlbaum, Mahwah
13. Levine JM, Moreland RL (1999) Knowledge transmission in work groups: helping newcomers succeed. In: *Shared cognition in organizations: the management of knowledge*, pp 267–296. Lawrence Erlbaum, Mahwah
14. Choi HS, Thomson L (2005) Old wine in a new bottle: impact of membership change on group creativity. *Organ Behav Hum Dec* 98:121–132
15. Gibson CB, Gibbs CL (2006) Unpacking the concept of virtuality: the effect of geographic dispersion, electronic dependence, dynamic structure, and national diversity on team innovation. *Admin Sci Quart* 51:451–495
16. Hirst G (2009) Effects of membership change on open discussion and team performance: the moderating role of team tenure. *EJWOP* 18:231–249
17. Badke-Schaub P, Neumann A, Lauche K, Mohammed S (2007) Mental models in design teams: a valid approach to performance in design collaboration? *CoDesign* 3:5–20
18. McComb C, Cagan J, Kotovsky K (2015) Lifting the veil: drawing insights about design teams from a cognitively-inspired computational model. *Des Stud* 40:119–142
19. Ambler NP (2015) Design in the modern age: investigating the role of complexity on the performance of collaborative engineering design teams. Ph.D. Thesis, Virginia Polytechnic Institute and State University
20. Singh V, Dong A, Gero JS (2013) Social learning in design teams: the importance of direct and indirect communications. *AIEDAM* 27:167–182
21. Kannengiesser U, Gero JS (2004) Modelling expertise of temporary design teams. *J Des Res* 4:1–13
22. Sosa R, Gero JS (2013) The creative values of bad ideas. *Proc CAADRIA* 2013:853–862
23. Van den Bossche P, Gijsselaers W, Segers M, Woltjer G, Kirschner P (2011) Team learning: building shared mental models. *Instr Sci* 29:283–301
24. Perisic MM, Storga M, Gero JS (2017) Building a computational laboratory for the study of team behaviour in product development. *Proc ICED'17* 8:189–198
25. Carley K, Gasser L (1999) *Computational organization theory, multiagent systems: a modern approach to distributed artificial intelligence*. MIT Press, Cambridge, pp 299–330
26. Anderson EG, Lewis K (2014) A dynamic model of individual and collaborative learning amid disruption. *Organ Sci* 25:356–376

An Exploration of the Effects of Managerial Intervention on Engineering Design Team Performance



Joshua T. Gyory, Jonathan Cagan and Kenneth Kotovsky

Engineering is often done in teams, but teams are not always efficient and maximally proficient. This work explores whether a portion of the resources used for solving a problem could instead be used to manage and control the design process of the remainder of a team. Consequently, this research begins to investigate whether a manager can improve the collective performance of engineering design teams. To study these manager–group interactions, a controlled human experiment was run in which novice engineering students solved a conceptual design problem under either the absence or guidance of a process manager. Results show that teams under the direction of a manager outperform unmanaged teams in terms of the novelty and quality of their final design solutions. Moreover, the types and motivation of the manager interventions are analyzed to begin to uncover some of the mechanisms and cognitive rationale that lead their teams to this superior performance.

Introduction

A vast majority of engineering problems are solved in teams, especially those that are complex and dynamically changing in nature. Research on team dynamics has roots grounded in the fields of cognitive and social psychology [1, 2]. In these areas, extensive research revolves around not only characteristics intrinsic to individuals themselves, but also to the interaction of the team as whole [3]. With an awareness of one's own abilities and the collective knowledge of the entire group, effective communication strategies can lead to a common, shared mental model of the problem and increase team synergy [4].

J. T. Gyory · J. Cagan (✉) · K. Kotovsky
Carnegie Mellon University, Pittsburgh, USA
e-mail: cagan@cmu.edu

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_33

Beyond its origins in psychology, strategies for team dynamics have become commonplace in organizational and behavioral management. For example, an effective approach for committee decision-making is to have individuals initially generate ideas and develop facts, before shifting toward group interaction and discourse for feedback [5]. Proper facilitation of group members is valuable not only for decision-making, but to also abate conflict in teams and organizations, which in turn, leverages team cohesion [6, 7]. Creative problem-solving can also benefit from effective management that fosters productive learning environments and engages its group members [8, 9]. A considerable amount of effort in the field of engineering design also focuses on the study of human behavior and creative problem-solving [10].

Engineering design teams may often reach impasses, especially while solving challenging design problems [11]. Additional aid in the form of analogies and solution examples stimulates group creativity and improves solution quality, particularly during the ideation phase of conceptual design [12, 13]. However, such aid can also lead to design fixation, which inhibits creativity and can lead to inferior overall group performance [14, 15]. Perhaps these difficulties can be ameliorated if some of the resources used for solving the problem are instead used to manage and control a team's design process. However, the question remains: under what circumstances and with what methods would management be most beneficial to an engineering design team?

Accordingly, the goal of this research is to assess the effect of managerial intervention on the performance of a design team, particularly when working on a conceptual engineering design problem. This research goal is empirically addressed by comparing the performance of managed teams, which have fewer members directly solving the problem but with a manager overseeing their design process, with teams that have more members devoted to directly solving the design problem and no manager. A supplementary analysis of the manager interventions is done to identify and provide insight into those effective management strategies. The first section of this paper provides an overview of the behavioral study, experiment methodology, and methods for data analysis. Preliminary results on team performance and managerial behavior are presented in the following sections, with a discussion of their implications. The paper concludes with some final remarks and directions for future work.

Methodology

Participants

To study these manager–group interactions, data was collected from a study run at Carnegie Mellon University in the mechanical engineering, introductory course, “*Fundamentals of Mechanical Engineering*,” with 72 freshman participants. Running the experiment in this specific course made it possible to control for a novice-level understanding of engineering design theory and methodology. These 72 freshman

students were the *novice designers*. Because the study was run during their scheduled class time, they were not monetarily compensated, but instead, were provided an educational lecture on engineering ethics during the portion of the class time they were not participating in the study. This study was approved by Carnegie Mellon University's institutional review board, and prior to taking part, all participants were asked to read and sign consent forms.

Mechanical engineering graduate students were also recruited for the study to serve as the *managers*. The graduate students were selected via a recruitment survey to ascertain whether they possessed two desired managerial attributes: design knowledge and leadership experience. Using a Likert scale assessment, questions on the survey asked the graduate students to self-assess their mechanical engineering design knowledge as well as their leadership experience. In order to minimize survey bias due to overconfidence, further questions asked them for specific details such as undergraduate/graduate classes they had taken in engineering design, the area of their primary research, interests outside of their primary research, and specific examples they had in leading a team. The graduate students with the highest levels of design knowledge and leadership experience were selected as the managers for the study. These participants were compensated \$10 per hour of their time not only for the experiment itself, but also for a manager training session and a post-study interview.

The novice designers were randomly assigned to one of two different team conditions: a managed team or an unmanaged team. Managed teams consisted of four novice designers and one manager, and unmanaged teams consisted of five novice designers and no manager. Thus, the resources of the fifth novice designer in the unmanaged teams were reallocated to the process manager in the managed teams. There were eight groups in each condition, totaling 72 freshman students and 8 graduate mechanical engineering student participants.

Experiment

During the experiment, teams in both conditions were given 30 min to solve the following conceptual engineering design problem [12, 16]:

Problem Statement – “Design a low-cost and easy to manufacture device that removes the outer shell from a peanut.”

To simulate a dynamically changing problem a design engineer would more likely experience in practice, two additional modifications to the original problem statement were introduced 10 and 20 min into the study in the form of constraints. These added constraints were also meant to exacerbate the problem difficulty throughout problem-solving, and to test whether the managers were better able to help their teams overcome such disruptions. Respectively, these constraints stated:

Constraint 1 – “The device is meant to be utilized in developing countries where electricity may not necessarily be available as a power source.”

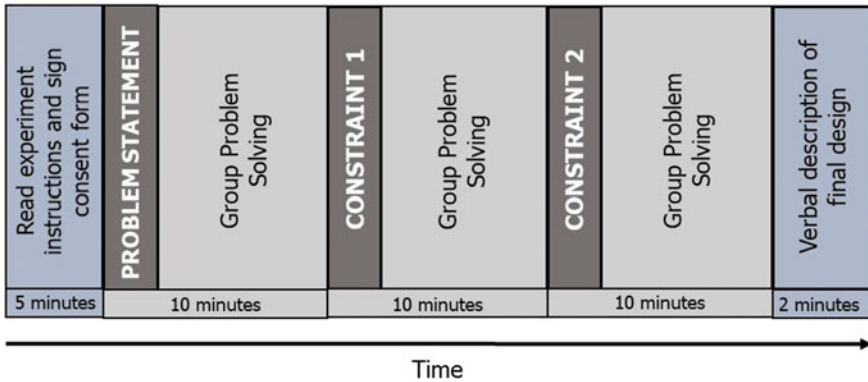


Fig. 1 A time line of the experiment illustrating when constraints were added to the original problem statement

Constraint 2 – “In addition to the previous constraint, the proposed design must be able to separate a large quantity of peanuts from their shells, while causing minimal damage to the inner peanut.”

The teams were encouraged to come up with and discuss as many design ideas as possible, but by the end of the experiment, they were told they had to agree upon a single, final design solution.

An overview of the experimental time line, including when the constraints were introduced, is shown in Fig. 1. Using audio recorders, the group discourse was collected throughout the experiment. Each time a constraint was introduced, the team was instructed to continue their sketches on a new sheet of paper which they were provided with, and so, the evolution of each team’s designs was also documented. By the end of the experiment, each team had three sheets of sketches, with the last sheet containing their final design solution. At the conclusion of the study, one team member was also asked to provide a brief verbal description of the group’s final design.

The unmanaged teams were under the supervision of passive experimenters who could only read the experiment instructions to their teams, monitor their time, provide experimental materials (sheets of paper, constraints to the problem), and work the audio recorder. The managed teams, on the other hand, were under the supervision of a graduate student manager who could intervene with their team to affect the solving process, but not directly contribute to the problem solution. Even though the managers were instructed to intervene when they felt it necessary, they were only allowed to interact with prescribed stimuli. Inspired by the approaches of design by analogy and metaphor [17], cognitive priming with solution examples [12], and different heuristics for creative problem-solving and management [8], the following intervention types were, respectively, chosen: keywords, design components, and design strategies. This collection of permissible stimuli (see Fig. 2) was created for this particular problem from questions in the manager recruitment survey. In that







MANAGER BANK		
Keywords	Design Components	Design Strategies
Separate		"Can you clearly identify the assumptions, constraints, and goals of the problem?"
Manufacturability		"Can you think of similar ideas that already exist?"
Affordable		"Are the requirements being met in your current design? Can you go back to a previous idea?"
Simple		"Why don't you split up and work on different components of the design?"
High Throughput		"Perhaps make a pro/con list of your design ideas?"
Crank		"Why don't you all go around and list your own suggestion on a final design?"

Fig. 2 The *manager bank* containing stimuli that the graduate student managers were allowed to intervene with

survey, each potential manager was asked to generate possible items, in each of those categories, which they might provide to a hypothetical engineering team in solving the specific design problem. After the graduate students with the highest level of engineering design knowledge and leadership experience were chosen through the survey, their answers to these items were compiled. For each of the three intervention categories, the six answers with the highest frequency between the chosen managers were aggregated to form this *manager bank*. During the ensuing study, the managers were allowed to select and intervene with any stimuli from these three categories when they saw fit, but only these 18 specific examples, controlling for consistency in the types of interventions. The keywords and design components in the bank were printed on cards that were physically handed to the design teams, while the design strategies were verbally spoken by the managers, when deemed appropriate.

Prior to the experiment, the graduate student managers were required to participate in a 30 min training session. During this session, which was led by one of the research investigators, the managers were trained on the experimental procedures. Other than reading the instructions, answering logistical questions regarding the experiment, and intervening with a design strategy, the managers were not permitted to speak during the experiment. The managers were also told to keep notes on the exact times and types of interventions they used during the study with their teams. It was also emphasized to them that they were not to help their team in directly solving the design problem, but were there only to manage their team’s design process.

Within a few days following the experiment, a post-study interview was also conducted with each manager. During these interviews, the researcher went through each intervention and asked the managers: “What made you interact [with item

$x]$,” “Why did you interact with what you did,” and “What was the effect of your interaction?” The goal of these interviews was to determine what prompted the managers to interact with their team, in order to gain a deeper understanding into the managers’ motivations. The managers were allowed to refer back to the notes they had taken during the study as well as listen to the audio recording to recollect their memory, if necessary.

Analysis

To compare the performance between the unmanaged and managed teams, their final designs from the end of the experiment were evaluated. According to Shah et al., accurate measures of ideation effectiveness, as well as how a team explores within the design space, can be seen in the quality and novelty of their design output [18]. Therefore, the teams’ final designs were evaluated based on these two metrics.

The novelty of a design defines how unconventional an idea is compared to other designs in a given set. Note that this is a measure of uniqueness and does not take into account solutions found beyond the study set. Thus, a posteriori evaluation was computed, where comparisons were made relative to the ideas generated by the participants during the experiment. To meet all the engineering requirements from the problem statement, an adequate solution can be broken down into five sub-functions/mechanisms that must be satisfied by a design, with associated weights f . These include an energy conversion (human/natural to mechanical) mechanism (f_1), transportation of the peanuts through or along the device (f_2), crushing/de-shelling of peanut shells (f_3), sorting of the intact peanuts from their crushed shells (f_4), and the collection of the harvested peanuts (f_5). Shah, et al., mathematically formulated the posteriori measure of the novelty, N , of a team’s design as

$$N = \sum_{j=1}^n f_j \sum_{i=1}^m \frac{T_{ji} - C_{ji}}{T_{ji}} \times 10 \times p_i, \quad (1)$$

where T_{ji} is the total number of ideas generated for sub-function j , C_{ji} is the count of the current solution for function j , f_j is the weight assigned to function j , signifying its importance, n is the total number of sub-functions (in this case, $n = 5$), m is the particular stage of the design process, and p_i is the weight associated with that stage. Because the focus was only on one phase of the design process (the ideation phase), the above equation reduced to the following:

$$N = \sum_{j=1}^n f_j \frac{T_{j1} - C_{j1}}{T_{j1}} \times 10. \quad (2)$$

The multiplication of the constant 10 was to normalize the novelty scores on a scale from 0 to 10. The weights, f_j , for each of the five sub-functions were subjec-

tively chosen based on the importance of their contribution to the problem statement. Accordingly, the chosen weights for the sub-functions used to compute the overall novelty scores were

$$f_j = [0.25, 0.10, 0.35, 0.20, 0.10], \quad \text{where } 1 \leq j \leq 5.$$

Next, the quality of the final designs between the unmanaged and managed groups was compared. The quality metric refers to how well a particular design satisfied the engineering requirements of the problem statement. Two mechanical engineering graduate students at Carnegie Mellon University, with extensive experience in design theory and methodology, rated the quality of the final designs based on how well they satisfied the constraints of the problem. To assist in their evaluations, the two raters used a combination of the teams' final design and the 2 min audio description taken from the end of the experiment. Ratings were coded into categories $\{0, 1, 2\}$, with "0" being that the design violated constraints, "1" being that the design poorly satisfied the constraints, and "2" being that the design effectively satisfied all constraints stated in the problem. The raters were given these category descriptions but did not receive any further training on how to score the designs.

The audio files were transcribed via an outsourced vendor. All the transcripts were then meticulously checked over by the researchers for proper speaker identification. Next, these transcriptions, along with the notes taken by the managers during the experiment, were examined to check for the exact timing and type of manager interventions. The analysis of these interventions was compared across three equal intervals of the experiment. These intervals were delineated based on when the constraints were added during problem-solving (i.e., before the first constraint was given, between the first and second constraints, or after the second constraint). Thus, the analysis of managerial behavior compared the interventions across these three, 10 min time intervals of the experiment.

Results

At the conclusion of the study, the design teams are told to circle their final designs. These marked designs are extracted from a team's last sheet of sketches and used in the computation of the ideation metrics. A sample final design solution from each team condition is shown below in Fig. 3.

It can be difficult to identify the sub-functions and design architecture from these design sketches alone, so the audio recordings are used to supplement the analysis. The final audio descriptions for the designs in Fig. 3 follow, with the bolded/underlined text highlighting the key components of their design, forming the basis of the novelty calculations.

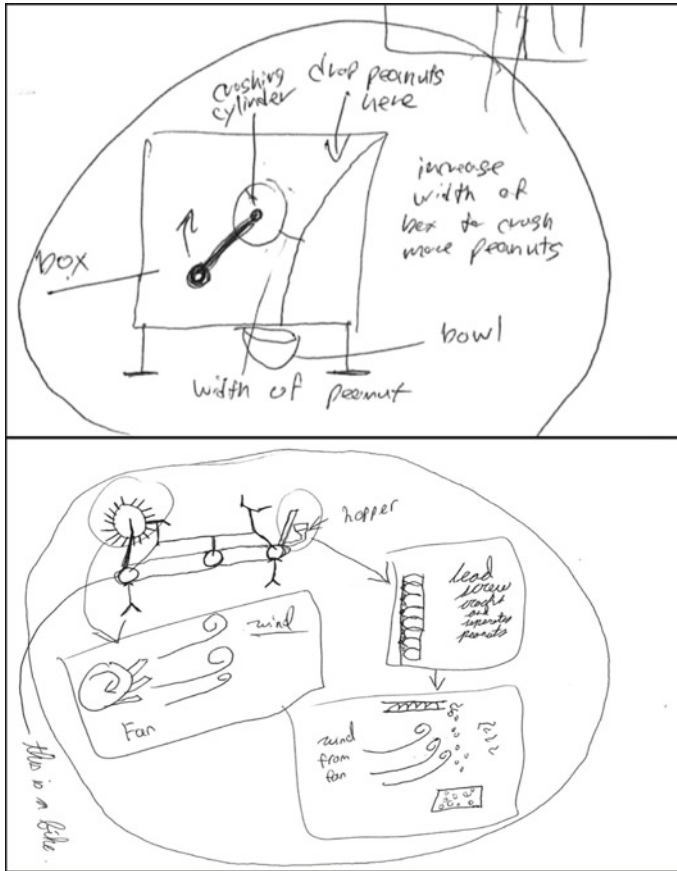


Fig. 3 Example final design solution from both an unmanaged (top) and a managed (bottom) engineering design team

For the first design, the unmanaged team's final description is:

It's basically a box with a uh, with kind of like a **sloped piece of wood** in order to ensure that when you drop the peanuts on the top of the slope they will actually slide down towards the **roller**, and not go flying past the roller. The roller is basically just like a **wood cylinder** so that way....And it's attached to a **crank**, so that way when you turn it, it will basically just crush the peanuts because the gap is the width, like the width of the peanut, so that would get crushed. Once they're crushed, the material falls into this **bowl** here, so, a little bit of manual labor is required to sort out the peanuts from the shell, but that's all that's required. And you're basically just dropping a bunch of peanuts and you can just keep crushing. Um, and the wider the box you make, and the longer the cylinder, the more peanuts you can crush at once.

The description from the managed team's final design is:

So using a fan, or I'm sorry, using a **bike** attached to the front tire having a **lead screw**, uh, with a hopper feeding into the lead screw mechanism would crush the peanuts, um, and

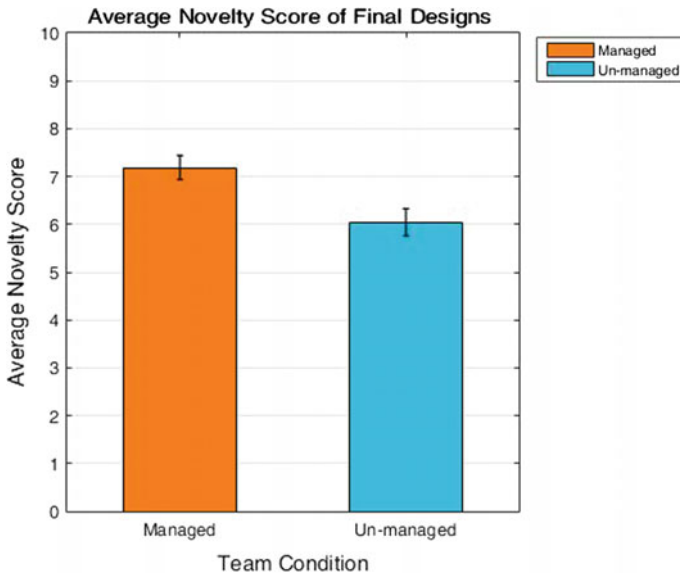


Fig. 4 The average novelty score of final designs for the managed and unmanaged teams. Error bars show ± 1 S.E

then at the end of the lead screw, they would fall out into a **box**, but there would be a **fan** in front of the box, also powered by the bicycle, which would blow off all the shells that would presumably have more drive than the peanuts.

As mentioned previously, the final designs are evaluated using a posteriori measure of novelty. The designs are evaluated based on how uniquely each of the five sub-functions is satisfied in a given solution, relative to all other design teams' solutions. Table 1 provides a summary for each of the five sub-functions and the count (C_j) for each mechanism among all the design teams. Summing the count for each mechanism shows how many teams in total, T_i , satisfy a particular sub-function, with i ranging from 0 to 16, for 16 total design teams. The highlighted mechanisms in Table 1 are those that are used only by managed teams. Out of the 14 most novel mechanisms, defined by their occurring only once (i.e., a count of 1 in Table 1), 9 of those happen to be used by managed teams.

Considering the total counts, T_j , the only sub-function that is satisfied by every single engineering design team is the crushing function, f_3 . This result is consistent with the chosen weights, f_j . For example, the sub-functions with the lower weights end up being fulfilled by fewer overall teams. Using the counts in Table 1 with Eq. 2, the novelty scores are calculated for each team. Figure 4 shows that the managed teams have a significantly higher measure of novelty than the unmanaged teams ($p < 0.012$).

To determine whether the chosen values of the weights significantly impact the results of novelty, a sensitivity analysis is performed. The functions are first weighted equally, $f_j = 0.2$ for each sub-function, which is essentially equivalent to no weight-

Table 1 The count of the different mechanisms used to satisfy the sub-functions identified in the teams' design solutions**Novelty Measurement**

<i>Energy Conversion</i> ($f_1 = 0.25$)		<i>Transportation</i> ($f_2 = 0.10$)		<i>Crushing</i> ($f_3 = 0.35$)	
<i>Mechanism</i>	C_1	<i>Mechanism</i>	C_2	<i>Mechanism</i>	C_3
Bike Pedal	3	Wind	1	Rollers	3
Crank	5	Conveyor Belt	3	Lead Screw	1
Lever	1	Inclined Slope	3	Hammer	1
Pivot	1	Gravity	1	Blade	1
Spring	2	Funnel/Cylinder	4	Cylinder	1
Hinge	3			Scissors	1
			$T_2 = 12$	Friction	2
	$T_1 = 15$			Crusher	5
				Claw	1
					$T_3 = 16$
<i>Sorting</i> ($f_4 = 0.20$)		<i>Collection</i> ($f_5 = 0.10$)			
<i>Mechanism</i>	C_4	<i>Mechanism</i>	C_5		
Sieve	6	Bucket/Basket	11		
Fan	1	Removable Base	1		
Mesh	2				
Mold	1				
Wedge	1				
	$T_4 = 11$		$T_5 = 12$		

Table 2 Quality ratings for the managed and unmanaged teams

		Rating categories		
		0	1	2
Team condition	Managed	3	3	10
	Unmanaged	4	10	2

ing at all. The results are identical to that found originally, with the managed teams having a significantly higher novelty score than the unmanaged teams ($p < 0.045$). Furthermore, each weight is successively perturbed $\pm 10\%$ from its original value, while readjusting the four remaining weights so that the total sum remains at 1 ($\sum f_j = 1$, for $1 \leq j \leq 5$). After recalculating the novelty score with each new combination of weights, in every case, the managed teams' novelty score remains higher than the unmanaged teams'. This analysis confirms that the scores are not sensitive to the weight values, so those originally chosen values will be the ones used for purposes of this work.

To calculate quality, each design is blindly rated by two mechanical engineering graduate students as either 0, 1, or 2, depending on how well the design constraints are satisfied. Table 2 shows the results for each team condition. All 16 designs are scored by both raters to gain objectivity in their scores, and their interrater reliability is used to measure the degree to which the raters are consistent with one another. The intraclass correlation coefficient (*ICC*) is used for this purpose and is computed as 0.632. Taking into account the small sample size ($n = 16$) as well as the fact that the raters are not previously trained, this result is satisfactory.

Table 2 shows that the quality data is skewed to the right; there are quite a few more designs in the 1 and 2 rating categories than 0. Because the data is ordinal and not normally distributed, a Mann–Whitney U-test, the nonparametric version of the t-test is run (which relaxes the assumption of normally distributed data). This test reveals that the managed teams have a significantly higher quality score than the unmanaged teams ($p < 0.02$), with almost all of the highly rated solutions (10/12) being managed teams. Thus, both the higher novelty and quality scores are indicative of better overall performance by the managed teams.

A preliminary analysis into managerial behavior is now done by examining the frequency and types of manager interventions. To recap, the allowable types of manager interventions are the keywords, design components, and design strategies that are depicted in the manager bank (Fig. 2). The interventions are counted for each 10 min experimental interval and binned according to type. The results are shown below in Fig. 5. In total, the managers intervened 52 times with 11 interventions in the first interval, 25 interventions in the second interval, and 16 during the final. The percentages shown in Fig. 5 are relative to the total number of interventions per interval. For example, 46% of the design strategy interventions in the first interval equate to five distinct interventions.

The post-study interviews are also analyzed to uncover the underlying motivations for what caused the managers to intervene. During these interviews, the managers

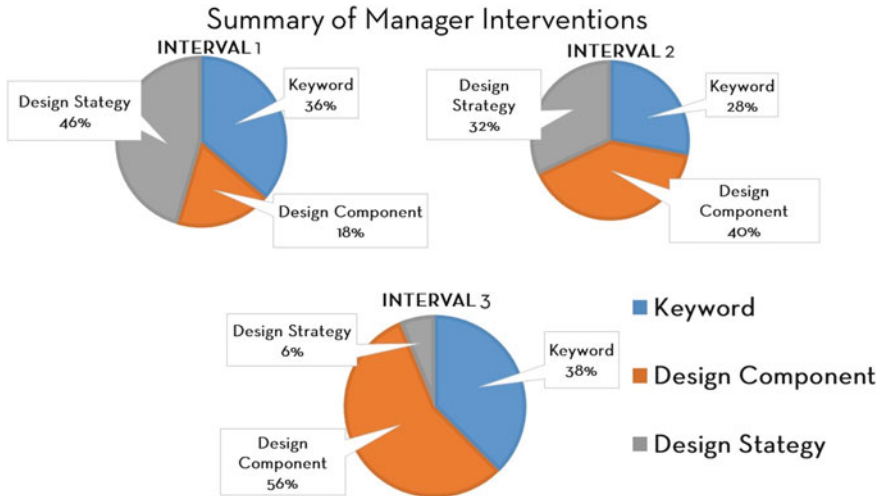


Fig. 5 Summary of manager interventions according to type, for all three experimental intervals

are reminded about each intervention and asked what caused them to intervene when they did and with what tool in the manager bank. After sorting all the motivations into common themes, four salient motivations emerged: assist the team in generating new ideas, help the team promote their current thought, remind the team of the engineering design requirements, and improve the team dynamics. To attain a better grasp of how these motives are characterized, consider this illustrative response from a graduate student manager on why they intervened with the “blade” design component:

They were getting really close to the idea. It was just to kind of push them a little bit farther in that direction.

This rationale constitutes a “promote current thought” motive. Furthermore, asking a manager why they intervened with the design strategy, “Can you think of similar ideas that already exist,” the response was:

There was no structure to their thought process, and there was no direction.

The above answer is characterized as “generate new ideas,” because the engineering team was not focused and having difficulty brainstorming initial concepts in the early stages of ideation.

Similar to the intervention types, these motivations are sorted with respect to each of the three experimental intervals. A summary of the managerial motivations is shown in Fig. 6. The percentages are relative to the total number of interventions per interval. For example, 18% of the “reminder of engineering design requirements” in the first interval equate to two distinct interventions.

One of the central trends captured in Fig. 6 is the steady increase of “reminder of engineering design requirements.” This development alludes to the idea that assisting the design teams in focusing on the constraints of the problem and attributes of their

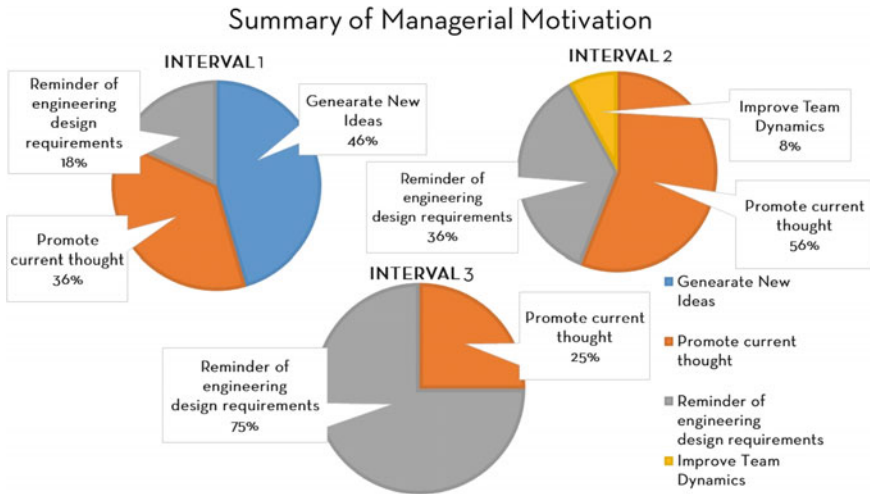


Fig. 6 Summary of manager interventions according to motivation, for all three experimental intervals

design is a major objective of the managers throughout the entirety of the study. Further analyses are discussed in the subsequent section.

Discussion

The results from this study illustrate that engineering design teams benefit from the guidance of a manager. Even though resources are reallocated from solving the problem to managing the design process, the managed teams, with one less member and therefore fewer direct problem-solving resources, still perform better. The managed teams produce final designs that are not only more novel but are also of higher quality than the unmanaged teams. Therefore, more team members may not always be the optimal circumstance, and perhaps, resources are better spent overseeing the process of design.

Generally, the managed teams satisfy more of the five sub-functions of the design problem and in more unconventional ways than the unmanaged teams, which results in higher novelty of their final design solutions. Of the 14 mechanisms (in Table 1) that are used by only a single design team, nine of those come from managed teams. Also, as depicted in Fig. 6, “reminder of engineering design requirements” is the only motivation to steadily increase in each consecutive interval of the entire experiment, suggesting that the managers play a significant (and increasing) role in guiding their team to think about all critical sub-functions of the design. During the post-study interviews, in response to the question, “What made you interact with [item x], and why did you interact with what you did,” some of the managers responded:

They had only focused on crushing the shell at that point and not thought about how to actually get the center of the peanut out of the shell.

Wanted to make sure they came up with a way to keep operation going if this was a process that was going to be scaled.

These responses underscore this incentive by the managers to remind their team of the engineering design requirements, and show that, throughout the entirety of the experiment, this was a major managerial strategy, particularly near the end when 75% of the interventions fell under this rationale.

Furthermore, the “design component” intervention type increases from 18% in the first interval to 56% of the manager interactions in the final interval of the experiment. Because the design components are all mechanical elements, this increasing trend in their usage is also consistent with the managers reminding their teams to consider different functions of the design. For example, providing a team with the “conveyor” stimuli reminds a team to consider the transportation of the peanuts through the device, and providing a team with the “sieve” component helps them to contemplate actually sorting the peanuts from their shells. One could argue that the managed teams become fixated and directly use the components that are provided to them during these interventions. Because the managers are trained not to speak when intervening with the design components, some of the teams may have perceived these particular interventions as additional requirements in their design. Even so, this does not undermine the fact that, overall, the managed teams’ designs are more novel. Also, as shown in Table 1, out of the most novel mechanisms (those with a count of 1), only the “hammer” is directly from the manager bank. As such, fixation is an unlikely implication from these interventions. The more comprehensive final designs from the managed teams also contribute to their higher quality ratings.

Now that the presence of a manager has shown to be beneficial to an engineering design team, the managing styles that led to this better performance can be studied. Broadly examining the trends in Fig. 6, one can study how the managers’ primary disposition shifts throughout the experiment, by identifying which motivations comprise the largest proportion in each experimental interval. In the first 10 min, 46% of the interventions are to guide their design teams in generating new ideas, which makes sense in the early stages of ideation when effective brainstorming is imperative. In the second interval, their behavior shifts to that of promoting their teams’ current thought (56% of the time). In the final experimental interval, for three-quarters of the interventions, the managers remind their teams of the engineering design requirements, which is critical in final design selection and refinement. Assisting teams to promote their current thought is another motivation to maintain a significant proportion through all three intervals (36% to 56% to 25%). This suggests that managers consistently contribute a supportive force in their efforts of managing their groups’ process of design.

Tracking the evolution of both the frequency and types of interventions can yield additional insight into this managerial behavior. Design strategies comprise the largest proportion of the interventions in the first interval of the experiment (46%) and the least in the final interval (6%). This result suggests that toward the

beginning of ideation, the managers want their teams to follow a more encompassing and exploratory search of the design space. Specifically, “*Can you think of similar ideas that already exist*” and “*Can you clearly identify the assumptions, constraints, and goals of the problem*” are the two significant design strategies in the early stages of the experiment. In contrast, the design components exhibit the opposite trend, amounting to the largest proportion in the last interval of the experiment (56%) and few in the beginning (18%). The increasing implementation of design components, particularly near the end of the experiment, indicates that the managers try to get their teams to hone in on a specific region, or subset of solutions, within the design space.

This exploratory-to-convergent funneling of design team efforts is also mirrored by the manager motivations. In the beginning of the experiment, the primary motivation is to help their team generate new ideas, encompassing 46% of the interventions. By the end of the study, the managers are more inclined to remind their teams of the engineering design requirements (75% of the interventions). Thus, the predominant tendency in managerial behavior is to push their teams to follow this exploratory-to-convergent search of the design space, which has been shown to be an effective strategy in design problem-solving [19].

Conclusions and Future Work

This study is an early exploration of managerial intervention in team-based engineering design. The primary research goal considers partially taking resources from problem-solving in teams and reallocating them to the management of the design process. Specifically, this behavioral study compares the performance of five-member teams with that of four-member teams under the guidance of a process manager, on a conceptual engineering design problem. The results in this paper demonstrate that teams, with resources partially devoted to managing the design process, even while shorthanded, are more effective than teams with an additional problem solver. This superior performance is reflected in the higher novelty (uniqueness) and quality of the managed teams’ final design solutions to the problem. Thus, process managers are beneficial in guiding a team through the ideation phase of the design process, even though the managers are specifically precluded from directly contributing to the design solutions.

After demonstrating the profitable effect of managing resources on engineering design teams, a preliminary analysis into managerial behavior is done by tracking the types and motivations of interventions throughout the experiment. The general pattern emerging in manager strategy is an exploratory-to-convergent managerial style. Future analysis could determine the extent of divergence in the group and when convergence actually occurs. Additionally, managers help their teams consider all functional aspects of the design problem, even those functions that are less perceivable from the problem statement, which allows them to generate solutions of both

higher novelty and quality. The managers also create a supportive force by providing interventions to bolster their teams' current thought throughout the experiment.

The data from this study presents numerous opportunities for future work. To gain a deeper understanding into managerial behavior, a more refined and in-depth analysis can be done to observe the specific modalities and timing of interventions that are most effective for engineering design teams. Perhaps an analysis of the underlying semantic structure of design team communication can augment the current findings, and provide additional insights into intervention effects. Furthermore, the evolution of each teams' designs can be tracked throughout the experiment, to see if the managers enable the teams to overcome many of the stumbling blocks of problem-solving such as fixation and lack of effective search.

Ultimately, the hope is to procure a deeper understanding of manager–group interaction strategies and their benefits to engineering design teams. Because this empirical study utilizes freshman engineering students from a university and limits the problem-solving process to a short time frame, there are still unanswered questions about the generalization of the results outside of the lab setting. Therefore, future work can also consider if the results and observations from this current work extend to a larger scale and apply to engineering design teams in industrial practice. This study is merely a first step toward uncovering approaches and methods that can build more focused and maximally efficient engineering design teams.

Acknowledgements This work was supported by the Air Force Office of Scientific Research (AFOSR) under grant No. FA9550-16-1-0049 and grant No. FA9550-18-1-0088. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the sponsors.

References

1. Laughlin PR (2011) Group problem solving. Princeton University Press
2. Hoffman LR (1966) Group problem solving. In: *Advances in experimental social psychology*, pp 99–132
3. Littlepage G, Robison W, Reddington K (1997) Effects of task experience and group experience on group performance, member ability, and recognition of expertise. *Organ Behav Hum Decis Process* 69(2):133–147
4. Kim MS (2007) Analysis of team interaction and team creativity of student design teams based on personal creativity modes. In: *Proceeding of ASME 2007 international design engineering technical conferences and computers and information in engineering conference*, pp 1–13
5. Ven A, Van De, Delbeco AL (2017) Nominal versus interacting group processes for committee decision-making effectiveness 14(2):203–212
6. Tesluk PE, Quigley NR, Tekleab AG (2009) A longitudinal study of team conflict, conflict management, cohesion, and team effectiveness. *Gr Organ Manage* 34(2):170–205
7. Turkalj Z, Fosic I, Dujak D (2008) Conflict management in organization. *Interdiscip Manag* (917340826)
8. Isaksen SG (2013) Facilitating creative problem-solving groups [Online]. Available: <http://www.cpsb.com/research/articles/creative-problem-solving/Facilitating-CPS-Groups.html>
9. Treffinger DJ, Isaksen SG (2009) Creative problem solving: the history, development, and implications for gifted education and talent development. *Gift Child Q* 49(4):342–353

10. McComb C, Cagan J, Kotovsky K (2015) Lifting the veil: drawing insights about design teams from a cognitively-inspired computational model. *Des Stud* 40:119–142
11. McComb C, Cagan J, Kotovsky K (2015) Rolling with the punches: an examination of team performance in a design task subject to drastic changes. *Des Stud* 36(C):99–121
12. Fu K, Cagan J, Kotovsky K (2010) Design team convergence: the influence of example solution quality. *J Mech Des* 132(11):111005
13. Linsey JS, Markman AB, Wood KL (2012) Design by analogy: a study of the wordtree method for problem re-representation. *J Mech Des* 134(4):041009
14. Jansson DG, Smith SM (1991) Design fixation. *Des Stud* 12(1):3–11
15. Sio UN, Kotovsky K, Cagan J (2015) Fixation or inspiration? A meta-analytic review of the role of examples on design processes. *Des Stud* 39:70–99
16. Linsey JS, Tseng I, Fu K, Cagan J, Wood KL, Schunn C (2010) A study of design fixation, its mitigation and perception in engineering design faculty. *J Mech Des* 132(4):41003
17. Hey J, Linsey J, Agogino aM, Wood KL (2008) Analogies and metaphors in creative design. *Int J Eng Educ* 24(2):283–294
18. Shah JJ, Smith SM, Vargas-Hernandez N (2003) Metrics for measuring ideation effectiveness. *Des Stud* 24(2):111–134
19. Dong A, Hill AW, Agogino AM (2004) A document analysis method for characterizing design team performance. *J Mech Des* 126(3):378

A Study in Function Modeling Preferences and its Variation with Designer Expertise and Product Types



Xiaoyang Mao, Chiradeep Sen and Cameron Turner

This paper presents a preliminary study of modeler preferences while constructing function models of technical systems. Function models of three product types distinguished by their main functions, each constructed by groups of expert and novice modelers, are compared. The topology of a model is codified as an adjacency matrix, whose top row and first column are the function verbs from the Functional Basis vocabulary and each cell shows the number of flows that go from a verb to another. The similarity between these matrices is then computed using matrix correlation functions. Analysis the data shows that the expert modelers show a stronger similarity of modeling preferences than the novices and that within each modeler group, modeling preferences are more similar within a produce type than across product types. The observations indicate that with increasing modeling expertise, designers tend to develop modeling preferences, which may vary with the product types modeled.

Introduction

Engineers relied on hand-sketches and drafted drawings to develop design concepts before Computer-Aided Design (CAD) systems emerged. CAD systems were built on geometric primitives, a vocabulary, combined with a grammar for developing high-order concepts from the primitives. They enable engineers to explore early design concepts on the computer. Even higher order reasoning about the performance or manufacturing of the design could be done in early design stages by combining

X. Mao · C. Sen (✉)
Florida Institute of Technology, Melbourne, USA
e-mail: csen@fit.edu

C. Turner
Clemson University, Clemson, USA

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_34

CAD systems with engineering analysis and manufacturing systems. However, CAD systems can be used only if the model's form-related information is available (form = geometry + material [1]). There is no CAD-equivalent system that provides for modeling, reasoning, and decision support, like that for the geometric CAD system, to the early design modeling approaches where the form of the model is not yet known.

In the abstract phase of the design process, the goal of the designer is to identify the functional underpinnings that will ultimately lead to the form of the design artifact. In order to help designers with identifying the functional relationships of the design, multiple representations have been developed to provide a connection between an abstracted system concept and the physical reality of the design artifact. This research recognizes that similar to CAD, where the same geometry could be produced using alternate correct approaches, such as building a block using the block feature or by extruding a rectangle, the abstract function models may also exist as multiple "correct" models, based on modeler preferences. These models are different in the level of abstraction and due to individual designer perspectives, but they all can provide the same behavior of the artifact. This phenomenon that two designers see the same system differently increases the complexity of developing any computational support tool for these modeling languages greatly and increases the difficulty to assess the true difference between two valid models significantly.

Background and Research Approach

It is well known in design that different designers can produce equally valid abstract models of the same design even though those models may have different analytical expressions [2]. These differences are probably the result of the "Veridicality of Perception" which is an underlying cognitive process related to designers' own experiences with illusion. One of the most common examples is shown in Fig. 1. Some people will see a young lady while others see an old woman at the first sight. However, if you look closely, you can see both a young lady and an old woman. Research reveals that the object you recognize from the image is related to the preconceptions that you have when you see the image [3]. Similarly, the preconceptions about the system and the modeling process of designers may influence the model they construct of the system.

In the case of function modeling in reverse engineering, the physical product is present and largely dictates the functions, flows, and topology that could be added to the model, since the task is to describe the product accurately rather than create new product ideas. In forward design, the model is still bounded by the overall functionality of the product but is more open to include preferences and biases as it reflects the designer's plan for the product's desired functionality. The similarity between two function models could then result from at least two factors:

Fig. 1 A young lady or an old woman illusion [4]



Factor 1: Products of similar overall functions often work in similar ways and thus could have similar models and

Factor 2: models created by designers of similar background, experience, or culture could display similarity owing to the preferences of the modelers, whether the modeled products are similar or not.

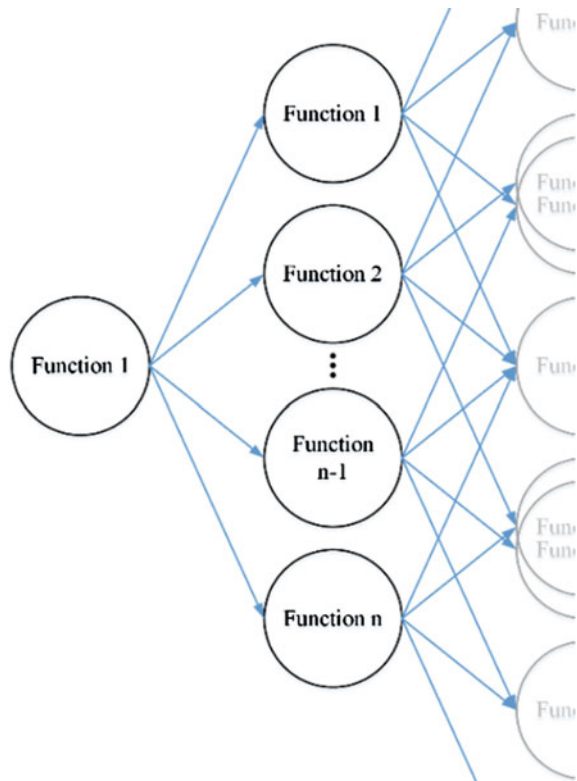
The goal of the study is to look for preliminary evidence for or against these factors from function models of different product types created by different designer groups.

It is emphasized that the study reported here is not a controlled experiment. The designers were not observed while creating the models, no data was collected during the modeling session that reflects the modeling process, and there was no protocol used to analyze the data. Instead, completed function models of three different product types were obtained from two designer groups of vastly different expertise levels, and compared in search of evidence of modeling preferences. Thus, this study could be better described as a case study rather than a protocol study or a user study. The conclusions could be used at the best to indicate the possibility of certain patterns or hypotheses, rather than to draw conclusive inferences.

For clarity's sake, "modeling" is the process of constructing a model and "model" is the end artifact of that process. The data available to conduct this study did not include the process, only the model. While preferences and stylistic variations of designers could be better studied with combined data from the process and the model, the basis of modeling preferences in this study is the similarity of modeling choices evident from the topological similarity of completed models without observing the process. The benefit of such an approach is that it could detect at least some preliminary evidence of preferential variations if they were strongly present, but with little investment in designing and conducting a proper protocol study that could potentially fail if no such pattern indeed existed. Thus, this study is a preliminary study, not a complete one.

When a certain function verb occurs in a model, esp. in forward design where the model is dictated more by the designer than a physical product, the designer is faced with choices for the neighboring function verbs that accept flows produced by the first verb or produce flows accepted by it, as shown in Fig. 2. For each topological connection, the designer chooses one of the multiple options of the neighboring functions available, whether up the flow or down the flow. Therefore, from the cumulative

Fig. 2 Conceptual representation of the network beginning with Function 1 [5]



behavior of one or more designers, one could ideally determine the probability of a particular function occurring in a function's immediate neighborhood.

Then, if models created by a group of designers of the similar background shows a certain topological similarity mutually that are absent in models from another group, a preferential pattern could be inferred. This pattern is an attribute of the end model and not of the process, at least to the extent data is available. It should not be extended to infer similarity of processes, such as "designer group A prefers to use the forward chaining process, while designer group B prefers backward chaining", etc.

Function-Based Design

The function-based design is a way of modeling technical systems in terms of their abstract functions. It can reason on the system's functionality in order to perform a variety of design tasks without commitment to a particular form. It is a well-recognized approach to model design concepts in an abstract form-neutral manner [6, 7], to study existing designs through reverse engineering [7–11], and to explore solution variants [6, 7, 12].

An engineered artifact is designed to address a particular need and the technical functions of the artifact is an abstract way to describe the artifact's actions which are selected by the designer to satisfy that need [6, 7]. Various design models have been produced by using the philosophy that design is a process to derive form from function [13–19]. Function was described as the goal or purpose of the product by Freeman and Newell in [13]. Analyzing and thinking about the design problem and product in terms of functions will help designers decompose the design problem [7, 20, 21], understand the workings of devices [7], expand the design search spaces [6, 7, 12], and archive existing designs for reuse [9, 22]. In order to do these activities, various representations and formalizations of functions have been proposed in Artificial Intelligence (AI) and engineering design research communities.

In the AI research area, function-based thinking and various approaches, in the range from representations [23, 24], ontologies [25, 26], languages [27, 28], and software tools [20, 29], have been proposed [30]. Device's functions are generally viewed as the result of the interaction between various aspects of the design, such as the device's structure and behavior, the device's interaction with its environment and, the user's intent [30]. For instance, the function has been defined as "the relation between the goal of a human user and the behavior of a system" [31]. Representations such as Function-Behavior-Structure (FBS) model [14] describing function as requirements of the device's performance [17, 32, 33], Function-Behavior-State (FBSt) model defining functions as "a description of behavior abstracted by human through recognition of the behavior in order to utilize it" [20, 24, 34, 35] and Structure-Behavior-Function (SBF) defining functions as sets of the input and output states and the behavior that cause state change [15, 30, 36] are broadly based on this view, although each of them supports a different type of reasoning. Notable examples are the FBS Modeler [20], which supports problem decomposition based

on the FBS, the IDeAL [15, 37], and Kritik [36] tools which support analogy-based search using the SBF, the Causal Function Representation Language (CFRL) [27, 38] which defines functions as cause-and-effect, and the Schemebuilder tool [29]. More function-based reasoning examples are provided in [39–41].

In engineering design, functions, in the systems-based view, are defined as transformations of material, energy, and signal flows through the system [6, 7, 12]. In the Contact-and-Channel model view, functions are described as the interaction between contacting surfaces [42, 43]. However, following discussions only rely on the transformative view. A graph-based representation, the function structure [6], where the nodes are the transformative actions and the edges are the flows of material, energy, and information subject to the transformations, is commonly used to this end [30]. Various tools and methods, using this representation, have been built to assist in conceptual design tasks. For example, problem exploration [44], problem decomposition [6, 7], solution search [12], solution synthesis [45–47], concept generation [48, 49], failure modeling [50–54], product similarity analysis [55], modeling signal flows [56, 57], and reverse engineering tasks such as design understanding and archiving [9, 22]—all leverage this representation. There are multiple levels of formalisms to model carrier flow relationships, computer-aided function design and form derivation from function description [56–61].

The Functional Basis (FB) [8], a popularly used vocabulary, consists of 53 function verbs and 45 flow nouns organized in a three-level hierarchy. This vocabulary was developed by empirical dissection of electromechanical products and cataloging their functions and flows through reverse engineering using function and flow keywords at Missouri University of Science and Technology [30].

The Design Repository at Oregon State University (OSU) [22, 62, 63] is a web-based archive storing the design information of a wide variety of product. This design information is obtained by tearing down the product through systematic reverse engineering. The Design Repository contains data about components, functions, and other aspects of the products. It currently contains information about 184 products and 6906 unique components and is used to support multiple research projects. In the Design Repository, the function models are modeled using the FB as the language. Although there were initial attempts to create grammars to accompany the FB, formal grammars have not been developed. The FB vocabulary and the Design Repository have been significant assets for design research over the past several decades. Various of the reasoning systems, methods, and tools related to the Engineering Design section are based on the FB vocabulary and the Design Repository as a source of function models. Notably, the development of the FB, especially converging it into a finite set of terms and reconciliation with other vocabularies [64], is a testimony for the preference of human designers to reuse function terms in models. Yet, these human designers can construct multiple correct function models, which are equally valid of the same system.

Methodology

Source of Raw Data and Modeler Groups

As stated earlier, the study included two sets of function models from two groups of designers with vastly different expertise levels. The first set was obtained from a graduate-level design textbook [7] written by members of a research group that co-authored the Functional Basis vocabulary and was, therefore, deemed experts. The second set was produced as a part of a design coursework by a class of mechanical engineering seniors at Clemson University, who were deemed novices. The students selected or were given product web pages that contain the products' names, photographs, descriptions, user reviews, and other technical information, from which they were instructed to construct function models describing how those products could work. The students did not use physical products in reverse engineering. The exact profiles of these populations, e.g., modeling experience or expertise, was neither collected, nor measured, nor was possible to know. However, the novice group members are expected to have exposure to function modeling only through a junior-level design course and have created 1–3 models each. By contrast, the experts were members of a leading research group in function-based design involved in authoring the Functional Basis vocabulary. Thus, the expertise difference between the two groups was large enough to support preliminary conclusions sought in this paper.

Modeling Vocabulary

Both groups used the Functional Basis vocabulary, shown in parts in Table 1. The experts' choice of the vocabulary was not possible to control but the novices were instructed to use this vocabulary as a part of design instruction. This vocabulary has verbs organized in a three-level taxonomy, the secondary and tertiary level of which are shown here. None of the studied models used any primary-level verb. Between the secondary and the tertiary levels, it is notable that each secondary term also appears as a tertiary subclass and, therefore, the models were considered as comprising of the tertiary terms only. Some natural-language terms did occur in the novice models and were manually replaced with closest-matching tertiary terms.

Function Models Used as Data

Set 1 includes seven models from the experts that were divided into three subsets by their main functions: cut material (pencil sharpener, knife, fruit and vegetable peeler, engraver and weed trimmer), shoot material (Supermax ball shooter), and collect debris (vacuum cleaner). Set 2 includes 84 models of totally 17 products from the

Table 1 Part of The functional basis function List

<i>Import</i>	<i>Import</i>
<i>Export</i>	<i>Export</i>
<i>Transfer</i>	<i>Transfer</i>
	<i>Transport</i>
	<i>Transmit</i>
<i>Guide</i>	<i>Guide</i>
	<i>Translate</i>
	<i>Rotate</i>
	<i>Allow DOF</i>

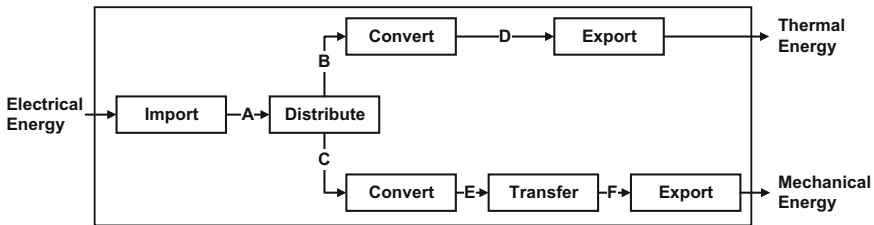


Fig. 3 Example: An example function model

Table 2 Example: the adjacency matrix of the example function model

		Flow goes to this function				
		Import	Export	Distribute	Convert	Transfer
Flow starts from this function	Import	0	0	1	0	0
	Export	0	0	0	0	0
	Distribute	0	0	0	2	0
	Convert	0	1	0	0	1
	Transfer	0	1	0	0	0

novices and include 2 products (11 function models) for cutting materials, nine products (45 function models) for shooting materials, and 6 products (28 function models) for collecting debris.

Data Processing Step 1: Translation to Adjacency Matrices

Each model is a graph, where the nodes and functions the edges are flows. Accordingly, each model was translated to its corresponding adjacency matrix. For example, the model in Fig. 3 produces the matrix in Table 2.

The top row and the first column lists all function in the model and each cell indicates the number of flows going from the verb on the first column of the cell to the verb on the top row of the cell. For example, the cell containing 2 implies that there are two flows (flows B and C in the figure) that go from Distribute to Convert, which in this case are two different instances of Convert but could be the same block as well. All other nonzero cells contain 1, reflecting the model topology, which can be easily verified from the figure. The zero cells imply no flow between the corresponding functions in the model. This process was repeated for all models, except that the top row and first column of the tables in those cases contained all 25 verbs of the tertiary-level vocabulary, thus yielding a 25×25 square matrix for each function model.

Data Processing Step 2: Model Correlations and Lumping

To determine the similarity between the function model topologies, the adjacency matrices for each model were compared within each set (expert, novice) and between the two sets using a correlation coefficient. The *CORREL* function in Excel was used to this effect and produced values between -100 and $+100\%$, which represents how strongly the two matrices are similar to each other. Table 3 is the correlation table of Set 1 (experts). The row and column headings are codes for function models. For example, “ESMA.16” reads as: “Expert, Shoot Material, model # A.16”. The 100% values on the diagonal indicate that each model is fully similar to itself, while the remaining cells are symmetric about this diagonal.

In order to see the pattern more clearly, the data was lumped to reveal the average correlation within and across product types, as shown in Fig. 4, which shows some operations done on the data of Table 3.

The diagonal row containing the 100% values was removed first. Then, the values within each of the five regions marked with thick lines and numbered 1 through 5 were averaged within each zone, thus producing the correlation table shown in Table 4. For example, the average value of box ① was calculated by dividing the sum of all values in the box (770%) by the count of the valued grids (30). The same lumping process was followed for the novice models in Set 2 as well.

Data Analysis: Within Expert and Within Novice Groups

Note that in Table 3 (expert designers), the model codes refer to product subsets, such as ECM=“expert, cut material”, ESM=“expert, shoot material”, and ECD=“expert, collect debris”. Reviewing the results from Table 3, we note that the correlation values within the same subset are generally higher than the correlation between the subsets. The minimum correlation value within the ECM subset is 22% and the maximum value is 53%. However, between different subsets, the minimum value is 13% and the

Table 3 Correlations between function models in Set_1

	Super Maxx ball shooter	Sharpener	Knife	Peeler	Engraver	Weed trimmer	Hand vaccum
	ESMA.16	ECMA.6	ECMA.7	ECMA.11	ECMA.12	ECMA.14	ECDA.5
ESMA.16	100%	17%	15%	16%	24%	27%	13%
ECMA.6	17%	100%	22%	29%	38%	41%	20%
ECMA.7	15%	22%	100%	53%	27%	36%	37%
ECMA.11	16%	29%	53%	100%	29%	35%	34%
ECMA.12	24%	38%	27%	29%	100%	34%	17%
ECMA.14	27%	41%	36%	35%	34%	100%	18%
ECDA.5	13%	20%	37%	34%	17%	18%	100%

	Super Maxx ball shooter	Sharpener	Knife	Peeler	Engraver	Weed trimmer	Hand vaccum
	ESMA.16	ECMA.6	ECMA.7	ECMA.11	ECMA.12	ECMA.14	ECDA.5
ESMA.16	100%	17%	15%	16%	24%	27%	13%
ECMA.6	17%		22%	29%	38%	41%	20%
ECMA.7	15%	22%		53%	27%	36%	37%
ECMA.11	16%	29%	53%	100%	29%	35%	34%
ECMA.12	24%	38%	27%	29%		34%	17%
ECMA.14	27%	41%	36%	35%	34%		18%
ECDA.5	13%	20%	37%	34%	17%	18%	100%

Fig. 4 Lumping process for Set 1

maximum value is 37%. The pattern can be clearly demonstrated by considering the lumped subsets as in Table 4. The correlations value within *ECM*, is 34% which is higher than the value between *ECM* and *ESM* and the value between *ECM* and *ECD* which are 20 and 25%, respectively. Therefore, we conclude that for models created by expert designers, models in the same product group agree with each other more than between different product groups. This pattern generally agrees with **Factor 1**

Table 4 Correlations between function models in Set 1 after lumping

	ESM(%)	ECM(%)	ECD(%)
ESM	100	20	13
ECM	20	34	25
ECD	13	25	100

stated earlier. In addition, the result also shows that when modeling products in the same product group, expert designers tend to follow similar topology choices.

The adjacency matrices of function models in Set 2 (novice designers) were generated and compared with each other in a similar fashion. The correlation between each of the adjacency matrices was calculated and used to generate Table 5. There are 84 novice models, which makes this model difficult to read but the color in Table 5 indicates the correlation between each model. The more red colors indicate lower correlations while the more green colors indicate higher correlations. The table shows that the correlation in the right bottom corner is higher than other places.

In order to see a clearer pattern, Table 5 was lumped by product types and then by student teams within the product groups, and the result is shown in Table 6. The thick lines show the lumping by product types: the NSM (novice, shoot materials) group is in the top-left, NCM (novice, cut materials) is in the middle L-shaped zone, and NCD (novice, collect debris) is in the lower right L-shaped zone. Within each zone, the first row and first column shows team ids and the number of models (equals the number of team members) within the team. For example, NSM5 in the top-left region stands for “novice, shoot material, team no. 5” and that it comprised of 5 models.

Patterns different than those in Set 1 are seen in Table 6. Most notably, it cannot be generally stated that models within the same product subset are generally correlated better than models across different subsets. Within the NSM subset, correlation ranges between 5 and 46%, but between the NSM subset and other subsets, it ranges between 9 and 37%, which is not significantly lower than the correlation within NSM. The same anomaly can be noticed in the NCM subset. However, the NCD subset has a similar pattern to Set 1, where the correlation within the subset is higher than across it. Within NCD, the correlation value ranges 15– 44%, whereas between NCD and the other subsets, it ranges between 9 and 37%. One more interesting phenomenon is that even within each team, the patterns are inconsistent. Some teams have low correlation values which mean they strongly disagree within themselves. However, some teams have high correlation values which mean they are more in agreement with themselves. These correlations are shown as small black boxes along the diagonal in Table 6.

In order to see the patterns of the novice set more clearly, the correlation table of Set 2 was lumped by subsets (product group) again as Table 7. The NSM subset contains 9 products with 45 models. The NCM subset contains 2 products with 11 models. The NCD subset contains 6 products with 28 models. The correlation value within NCD is 33%, which is higher than the value between NCD and NSM (19%) and the value between NCD and NCM (27%). However, the correlation value within

Table 5 Correlations between function models within novices group

Table 6 Correlations between function models in Set 2 after lumping by team

	5 Models		5 Models		6 Models		5 Models		5 Models		4 Models		6 Models		4 Models		5 Models		6 Models		5 Models		4 Models		3 Models		6 Models		5 Models		
	NSM3	NSM5	NSM7	NSM11	NSM14	NSM16	NSM19	NSM25	NSM27	NCM9	NCM17	NCD1	NCD2	NCD15	NCD20	NCD21	NCD22														
NSM3	10%	13%	7%	11%	11%	16%	9%	12%	12%	10%	10%	13%	12%	12%	9%	9%	11%														
NSM5	13%	5%	9%	11%	11%	18%	13%	15%	12%	14%	11%	14%	13%	15%	11%	12%	9%														
NSM7	7%	9%	11%	10%	11%	17%	12%	14%	15%	15%	16%	13%	14%	18%	18%	14%	10%														
NSM11	11%	11%	10%	8%	17%	18%	10%	11%	16%	11%	17%	13%	15%	13%	15%	13%	11%														
NSM14	11%	11%	11%	17%	46%	25%	14%	22%	28%	20%	30%	31%	37%	27%	26%	32%	33%														
NSM16	16%	18%	17%	18%	25%	32%	26%	31%	23%	25%	25%	24%	27%	28%	21%	24%	17%														
NSM19	9%	13%	12%	10%	14%	26%	21%	18%	18%	16%	16%	15%	18%	16%	12%	16%	16%														
NSM25	12%	15%	14%	11%	22%	31%	18%	21%	18%	22%	22%	22%	23%	23%	22%	19%	15%														
NSM27	12%	12%	15%	16%	28%	23%	18%	18%	25%	22%	29%	31%	31%	28%	23%	29%	34%														
NCM9	10%	14%	15%	11%	20%	25%	16%	22%	22%	18%	24%	24%	24%	24%	23%	20%	22%														
NCM17	10%	11%	16%	17%	30%	25%	16%	22%	29%	24%	35%	34%	30%	28%	30%	28%	33%														
NCD1	13%	14%	13%	13%	31%	24%	15%	22%	31%	24%	34%	41%	34%	36%	33%	36%	42%														
NCD2	12%	13%	14%	15%	37%	27%	18%	23%	31%	23%	30%	34%	36%	32%	27%	33%	36%														
NCD15	12%	15%	18%	13%	27%	28%	16%	23%	28%	24%	28%	36%	32%	32%	29%	32%	32%														
NCD20	9%	11%	18%	15%	26%	21%	12%	22%	23%	23%	30%	33%	27%	29%	15%	27%	29%														
NCD21	9%	12%	14%	13%	32%	24%	16%	19%	29%	20%	28%	36%	33%	32%	27%	34%	35%														
NCD22	11%	9%	10%	11%	33%	17%	16%	15%	34%	22%	33%	42%	36%	32%	29%	35%	44%														

Table 7 Correlations between function models in Set_2 after lumping by subset

	9 Products 45 models	2 Products 11 models	6 Products 28 models
	NSM(%)	NCM(%)	NCD(%)
NSM	16	18	19
NCM	18	26	27
NCD	19	27	34

Table 8 Correlations between expert and novice models for cutting materials after lumping

	5 Products 5 models	2 Products 11 models
	ECM(%)	NCM(%)
ECM	34	7
NCM	7	26

NSM is 16%, which is lower than the value between NSM and NCM (18%) and the value between NSM and NCD (19%).

Data Analysis: Between Expert and Novice Groups

Next, expert models and novice models were compared. The correlation between expert models and novice models in the same product group was calculated and the lumped values are shown in Table 8. The product subset for cutting materials was selected for this analysis since the expert models in the other two subsets contain only a one model each. Within “cut material”, the ECM subset contains 5 products with 5 models and the NCM subset contains 2 products with 11 models.

The pattern in Table 8 shows that within the same product group, the expert models are in more agreement with each other than novice models. The correlation value of the ECM with itself is 34% and the value of the NCM with itself is 26%. In addition, for the same product group, expert models and novice models strongly disagree with each other (correlation value of 7%). These observations provide some evidence in favor of **Factor 2** stated earlier, that models within designer groups would show more preferential similarity than those across designer groups, although the data set is not large enough for drawing conclusive inferences.

Conclusions and Future Work

This paper presents an approach for studying the similarity between function model graphs using the correlation between their adjacency matrices, which can be used to study the trend of designers' preferences and stylistic choices during function modeling. The presented case study includes three product types and two designer groups of vastly different experience levels.

The results provide preliminary evidence that expert designers of similar backgrounds tend to show similar modeling preferences and that this preference are more strongly aligned within a given product type than between product types. This observation suggests that modeling preference could be an effect of both background (expertise, training, etc.) and the product type modeled. Conversely, novice designers do not show any strong pattern at all. Across all products, their modeling preferences are strongly dissimilar with that of the experts, and often even dissimilar among the designers within a team. For a given product type, the modeling preferences are more similar within the expert group than within the novice group.

Collectively, these observations lead to the following hypothesis: "with experience, modelers develop preferential biases and these biases could be functions of their personality, the product domain they are familiar with, or the training they received". To test this hypothesis, several studies into the experience, grade, background knowledge, and product familiarity of novice and expert designers will be conducted in the future. If this hypothesis was supported by data, one could probably evaluate the modeling expertise of a novice engineering by looking for the similarity score of her model's adjacency matrices with expert matrices, and this method could provide a basis for formalizing the grading of function models in engineering design courses.

Acknowledgements This research is supported by National Science Foundation grant no. CMMI-1532894. The authors are thankful to their respective departments and colleagues for their assistance in this research.

References

1. Fenves SJ, Foufou S, Bock C, Sudarsan R, Sriram RD CPM: A core product model for PLM support. In: *Frontiers in design simulation and research*, Alanta, Georgia, USA
2. Khalil H, Kokotovic P (1978) Control strategies for decision makers using different models of the same system. *IEEE Trans Autom Control*, AC- 23(2):289–298
3. Carbon C (2014) Understanding human perception by human-made illusions. *Front Hum Neurosci* 8(7), Article 506
4. Boring EG (1930) A new ambiguous figure. *Am J Psychol* 42:444–445
5. Turner C, Sen C (2018) Modelling differences in individual perception of abstracted system models. In: *Proceedings of the international conference on design creativity*, Bath, United Kingdom, January 31–February 2, 2018
6. Pahl G, Beitz W, Feldhusen J, Grote KH (2007) *Engineering design: a systematic approach*. Springer, London Limited, London, UK
7. Otto KN, Wood KL (2001) *Product design: techniques in reverse engineering and new product development*. Prentice Hall, Upper Saddle River, NJ
8. Stone R, Wood K (2000) Development of a functional basis for design. *J Mech Design* 122(4):359–370
9. Bohm MR, Stone RB (2004) Product design support: exploring a design repository system. *ASME International Mechanical Engineering Congress*, Anaheim, CA, USA
10. Bohm M, Stone RB, Simpson T (2008) Introduction of a data schema: to support a design repository. *Comput Aided Des Appl* 40(7):801–811
11. Bohm MR, Stone RB, Szykman S (2005) Enhancing virtual product representations for advanced design repository systems. *J Comput Inf Sci Eng* 5(4):360–372
12. Ullman DG (1992) *The mechanical design process*. McGraw-Hill, New York
13. Freeman P, Newell A (1971) A model for functional reasoning in design. In: *Proceedings of the international joint conference for artificial intelligence*, London, UK
14. Gero JS (1990) Design prototypes: a knowledge representation schema for design. *AI Mag* 11(4):26–36
15. Goel AK, Bhatta SR (2004) Use of design patterns in analogy-based design. *Adv Eng Inform* 18(2):85–94
16. Umeda Y, Takeda, H., Tomiyama, T., and Yoshikawa, H. (1990) *Function, behaviour, and structure. Applications of art. intelligence V. Vol 1*. Springer Verlag, Boston, MA. 177–193
17. Gero JS, Kannengiesser U (2002) The situated function-behaviour-structure framework. In: Gero JS (ed) *Artificial intelligence in design*. Kluwer Academic Publishers, Norwell, MA, USA, pp 89–104
18. Sembugamoorthy V, Chandrasekaran B (1986) Functional representation of devices and compilation of diagnostic problem-solving systems. In: Kolodner J, Riesbeck CK (eds) *Experience, memory, and reasoning*. Lawrence Erlbaum Associates, Hillsdale, NJ, pp 47–53
19. Bhatta SR, Goel AK A functional theory of design patterns. In: *Proceedings of the 15th international joint conference on artificial intelligence*, vol 1, Nagoya, Japan
20. Umeda Y, Ishii M, Yoshioka M, Shimomura Y, Tomiyama T (1996) Supporting conceptual design based on the function-behaviour-state modeler. *Artif Intell Eng Des Anal Manuf* 10(4):275–288
21. Umeda Y, Tomiyama T (1995) FBS modeling: modeling scheme of function for conceptual design. In: *Proceedings of the 9th. international workshop on qualitative reasoning*, Amsterdam, Netherlands
22. Bohm MR, Stone RB (2004) Representing functionality to support reuse: conceptual and supporting functions. In: *Proceedings of the ASME 2004 design engineering technical conferences and computers and information in engineering conference*, Salt Lake City, UT, USA
23. Chandrasekaran B, Josephson JR (2000) Function in device representation. *Engineering with Computers* 16(3–4):162–177

24. Umeda Y, Takeda H, Tomiyama T, Yoshikawa H (1990) Function behaviour and structure. In: Gero JS (ed) *Applications of artificial intelligence V*, Vol 1, vol Design. Springer Verlag, Boston, MA, pp 177–193
25. Garbacz P (2005–2006) Towards a standard taxonomy of artifact functions. *Appl Ontol* 1: 221–236
26. Kitamura Y, Mizoguchi R (2003) Ontology-based description of functional design knowledge and its use in a functional way server. *Expert Syst Appl* 24(2003):153–166
27. Vescovi M, Iwasaki Y, Fikes R, Chandrasekaran B (1993) CFRL: A language for specifying the causal functionality of engineered devices. In: *Proceedings of the eleventh national conference on artificial intelligence*, American Association for Artificial Intelligence, Washington, D.C
28. Sasajima M, Kitamura Y, Ikeda M, Mizoguchi R (1995) FBRL: A function and behaviour representation language. In: *Proceedings of the international joint conferences on artificial intelligence montreal, Quebec, Canada*
29. Bracewell RH, Sharpe JEE (1996) Functional descriptions used in computer support for qualitative scheme generation—“Schemebuilder”. *Artif Intell Eng Des Anal Manuf* 10(4):333–345
30. Sen C, Summers JD, Mocko GM (2011) Exploring potentials for conservational reasoning using topologic rules of function structure graphs. In: *Proceedings of the international conference on engineering design*, Technical University of Denmark
31. Bobrow DG (1984) Qualitative reasoning about physical systems: an introduction. *Artif Intell* 24(1–3):1–5
32. Gero JS, Kannengiesser U (2000) Towards a situated function-behaviour-structure framework as the basis for a theory of designing. In: *Proceedings of the workshop on development and application of design theories in AI in design research*, sixth international conference on artificial intelligence in design, worcester, MA, USA
33. Dorst K, Vermaas PE (2005) John Gero’s function-behaviour-structure model of designing: a critical analysis. *Res Eng Design* 16:17–26
34. Umeda Y, Kondoh S, Shimomura Y, Tomiyama T (2005) Development of design methodology for upgradable products based on function–behaviour–state modeling. *Artif Intell Eng Des Anal Manuf* 19:161–182
35. Erden MS, Komoto H, VanBeeK TJ, D’Amelio V, Echavarria E, Tomiyama T (2008) A review of function modeling: approaches and applications. *Artif Intell Eng Des Anal Manuf* 22(2):147–169
36. Goel A, Bhatta S, Stroulia E (1997) Kritik: An early case-based design system. In: Maher ML, Pu P (eds) *Issues and applications of case-based reasoning in design*. Erlbaum, Mahwah, NJ, pp 87–132
37. Bhatta, S., Goel, A., and Prabhakar, S. (1994) Innovation in analogical design: A model-based approach. In: *Artificial intelligence in design*, Dordrecht, The Netherlands
38. Iwasaki Y, Fikes R, Vescovi M, Chandrasekaran B How things are intended to work: capturing functional knowledge in device design. In: *Proceedings of the international joint conference on AI*, Menlo Park, CA
39. Umeda Y, Tomiyama T (1997) Functional reasoning in design. *IEEE Intell Syst* 12(2):42–48
40. Sen C, Summers JD (2013) Identifying requirements for physics-based reasoning on function structure graphs. *Artif Intell Des Anal Manuf AIEDAM* 27(3):219–299
41. Summers JD, Eckert C, Goel A (2013) Function in engineering: benchmarking representations and models. In: *Proceedings of the international conference on engineering design*, ICED13, Seoul, Korea
42. Albers A, Matthiesen S, Thau S, Alink T (2008) Support of design engineering activity through C&CM - temporal decomposition of design problems. In: *Proceedings of the TMCE 2008 symposium*, Izmir, Turkey
43. Albers A, Burkardt N, Ohmer M (2004) Principles for design on the abstract level of the contact & channel model. In: *Proceedings of the TMCE 2004*, Lausanne, Switzerland
44. Sridharan P, Campbell MI (2005) A study on the grammatical construction of function structures. *Artif Intell Eng Des Anal Manuf* 19(3):139–160

45. Kurtoglu T, Campbell MI, Gonzales J, Bryant CR, Stone RB (2005) Capturing empirically derived design knowledge for creating conceptual design configurations. In: Proceedings of the ASME 2005 international design engineering and technical conferences and computers and information in engineering conference, Long Beach, CA, USA
46. Kurtoglu T (2007) A computational approach to innovative conceptual design. University of Texas, Austin
47. Kurtoglu T, Campbell MI (2009) Automated synthesis of electromechanical design configurations from empirical analysis of function to form mapping. *J Eng Des.* 20(1), <https://doi.org/10.1080/09544820701546165>
48. Vucovich J, Bhardwaj N, Ho HH, Ramakrishna M, Thakur M, Stone R (2006) Concept generation algorithms for repository-based early design. In: ASME 2006 international design engineering technical conferences and computers and information in engineering conference, Philadelphia, PA, USA
49. Bryant CR, Stone RB, McAdams DA, Kurtoglu T, Campbell MI (2005) Concept generation from the functional basis of design. In: Proceedings of the international conference on engineering design, ICED 05, Melbourne
50. Stone RB, Tumer IY, Stock ME (2005) Linking product functionality to historic failures to improve failure analysis in design. *Res Eng Design* 16(2):96–108
51. Stone RB, Tumer IY, Wie MV (2005) The function-failure design method. *J Mech Des.* 127(3): 397 (311 pages)
52. Papakonstantinou N, Sierla S, Tumer IY, Jensen D (2012) Multi-scale simulation on interactions and emergent failure behaviour during complex system design. *ASME J Comput Inf Sci Eng.* In press
53. Sierla S, Tumer IY, Papakonstantinou N, Koskinen K, Jensen D (2012) Early integration of safety to the mechatronic system design process for the functional failure identification and propagation framework. *Mechatronics* 22(2):137–151
54. VanBossuyt DL, Hoyle C, Tumer IY, Dong A (2012) Considering risk attitude using utility theory in risk-based design. *Artif Intell Eng Des Anal Manuf*, 26(4), In Press
55. McAdams DA, Wood K (2002) A quantitative similarity metric for design-by-analogy. *J Mech Des* 124(2):173–182
56. Nagel RL, Vucovich JP, Stone RB, McAdams DA (2007) Signal flow grammar from the functional basis. In: Proceedings of the international conference on engineering design, ICED '07, Prais, France
57. Nagel RL, Stone RB, Hutcheson RS, McAdams DA, Donndelinger JA (2008) Function design framework (FDF): integrated process and function modeling for complex systems. In: Proceedings of the ASME 2008 international design engineering technical conferences & computers and information in engineering conference, Brooklyn, New York, USA
58. Bohm M, Stone RB, Nagel R (2009) Form follows form-is a new paradigm needed? In: Proceedings of the ASME 2009 international mechanical engineering congress and exposition (IMECE2009), Lake Buena Vista, Florida, USA
59. Nagel RL (2011) A design framework for identifying automation opportunities. Doctoral dissertation, Oregon State University, Corvallis, OR
60. Nagel RL, Bohm MR, Stone RB, McAdams DA (2007) A representation of carrier flows for functional design. In: Proceedings of the international conference on engineering design ICED 07, Paris, France
61. Nagel RL, Perry KL, Stone RB, McAdams DA, FunctionCAD: A functional modeling application based on the function design framework. In: Proceedings of the international design engineering technical conferences and computers and information in engineering conference, San Diego, CA
62. Bohm MR, Stone RB, Simpson TW, Steva ED (2006) Introduction of a data schema: the inner workings of a design repository. In: Proceedings of the ASME 2006 international design engineering technical conferences and computers and information in engineering conference, Philadelphia, PA, USA

63. Bohm MR, Stone RB, Szykman S (2003) Enhancing virtual product representations for advanced design repository systems. In: Proceedings of the ASME 2003 design engineering technical conferences and computers and information in engineering conference, Chicago, IL, USA
64. Szykman S, Racz JW, Sriram RD (1999) The representation of function in computer-based design. In: Proceedings of the 1999 ASME design engineering technical conferences, Las Vegas, NV, USA

Part IX
Design and Visualization

Information Visualisation for Project Management: Case Study of Bath Formula Student Project



Nataliya Mogles, Lia Emanuel, Chris Snider, James Gopsill,
Sian Joel-Edgar, Kevin Robinson, Ben Hicks, David Jones
and Linda Newnes

This paper contributes to a better understanding and design of dashboards for monitoring of engineering projects based on the projects' digital footprint and user-centered design approach. The paper presents an explicit insight-based framework for the evaluation of dashboard visualisations and compares the performance of two groups of student engineering project managers against the framework: a group with the dashboard visualisations and a group without the dashboard. The results of our exploratory study demonstrate that student project managers, who used the dashboard generated more useful information and exhibited more complex reasoning on the project progress, thus informing knowledge of the provision of information to engineers in support of their project understanding.

Introduction

Organisations across diverse sectors are increasingly turning to information visualisation tools to leverage the growing availability of large data sets for insights to improve their work practices and performance [2, 10, 17, 28, 31]. Within the engineering domain, projects are becoming ever more complex and highly distributed, creating immense digital footprints as part of their project lifecycle

N. Mogles (✉) · C. Snider · B. Hicks · D. Jones
University of Bristol, Bristol, UK
e-mail: mogles1@yahoo.com

J. Gopsill · K. Robinson · L. Newnes
University of Bath, Bath, UK

L. Emanuel
Nomensa Company, Bristol, UK

S. Joel-Edgar
Aston Business School, Aston University, Birmingham, UK

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_35

[3, 11]. Unsurprisingly, research has begun exploring ways to leverage that digital footprint through analytics and information visualisation as a means to monitor engineering activity and progress [11, 33] to support project delivery on time and at cost. Certainly, the use of such analytics and information visualisations can be particularly beneficial for managers in better understanding organisational processes, performance and improve their decision-making [23, 30].

However, researchers and practitioners alike have been calling for the establishment of information visualisation appropriate evaluation methods and guidelines. An often-recurring critique is the lack of rigorous qualitative user-centred design approaches being applied in the design and evaluation of information visualisation [1, 5, 14, 19, 26]. In line with this, user-centred design studies have become increasingly popular in problem-driven visualisation research; whereby researchers work with real users to understand the context of their problem, the tasks and data they work with, implementing and evaluating a visualisation solution in a practical context [25].

The process of evaluating information visualisation tools, both in examining the usability of the visualisations as well as the utility of the tool to support complex user processes or tasks, is an imperative and often difficult step, see e.g., [9], in validating and encouraging wider adoption of those tools. In systematic reviews of the evaluation practices in the visualisation and information visualisation research community over the past 10 years, Lam et al. [16] and Isenberg et al. [14] uncovered a striking trend in evaluation techniques. Both the sets of authors found that the majority of evaluation research has focused on understanding the visualisation systems and underlying algorithms, for instance through user and algorithm performance, accuracy and efficiency metrics. On the other hand, they found there were surprisingly low instances of evaluation approaches which focused on understanding the user's process. These include evaluation methods which aim to uncover how visualisation tools can be integrated and used in the user's work environment, and how user-centric tasks such as reasoning, knowledge discovery, or decision-making are supported by visualisation tools [14, 16].

While the more frequently undertaken evaluation of visualisation and algorithm systems, largely quantitative controlled experiments using predetermined tasks, can help us understand the boundary factors for a visualisation tool's capabilities; successful application and adoption of these tools lies in grounding evaluation in the contextual needs and tasks of the end user. [14, 21, 22]. In Isenberg et al.'s review [14], they suggest three pathways to aid in the wider uptake of grounded evaluation approaches: (1) less emphasis on quantitative significance testing, and greater acceptance and application of qualitative evaluation methods, (2) more detailed reporting about the methods used to gather insights from expert users, and (3) more rigorous and in-depth evaluation of feedback from users. Encouragingly, there is an increasing body of work highlighting the potential benefits of integrating qualitative enquiry into visualisation research (e.g., [15, 25, 32])-particularly, in insight-based evaluation methods and frameworks.

The aim of this work is to evaluate the effect of a dashboard visualisation tool on engineering project progress understanding and knowledge discovery. To achieve

this, we briefly discuss information visualisation evaluation approaches and propose an insight-based evaluation framework for visualisations geared towards the engineering domain. This framework was applied in a case study with an engineering project management team, in which information visualisation tools were evaluated on how they supported user understanding of project activities and events. The results of the study are presented, followed by a discussion of the proposed framework and its implications in how information visualisation is applied to support engineering project activities.

Insight-Based Evaluation

The main tenant of visualisation is to display data in a way that maximises comprehension [8] and enables the viewer to gain insight into the data [4, 27]. Though there are a few definitions of insight extant in the information visualisation (InfoVis) literature, the concept of insight is still not well understood [34]. North [20] argues that if the purpose of visualisation is to provide insight then evaluations of visualisation should aim to understand the degree of insight achieved by the end user. Towards this end, North [20, p. 20] broadly characterised insights as:

- **Complex:** Insight is complex, involving all or large amounts of the given data in a synergistic way, not simply individual data values.
- **Deep:** Insight builds over time, accumulating and building on itself to create depth. Insight often generates further questions and, hence, further insight.
- **Qualitative:** Insight is not exact, can be uncertain and subjective, and can have multiple levels of resolution.
- **Unexpected:** Insights is often unpredictable, serendipitous and creative
- **Relevant:** Insight is deeply embedded in the data domain, connecting the data to existing domain knowledge and giving it relevant meaning.

Indeed, a more insight-based evaluation of visualisations has recently received a lot of interest, with researchers and practitioners championing qualitative evaluation methods [1], embedding evaluation in relevant domain impact [25], and exploring how to leverage the unexpected [32]. Unsurprisingly, this increase in insight-based evaluation has led to the proposal of several frameworks to help categorise and quantify user insights in a meaningful way. For instance, [6] proposed a generalised non-domain specific ‘fact taxonomy’ drawn from literature review, user studies and expert reviews from a wide range of domains. The authors generated 12 different categories for characterising user insights, including trend, clustering, distribution, outliers, ranking, and associations. Similarly, [7] identified 8 categories of the types of insights generated by participants presenting visualisations of personal data from self-tracking technology. Similar to [6], insight categories were largely based on how data points were discussed. For instance, categories included data summary, distribution, trends, comparison, and outliers [7]. While these frameworks certainly contribute to a better understanding of different ways visualisation data is leveraged to reach

insights, we argue that these frameworks tell us very little about how visualisation generated insights may support domain specific, in our case engineering, tasks and knowledge discovery as outlined by North [20].

Saraiya et al. [24] developed a more domain specific compared to North [20] insight-based coding framework, which utilised an open-ended user testing approach. Specifically, the authors asked users familiar with biological data analysis to explore bioinformatics visualisation tools. Rather than asking users to follow a strict protocol or to complete predetermined tasks with the tool, the authors encouraged the users to explore and analyse the data represented in the tool as they normally would in their roles. Users were asked to think aloud throughout this process and verbalise their thoughts and findings of the data set. By inductively categorising user comments from this open-ended analysis process, the authors developed eight broad insight dimensions that focused more heavily on user's work processes, behaviour and domain value. We argue this method, and resulting insight framework, is more in line with a context-driven evaluation approach.

Therefore, we adopted Saraiya's methods to formalise an insight-based evaluation framework for the engineering domain, particularly, to evaluate the project analytics visualisation tools we developed and their ability to support insight generation for engineer project managers. We introduce the engineering user group and project management visualisation tools evaluated within the case study below, and describe the development and application of our engineering domain insight evaluation framework to better understand how project analytics visualisations can support project management.

Case Study: Engineering Project Management—Bath Formula Student

This case study is focused on understanding and utilising the digital footprint generated by engineering project work, such as digital communications (e.g., email, social media), records (e.g., reports, documents, presentations) and design representations (e.g., Computer-Aided Design (CAD) models) in the context of the Language of Collaborative Manufacturing (LOCM) project.¹ Using this low-level output data to provide student project managers from the University of Bath with dashboards supporting high-level insights into project changes and progress, we evaluated the effect of an information visualisation tool on project progress understanding and knowledge discovery.

Participants

This evaluation work was performed with a project team of the University of Bath engaged in Formula Student (FS) competition [13]—Team Bath Racing. The FS is a yearly international competition where project teams of approximately 25 multidis-

¹<http://locm.blogs.ilrt.org/>.

Table 1 Number of files on the shared drive and their updates per project activity

Activity	Number of files	Number of updates
Simulation	23,694	103,481
Software	4,755	15,762
Spreadsheet	2,282	11,938
Testing	868	5,226
Video	452	1,804
Website	290	1,102
Design	12,590	53,665
Documentation	7,220	23,993
Images	21,539	90,895
Management	36	231

cipline engineering university students design, manufacture, and race a single-seat racing car. This user group was selected as it encompassed an entire engineering project lifecycle, and its related digital outputs, in a rapid time-frame (22 weeks) and re-occurs annually. Six project managers, all male, from one team took part in the study. Each manager was responsible for managing different sub-teams across the project. Participants received £10 for each session they took part in.

Project Management Dashboard Tool

An FS team will generate approximately 8–9 terabytes of project-related data over the course of their project lifecycle. In developing a dashboard tool, our aim was to develop automated analytic and information visualisation approaches using this low-level output data to provide project managers with dashboards supporting high-level insights into project changes and progress. Ultimately, to support informed decision-making towards optimal performance and productivity. In the development of the project management dashboard analytics, for explorative purposes, we used project data generated across three different FS project teams over a 3-year period. Exploration of this dataset was undertaken by engineering researchers to understand what type of data was created and how it was organised.

The monitoring of the digital footprint was performed using a custom software tool that monitored the activity of the Formula Teams shared network drive (<https://www.npmjs.com/package/fal>). Over the course of the project, 129,377 files were created and 870,134 updates made. This includes the creation, deletion and modification of the files on the shared drive. The shared drive contains files pertaining to all activities of the project. The files were further classified by engineering activity defined by file type, with activities associated with engineering activities where software use is specific to an activity type (e.g. CAD files—Design), or to a general form of activity where software use may be for multiple purposes (e.g. documents, presentation slides—Documentation). Table 1 shows the volume and level of activity in each area.

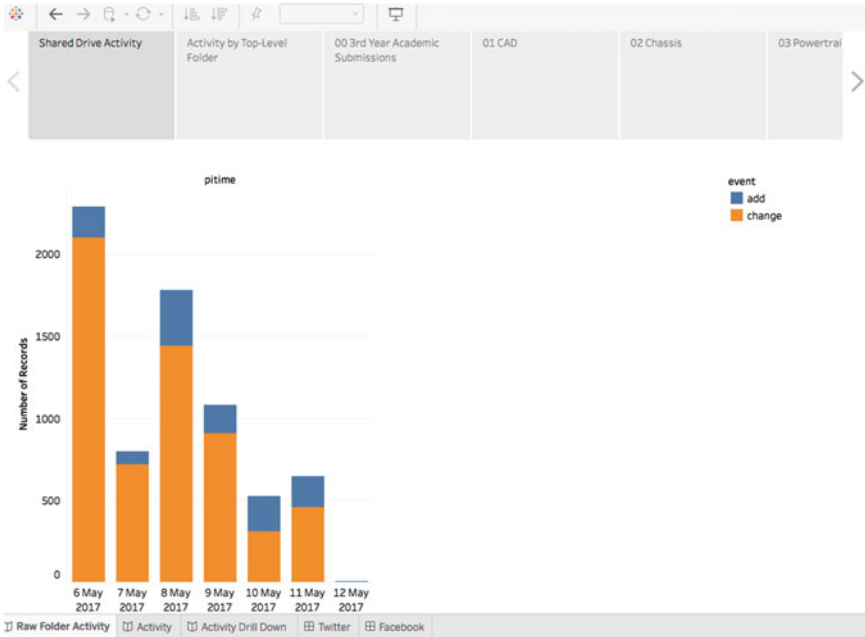


Fig. 1 Total files added and changed on the shared x-drive by day

In addition to the shared drive, the Social Media communications of the team were also recorded. This was achieved by recording the public tweets and Facebook posts of the team and placing them in the context of all other FS national teams. A total of 1341 public tweets were captured for all teams during this project.

From this initial exploration, nine broad data analytic metrics emerged (discussed further in [11]), which could be leveraged to support the monitoring of project activities. Through a series of iterative user-centred design interviews, focus groups and workshops with stakeholders and FS user-groups, a suite of initial interactive information visualisations were designed and developed using free Tableau software [29] for data visualisation. Dashboard design requirements and principles were formulated based on users needs and available data. A dashboard consisted of five data tabs with one data visualisation tab presented at a time via a web-based Tableau application. The tabs were presented in the following order: Raw Folder Activity, Activity, Activity Drill Down, Twitter, Facebook (see Figs. 1, 2, 3 and 4). The users were able to navigate through the tabs at the bottom of the display in order to access different data analytics and visuals developed from the data on the project digital footprint. The dashboard was presented on a laptop computer with the 27” touchscreen monitor during interviews with half of the project managers (Dashboard experimental group).

The following information on the project digital footprint was presented to the users: **Raw Folder Activity tab**—total files added and changed on the shared X-

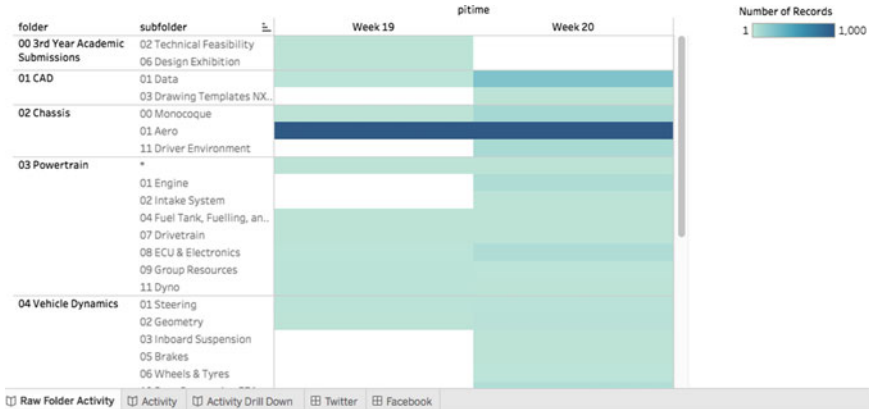


Fig. 2 Number of files added in the week—top and sublevel folders

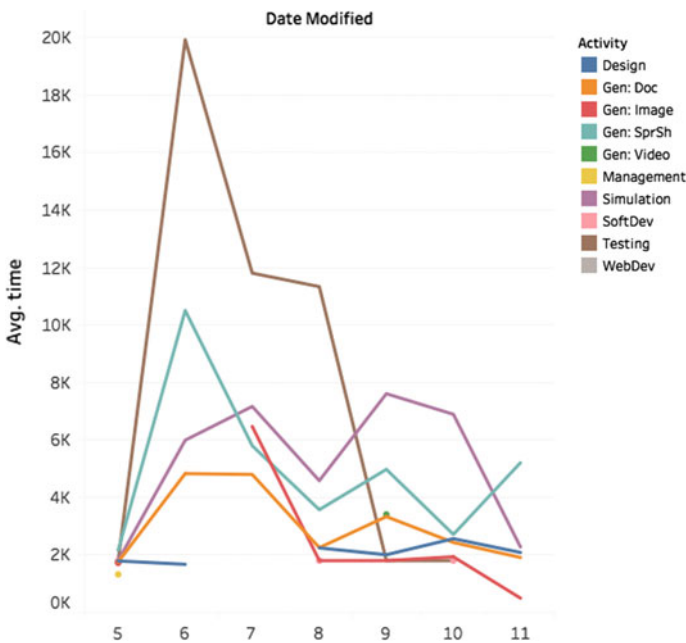


Fig. 3 Type of activity by day: design, reports, images, video, management, simulation, software development, testing, web development

drive by day (bar chart—see Fig. 1), number of files added in the previous week—top and sub-level folders (heat map—see Fig. 2), activity by top and sub-folders and the number of files added or changed for each top and sub-level folder, time spent on activity types and number of files worked on—top and sub-folder (heat map similar to the one in Fig. 2), **Activity tab**—type of activity analytics by day derived from

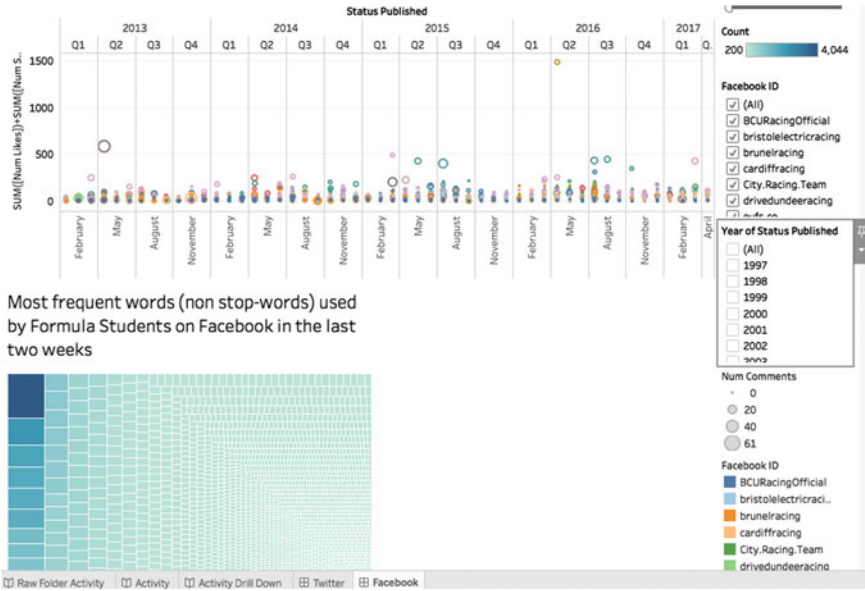


Fig. 4 Facebook analytics dashboard: impact (likes and shares), engagement (comments) across 43 FS teams’ posts, trending words/topics being used by FS in the last month

the files extensions, including time spent on activity and number of files worked on (line graph, see Fig. 3), **Activity Drill Down tab**—with information on time spent on activity and number of files worked on within sub-folders (line graph similar to the one in Fig. 3), **Facebook** and **Twitter tabs**—information on social media which consisted of Facebook and Twitter dashboards. On the Facebook tab: impact (likes and shares), engagement (comments) across 43 FS teams’ posts, trending words/topics being used by Formula Students in the last month (treemap). On the twitter tab: the top Formula Student Accounts currently being followed (treemap), network of top users interacting with @TeamBathRacing handle (EgoNet), reach (retweets) and size of reaction (favourites) across 43 FS teams’ tweets. An example of Facebook activity visualisations is represented in Fig. 4. A visualisation similar to Fig. 4 was developed for Twitter-related analytics.

Each tab contained further up to 16 lower level sub-tabs. All main tabs except Activity Drill Down were used by the participants, though only three types of data visualisations across all tabs were examined: activity type heat map, number of records by the shared X-drive folder structure over time and social media visualisations. The participants were only provided with the visualisations of the data from their project.

Methods

The aim of the experiment was to evaluate how the provision of the project management analytics dashboard affected FS project managers’ interpretation of project

activities and events. A dashboard present versus absent mixed design was employed. The dashboard was tested with the help of semi-structured interviews with 6 project managers. Managers took part in four evaluation sessions, one every 2 weeks over an 8-week period. Each session comprised of a semi-structured think-aloud task in which managers were asked to consider and verbally walk through their thought process around the project's progress and performance over the past two weeks. Experimenter prompts contained project's main goal, activities and issues encountered. An example of the interview prompts is: 'What have been the main activities and goals you were working towards this week?'

As this type of project review activity was generally performed as a group within the team, half of the evaluation sessions were group sessions and were conducted with a maximum of four managers in the dashboard present group, and maximum two managers in the No dashboard group. Across the four sessions, the dashboard present group had access to the developed dashboard and was encouraged to use and explore the data to help them reflect on project activities. Prior to the first session, this group was given a brief training session, walking them through what data and visualisations were available to them in the dashboard. Evaluated separately, the No dashboard group did not have access to the dashboard and was asked to simply reflect on and discuss their project activities.

Each session lasted between 20 and 45 min, with both groups' comments audio recorded and the dashboard present group's interaction with the dashboard was video recorded using screen capture software.

Proposed Insight Framework

User comments recorded across the four evaluation sessions were transcribed for coding. An insight, in this case, has been defined as an individual observation about project activity by the participant [24]. An inductive and iterative coding process was used to develop the insight framework. Specifically, two coders used Saraiya et al.'s eight insight dimensions, which focuses on user's work processes, behaviour and domain value, as an initial coding template. As categories of our users' domain-specific processes and tasks emerged, insight dimensions were adapted and added to. This process led to the final insight-based evaluation framework, which included nine insight dimensions (see Table 2 below).

Average Huberman's inter-coder reliability [18] across the first two sessions was 70% for the first four dimensions (*Observation, Comparison, Hypothesis, Judgment*) and 100% for the other, more straightforward dimensions.

Table 2 Insight-based evaluation framework

Dimension	Value	Description
Observation	Numerical	Frequency of insights made by the participants—this dimension is a direct match with an element of the framework of Saraiya et al. [24]
Comparison	Numerical	The insight discussed the similarities/differences between pieces of information (e.g. objects, people, activities, etc.) This insight dimension is adapted from Saraiya ‘category’ characterisation [24], it is also critical for information processing in the context of engineering design [12]
Hypothesis	Numerical	Suggests an in-depth data understanding and inference; it is adapted directly from [24] and regarded as the most critical dimension according to Saraiya et al. [24]. Hypothesis can be: <ul style="list-style-type: none"> ○ <i>Causal</i>: Linking pieces of information to explain causal relations, the ability of the individual to understand information [12] ○ <i>Proposed further enquiry</i>: generates or identifies a new question/hypothesis [24]
Judgement/valence	Numerical	Whether an opinion or judgement was made about the value of the insight made (e.g. evaluation of information, domain value of information [24], subjective interpretation [12])
Information granularity (breadth or depth)	Categorical	Indicates the level of granularity or detail in the statement (directly adapted from [24])
Project context	Categorical	Insights were grouped based on the area of the project that they pertained to. This dimension emerged in the process of interviews coding as domain-specific project activities according to Saraiya’s inductive coding methodology [24]
Project aspect (managerial or technical)	Categorical	Whether the insight was related to managerial or technical activities. This dimension also emerged in the process of interviews coding as domain-specific project aspect according to Saraiya’s inductive coding methodology [24]
Information usage behaviour (confirmatory or exploratory)	Categorical	Whether the dashboard is used to confirm an insight generated by memory; or the dashboard is used in an exploratory way unrelated to a priori ideas [24]. This insight dimension is relevant only to the dashboard group
Information source (self or dashboard generated)	Categorical	Whether the user generated the insight from memory or from interacting with the dashboard. This description is relevant only to the dashboard group. It emerged in the process of interviews coding as domain-specific project activities according to Saraiya’s inductive coding methodology [24]

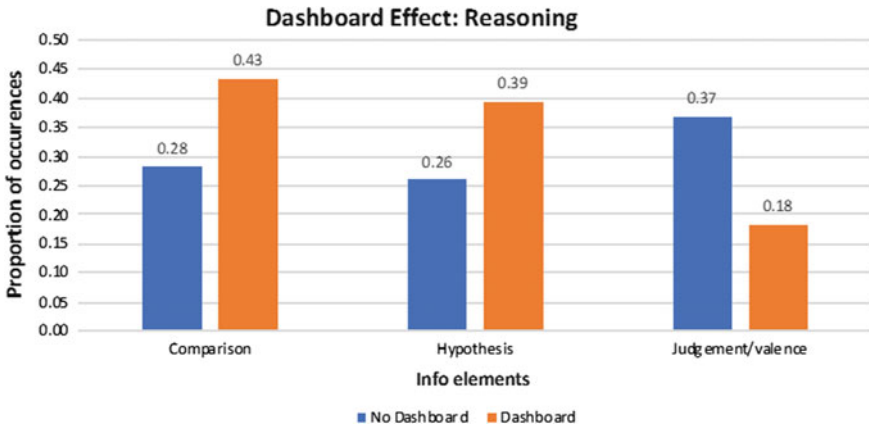


Fig. 5 Proportion of occurrences of cognitive elements: comparison, hypothesis and judgement/valence—for dashboard and No dashboard groups

Case Study Results

In this section comparison results between dashboard and No dashboard groups are presented across the nine insight categories of the proposed framework. Further, a closer look is taken at the dashboard group to investigate how they used dashboard visualisations. Note that the last two dimensions mentioned in the previous section—Information usage and Information source—refer only to the dashboard users as they characterise dashboard interaction behaviour.

Dashboard versus No Dashboard groups

Observation: In the dashboard group 76 meaningful project-related observations with different topics were identified, while in the No dashboard group there were only 46 which is 40% less compared to the dashboard group (see Fig. 5 for the proportions).

Comparison: There are 15% more comparisons identified in the dashboard group compared to the No dashboard (see Fig. 5).

Hypothesis: 13% more hypotheses were generated by the dashboard group (see Fig. 5).

Judgement/valence: 19% less judgmental statements were generated in the dashboard group compared to the No dashboard group (see Fig. 5).

Information granularity: there are 23% more occurrences of statements with specific information and 15% less occurrences with mixed (both specific and general) statements in the dashboard group (see Fig. 6). The difference for general information statements between the groups is not so substantial: there are 8% more general statements in the No dashboard group.

Project context: in both the groups, 8 project-related topics, or activities, were identified: social media, manufacturing, simulation, general trends, academic/Final

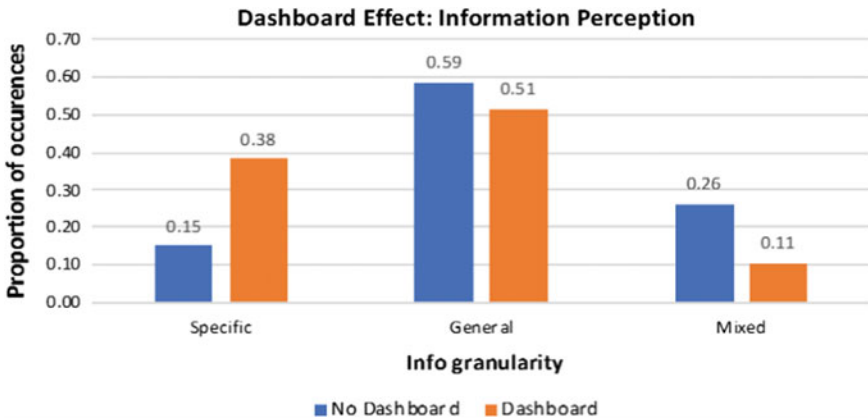


Fig. 6 Proportion of occurrences of three types of information granularity: Specific, General and Mixed information—for dashboard and No dashboard groups

Table 3 Project-related activities proportions mentioned in the statements in dashboard and no dashboard conditions

Project activities	Dashboard	No dashboard	Difference
Social media	0.21	0.11	0.10
Manufacturing	0.29	0.39	-0.10
Simulation	0.07	0	0.07
General trends	0.07	0.04	0.03
Academic/FYP	0.03	0.02	0.01
CAD/Design	0.08	0.09	-0.01
Static events	0.21	0.2	0.01
Admin events	0.05	0.15	-0.10

Year Projects (FYP), Computer-Aided Design (CAD)/Design, Static events such as business, finance, design reports, etc., and administration-related events (see Table 3). The heat map of proportions occurrences of these activities in the interview scripts of both dashboard and No dashboard groups is represented in Table 3.

The majority of statements in the No dashboard condition mention manufacturing: 39% of statements. 20% of statements in the same group contain information about static events, such as business, finance-related events, or design reports. Within the dashboard group, only 29% of statements mention manufacturing activity and the general distribution of project activities is more even compared to the No dashboard group. If we compare proportions of occurrences of different project activities discussions across the two groups, it can be seen in Table 3 that the biggest differences are related to such activities as social media, manufacturing, and admin events (10% difference per each of these three activities).

Project aspect: there is no substantial difference across the two groups with respect of two main project management areas: technical aspects are mentioned 4%

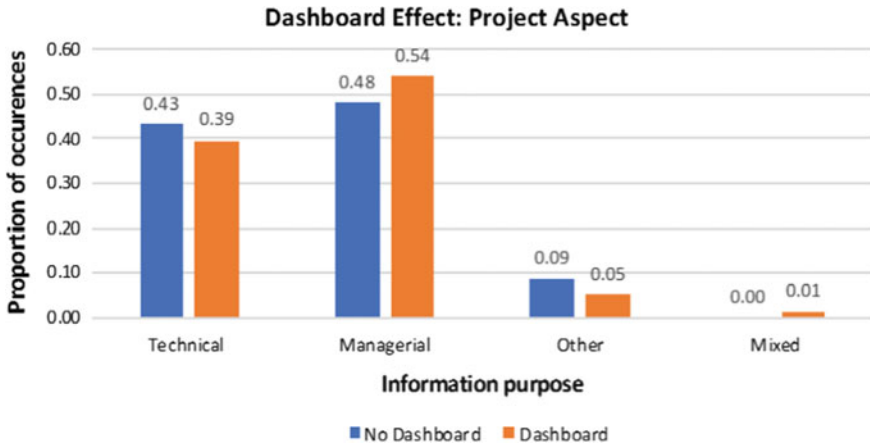


Fig. 7 Proportion of occurrences of three types of information granularity: Specific, General and Mixed information—for dashboard and No dashboard groups

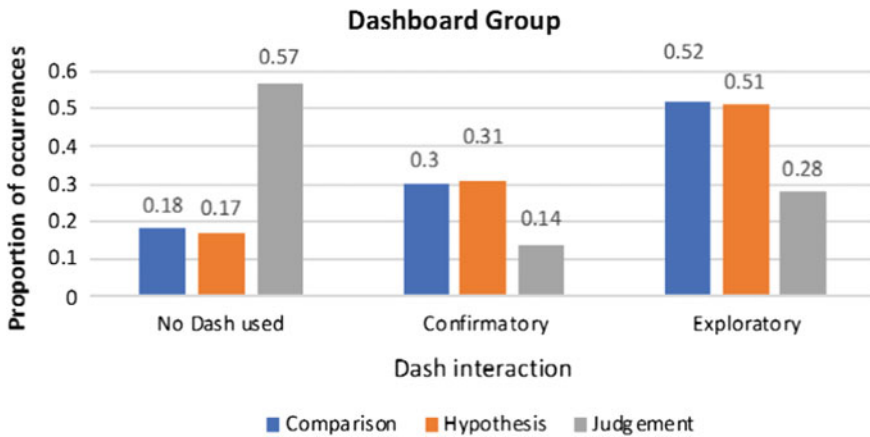


Fig. 8 Proportion of occurrences of cognitive elements: comparison, hypothesis and judgement/evaluation—across three types of dashboard interaction behaviour (no dash, using to confirm information, using to explore) within the dashboard group

less times in the dashboard group and managerial aspects are mentioned 6% more in the dashboard group (see Fig. 7).

Dashboard group

Information usage behaviour and **information source** dimensions of the evaluation framework refer only to the dashboard group since they describe how users interact with the dashboard. Dashboard exploratory behaviour contained more comparisons and hypothesis and statements generated without using the dashboard contained more subjective judgements and evaluations (see Fig. 8).

Table 4 Heat map of project-related activities proportions across three dashboard analytics used in the dashboard condition

Project Activities	Post impact	Type of activity	# Records by x-drive folder
Social media	1.00	0.00	0.00
Manufacture/Build	0.00	0.08	0.38
Static events	0.00	0.33	0.13
General trends	0.00	0.25	0.13
Simulation	0.00	0.25	0.06
CAD/Design	0.00	0.08	0.19
Admin - events	0.00	0.00	0.06
Academic/FYP	0.00	0.00	0.06

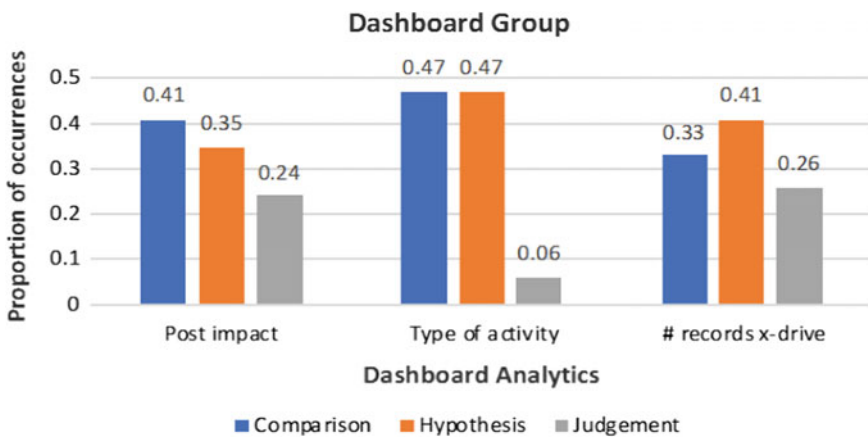


Fig. 9 Proportion of occurrences of cognitive elements: comparison, hypothesis and judgement/evaluation—for dashboard group across three types of analytics used by the participants

Participants in the dashboard condition used only three types of analytics out of four given to them by the experimenters: types of project activity derived from X-drive files extensions, number of records by each X-drive folder and social media impact posts. The heat map in Table 4 demonstrates the proportion of frequencies of project activities mentioned by the participants across these three types of visual analytics. As it can be seen in Table 4, posts impact visualisation was used only in discussions about social media activity, project activity analytics was mainly used to discuss static events (33%), general trends (25%) and simulation (25%) activities, number of records in folders on the shared X-drive analytics was used to discuss manufacturing and build activity (38%) and CAD design (19%).

The least number of judgemental statements were generated while using the visualisation on project activities (see Fig. 9) compared to the other two visualisations.

Table 5 Comparison results across the seven out of nine insight dimensions for dashboard and No dashboard conditions

Insight dimension	Comparison results
Observation	More in the Dashboard group
Comparison	More in the Dashboard group
Hypothesis	More in the Dashboard group
Judgement/valence	More in the No Dashboard group
Information granularity (breadth or depth)	More <i>specific</i> information in the Dashboard group, no difference for <i>general</i> information
Project context	The topics are more evenly distributed across all project areas in the Dashboard group, focused on one area (Manufacturing) in the No Dashboard group
Project aspect (managerial or technical)	No difference

Discussion and Conclusions

One of the findings of the current study is that only three types of dashboard analytics out of four available were used. It can be explained by the fact that the users selected those analytics, which matched the questions of the interviewer. With these three dashboard analytics types for project management based on project digital footprint, there are some implications of a positive dashboard effect on participants’ reasoning about the status of the project. The comparison results between dashboard and No dashboard conditions are summarised in Table 5 below:

The main positive effects of the dashboard tool can be summarised as follows:

- Dashboard visualisations possibly broadened participants’ attention and attracted it to different project activities and aspects. This conclusion is based on the number of observations and topics distribution in dashboard and No dashboard groups (Fig. 5, Table 3);
- Dashboard changed participants’ reasoning and facilitated higher value reasoning elements, such as comparisons and hypothesis generation (Fig. 5);
- Dashboard provided more specific information and helped to focus on lower granularity of information without losing general information of higher granularity (Fig. 6).

Based on the above-described findings of this exploratory study, we can suggest that digital footprint analytics has a good potential and can be a useful measure which can assist project managers and participants in project status analysis. The next steps for the future work can be the exploration of the effect a project digital footprint analytics dashboard on decision-making and actual project outcomes. Cognitive benefits of using a dashboard in this study do not directly imply better project outcomes and more research is needed to examine this connection. Further, new project digital footprint analytics based on people and team aspects can be developed and tested.

There are several limitations of the current study which should be mentioned. First, this dashboard evaluation was conducted with a relatively novice and small team of engineers. Further work is needed to examine how the beneficial insights into project activity observed here may scale up to larger projects and organisations. Second, interview questions and prompts were mainly focused on project activities which could define the usage of specific dashboard analytics. Third, the present study might not represent naturalistic usage of project-related dashboard analytics, but rather an off usage of dashboards. The study aimed to simulate relatively naturalistic review of project progress and activities for this user group (e.g. held in work environment, in groups rather than individually, applying an open-ended task methodology). However, based on user feedback and the real-time project statistics provided by the project status monitoring dashboard, this particular visualisation tool may have more impact on insight generation if interacted with more consistently over time. While this was not possible in this phase of testing due to the stability of the prototype dashboard, further evaluation of this dashboard tool will entail field trial testing in everyday usage of the tool.

Acknowledgements This work was done under the auspices of the Language of Collaborative Manufacturing (LOCM) project funded by the Engineering and Physical Sciences Research Council (EPSRC), UK, grant reference EP/K014196/2. The study was conducted in accordance with the local departmental codes of ethics (Department of Computer Science, University of Bath) and the University of Bath's concordat for research integrity. All data, including the consent forms, was stored and processed in line with the EU General Data Protection Regulation (GDPR) guidelines of anonymity.

References

1. Arias-Hernandez R, Fisher B (2013) A qualitative methodology for design of visual analytic tools for emergency operation centers. In: proceedings of the international conference on system sciences, 126–135
2. Bačić D, Fadlalla A (2016) Business information visualization intellectual contributions: an integrative framework of visualization capabilities and dimensions of visual intelligence. *Decis Support Syst* 89:77–86
3. Briggs D (2012) Data distribution on the 787, bringing model based definition to the world. In: Collaboration and interoperability congress, Denver, 21–23 May 2012
4. Card SK, Mackinlay JD, Shneiderman B (1999) Readings in information visualization: using vision to think. Morgan Kaufmann, San Diego, USA
5. Carpendale S (2008) Evaluating information visualizations. In: *Information visualization: human-centered issues and perspectives*, 19–45
6. Chen Y, Yang J, Ribarsky W (2009) Toward effective insight management in visual analytics systems. In: *IEEE pacific visualization symposium 2009*, 49–56
7. Choe EK, Lee B, Schraefel MC (2015) Characterizing visualization insights from quantified selfers' personal data presentations. *IEEE Comput Graphics Appl* 35(4):28–37
8. Daintith J, Wright E (2008) *A dictionary of computing*, 6 edn. Oxford university press
9. Ellis G, Dix A (2006) An explorative analysis of user evaluation studies in information visualisation. In: *Proceedings of the BELIV'2006*, ACM press, 1–7
10. Gaardboe R, SVARRE T, Kanstrup AM (2015) Characteristics of business intelligence and big data in e-government: preliminary findings. In: *Innovation and the public sector*, p. 109

11. Hicks B, McAlpine H, Gopsill J, Snider C (2016) The digital footprint of engineering design projects: sensors for project health monitoring. In: International conference on design computing and cognition (DCC 2016)
12. Hicks BJ, Culley SJ, Allen RD, Mullineux G (2002) A framework for the requirements of capturing, storing and reusing information and knowledge in engineering design. *Int J Inf Manage* 22:263–280
13. IMechE (2017) Formula Student: <http://events.imeche.org/formula-student> Accessed December 14, 2017
14. Isenberg T, Isenberg P, Chen J, Sedlmair M, Moller T (2013) A systematic review on the practice of evaluating visualization. *IEEE Trans Visual Comput Graphics* 19(12):2818–2827
15. Jackson B, Coffey D, Thorson L, Schroeder D, Ellingson AM, Nuckley DJ Keefe, DF (2012). Toward mixed method evaluations of scientific visualisations and design process an evaluation tool. In Proceedings of the BELIV'2012, ACM press, 4
16. Lam H, Bertini E, Isenberg P, Plaisant C, Carpendale S (2012) Empirical studies in information visualization: seven scenarios. *IEEE Trans Visual Comput Graphics* 18(9):1520–1536
17. Lurie NH, Mason CH (2007) Visual representation: implications for decision making. *J Mark* 71(1):160–177
18. Miles MB, Huberman AM (1994) *Qualitative data analysis: an expanded sourcebook*. Sage
19. Munzner T (2009) A nested model for visualization design and validation. *IEEE Trans Visual Comput Graphics* 15(6):921–928
20. North C (2006). Toward measuring visualization insight. *IEEE Comput Graph Appl: Vis Viewpoints*, May/June 2006, 20–23
21. Plaisant C (2004) The challenge of information visualization evaluation. In: *IEEE Proceedings of the advanced visual interfaces 04*, 109–116
22. Plaisant C, Fekete JD, Grinstein G (2008) Promoting insight-based evaluation of visualizations: from contest to benchmark repository. *IEEE Trans Visual Comput Graphics* 14(1):120–134
23. Raymond L, Bergeron F (2008) Project management information systems: an empirical study of their impact on project managers and project success. *Int J Project Manage* 26:213–220
24. Saraiya P, North C, Duca K (2005) An insight-based methodology for evaluation bioinformatics visualizations. *IEEE Trans Visual Comput Graphics* 11(4):1–14
25. Sedlmair M, Meyer M, Munzner T (2012) Design study methodology: reflections from the trenches and the stacks. *IEEE Trans Visual Comput Graphics* 18(12):2431–2440
26. Shneiderman B, Plaisant C (2006) Strategies for evaluating information visualization tools: multi-dimensional in-depth long-term case studies. In *Proceedings of the BELIV'2006*, ACM press, 1–7
27. Spence R (2001) *Information visualization*. Addison-Wesley
28. Trieu VH (2017) Getting value from business intelligence systems: a review and research agenda. *Decis Support Syst* 93:111–124
29. Tableau software (2013) Technical specifications—Tableau desktop <http://www.tableausoftware.com/products/desktop/techspecs> Accessed December 6, 2017
30. Watson HJ, Wixom BH (2007) The current state of business intelligence. *IEEE IT Syst Perspect*, 96–99
31. Wixom BH, Watson HJ, Reynolds AM, Hoffer JA (2008) Continental airlines continues to soar with business intelligence. *Inform Syst Manage* 25(2):102–112
32. Woźniak P, Valton R, Fjeld M (2015) Volvo single view of vehicle: building big data service from scratch in the automotive industry. In *Proceedings of the CHI 2015*, ACM press, 671–678
33. Wu I, Hsieh S (2012) A framework for facilitating multi-dimensional information integration, management and visualization in engineering projects. *Autom Constr* 23:71–86
34. Yi JS, Kang YA, Stasko JT, Jacko JA (2008) Understanding and characterizing insights: how do people gain insights using information visualization?. In: *Proceedings of the 2008 workshop on BEyond time and errors: novel evaluation methods for Information Visualization ACM*, April 2008, (p. 4)

A Visualization Tool to Investigate the Interplay of External and Internal Processes



Mia A. Tedjosaputro and Yi-Teng Shih

This paper explores the potential of a newly devised visualization tool to facilitate empirical studies related to external and internal design processes. It also highlights the importance of viewing the designing process through the lens of embodied cognition, how designer's mind-body (design) environment system and its interactions should be considered as holistic design strategies. Protocol data of one forty-minute sketching session and twenty-four design sessions of novice designers are utilized to illustrate the possible use of the visualization tool in different grains of analysis. It is observed that with access to externalizations, the cognition boundary between internal and external might be shifted. This is due to design tools (in this paper, the pen) which play a significant role in how designers can recognize affordances exhibited by drawing marks or design environment. It is proposed that *design affordances* and *design effectivities* are fundamental notions to the interplay between internal and external processes.

Introduction

Bilda et al's seminal studies posited that expert designers were able to satisfactorily design without access to sketches [1–3], only using internal representations. An inclination towards assumptions that sole mental processes are as germane as processes which are aided with externalizations was revealed. Sketches and drawings are the “percept” half of a hybrid percept mental image which amplifies the mind's capacity to make descriptive to depictive translations [4], and this interplaying nature of external and internal processes has not been exhaustively studied. This paper aims to present an online visualization tool to facilitate an empirical investigation of the

M. A. Tedjosaputro (✉) · Y.-T. Shih
University of Nottingham, Ningbo, China
e-mail: mia@miatedjosaputro.com

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_36

Table 1 List of abbreviations

Acronym	Definition
SK	Sketching session, a pen-and-paper on special replayable dotted paper session
MI	Mental imagery session, which consists of BF (blindfolded) and EXT (externalization) sessions
D	Design session, the control group which allows designers to use provided design tools
P1	Participant number 1
CO	Cognition category, one of three-axis action codes. They will be exhibited in this format "Category(Code)"
BO	Body category, one of three-axis action codes
ENV	Environment category, one of three-axis action codes

interplay, by illustrating possible use in different detail levels of designers' activities (macro, in terms of an entire design session and micro, in terms of a specific number of utterances). To do so, a hypothesis is tested, aided by the newly devised tool. The study is in the prototype state of development and focuses on the ideation phase when novice designers first become aware of design briefs; in particular, related to the *problem* intention, designers' bodily, and mental interactions with the design world and processes which occur simultaneously. The research hypothesis is

When access to external representations (sketches) is unlimited; the boundary between internal and external processes can be shifted. Also, when access to external representations is limited and staged, limitations are overcome by the use of bodily movements. The boundary remains.

Abbreviations used throughout the paper are as follows (Table 1).

Relevant Literature

External and Internal Processes

Previously it was learnt that sketches and mental imagery activities are intertwined. Sketches are perceived as briefly stored and highly processed mental representations and contain results of mental analysis [4]. They leave visible marks and give access to various mental images (figural or conceptual) [5]. In return, they stimulate the generation of new sketches and the iterative process begins. This ongoing dialectic reflects how they feed directly off each other [6].

"A pencil is a bridge between imagining mind, image on paper is an automatic projection of mind. Or the hand that really imagines" [7]. The authors believe that mental imagery is not a process of retrieving features from long-term memory and assembling a mental image only; but the generation of novel solutions requires manip-

ulation and interpretation. This is in accord with the findings of Goldschmidt [8]. This “*imagining mind*” does more than simply projecting, it also alternates patterns, relations, and interpretations. What may be inferred is that external processes such as hand activities give rise to the mind. This dialectic interaction of external and internal representations is often touched upon but has not been thoroughly or empirically studied. This hand–eye–mind proposition sees the pencil as a bridge which meditates between two realities, and focus can constantly change between physical drawings and nonexistent objects in the mental space depicted in drawings [7]. The authors argue that it transcends the designers’ hands, into their bodily actions.

In this paper, internal and external processes are explored through aspects related to externalizations (drawings, design tools, hands and environmental aspects where designing takes place) and mental processes.

Design Intentions

Design intentions are embedded in each design move, although often a designer does not consciously realize the intention at the moment a move is executed. They are derived from classifications of human-based designing actions mapped by Bernal et al. [9]. These intentions have specific aims related to design: situations, problems, patterns, solutions, and domain intentions. This paper only focuses on *problem* intention-related design moves, other intentions were studied but will not be presented here. *Problem* intentions are classified under three sub-intentions [9]. First, *Problem (1)—Framing* intention. Designers often frame the focus of interest by setting boundaries of the design situation, selecting the focus of attention, and imposing coherence in decisions. Second, *Problem (2)—Ill-definition*, when designers build ill-defined problems to preserve the openness of the process. Third, *Problem (3)—Co-evolution*. Designers engage in evolving problems and solutions concurrently to better understand the nature of the problem. Only the latter is explored in this paper.

Embodied Cognition in Design Studies

Embodied cognition differs from the standard cognitive science views in that the first view includes phenomena in which the latter have little interest; the body and world play a crucial role in cognitive processes. It is a constitutive role, rather than merely causal [10]. While standard cognitive science views that the brain receives input from sensory systems, has a motor cortex and its job is to organize action [11], and that the brain serves an intermediating role between inputs from sensory systems and outputs to the motor systems; embodied cognition perceives thinking as an active exploration with the use of “body with things in environment”. Simply put, the mind not only senses the world but also confronts it with its own representation models. The key assumption is that the body functions as a constituent of the mind rather than

a passive perceiver and actor serving the mind [12]. This concept is applicable to designing in that the previously mentioned hand–eye–mind conception is extended to the designing environment, and the feedback loops between mind–body–environment feed each other.

Different accounts and views on embodied cognition were posited by Wilson [13], Wilson and Golonka [14], Clark [15] and Shapiro [10]; to name but a few. Shapiro outlines three general themes of embodied cognition: (1) *Replacement* view, which views an organism's body in interaction replaces the need for representational processes; (2) *Conceptualisation* view, sees cognition as depending on the kind of body an organism possesses; and (3) *Constitution* view sees body or wordplay a constitutive rather than causal role in cognitive processes. Wilson and Golonka posited four questions [14] as a strategy to explore notions of embodied cognition. In this paper, the questions inspired the authors to explain the nature of certain kinds of interplay between internal and external representations. They are: (1) What is the task to be solved? (2) What are resources that the organism has access to for solving the task? (3) How can these resources be assembled to solve the task? and (4) Does the organism, in fact, assemble and use these resources?

To say that cognition is embodied means it arises from bodily interactions with the world [15]. In design research, similar notions related to the outside world of a designer—*situatedness* and *constructive memory*—were observed, namely *External world*, *interpreted world* and *expected world* were introduced and examined: by observing and interpreting the results of their actions, new actions can be executed [16]. Also, embodied cognition principles were used to highlight the relevance of mind–body–environment system in the manufacturing environment [17] and the findings were analyzed with influence from Wilson and Golonka's key questions.

Affordances and *effectivities* are important concepts from the ecological view of embodied cognition. *Affordances* (coined by Gibson) of environment is what the environment offers/provides animals [18]. *Effectivities* are ways for acting that an animal can use to realize specific affordances, coined by Turvey and Shaw [19], and they can be extended or enhanced. Tool use is the common illustration. Before tools are used, they are separate from the user's body. Once used, a tool is treated as a functional extension of the user (ibid). Tools shift the boundary between body and environment. Drawing upon this, *design affordances* in a similar context are what the design environment offers designers. The environment in this case can be visual marks a designer made in their previous design moves or the design setting (the room for instance). *Design effectivities* are designing acts which designers take to realize *design affordances*. However, the mentioned concept of affordances in design in this paper is slightly different to what is commonly understood: as possible uses of an object, "*clues to operations of things*"—a concept introduced by Norman [20]. For instance, a chair affords support and therefore enables sitting, it can also be carried (ibid). The influence of embodied cognition in this paper is limited to designers and their designing environment.

To clarify, the proposition of mind–body–environment in this paper consists of interaction between mind, gesture, and designing environment of the designer which is limited to where the experiment was conducted.

Table 2 List of participants and design environment

EG1 (4 participants)	EG2 (4 participants)	CG (4 participants)
Archi	PDM	Archi and PDM
SK environment: 45 min of SK session		D environment: 45 min of D session (design tools)
MI environment: 35 min of BF session and 10 min of EXT session		

Method of Data Collection

Protocol data was obtained through a contrived setting with two controlled independent variables, the SK (sketching) and MI (mental imagery) design conditions. There was also a control group design condition, D (design) session condition. There were five steps to obtain the protocol data.

First, 12 novice designers participated on a voluntary basis, they were final-year design architecture and product design students in a Sino-British university. Each participant did two design sessions, in total there were twenty-four forty-five-minute sessions. The rationale for studying novice designers rather than expert designers was: (1) previously related research was focused on experts and (2) experts have a bigger “bank of mental images” through previous experiences, hence the way images are manipulated in novice designers is potentially different. The study has been reviewed according to the university’s code of research conduct and research ethics. It was part of first author’s Ph.D. study and was under the supervision of the second author. Consent was obtained from all participants in written form.

Each participant was given two multifunctional design tasks during two different sessions (with one month’s gap in between). A summary of briefs is (1) to design a convertible space for a creative industry company in an open-plan office, less than 100 m² footprint and (2) to design a hybrid system of sitting space and dining setup for adults and toddlers with a maximum footprint of 3 m × 3 m. Participants were asked to think aloud while designing, and the method was rehearsed prior to the session. They were asked to keep the verbalization to level 1 and 2 [21] as in externalizing inner speech rather than involves describing and explaining. In the SK environment, participants were asked to design using a pen-and-paper-based smartpen for 45 min. In the MI environment, participants were engaged in a BF (blindfolded) session for 35 and 10 min of EXT (Externalization) session. The arrangement of SK and BF sessions were adapted from previous studies by Bilda et al. [22]. In the D environment, participants were engaged in 45 min design sessions using offline design tools (watercolors, markers, color pencils, model making equipment, etc.). P1’s session was an SK session with the first design brief (Table 2).

Second, three to four audio/video recordings were obtained. They are: front, top, side, and a replayed pencast from a smartpen to capture design processes. Third, in the transcription stage, an exact reproduction of spoken words was necessary. Fourth, during the segmentation step, verbal data was parsed into smaller chunks of

Action ID	Actions	Description
COGNITION		
C-re	Retrieval	Recall and recognise existing structures from memory
C-as	Association	Connect two images, thoughts, ideas or psychological phenomena
C-sy	Mental synthesis	Combine objects of thought, images, scenes or concepts
C-tr	Mental transformation	Mentally rearrange, reassemble and alter parts
C-an	Analogical transfer	Transfer relationship or a set of relationships from one context to
C-ca	Categorical reduction	Mentally reduce objects or elements to more primitive categorical descriptions.
C-at	Attribute finding	Systematically search for emergent features in preinventive structures.
C-co	Conceptual interpretation	Take a preinventive structure and find an abstract or metaphorical / theoretical interpretation of it.
C-fu	Functional inference	Explore potential uses or functions of a preinventive structure
C-tx	Contextual shifting	Consider a preinventive structure in a new or different context to gain insights about other possible uses or meanings.
C-hy	Hypothesis testing	Seek to interpret the structure as representing possible solutions to a problem.
C-se	Searching for limitations	Discover limitations to provide insights into which ideas will not work or what types of solutions are not feasible.
BODY		
D-ts	Tracing (same sheet)	Trace over a depiction on the same sheet of paper
D-dt	Tracing (new sheet)	Trace over a depiction on a new sheet of paper
D-sy	Symbol depiction	Depict a symbol that represents a relation
D-wo	Textual aid	Write sentences or words that express ideas
L	Previous depiction	Look or attend to previous depiction
M-od	Movement (over previous depiction)	Move a pencil to the previous depiction
M-a	Movement (sheet beneath)	Move a depiction against the sheet beneath
M-hg	Hand gesture	Use hand gesture
M-og	Other gesture	Use gesture other than hand gesture
ENVIRONMENT		
E-db	Written design brief	Look at given design brief
E-dt	Design tools	Utilise design tool
E-p	Paper	Adjust paper or representation medium
E-ps	Physical surrounding	Refer to physical setting of experiment room
E-b	Blindfold	Adjust blindfold

Fig. 1 Categories and coding scheme

activities. In sum, there are 4522 design moves spanning a total of 960 min of design sessions.

Fifth, the encoding step. Three categories (*Cognition*, *Body*, and *Environment*) of codes were adopted. *Cognition* codes were adapted from creative cognition, the Geneptore (generative and exploratory) processes [23]. *Body* codes were adapted and revised from the *physical* category of Suwa et al.’s [24] coding scheme. *Environment* codes were constructed during pilot studies of 14 participants. The list of codes can be found in Fig. 1. Within each category, linkography [25], a notation system that focuses on links between design moves was derived.

Method of Data Analysis

An online program was developed particularly for this exploration and was implemented in JavaScript (with AngularJS and paper.js frameworks). The input is in .csv file format, which can be uploaded using the template file. The online program can be found at <https://bigzhe.github.io/coded-design-analysis/>. An excerpt of the input file is illustrated in Fig. 2. There are four possible outputs: Output 1 depicts unfolded macro (and micro) design processes: coded design moves in three categories, the interactions (of the categories) within the same utterance and linkograph of each category. Output 2 presents a combined linkograph of three categories. Output 3 shows micro processes within a design move, and the basic statistics provide a snapshot of the session.

The program is flexible in terms of input selection, for instance, range of utterance number, options to turn on/off the linkographs and fonts and dot sizes according to the scale of data needed. Possible outputs at a more detailed scale will be illustrated using P1 (Participant 1)'s sessions by using outputs generated from the online program. In addition, within a larger sample of 24 design sessions, *problem* intentions, particularly problem (3) sub-intentions related to co-evolution will be explored.

Results

Participant 1 (P1)-SK Session

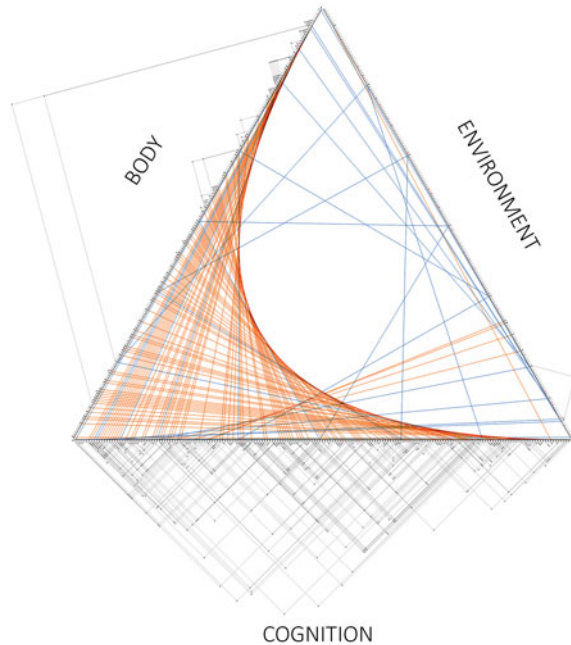
Figure 3 displays an overall view of how the three categories are interrelated in P1's entire SK session (utterance #1-#189), especially the frequent concurrence of *Cognition* and *Body* moves. This is obtained as Output 1 from the program. It shows moves which were coded in two (orange color lines) and three (blue color triangles) categories. If a design move falls into one category only, there will be no lines drawn in the inner triangle. In the outer triangle; utterance number, codes and linkographs are illustrated. Red utterance numbers refer to critical ideas identified by the researcher. At this scale of presentation, an overall representation of how the session progressed throughout time is obtained. In terms of the relationship between categories, in the SK session, P1 was engaged in more Co-Bo (orange color lines) activities in comparison with three-coded categories (Co-Bo-Env, the blue triangles). The Co-Bo-Env activities occurred mostly at the beginning of the session and sparsely throughout the session which might suggest that feedback between mind-body-environment exists to facilitate the design process. Sometimes P1 realized *affordances* from her own marks she left on paper, on other occasion she picked up dimension cues she saw in the room. However, often this scale of representation is not adequate, for instance, if a more detailed look at design moves at a specific time is preferable.

Output 1 can also be modified to a smaller number of utterances (Fig. 4) spanning from utterance number #6-#25, which is identified as exploring the *problem* inten-

No	Time	Transcript	Critical Ideas	Cog code	Links (Cog)	Bo code	Links (Bo)	Env code	Links (Env)
6	0:57.6 - 1:09.1	Let's say that if, firstly I need to have the client analysis and what do they really want, I think.	CI	C-hy		D-sy			
7	1:08.4 - 1:19.7	What they really want? Please read..		C-hy	6	D-sy	6		
8	1:19.0 - 1:38.6	And also this like, dining set up. A place to eat and-- and entertain.. Entertainment. And what else?		C-re	6,7	D-wo	6,7	E-db	
9	1:37.6 - 1:50.9	Learning space. Ok, I think it's normally the same with the adults, like. What's for the children?		C-as	6,7	D-wo	6,7,8	E-db	8
10	1:49.8 - 2:05.9	(long pause) What a child need, for you? Play around..		C-re	9,7	D-sy	6		
11	2:04.9 - 2:12.7	What else, read? Yes..		C-re	7	D-wo	10		
12	2:12.0 - 2:18.0	For study? Dining is for eating right, of course..		C-re	9,8,7	D-wo	11,10,11		
13	2:18.0 - 2:33.6	What else, is there any special one? (long pause) Um.., I think there is nothing special actually.		C-re	6	M-hg	10		
14	2:32.4 - 2:42.8	Play around.. Oh, all the children did before. And um, alright.		C-as	10,9	M-og	10		
15	2:41.8 - 3:01.9	Mixed use.. (long pause). Mixed use place of what? (long pause)	CI	C-sy		D-wo			
16	3:01.9 - 3:05.9	What if I seperated the needs and say that..	CI	C-tr	6	D-c			
17	3:04.4 - 3:15.4	One part is for children. And um, let's say, if we leave one-point-five meters for children		C-hy	6,7,8	D-wo	16		
18	3:15.4 - 3:24.6	And another one-point-five meters for the adult? Adult..		C-hy	6,7,8	D-wo	17	E-db	
19	3:24.6 - 3:36.8	Adults.. Or mixed them together, they have the same-- same need?		C-tx	17,18,15				
20	3:36.8 - 3:42.7	They don't have the same need but they can play at the same space.		C-sy	17,18,15,19	D-wo	15		
21	3:41.7 - 3:53.1	If I imagine the furniture as space or [inaudible mumble]. What it could be?	CI	C-tr	5				
22	3:52.6 - 4:07.7	Let's say, it's quite like um, a sofa or something like that (long pause).		C-at	21	D-c			
23	4:07.0 - 4:17.7	What if the sofa have--um, two sides? (long pause)		C-co	22,17,18	M-og	22		
24	4:16.4 - 4:44.6	If we put the sofa into-- select-- dividing into two part, and then one is for adult one is for children. How to make it interesting? (long pause)		C-fu	16,23				
25	4:44.0 - 4:51.6	But this is too close-- too close. Is not-- what it should be?		C-se	24				

Fig. 2 Program input example (P1-SK session, utterance #6-#25)

Fig. 3 Output 1 of P1-SK session



tion. In addition, Output 2 produces a merged linkograph of the three categories. This is particularly useful to visualize design processes in a singular axis. Figure 5 presents the same excerpt of P1 SK's session. Colors are used to differentiate Body's (red), Environment's (green) and Cognition's linkographs (gray). In comparison with P1's output 1, although sometimes Body links appeared at the same time as Cognition links, from move #10–#13, there were Body links only. These four links were edited and shaded in Fig. 5. This suggests the importance of distinguishing external and internal processes.

Output 3 provides a more detailed view by attaining a graphical representation of each move: categories, codes, and utterance numbers. Figure 6 was partially generated by the program. The program generates two-coded (example: #6) and three-coded (example: #8) Output 3 visualizations only. One-coded moves and explanations (both in navy blue color) were added by the authors for illustration. In terms of *problem* intentions, P1 used the first hypothesizing stage (#6–#16) to frame functions of her intended design with the use of textual aids and symbols while making sure they fit into the provided brief. She then retrieved an idea with the use of prominent gestures. Then she produced an initial idea (#15) and wrote “mixed use space”, then quickly and in contrast, transformed the idea to two separated areas (#16). Subsequently, in move #17 she started a second hypothesis with a new interpretation of a previous idea, to a “mixed use space with separated function”, which then was transformed into the notion of “sofa with two sides” at the end of move #23. In three-coded moves, such as #8, #9 and #18; at approximately the beginning of new ideas

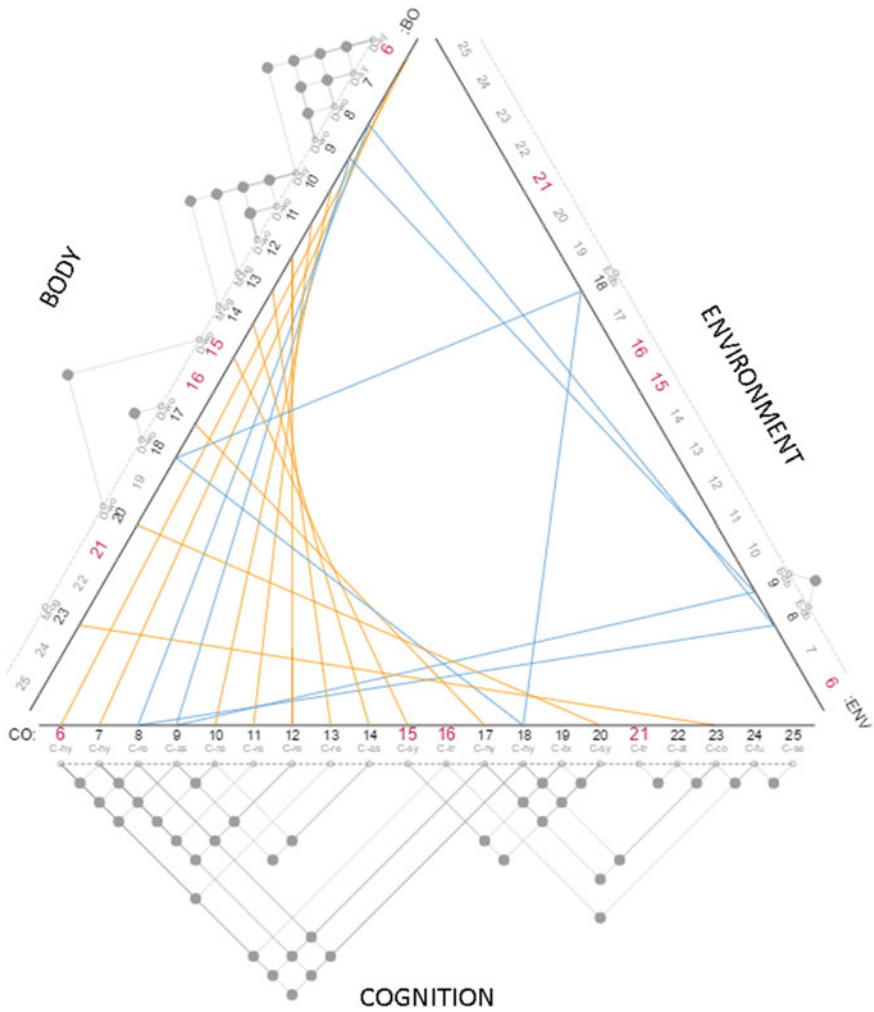


Fig. 4 Output 1 of P1-SK session, utterance #6–#25

being developed; *textual aids*, activity Bo(D-wo) occurred in conjunction with looking at the given *design brief*, activity Env(E-db) through retrieval and association activities. With the three-axis mind–body–environment actions, the ability to visualize each move’s concurrent actions yields the possibility to investigate the feedback loop of cognitive strategies in a designing environment.

Basic statistics generated by the script offer a snapshot of selected preferred data, the whole session or partial. They are total segments, total links, link index, distribution of moves in phase 1–4, top three critical moves, distribution of issues and number of links. Table 3 shows the issue distribution of three categories. The

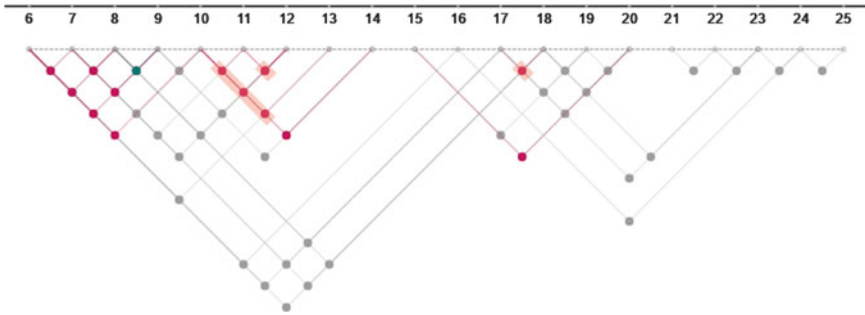


Fig. 5 Output 2 of P1-SK session, utterance #6–#25

Table 3 Issue distribution of moves #6–#25

Co issues	X(%) of total Co issues	Bo issues	X(%) of total Bo issues	Env issues	X(%) of total env issues
C-hy	20	D-sy	21.43	E-db	100
C-re	25	D-wo	57.14		
C-as	10	M-hg	7.14		
C-sy	10	M-og	14.29		
C-tr	10				
C-tx	5				
C-at	5				
C-co	5				
C-fu	5				
C-se	5				

Co(C-re)-retrieval strategy comprised 25% of total Cognition issues, whereas Bo(D-wo)-textual aid embodied more than half of the total Body issues; while 100% of issues occurring in the Environment category were Env(E-db), looking at the design brief.

This short excerpt from P1 shows that the interplay related to *problem* intentions is that external processes mutually aid internal thoughts to bring design knowledge to the fore. As products of Body activities, P1 left visual marks on the paper. In #6–#7 for example, she hypothesized with symbol depiction. These depictions acted as *design affordances*, which are the design environment’s property. P1 needed the ability to utilize the resources in the environment to regulate her behavior. Subsequently, in moves #8–#9 she looked at the design brief. She then made associations and with the use of a pen, she created textual aids. In moves #8–#9, *design effectivities* were displayed, an act which P1 took to realize *affordances* which the symbol depiction facilitated. The pen is crucial to this process because it functions to realize the *affordance* (provided by symbol depiction in moves #6–#7). P1 used a pen as an extension of her hand and therefore extended her bodily space. The pen provides new

	<p>#6-#7: Created first hypothesis with symbol depiction.</p>
	<p>#8-#9: Created textual aids and looked at design brief while retrieving and making an association.</p>
	<p>#10-#12: Retrieved brief content in form of texts and symbols.</p>
	<p>#13-#14: Retrieved and made an association with gestures.</p>
	<p>#15-#16: Synthesized, created text and mentally transform the idea (<i>separation of functions</i>)</p>
	<p>#17-#19: Created second hypothesis by making inferences from design brief, noted down and considered preinventive structure in different context (<i>mixed use space</i>).</p>
	<p>#20-#23: Synthesized with aids of text, transformed an idea mentally and searched emergent feature (<i>'sofa' feature</i>) and made an interpretation of feature (<i>two sides of sofa</i>).</p>
	<p>#24-#25: Inferring thought functions to design and searched for limitations of idea.</p>

Fig. 6 Edited output 3 moves #6-#25

opportunities for action. If the pen did not exist (more like MI session environment), *design affordances* could not be precisely exhibited.

Related to the research hypothesis, it is too early to assume that the boundary between internal and external processes is always shifted with the use of a design tool (a pen in the case of P1-SK session's excerpt) using this small dataset. However, this excerpt provides an early understanding of how a design tool is perhaps no longer acting as an object, but as a functional extension of the mind.

Related to the adapted four questions from the perspective of embodied cognition lens:

“*What is the task to be solved?*” A specific task at the beginning of the design session, in which P1 decided to make boundaries of potential functions as design guidelines. Two important stages were noted, the first hypothesis testing (#6–#16) and the second one (#17–23).

“*What are the resources to solve the task?*” The distribution of resources can be found in Table 3. Apart from the cognitive activities; bodily experience such as giving textual aid, depicting symbols and gesture are fundamental to design exploration and should not be under-valued. In addition, in terms of environmental experience, looking at the design brief played an important role at the beginning of a new idea.

“*How can the resources be assembled?*” From the segment above, the distribution over mind, body, and environment could appear in ways such as: first, in its singularity (one-coded moves); second, two-coded moves with three possible occurrences between Co-Body, Body-Env and Co-Env; and third, three-coded moves.

“*Does the organism, in fact, assemble and use these resources?*” This last question proposes to test whether P1 actually used the solution identified in the previous step (3). With this very limited excerpt, it appears that three-coded moves occur at the beginning of a new idea while structuring a new hypothesis (#18), or after a new hypothesis (#8 and #9). Two-coded moves occurred regularly throughout the segment, with only one type of relation, Co↔Bo (for instance, #6 and #7). ‘↔’ shows concurrent moves across different categories. This preliminary observation might suggest that states of the body affect states of mind. One-coded moves (only Co) suggests that sometimes bodily and environmental experiences were on idle.

Problem Intentions

As mentioned in the relevant literature section, verbal data was categorized into five design intentions with 16 sub-design intentions based on observations of verbal data. This paper focuses on *problem* intentions, particularly *problem* (3) sub-intention which is related to *co-evolution*. An illustration of P1’s *problem* intentions during her session is presented in Fig. 7. It exemplifies how intentions are observed from the audio/video recording, transcript and output 3 derived from the online script (see dotted box in Fig. 7). There are five *problem* intentions which occurred in P1’s SK session, and four include *critical ideas*. These *critical ideas* were derived from observation.

Figure 8 shows the frequency of this particular intention across twenty-four design sessions in the form of a timeline. The length of *problem* intention varies, from one design move to twenty-one design moves, as shown in Table 4. In SK sessions, most design problems are explored during phase 1, thinned out during the second and third phases, and start to appear more towards the end of the session (phase 4). In D sessions, a session where designers are allowed to use provided design tools, designers deal with design problems mostly in phase 1–3. In phase 4, there are none recorded. In MI sessions, however, shorter bursts of activities related to problems occur with higher frequency in the first two phases. Phase 1 is longer than

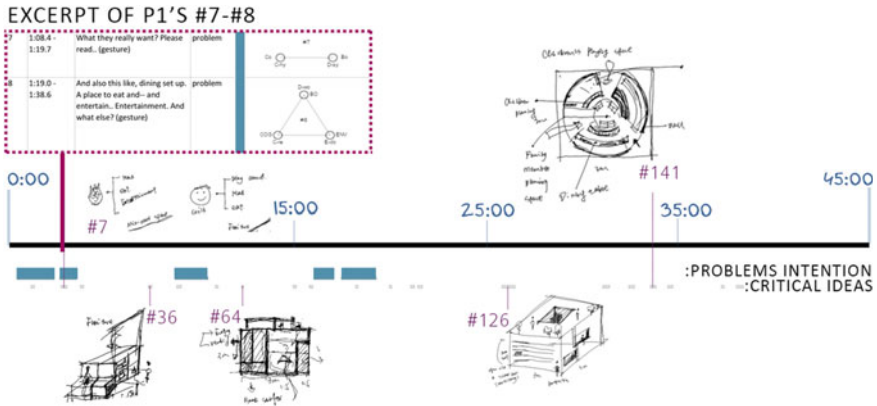


Fig. 7 An example of P1's problem intention (SK session)

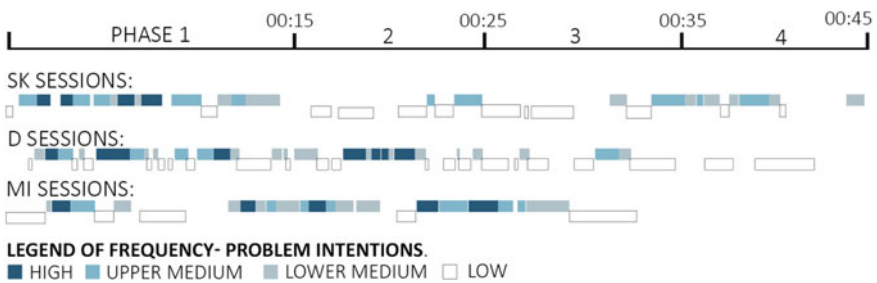


Fig. 8 Problem intentions of 24 sessions and frequency

Table 4 Problem sub-intentions distribution in 24 sessions

Sub-intentions	No of occurrences					
	SK sessions	No. of moves	D sessions	No. of moves	MI sessions	No. of moves
Problems (1)-framing	10	3-17	8	2-25	20	3-20
Problems (2)-III-definitions	5	7-13	1	22	8	3-13
Problems (3)-Co-evolution	22	4-21	15	5-27	23	2-18

the other phases due to the greater occurrence of idea generation while participants are relatively not tired, and this also gives participants time to acclimatize to the experimental setting.

Designers seem to engage in *co-evolution* intentions between design problems and solutions more than other sub-intentions in all the environments (SK, D, and MI). The MI environment seems to be as conducive as the SK environment in terms of facilitating this sub-intention. The activity ranges from two design moves up to twenty-seven moves. This iterative dialogue happens until designers achieve matching problem-solution [9]. Observations of this sub-intention will be presented in the next section, due to it having the highest number of occurrences in comparison with the other two sub-intentions.

Problem (3)-Co-evolution Intention

First, the way designers deal with *co-evolution* strategy in SK sessions is related to function, Co(C-fu). Tentative solutions are explored through possible functions. In D sessions, designers tend to systematically search for emergent features, Co(C-at), while in MI sessions when externalization is limited, by synthesizing (combining objects of thoughts), shown by the code Co(C-hy).

Second, a more balanced percentage between two-coded (Co \leftrightarrow Bo or Bo \leftrightarrow Env or Co \leftrightarrow Env) and three-coded (Co \leftrightarrow Bo \leftrightarrow Env) in comparison with sole activities (either Co, Bo or Env only) is significantly exhibited in SK sessions. A small number of two-coded and three-coded design moves are observed in D and MI sessions. This might suggest that in the SK environment, designers have more access to extend their cognition boundaries through tool use (in this case, pen) and their surroundings. Although in D sessions technically it is possible to simulate the same condition, interestingly it does not happen. There are properties in this pen-and-paper interaction which might not be replaceable.

Third, there are moves which are identified as occurring simultaneously or close in terms of occurrences. In SK sessions, there are two. First, one is Co(C-fu) \leftrightarrow Bo(D-wo) which suggests designers explore functions through textual aids concurrently. ' \leftrightarrow ' shows concurrent moves across different categories, and "—" refers to actions occurring chronologically closely to each other, within the same category. Second is related to the way designers *co-evolve* problems and solutions by hypothesizing emergent features of a design object and through possible functions. The illustration of this strategy is as follows: Co(C-hy)—Co(C-at)—Co(C-fu), refer to Fig. 9 (highlighted in cyan circles). In the D session, Co(C-fu)—Co(C-at) often occur together. This phenomenon is quite similar to SK sessions, and suggests that with unlimited access to externalization (pen and available design tools), designers are facilitated in searching for emergent features. In MI sessions when externalization opportunity is limited, designers compensate for the withdrawal by synthesizing design conditions, Co(C-hy)—Co(C-sy) and test the hypotheses through functions, Co(C-hy)—Co(C-fu).

Fourth, with agreement in three different designing environments, designers mostly engage in co-evolving strategy through the act of exploring functions and uses.

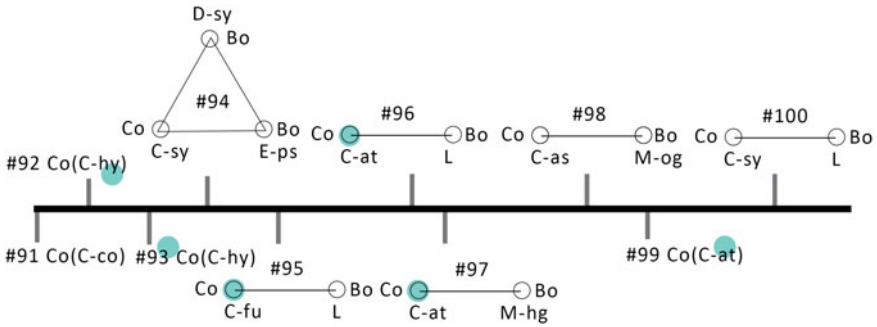


Fig. 9 Illustration of iterative C-hy, C-at and C-fu in P1-SK session

Next, related to the posited research hypothesis

When access to external representations (sketches) is unlimited; the boundary between internal and external processes can be shifted. And when access to external representations is limited and staged, limitations are overcome by the use of bodily movements. The boundary remains.

To reiterate, this explanation is limited to *problem* (3) sub-intentions, dealing with co-evolution. One appealing notion in SK sessions is how designers engage in moves related to a previously made depiction; particularly these three activities: Bo(M-od)-movement over previous depiction, Bo(D-rf)-revising depiction and Bo(L)-looking at previous depiction. Similarly, in D sessions, Bo(L) activity is significantly high. On four occasions, the notion of looking at previous depictions (on the same designing page or different pages) appears simultaneously with the act of contextual shifting to gain new insight into other possible uses, Bo(L) ↔ Co(C-tx). The use of tools does not always extend the cognition boundary as illustrated in P1-SK session. Although the cognition boundary moves (towards the depiction designers previously produced), it is not enhanced by tool use. *Design affordances* provided by previously made depictions assist Co(C-at) and Co(C-fu) activities which were identified as commonly occurring together in both SK and D sessions. It can be assumed that these acts of Co(C-at) and Co(C-fu) are the *design effectivities*. Unlike in MI sessions when externalizations are limited, limitations are not compensated by bodily movement. In this sub-intention, designers deal with synthesizing design conditions. The adapted four questions will now be explored, focusing on the *Co-evolution* sub-intention.

“*What is the task to be solved?*” During the co-evolving problem and solution, the more specific task is to use tentative solutions to gain a better understanding of problems.

“*What are the resources to solve the task?*” Typically, design tasks are ill-defined or wicked problems. There is no right or wrong solution. Resources are an exhaustive list of processes and their combinations; happening in the main (creative cognition consisting of processes used to generate creative structures such as Co(C-re), Co(C-sy) and the exploratory processes such as Co(C-co) process); body (related to

physical movements of pen and paper); designing environment and relations between brain–body environment.

“*How can the resources be assembled?*” and “*Does the organism, in fact, assemble and use these resources?*” There are different ways—almost unlimited ways—to assemble the resources. It is thus impossible to list possible ways or design solutions. However, some common patterns can be identified. First, designers deal with the functions of the design object when they co-evolve problems and solutions, often Co(C-fu) activity. Secondly, iterative processes such as Co(C-fu)—Co(C-at) happen when designers have access to externalization. And iterative Co(C-hy)—Co(C-sy) and Co(C-hy)—Co(C-fu) to compensate for the limitation of externalizations in MI sessions. Third, previously made depictions provide assistance to extend the cognition boundary when externalizations are possible.

It is noted that these four key questions cannot be satisfactorily answered in the designing situation and should be adopted, due to the nature of openness of design problems and solutions.

Conclusions

This paper has demonstrated an online visualization tool to possibly facilitate empirical investigation of the interplay between internal and external processes through the lens of embodied cognition. By considering gesture, movement, tools designers use and designing environment as a constituent of rather than causal, it provides understanding on how the interplay come into play. It is concluded that when access to external representations (sketches) is unlimited, the boundary between internal and external processes might be shifted, but not always. This is illustrated in P1-SK excerpt. Once P1 used the pen to make new depiction, it served as an extension of the hand, therefore the boundary between internal and external was no longer fixed at the surface of her skin. The pen became part of the designer rather than the environment. In situations when externalizations are limited, the boundary between internal and external processes remains. As there is no feedback loop between internal and external processes due to the blindfold, in the *problem* (3) sub-intention related to co-evolution, the withdrawal is compensated for by internal processes such as synthesizing and hypothesizing rather than bodily movements. Regarding future recommendations, this study has provided a groundwork for further studies in real design settings. It has also demonstrated the potential of both linkography and the publicly available computer script to simplify data analysis in this and related fields.

References

1. Bilda Z, Gero JS, Purcell T (2006) To sketch or not to sketch? That is the question. *Des Stud* 27:587–613
2. Bilda Z, Gero JS (2007) The impact of working memory limitations on the design process during conceptualization. *Des Stud* 28:343–367
3. Bilda Z, Gero JS (2008) Idea development can occur using imagery only. In: Gero JS, Goel AK (eds). Springer, Netherlands, Dordrecht, pp 303–320
4. Fish J, Scrivener S (1990) Amplifying the mind's eye: sketching and visual cognition. *Leonardo* 23:117–126
5. Goldschmidt G (1991) The dialectics of sketching. *Creativity Res J* 4:123–143
6. Kavakli M, Scrivener SAR, Ball LJ (1998) Structure in idea sketching behaviour. *Des Stud* 19:485–517
7. Pallasmaa J (2009) *The thinking hand: existential and embodied wisdom in architecture*. Wiley [distributor]
8. Goldschmidt G (2003) The backtalk of self-generated sketches. *Des Issues* 19:72–88
9. Bernal M, Haymaker JR, Eastman C (2015) On the role of computational support for designers in action. *Des Stud* 41(Part B):163–182
10. Shapiro L (2010) *Embodied cognition*. Routledge
11. Campbell G, Shapiro L (2011) Embodied cognition with Lawrence Shapiro. In: Campbell G (ed) *Brain science podcast*
12. Leitan ND, Chaffey L (2014) Embodied cognition and its applications: a brief review. *Sensoria: A Jof Mind Brain Culture* 10:3–10
13. Wilson M (2002) Six views of embodied cognition. *Psychon Bull Rev* 9:625–636
14. Wilson A, Golonka S (2013) Embodied cognition is not what you think it is. *Frontiers in Psychol* 4:1–13
15. Clark A (2008) *Supersizing the mind: embodiment, action, and cognitive extension*. OUP USA
16. Gero JS, Kannengiesser U (2004) The situated function–behaviour–structure framework. *Des Stud* 25:373–391
17. Kolbeinsson A, Lindblom J (2015) Mind the body: how embodied cognition matters in manufacturing. *Procedia Manuf* 3:5184–5191
18. Gibson JJ (1986) *The ecological approach to visual perception*. Lawrence Erlbaum Associates
19. Hirose N (2002) An ecological approach to embodiment and cognition. *Cogn Syst Res* 3:289–299
20. Norman DA (2002) *The design of everyday things*. Basic Books, Norman
21. Ericsson KA, Simon HA (1993) *Protocol analysis: verbal reports as data revised edition*. MIT Press, Cambridge, Mass
22. Bilda Z, Gero J (2006) Reasoning with internal and external representations: a case study with expert architects. In: Sun R (ed) *The annual meeting of cognitive science society*. Lawrence Erlbaum Associates, pp 1020–1026
23. Finke RA, Ward TB, Smith SM (1992) *Creative cognition: theory, research and applications*. MIT Press, Cambridge
24. Suwa M, Purcell T, Gero J (1998) Macroscopic analysis of design processes based on a scheme for coding designers' cognitive actions. *Des Stud* 19:455–483
25. Goldschmidt G (2014) *Linkography [electronic resource]: unfolding the design process*. MIT Press

Visual Interactivity to Make Sense of Heterogeneous Streams of Design Activity Data



Yasuhiro Yamamoto and Kumiyo Nakakoji

The goal of our project is to design computational environments to help understand or presume the origin, trajectories, or relationships of a certain part or aspect of an existing designed artifact. Our approach is to visualize multiple heterogeneous streams of design activity data related to the artifact, which had been naturally accumulated as a result of designers' engaging in a wide range of activities using a variety of computational tools during the course of designing the artifact. This paper describes the motivation for the approach and presents the visual interaction design for such computational environments, including linear browsing within a single stream, temporal alignment of multiple streams, and symbolic and semantic associations across different streams of data. We demonstrate two of the prototyped systems to reveal the notion of clusters introduced in the interaction design, and illustrate a cyclic process of trace, focus, and highlight as a core user interaction model commonly employed in the two environments. The paper concludes by discussing how our approach of using design activity history data supports investigating the provenance of some elements of a designed artifact in a variety of ways.

Introduction

We have come to use computational tools to perform a variety of design-related activities, such as investigating and analyzing user information, sketching ideas, taking notes during stakeholder interviews, discussing ideas through email and conversational media, making presentations, conducting remote meetings, prototyping, versioning, keeping alternatives, or analyzing study participants by video-recording the study sessions. Such activities and representations produced over a long period

Y. Yamamoto · K. Nakakoji (✉)
Kyoto University, Kyoto, Japan
e-mail: kumiyo@acm.org

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_37

687

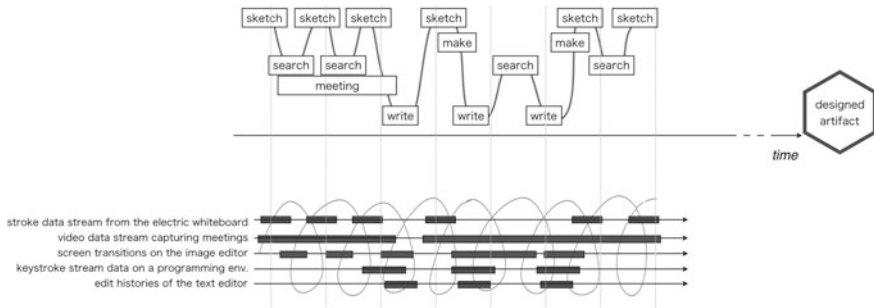


Fig. 1 Naturally accumulating design activity data of an artifact

of a design project duration (ranging from a week to months to sometimes years, for instance, in the case of city development) often result in a potentially massive amount of the collection of a variety of time-stamped process and object records; they become heterogeneous multiple streams of design activity data accumulated in individual tools (Fig. 1).

We postulate that more and more artifacts existing in the coming future would be accompanied with such accumulated design process data. Such data is likely to contain some pieces of information or clue to understand or presume produced ideas and artifacts, decisions made and knowledge involved, and external or social context that might have affected the deliberation and decision-making processes. Each of such pieces of information is time-stamped, telling when it was happening.

If one becomes interested in knowing why a certain part of an existing artifact carries a particular feature, or what is the origin of some concepts of the artifact, such accumulated design activity data could serve as a rich source of information to investigate. The goal of our project is to design computational environments that make use of the multiple time-stamped heterogeneous streams of design activity data of a designed artifact to help us address the questions about the artifact, by making sense of the data, during the design project as well as long after the project is over.

In what follows, we first illustrate the motivation of our approach, which uses accumulating activity traces in computational tools. The field of MSR (Mining Software Repository) has long been investigating software development process activity and outcome histories in a variety of purposes. The following section discusses existing research related to our approach, including design rationale, design process history, and visual analytics provenance. We present the visual interaction design in our approach including linear browsing within a single stream, temporal alignment of multiple streams, and symbolic and semantic associations across different streams of data. We demonstrate two prototyped systems, DPS (Design Practice Streams) and Orca (Observation-Representation-Communication-Archival), as the instantiation of the visual interaction design. The reflection on the instantiation reveals the notion of clusters introduced in the interaction design, and identifies a cyclic process of trace, focus, and highlight as a core user interaction model commonly employed in the two

environments. The paper concludes by discussing how our approach of using design activity history data supports investigating the provenance of some elements of a designed artifact in a variety of ways.

Motivation: Using Accumulating Activity Traces

As a variety of design activities have started taking place on computational environments, some aspects of design activities are naturally accumulated and become available in individual tools. Software development has long demonstrated the use of data accumulated in computational environments used for various collaborative creative knowledge activities [1].

Software developers use programming environments to keep track of edit histories and store different versions of program components on version control systems. Project members communicate with one another through email, mailing lists, and other social media such as Slack. They use project management systems to organize a wide range of tasks among the project members by articulating each task on a ticket assigning developers to be in charge and tracking the progress for the task. Program testing is performed on testing tools. Reporting and fixing bugs is managed on bug management tools such as Bugzilla. Documents are shared on shared document repositories using Wiki.

Each of such computational tools individually supports the development of its own type of artifact. A programming environment houses the gradual growth of program components, Bugzilla houses the generation and resolution of bugs, and a mailing list houses the peer-to-peer communication histories including question-answering dialogues on particular code snippets, or task coordination announcements. Although those tools individually store activity logs separately, they are typically time-stamped and often associated with the developer information (i.e., a developer's email account). Such temporal and developer information serve as keys to associate multiple data segments stored among different repositories to derive some stories, for example, to make sense of the historical development of a particular revision of source code; such as "Tom changed the code which was originally programmed by David, after communicating with Jack and Jill about the data update timing, around the time when the major scheduling change was announced by the project leader Kate."

MSR (Mining Software Repository) is a research community that analyzes the rich data available in software repositories to "uncover interesting and actionable information about software systems and projects" [2]. A variety of tools have been developed to investigate ways to discover relationships in different types of software repositories, such as to predict bugs, to detect fault-prone modules, or to analyze social networks among developers [3].

Related Work

Three areas of existing research relevant to the use of accumulated design activities include design rationale, design process history, and visual analytics provenance.

Design Rationale

Design rationale refers to the explanation, reasoning, and internal logic of a decision made during a design project [4]. Design rationale research aims at articulating and externalizing the reasoning and logic behind a decision made to help design project members understand the current and the past design situations better. Design rationale would remind designers of why certain decisions were made in such a way sometime ago in an earlier design process phase. It would also inform those who later joined the project of the ongoing design project status, thus helping design project members synchronously and asynchronously collaborate more effectively. Understanding the current design situation would help designers listen to the backtalk of the situation better [5], and guide them toward more creative design process and product [6].

One of the two major approaches on design rationale is to record and structure design discussions and arguments carried out during a design process in a form of argumentation-base. The other approach is to write down comments and concerns raised during a decision-making process and associate them with relevant parts of an evolving artifact to be designed.

Argumentation-Based Design Rational

Early approaches on recording decision-making processes in design to reflect on them include argumentation-based methods, the paradigmatic system of which was IBIS (Issue-Based Information Systems) [7]. The basic idea of IBIS is to represent design discussions and ideas as a network of argument structures, consisting of Issues (i.e., a design concern or a partial problem), Answers (i.e., a response, a claim, or a partial solution), and Arguments (i.e., pro- and con-reasoning, an explanation, or a ground). IBIS served as a notation to record and structure design concerns and discussions raised during a design process and to read and reexamine them later in the process.

A computational tool, gIBIS (Graphical IBIS), was then proposed as a computational environment to support IBIS [8]. Compendium succeeded the approach and evolved it as software service [9], and various tools that visualize argumentation have been proposed in the context of helping sense-making in multi-stakeholder, ill-structured problems [10, 11].

Many of early approaches express, record, and browse discussions independently what is actually produced as design artifacts, and efforts have been made to integrate the argumentation space with artifacts [12].

Design Annotation

Design annotation directly associates ideas and comments with the relevant part of a design component represented in a digital form or modeled on a computational environment.

Studies on supporting designers to annotate some parts of an intermediate design state during a design process have started in the early 90s. An example of such early work includes an interactive environment for designing a LAN (Local-Area Network), where a user adds comments to a component of the evolving LAN design [13]. Some members of the design project then could read such comments later in the design process supporting asynchronous communication among the design project members [14]. A later example is a tool that allows a user to examine a building being designed by walking through its 3D VR (Virtual Reality) model, to write down a concern or comment on a VR post-it note, and to attach it to the relevant part of the building, such as a wall, shelf, window, or a door [15].

Design Process History

Existing studies explored ways to collect and store design activities that are engaged on a computational environment during a design process [16]. Examples include tools that record file editing and browsing histories [17, 18], collect whiteboard drawing and erasing actions [19], or accumulate web browsing histories [20]. Such systems typically provide undo (to roll back to a previous state before the operation was performed) or redo (to repeat the same operation to a different part) operations, and may identify parts that had been frequently modified or referred to by using such process histories.

A variety of tools have been developed for videotaping design meetings. Such recorded design meetings would help design project members to asynchronously collaborate, and serve as a source for providing design rationale for particular design decisions. The major challenge of the approach has been how to extract particular parts of the recorded meetings relevant to the current interest. To address the challenge, mechanisms have been proposed to automatically or semiautomatically summarize video-recorded meetings, or to structure the video data by annotating or adding links to its parts while browsing the data. One of recent example is a HyperMeeting, which supports a chain of geographically and temporally distributed meetings in the form of a hypervideo with an automatically generated playback plans based on the user's interests or prior meeting attendance [21].

Many of such design process histories and design meeting record approaches deal with a single medium where design activities take place; a tool collects and stores design actions and operations, and helps designers browse, explore, and reuse the recorded process or the product.

Provenance in Visual Analytics

The notion of provenance has been studied in the field of visual analytics since the early 2000s [22]. The term provenance means the place of origin or earliest known history of something. It is often used to describe where an artwork has been after the artist created the artwork, guiding to authenticity or quality. Provenance comes from Latin *provenire*, meaning “come or stem from.”

Data analysts engage in data visualization to make assumptions and draw conclusions through sense-making by zooming in and out, shifting display regions, and changing focal points. Provenance in visual analytics typically consists of a series of tables, graphs, and nodes, displayed and examined by a user. It helps analysts to inspect and validate the assumptions and conclusions made during a prior visual analytics process and supports collaboration among visual analytics team members [23]. Computational tools and environments have been employed and studied to support visual analytics provenance. Feire et al. [24] propose the notion of provenance-enabled systems, which embed provenance support mechanisms in visual analytics workflow environments. The provenance-support mechanism is composed of collecting and storing activity histories and providing them as provenance for a user’s perusal. They argue that supporting visual analytics provenance requires three components: a mechanism to capture provenance, a model to represent provenance, and an infrastructure that stores, accesses, and searches provenance.

VisTrails is an example of provenance-enabled system that supports the workflow of science discovery while managing discovery provenance [<http://www.vistrails.org>]. Recent work on visual analytics provenance includes an approach that integrates external information (such as a user’s notes and comments) with operation histories [25]. Insight provenance captures a user’s thought externalization process together with operation histories and records a data analysis process at a higher level of abstraction (called actions) by observing the moments of a user getting insights while engaging in data visualization [26].

The Design of Visual Interactivity

This section describes the design of visual interactivity for our environments that support users to engage in interconnected historical elements in various manners. Our approach proceeds through the cycle of identifying core visual interactivity, instanti-

ating the interactivity in prototyping, producing the actual user experience by using the prototyped tools, and examining the experience and refining the interactivity.

We have been prototyping interactive tools to deal with particular chronological data streams, including tools for visualizing the chronological historical tables of the city of Kyoto over 1400 years [27], interactive visualization tools for several thousands stock prices of Tokyo stock exchange over the period of 25 years [28], and browsers for scanned drawings produced every day by an artist over the last 30 years.

Making sense of data starts by either wandering over the entire data space or by focusing on one element and investigating its relevant elements in interacting with those prototyped tools. The reflections on the experience of interacting with them help us identify the following three elements of interactivity for dynamically identifying elements relevant to the user's current focus among multiple heterogeneous streams of design activity data.

- *linear browsing within a single stream*: the user sequentially traces a single data stream in the chronological order by focusing on each of its elements (forward or backward).
- *temporal alignment of multiple streams*: the user chronologically compares elements of different data streams.
- *symbolic and semantic associations across different streams of data*: the user finds relevant elements scattered over time and across multiple data streams.

The temporal order implies the cause-effect relationship. We have found that it is quintessential to make it obvious for the user which data elements chronologically precede or succeed the element under focus. It has also become apparent that the user needs a systematic way to browse each element of a data element to focus. Once having focused on one element, the user needs to explore its relevant elements not only temporarily relevant ones but also semantically related ones.

Instantiated Visual Interactivity

This section demonstrates two prototyped systems that instantiated the visual interactivity for browsing heterogeneous streams of data described in the previous section: DPS (Design Practice Streams) and Orca (Observation-Representation-Communication-Archival).

DPS (Design Practice Streams)

DPS (Design Practice Streams) (Fig. 2) aims at helping a user extract the relevant portions of the design process data to the point of concern. DPS supports browsing

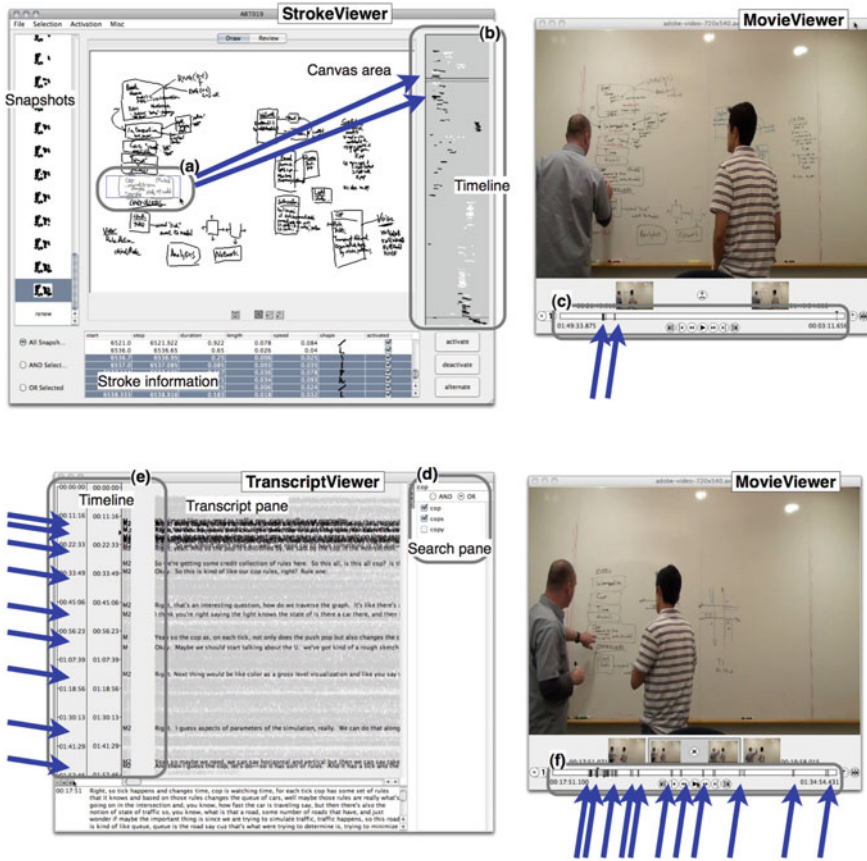


Fig. 2 DPS (design practice streams) consisting of stroke viewer (top left), movie viewer (right), and transcript viewer (bottom left)

the process of a software design meeting captured in the video-recording, transcribed utterances (text), and visual diagrams (whiteboard drawings).

DPS does not require any tagging, annotations, or semantic analysis of the recorded meeting captured in video, transcripts, and stroke data. DPS provides basic search mechanisms over each medium. It uses time stamps to glue segments in different media together. DPS has been developed since 2010, and the earlier version was originally reported in [29]. This section gives its overview by illustrating how the designed visual interactivity described in the previous section is instantiated.

The interaction design of DPS is such that it would help the user (1) search for points of concern through diagram-based search over the recorded stroke data drawn on the whiteboard, or through text-based search over the transcript data, then (2) hop over and browse related segments of stroke data, transcript data, and video data through their time stamps. By integrating the three media, the user of DPS explores

the recorded data by switching between temporal proximity, spatial proximity, and symbolic proximity of the focal point.

DPS consists of three components: the Movie Viewer for the video data, the Transcript Viewer for the transcript data, and the Stroke Viewer for the stroke data (Fig. 2). Diagram-based search uses temporal proximity and spatial proximity to specify the point of concern. Text-based search uses incremental text matching over each sentence in the transcript. A search result often involves multiple strokes, or multiple sentences, each of which is associated with a time stamp. We use such a time stamp to associate each stroke and sentence with the corresponding video segment with the same time stamp. Thus, a search result may identify multiple segments of the video, all of which the user may play at a time in a sequential manner.

When a user specifies a rectangular region in the Canvas area of the Stroke Viewer (Fig. 2a, b), the system identifies a set of strokes that are included in the specified area, and the Movie Viewer identifies the video segments that have the same set of time stamps (Fig. 2c). The transcript viewer also identifies a set of sentences that have the same time stamps, showing what dialogues were taking place while drawing the region possible over scattered periods of time.

Similarly, when a user specifies a phrase as a search query in the Transcript Viewer (Fig. 2d), the system locates one or more sentences among the transcript data that have the queried phrase (Fig. 2e). Then, the Movie Viewer identifies the video segments that have the same set of time stamps, and a user can watch all the video segments that have the utterances that have the queried phrase in a sequential manner (Fig. 2f). The corresponding strokes that have their time stamps within the selected time stamp periods are also identified and selected in the Stroke Viewer, emphasizing which parts were drawn while uttering the queried phrase.

Orca

Orca (Observation-Representation-Communication-Archival) is a Web browser based viewer that displays different streams of time series data temporally aligned either vertically or horizontally in a chronological table form (Fig. 3). A stream of time-stamped object data is displayed either in each column in a vertical display, or in each row in a horizontal display. Types of objects that Orca is currently able to deal with are text, numbers, image files, and movie files.

The timeline goes from top to bottom in the vertical display, or from right to left in a horizontal display (corresponding to the Japanese vertical writing). Each column (or row) may be assigned with a label, which is displayed at the beginning of the column (or row). Objects in each column (or row) are positioned so that those have the same time stamp are aligned horizontally (or vertically) thereby all the columns (or rows) share the common timeline. The timeline may be sectioned while each section may be assigned with a label.

Figure 3 left shows a part of the Orca displaying the history of periodical internal newspaper articles published by a company over nine years. It shows a list of titles

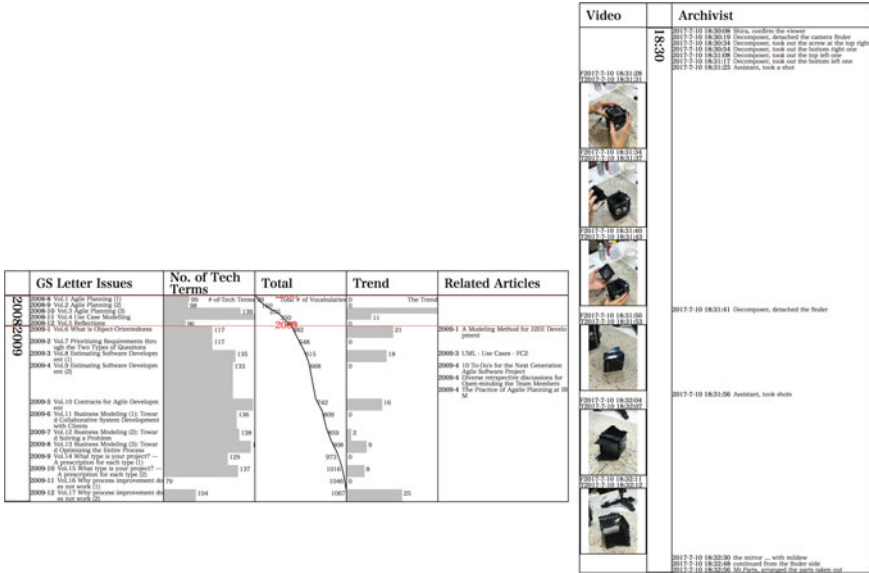


Fig. 3 Orca with the vertical display mode where the timeline goes from top to bottom (left: about a company’s periodical newspaper articles; right: about the process decomposing a classical camera)

each with its publication date, accompanied by the number of technical terms contained in the article, alighted with technical newspaper webpage available at the time of the publication collected from the public news server. Figure 3 right shows a part of the Orca displaying the recorded activities of a university course project, where the enrolled students decomposed a classical twin lens reflex camera. The activity was recorded by using two video cameras to take videos, mobile phone camera to take photos, and time-stamped text notes composed by one of the students.

Orca markup language is designed to display a series of time-stamped object by combining the HTML and Markdown notations and formatting syntax. The basic form of an orca markup file consists of stream specifications (Fig. 4). There are five types for specifying streams: *timeline*, *graph*, *ruler*, *timeseparator*, and *spawn*. An orca markup file can contain more than one stream specifications of the same type specified in the markup file.

The *timeline* type is for displaying one type of time series data, which would then be mapped to a column (or a row). The first line starts with “#” serves for the label to display in the beginning of the column (or the row). Following the first line, each time-stamped object must be specified in a single line starting with a time stamp. If the object is text, then text content should follow the time stamp. A long sentence may be wrapped to fit in the width of the column (or the row). If the object is an image or a movie, then a path to the location of the image or movie file should be put. When positioning an image, its thumbnail image is displayed at the corresponding timeline, which is clickable to enlarge and displays the original image file. When

Fig. 4 Orca stream specification

```

1 ---
2
3 timeline
4
5 ---
6
7 # Video
8
9 f-yyyy-mm-dd f-hh:mm:ss t-yyyy-mm-dd t-hh:mm:ss image video
10 2017-07-10 18:31:28 2017-07-10 18:31:31 iPhone/IMG_0183.MOV
11 2017-07-10 18:31:34 2017-07-10 18:31:37 iPhone/IMG_0184.MOV
12 2017-07-10 18:31:40 2017-07-10 18:31:43 iPhone/IMG_0185.MOV
13 2017-07-10 18:31:50 2017-07-10 18:31:53 iPhone/IMG_0186.MOV
14 2017-07-10 18:32:04 2017-07-10 18:32:07 iPhone/IMG_0187.MOV
15 2017-07-10 18:32:11 2017-07-10 18:32:12 iPhone/IMG_0188.MOV
16
17 ---
18
19 ruler
20
21 ---
22
23 f-yyyy-mm-dd f-hh:mm:ss t-yyyy-mm-dd t-hh:mm:ss text
24 2017-07-10 18:30:01 2017-07-10 18:45:00 18:30
25
26 -----
27
28 timeline
29
30 -----
31
32 # Archivist
33
34 yyyy-mm-dd hh:mm:ss text
35 2017-07-10 18:30:08 Shira, confirm the viewer
36 2017-07-10 18:30:19 Decomposer, detached the camera finder
37 2017-07-10 18:30:34 Decomposer, took out the screw at the top right
38 2017-07-10 18:30:54 Decomposer, took out the bottom right one
39 2017-07-10 18:31:08 Decomposer, took out the top left one
40 2017-07-10 18:31:17 Decomposer, took out the bottom left one
41 2017-07-10 18:31:25 Assistant, took a shot
42 2017-07-10 18:31:41 Decomposer, detached the finder
43 2017-07-10 18:31:56 Assistant, took shots
44 2017-07-10 18:32:30 the mirror ... with mildew
45 2017-07-10 18:32:48 continued from the finder side
46 2017-07-10 18:32:56 Mr.Parts, arranged the parts taken out
47 2017-07-10 18:33:08 Decomposer, decomposed
    
```

positioning a movie, thumbnail scenes of every minute during the movie duration are generated and displayed at the corresponding timeline. Each thumbnail is clickable to start playing the video from the point of time when the thumbnail is positioned. The *graph* type is similar to timeline but for displaying numbers in a form of a line graph instead of text.

The *ruler* type is for segmenting a timeline and displaying a label for each segment, which is written in each line. Following the first line, which serves for labeling the ruler, each line starts with two time stamps, corresponding to the beginning and the end of the segment, followed by a label for its segment. The *timeseparator* type is for inserting align lines in red at the location of the time stamps specified in each line, followed by a label. Finally, the *spawn* type is for extracting some parts of the timeline to display by specifying the beginning and end of the time to extract.

Reflections on the Prototype Instantiation

While instantiating DPS and Orca, we introduced the notion of clusters over design activity data streams (see Fig. 5). Our original interaction design (as described above) postulates that a user would follow a single data stream in the chronological order. DPS and Orca reveals that other types of sequential tracking were also found to be necessary, so we have introduced clusters.

A cluster is a collection of time-stamped data elements related with each other in some ways. There are many ways to formulate clusters. A cluster may be a single data stream accumulated in a tool (e.g., surrounded by a dotted horizontally long rectangle in Fig. 5), a segmented part of a data stream by sharing the same time stamps (e.g., surrounded by a dotted vertically long rectangle in Fig. 5), a collection of a keyword search results over data streams (e.g., connected by dotted curved lines in Fig. 5), or generated by applying data clustering algorithms. Some clusters are statically determined, and others may be dynamically computed (i.e., by search results or by changing parameters for clustering algorithms). A data element belongs to at least one cluster (the original data stream source), and is likely to belong to many other clusters. When a user focuses on a data element, each of its belonging clusters contains the related elements in corresponding contexts.

In interacting with clusters, there was a cyclic process of *trace*, *focus*, and *highlight* as a core user interaction model commonly employed in the two environments:

- *trace*: a user visits (i.e., browses) each element one by one in a chronological order within a cluster (e.g., a single data stream). Where the element is in terms of the whole (i.e., the cluster, the time, and the original data stream source) ought to be visually emphasized, and the timeline must be zoomable. Changing the direction of the trace (chronologically forward or backward) within the cluster ought to be easy to perform.
- *focus*: a user shows some interest on one element while tracing the cluster, probably wanting to explore its related information spaces. A user ought to be able to easily examine details of the focused element and to identify the other clusters, the focused element belongs to.
- *highlight*: a user chooses one of the clusters the focused element belongs to. The system visually stresses the elements of the chosen cluster as related elements to the focused.

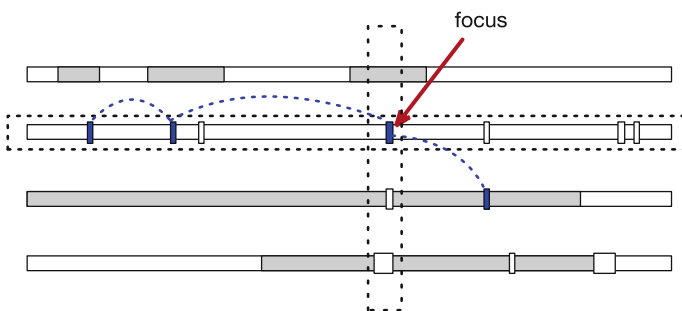


Fig. 5 User-interaction with multiple streams of data through clusters

Discussion

This paper presented our approach to build computational environments that provide trajectories, clues, and plausible relationships relevant to a user's current concern about an existing artifact from the rich source of its historical information spaces. We view that our environments help enriching the interpretation of an existing artifact by adding the temporal or historical dimension to the artifact.

An artifact, as the designed outcome, is present-at-hand as a result of a variety of activities and situations intertwined over the course of the design evolutionary period. Design is a result of people engaging in creative knowledge work. The design carries the knowledge, deliberation, experience, creativity, and enthusiasm of those who have engaged during the design process. At each point of time, a certain decision was explicitly made, temporarily decided, unwillingly accepted, or unknowingly implied by design team members at the time, each of whom was situated in the context, such as what experience he/she had been coming through by that time, what was available, was popular to employ, was commonly trusted, was uniquely prominent, was believed to be promising, etc. Each member's ideas and thoughts at the point of time were interwoven into the decision-making process, making the design as a result of piled-up socio-technical systems.

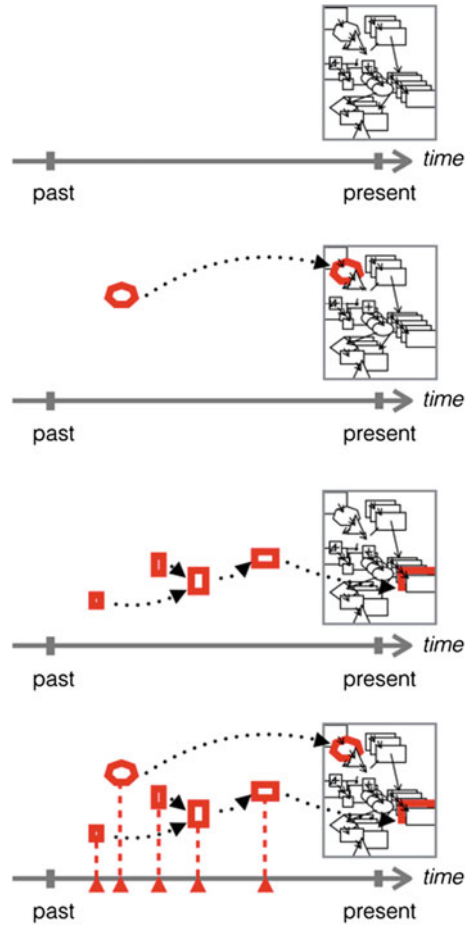
Design Provenance

Understanding the evolutionary course of a certain design element, the surrounding conditions and situations at each moment during the course, and then-existing relationships with other elements that might have diminished or become irrelevant over the time period, would help us better understand the value of the artifact and its *raison d'être*. Our environments help the user investigate the provenance of a certain aspect of an artifact. Such design provenance carries two notable characteristics: contextuality and indeterminacy.

Contextuality

The provenance of a design may exist in infinite number of ways depending on which component, or which aspect of the artifact to focus on (Fig. 6). When one's interest is on a few elements of the artifact, the one may want to know how each of the elements came into existence or became relevant to the final design. If focusing on the relationship among few elements, one may want to know why and how the relationship emerged and became significant. Focusing on elements of varying granularity would make one interested in knowing when, how, and why those elements emerged and coexisted over a long period of time during the lifetime of the artifact.

Fig. 6 Design provenance explored through our environments



Indeterminacy

Users would engage in completely different experience of provenance depending on which design element to start focusing on in using our systems. What is presented to the user is not the “right” provenance. Rather, having the object or the area of interest as a starting point, the user hauls in its relevant pieces of information in order. By so doing, the user may cognitively supply a seemingly missing piece of information or fill a contradictory information chasm through one’s simulation and imagination by picturing a story or plausible explanation for the focused topic. It depends on the user’s cognitive simulation, imagination, and inspiration to turn them into provenance.

This is in contrast with design history studies or design archeology. The primary goal of the historical research is to organize and structure the historical process and

its surrounding environmental changes from an orderly perspective; they investigate the evolutionary process of a design, seeking for the right or correct provenance by structuring and formalizing the historical information space.

Line-Ness

Ingold [30] uses the concept of a line to depict the life of a person, of a thing, of a city, or a wide variety of phenomena such as music, speech, or travel. A line may be drawn from left to right where the horizontal axis corresponds to the flow of time. As a number of lines, each of which corresponds to a person, an object, music or a book, coexist by sharing the same period of time (the same vertical area), some of the lines may be intertwined, producing knots. Such knots represent meeting points, encounters, or interactions among the intertwined lines.

We are inspired by this notion of lines as a way to understand how the collection of a variety of time series data streams each accumulated in individual tools and environments serve as the source for investigating the provenance of a designed artifact. Each time series data stream is a line. Notions such as continuation, duration, and coexistence, which are naturally expressed in lines are applicable to the space of multiple time series data streams.

Aligning points or sections on different data streams through their time stamps is like creating a knot with the lines. Ingold calls a large number of intertwined lines through knots a meshwork. Associating points and segments of different data streams through time stamps may be viewed as producing a meshwork of design historical activities [30].

The notion of lines is similar to narratives, as the word storyline implies. A user of our environment traversing the space of a variety of time series data streams like a meshwork would help him/her find or generate a story relevant to the current focus.

Concluding Remarks

The research presented in this paper serves for the same purpose as design rationale research has been aimed at but through a different means. Instead of trying to compose structured arguments or asking designers to annotate ongoing design for later use, our approach uses activity histories accumulating in a variety of individual tools. Our tools are to help people answering the question they have got about an artifact by interacting with its design history spaces while imagining or presuming plausible account behind it, such as how particular information has grown or why a certain idea has evolved, by tracing the transitional processes of a certain aspect of the artifact. The designed visual interactivity for browsing accumulated activity histories would allow a user to feel as if he/she is investigating the provenance of the current concern. Our future work includes to incorporate more long term design process activity data

into our prototyped systems, and to conduct empirical studies to analyze how users' sense-making and investigation proceeds through the designed visual interactivity.

Acknowledgements The research presented in this paper is partially supported by JST CREST Grant Number JPMJCR1401 Japan. We would like to thank Nobuto Matsubara for his technical discussions and support.

References

1. Ye Y, Yamamoto Y, Nakakoji K (2008) Expanding the knowing capability of software developers through knowledge collaboration. *IJTPM (Int J Technol, Policy Manag)* 8(1):41–58 (Special Issue on Human Aspects of Information Technology Development, Inderscience Publishers)
2. Mining Software Repositories. <http://www.msrconf.org> (visited 2017-12-01)
3. Chaturvedi KK, Sing VB, Singh P (2013) Tools in mining software repositories. In: Proceedings of 13th international conference on computational science and its applications 2013. Ho Chi Minh City, 2013, pp 89–98
4. Moran TP, Carroll JM (eds) (1996) *Design rationale: concepts, techniques, and use*. CRC Press
5. Schoen DA (1983) *The reflective practitioner: how professionals think in action*. Basic Books, New York
6. Carroll JM (ed) (2013) *Creativity and rationale: enhancing human experience by design*. Human-Computer Interaction Series, Springer, London
7. Rittel HWJ (1972) On the planning crisis: systems analysis of the first and second generations. *Bedriftsokonomien* 8:390–396
8. Conklin J, Begeman M (1988) gIBIS: a hypertext tool for exploratory policy discussion. *Transactions of Office Injonnation Systems* 6(4):303–331
9. Compendium. [<http://www.compendiumng.org/>] (visited 2017–12)
10. Bracewell R, Wallace K (2003) A tool for capturing design rationale. In: Proceedings of international conference on engineering design (ICED'03), Stockholm, Sweden, pp 185–186
11. Kirschner PA, Buckingham Shum SJ, Carr CS (eds) (2003) *Visualizing argumentation: software tools for collaborative and educational sense-making*. Springer, London, UK
12. Hassanaly P, Herrmann T, Kunau G, Zacklad M (eds) (2006) *Cooperative systems design: seamless integration of artifacts and conversations—enhanced concepts of infrastructure for communication*. In: *Frontiers in artificial intelligence and applications*, IOS Press, p 137
13. Fischer G, Grudin J, Lemke AC, McCall R, Ostwald J, Reeves BN, Shipman F (1992) Supporting indirect, collaborative design with integrated knowledge-based design environments. *Human Comput Interact* 7(3):281–314
14. Hisarciklilar O, Boujut J-F (2007) An annotation based approach to support design communication. In: Proceedings of international conference on engineering design (ICED'07). Paris, France, pp 170–180
15. Do EY-L (2011) Sketch that scene for me and meet me in cyberspace. In: Wang X, Tsai JJ-H (eds) *Collaborative design in virtual environments*, ISCA 48. Springer, pp 121–130
16. Rekimoto J (1999) Time-machine computing: a time-centric approach for the information environment. In: Proceedings of the ACM symposium on user interface software and technology (UIST'99), pp 45–54
17. Hill WC, Hollan JD, Wroblewski D, McCandless T (1992) Edit wear and read wear. In: Proceedings of ACM conference on human factors in computing systems (CHI'92), pp 3–9
18. Shirai Y, Yamamoto Y, Nakakoji K (2009) Time-based authoring tools for informal information management. In: Symposium on interactive visual information collections and activity (IVICA2009). Austin, TX, pp 1–6

19. Mangano N, LaToza TD, Petre M, van der Hoek (2014) Supporting informal design with interactive whiteboards. In: Conference on human factors in computing systems (CHI2014), April 2014, pp 331–340
20. Shirai Y, Yamamoto Y, Nakakoji K (2006) A history-centric approach for enhancing web browsing experiences. In: Extended abstracts of conference on human factors in computing systems (CHI2006), pp 1319–1324
21. Girgensohn A, Marlow J, Shipman F, Wilcox L, (2016) Guiding users through asynchronous meeting content with hypervideo playback plans. In: Proceedings of the 27th ACM conference on hypertext and social media (HT'16). ACM, New York, NY, USA, pp 49–59
22. Mattoso M, Glavic B (eds) (2016) Proceedings of 6th international provenance and annotation workshop, IPAW 2016. McLean, VA, USA, June 7–8, 2016; Proceedings, lecture notes in computer science, vol 9672. Springer
23. Silva C, Freire J, Santos E, Anderson E (2010) Provenance-enabled data exploration and visualization with VisTrails. In: 2010 23rd SIBGRAPI conference on Graphics, patterns and images tutorials (SIBGRAPI-T), pp 1–9
24. Freire J, Koop D, Santos E, Silva CT (2008) Provenance for computational tasks: a survey. *Comput Sci Eng* 10(3):11–21
25. Wheat AJ (2015) External representation of provenance in intelligence analysis, In: *Infovis sensemaking workshop*
26. Gotz D, Zhou MX (2009) Characterizing users visual analytic activity for insight provenance. *Inf Visual* 8(1):42–55
27. Nakakoji K, Yamamoto Y, Kita Y (2016) Visual interaction design for experiencing and engaging with a large chronological table. In: Proceedings of the 3rd HistoInformatics workshop on computational history (HistoInformatics 2016). Krakow, Poland, pp 47–51
28. Nakakoji K, Yamamoto Y, Matsubara N, Kawashima T, Hamuro N, Uno T (2017) Visual interactivity for understanding relationships and uncovering correlations among temporal numerical data of thousands of elements. In: Proceedings of the 31st annual conference of the Japanese society for artificial intelligence (JSAI2017), 2D2-4in2. Nagoya, Japan, pp 1–3 (in Japanese)
29. Nakakoji K, Yamamoto Y, Matsubara N, Shirai Y (2012) Toward unweaving streams of thought for reflection in early stages of software design. *IEEE Softw Spec Issue Studying Prof Softw Des* 29(1):34–38
30. Ingold T (2007) *Lines: a brief history*. Routledge

Style-Oriented Evolutionary Design of Architectural Forms Directed by Aesthetic Measure



Agnieszka Mars, Ewa Grabska, Grażyna Ślusarczyk
and Barbara Strug

This paper deals with an aesthetic and style-oriented approach to architectural design based on the combination of three theories—recognition, generation and evaluation. The Biederman's recognition-by-components theory is used to recognize the design structure. An evolutionary algorithm serves as a generative tool. The design evaluation function is based on Birkhoff's aesthetic measure. Phenotypes of architectural objects are seen as configurations of Biederman's basic components essential for visual perception. Genotypes of these objects are represented by graphs with bonds, where nodes represent object components, node bonds represent component surfaces, while graph edges represent relations between surfaces. Graph evolutionary operators, i.e. crossover and mutation, are defined in such a way that they preserve features characteristic for styles. The fitness function is based on Birkhoff's aesthetic measure for polygons adapted for 3D solids. The approach is illustrated by examples of designing objects in the Neoclassical style using the encoded aesthetic evaluation mechanism.

Introduction

Computer-Aided Design (CAD) belongs to well-established research areas. But aesthetic evaluation of architecture in context is very rarely supported by the computer. This topic will be discussed in this paper.

The process of architectural design involves self-expression of the architect and requires a lot of imagination to pass on ideas and values that have social significance. Architectural objects also often refer to cultural and historical context. Therefore, it is difficult to equip computer programs with knowledge needed to imitate human way

A. Mars · E. Grabska · G. Ślusarczyk (✉) · B. Strug
Jagiellonian University, Kraków, Poland
e-mail: grazyna.slusarczyk@uj.edu.pl

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_38

of thinking during the automatic design process. However, living organisms, being products of evolution, disregard human culture and do not fit into the definition of art, but they are still appreciated by people for their beauty comprising functionality and harmony of shapes. Therefore, an evolutionary approach to design gives a chance to imitate to some degree the biological processes in order to obtain objects with high aesthetic values.

As evolutionary search consists of evaluating and refining possible solutions, it is highly analogous to a human design iterative process of analysis, testing and optimization [1]. Similarly to the refinement step in human design, in evolutionary search, designs to be modified are determined according to their evaluation (fitness). The refinement step is often performed not on actual solutions (phenotypes) but on their coded equivalents (genotypes). Yet, in human design, the process is usually directed not only by the desire to obtain an optimal artefact but also such a one that meets certain requirements. Design requirements are often related to styles [2]. In [3], genotypes are used as a representation of architectural layout components in evolutionary design, while the representation of style in design is described in [4].

The aim of this paper is to present a new approach to aesthetic-oriented and style-directed evolutionary design of architectural object prototypes. As an aesthetic evaluation of architectural objects is associated with visual perception, we propose to take as the foundation for automatic assessment of generated models a human perception model. The presented method is based on the Biederman's visual perception model, in which object recognition is assumed to be performed by exploration of component shapes and relations between them. In our approach, architectural object prototypes are generated as configurations of some basic solids. Each prototype has its representation in the form of an attributed composition graph, where nodes denote components, bonds assigned to nodes represent component surfaces, while edges describe spatial relations between these surfaces. Such a representation enables us not only to express geometrical properties of an object but also its attributes (like size, material, etc.).

In genetic algorithms, considered in this paper, all prototypes are represented in two forms: in an encoded graph-based form of genotypes and in the decoded form of phenotypes. During the process of evolution, the graphs representing designs (genotypes) are modified in the result of mutation and crossover operations. After each step of evolution, a new generation of 3D models (phenotypes) is rendered.

Using graph representations of design objects during evolutionary search process requires the adaptation of traditional evolutionary operators as well as defining an appropriate fitness function. Moreover, the desired/required style of the object being designed have to be taken into account. As the graphs selected to be transformed by these operators during the evolution and their structures are not known a priori, the operators must be defined in a way which allows for a dynamic computation of resulting graphs. In our approach, a crossover operation can only exchange subgraphs, while mutation affects local and global attributes as well as the graph structure (by adding or deleting subgraphs). Moreover, a mutation operator is designed in such a way that it allows only changes within a range suitable for a given style of designs. A fitness function is mainly based on the Birkhoff's aesthetic measure for polygons

adapted for 3D solids. Symmetrical and harmonic forms with optimal equilibrium are preferred, however, some elements of chaos, that make the shape more interesting, may occur. It also prefers solutions adhering to the rules of a given style. A selection function prefers objects with higher aesthetic values as it is supposed to imitate natural processes of evolution.

The successive steps of the proposed approach are as follows:

- phenotypes of architectural objects are described as configurations of basic solids (geons),
- each phenotype is represented in the form of an attributed composition graph being its genotype,
- evolutionary algorithms act on genotypes,
- evolutionary operators (crossover and mutation), which are introduced to act on the graphs, are based on stylistic constraints,
- evaluation is performed based on the fitness function, which involves an encoding of the Birkhoff's aesthetic measure and
- individuals with highest scores (best fitted) are chosen for reproduction in the selection process.

The paper is organized as follows. At first, the Recognition-by-Components perception model is explained and phenotypes of architectural objects in the Neoclassical style are presented as configurations of elementary shapes. Then, the representation of objects in the form of attributed composition graphs, which constitute genotypes for the evolutionary algorithm, is presented. Further sections contain the specification of mutation and crossover operators, as well as the fitness and selection functions. The next section presents examples of designing buildings in the Neoclassical style, and, finally, the conclusion is drawn.

Phenotypes

As we do not know how exactly aesthetic evaluation is performed by a human brain, it is very difficult to equip computer tools with proper rules of computing aesthetic values of architectural objects. Because aesthetic evaluation is related to perception, it seems a promising solution to use a human visual perception model in order to assess quality of phenotypes in an evolutionary algorithm focused on aesthetic values. There are two main perception theories. The view-dependent model concentrates on recognition based on memorized views of an object—identification occurs when the most similar view is found. The view-independent model assumes that object recognition is performed by dividing a perceived form into basic components and exploring their shapes and relations between them. Although probably both models take part in human perception, the second one appears to be more appropriate for the purpose of computational design [5].

In our approach, an alphabet of elementary shapes is used to construct phenotypes of architectural objects. This enables us to analyze properties of object components

and relations between them and on this basis specify the fitness function, which prefers such hard-to-define elements of beauty like order, harmony, rhythm, coherence, etc.

Recognition-By-Components

The Recognition-By-Components (RBC) theory was developed by Biederman [6] on the basis of the Marr's view-independent perception model [7]. RBC assumes that most objects can be divided into elementary shapes called geons, characterized by lack of sharp concavities. Biederman distinguished 36 geon types, each of them defined by four non-accidental properties: cross-section edges (curved or straight), cross-section size change (constant, contract or expand and contract), cross-section symmetry (vertical, vertical and rotational, or none) and axis type (straight or curved). These properties are easy to recognize independently of the point of view and cost of their perception is low. Except that, each geon can be also described by a set of metric properties, like size or location, which take longer time to be processed and perception of them is prone to errors.

Figure 1 presents exemplary geon types with different non-accidental properties. In Fig. 1a, the solid has straight edges, vertical and rotational cross-section symmetry, the constant cross-section size and the straight axis. Figure 1b shows a geon with curved edges, vertical and rotational symmetry, the contracting cross-section size and the straight axis. Finally, the geon presented in Fig. 1c has straight edges, no symmetry, the constant cross-section size and the curved axis.

Object recognition is performed by the analysis of geon types that compose an object, and exploration of relations between geons. There are two main types of relations recognizable independently of the point of view: the end-to-end relation, in which two geons have a common surface (Fig. 2a), and the end-to-side relation, in which a surface of one geon is attached to the larger surface of another geon (Fig. 2b). For the purpose of computer-aided architectural design, we have introduced another type of relation, overlap, which simplifies the object structural representation. For

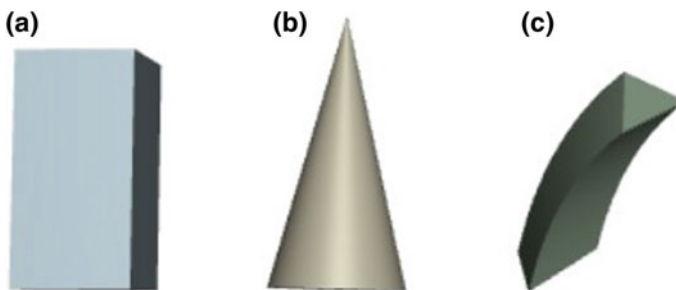


Fig. 1 Exemplary geons

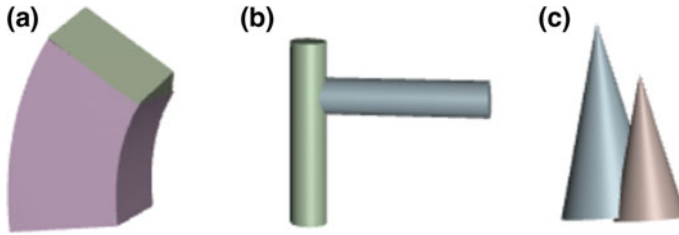


Fig. 2 Relations between geons: **a** end-to-end, **b** end-to-side, **c** overlap

example, we can assume that Fig. 2c presents a solid composed of two overlapping cones instead of one cone and one quite complex solid, which is a cone without one bottom corner.

It is worth noticing that the geon theory is one of the most controversial theories of identifying structured objects. Few researchers believe that the brain contains mechanism to exactly identify the set of three-dimensional geons proposed by Biederman. However, there is a strong evidence that we perceive the way components of objects are interconnected forming the structural skeleton [8].

Using geons as basic primitives offers at least two advantages. As geons are based on object properties that are viewpoint independent, a single geon description describes an object from all possible viewpoints. Moreover, a relatively small set of geons forms an alphabet of shapes that can be combined to form complex objects, so the representation is efficient. On the other hand, using RBC theory it is very difficult to produce a geons-and-relations description of a real object, as it does not provide a mechanism to reduce the complexities of real objects to geon shapes.

Style Representation

In the proposed approach, phenotypes of design buildings are composed of geons. It is assumed that in most cases aesthetic evaluation is based on non-accidental properties, i.e. only geon types are taken into account, disregarding metric information. Therefore, the most important part of the phenotype description constitutes non-accidental attributes and relation types, although metric parameters are of course necessary to visualize a designed object.

Our approach to evolutionary design is style-oriented. It is known that “an architectural style is characterized by the features that make a building or other structure notable and historically identifiable”. In this paper, we consider style in the context of building forms. It means that we aim at the automatic design process which preserves a set of elements characteristic for the form in a given style. It should be noted that in many cases these elements must be arranged in some predefined way [9, 10]. A method for classifying artefacts as belonging to the same style, which is based on the degree of their similarity assessed using Self Organizing Maps (SOM) networks,

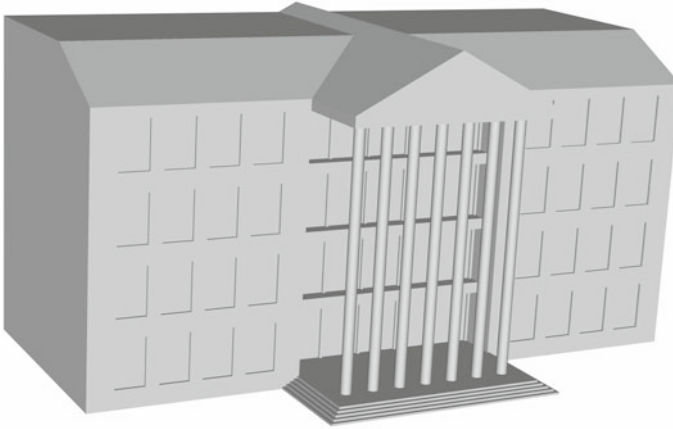


Fig. 3 An example of a building in a Neoclassical style

is developed in [2]. Other models of style, which focus on distance measures to compare objects, are described in [11–14].

In this paper, we formalize rules related to the composition of elements determining the Neoclassical style. Neoclassical architecture was an architectural style that began in the mid-eighteenth century. It was a reaction against the Rococo style of naturalistic ornament. Neoclassicism draws inspiration from the architecture of Classical antiquity, the Vitruvian principles and the architecture of the Italian architect Andrea Palladio. This style is characterized by removal of all unnecessary ornamentation, enhancement of geometric forms, and repeating series of arches and/or columns. There exist three types of neoclassical architecture: temple style based on an ancient temple, Palladian style and classical block style [15]. We consider Neoclassicism, as objects in this style can be represented by compositions of distinct forms, which are arranged using an explicit set of rules. It is assumed here that a building is in the Neoclassical style if it has a façade with a porch having at least two columns and exactly one pediment [16].

An example building which is shown in Fig. 3 has all characteristic features of the classical block style.

Genotypes

The design process can be modelled by a search process. As evolutionary search consists of evaluating and refining possible solutions it can be seen as analogous to a human design iterative process of analysis, testing and optimization [1, 17, 18]. In many types of evolutionary search, the refinement step is often performed not on actual phenotypes, but on their coded equivalents called genotypes.

The proposed approach uses composition graphs with bonds [19] as the representation of design object phenotypes. Graph nodes represent components (parts) of the object being designed, bonds assigned to nodes represent component surfaces, while directed edges connecting bonds express relations between these surfaces. Nodes are labelled by names of components (or types of components) and edges are labelled by names of relations between them.

In case of designing buildings, graphs encode building genotypes. The graph nodes are labelled by the names of the building elements represented by geons. To each node, two groups of attributes are assigned. The attributes of the first group describe non-accidental properties of geons, while attributes of the second group describe their metric parameters. Node bonds represent types of geon surfaces and their number varies depending on the cross-section shape. The edges labelled *e-to-e*

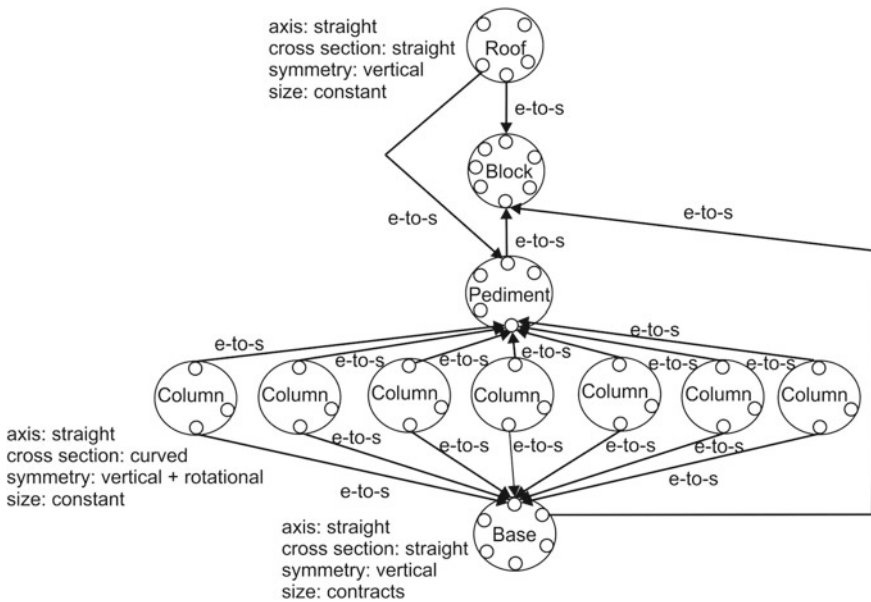


Fig. 4 A composition graph representation of a building from Fig. 3

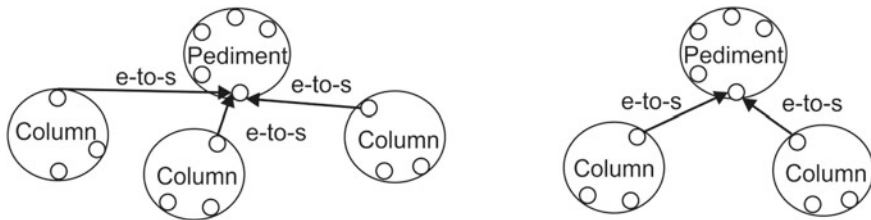


Fig. 5 Examples of composition graphs representing style requirements

represent the end-to-end relation, the edges labelled *e-to-s* represent the end-to-side relation, while the third type of relation is represented by the edges labelled *overlap*.

Figure 4 presents a composition graph representing the building phenotype shown in Fig. 3. The nodes labelled *Block* and *Base* represent cuboids, nodes labelled *Roof* and *Pediment* represent prisms, while nodes labelled *Column* represent cylinders. All edges are labelled *e-to-s*. They connect bonds representing the basis of the prism and the top basis of the bigger cuboid, the front side of the prism and the slanted side of this cuboid with the back side of the prism. They connect also bonds representing the back side of the small cuboid and the front side of the bigger cuboid, as well as top bases of all cylinders and the base of the prism, and bottom bases of all cylinders and the top basis of the small cuboid. To each node, a set of attributes is assigned. The non-accidental attributes for the cuboid, the prism, and the cylinder are shown.

As design objects are represented by composition graphs, also style requirements should be represented by composition (sub)graphs [20, 21]. Style requirements are specified by the designer on the basis of his background knowledge. For example, if the designer wants a building to be in the Neoclassical style the main building block must be connected with a porch composed of a pediment and at least two columns.

In Fig. 5, two examples of composition graphs representing Neoclassical style requirements are presented. The first one corresponds to a porch with three columns, while the second one represents a porch with two columns.

Evolutionary Operators on Graphs

As in this paper, an evolutionary algorithm acts on designed objects genotypes in the form of composition graphs, the genetic operators for such graphs have to be defined. The graph-based equivalent of a standard crossover operator requires establishing subgraphs that would be exchanged during the process of evolution. When a crossover is performed on two selected graphs, G and H , the subgraphs g and h , respectively, are selected in these graphs. Then each subgraph is removed from a graph and inserted into the second one. As a result, two new graphs are generated. However, there may exist edges connecting nodes belonging to a chosen subgraph with nodes which do not belong to it. Such edges are called embedding of a subgraph. So removing a subgraph from a graph and placing it in another one requires a method allowing for proper reconnection of these edges. The underlying idea is that all edges should be reconnected to nodes similar to those they were connected to in the graph from which they were removed.

There is probably more than one possibility of defining similarity of nodes. In this paper, a similarity-like relation is used. Its definition is based upon the assumption that graphs selected for crossover code designs consisting of parts with similar or even identical functions (even if these parts have a different internal structure, material or/and geometrical properties). Thus, we can define the similarity on the basis of the node and edge labels.

It is important to notice, however, that the graphs to be crossed over and their respective subgraphs are selected during the execution of the evolutionary algorithms, so the embedding transformations cannot be defined a priori (as it is in graph grammars [22]). The idea behind the algorithm that generates automatically such an embedding transformation is to preserve the relations between the nodes as much as possible, i.e. to connect each edge removed from one graph to a node in the second graph that represents the same or similar object (i.e. has the same label) [23].

In addition to dealing with the graph embedding problem, in case of using the evolutionary process to generate designs adhering to a particular style an additional step must be performed. During the process of selecting subgraphs in graphs to be crossed over it is possible that the patterns representing style requirements will be broken. As the result, new graphs generated by the genetic operator could not represent designs in a required style. To prevent such a situation, we introduce the notion of an unbreakable subgraph. An unbreakable subgraph is a subgraph which represents a predefined requirement, for example, a style component. At the outset of a design process, a set of unbreakable subgraphs associated with a given style is specified. Then, in each graph G representing a design, all unbreakable subgraphs are found and stored together with their position in the design in a set B_G .

After selecting two graphs G and H to be crossed over, its subgraphs g and h are selected. In the first step, a starting node v is selected in graph G and a similar node w is selected in graph H . Each of these nodes represents a geon located on the ground (a ground geon), which is indicated by a metric attribute defining location of its bottom basis. Then two numbers, i and j , are randomly chosen for the size of the subgraphs in both G and H . Then in graph G starting from node v we select adjacent nodes until the subgraph built reaches i nodes. Each time a node x is selected in G to be added to the subgraph g it is checked against the set B_G to verify if it belongs to any of the unbreakable patterns. If no, it is added to subgraph g and the selection of the subsequent node is performed. If node x belongs to some pattern either the whole pattern has to be added to subgraph g being generated or none of its nodes. This decision is based on the size of the subgraph. If adding the whole pattern would not exceed the selected size i of subgraph g it can be added, otherwise node x is not added to subgraph g and the selection process is continued. If the whole pattern is added to g , it is also added to the set of unbreakable patterns B_g associated with g and removed from set B_G associated with G . Similarly in graph H a subgraph is built starting from node w . As a result, we obtain two subgraphs, g and h , and at the same time two sets of unbreakable patterns B_g and B_h .

Formally, a crossover operator cx is defined as a 6-tuple (G, H, g, h, T, U) , where G, H, g, h are graphs and their subgraphs, respectively. So we have $G=(V_G, E_G)$, $H=(V_H, E_H)$, $g=(V_g, E_g)$, $h=(V_h, E_h)$, where $V_g \subset V_G$, $V_h \subset V_H$, $E_g \subset E_G$, $E_h \subset E_H$. The crucial elements of this operator are T and U that are called embedding transformations, i.e., they describe how edges of the embedding are to be reconnected. They are sets of pairs of the form (n, n') , where n denotes a node to which an edge was assigned originally and n' —the one to which it will be assigned in a new graph.

As the result of the crossover, we obtain two graphs G' and H' . Graph G' is constructed in such a way that it contains all nodes and edges remaining from G after

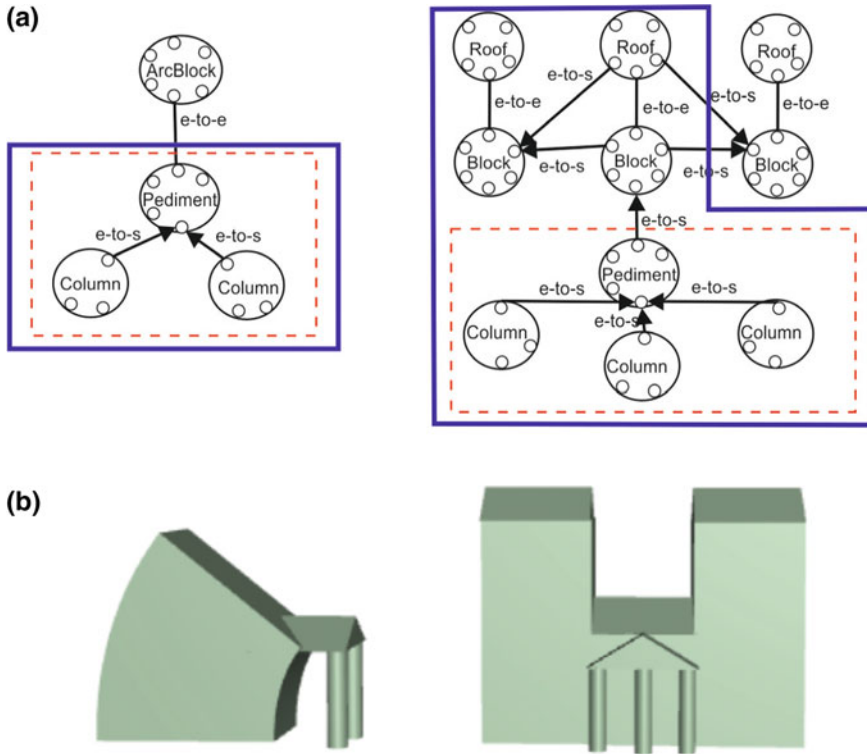


Fig. 6 **a** Two composition graphs chosen for reproduction and **b** their corresponding building phenotypes

removing g , all nodes and edges from h and edges connecting nodes from $G-g$ and h obtained by applying transformation T . Thus, we have graphs $G' = (V_G - V_g \cup V_h, E_G - E_g - Emb(g, G) \cup E_h \cup E_T)$ and $H' = (V_H - V_h \cup V_g, E_H - E_h - Emb(h, H) \cup E_g \cup E_U)$, where $Emb(x, X)$ is the set of all edges having one end in set x and another one in set X , that is the embedding of the subgraph x in the graph X . Sets E_T and E_U represent sets of new edges generated by the application of the embedding transformations U and T , respectively.

Moreover, the set $B_{G'}$ is obtained by summing sets B_G and B_h . In an identical way graph H' is constructed from $H-h$ and g and set $B_{H'}$ from sets B_H and B_g .

In Fig. 6a, two composition graphs chosen for reproduction are depicted. On both of them, style patterns are marked by the red-dashed line and subgraphs g and h selected for the crossover operation are marked with the thick blue line. The prototype buildings in the Neoclassical style represented by these graphs are shown in Fig. 6b. In both composition graphs, the selected subgraphs contain one of the unbreakable patterns. After the crossover, two new graphs are constructed according to the method described above.

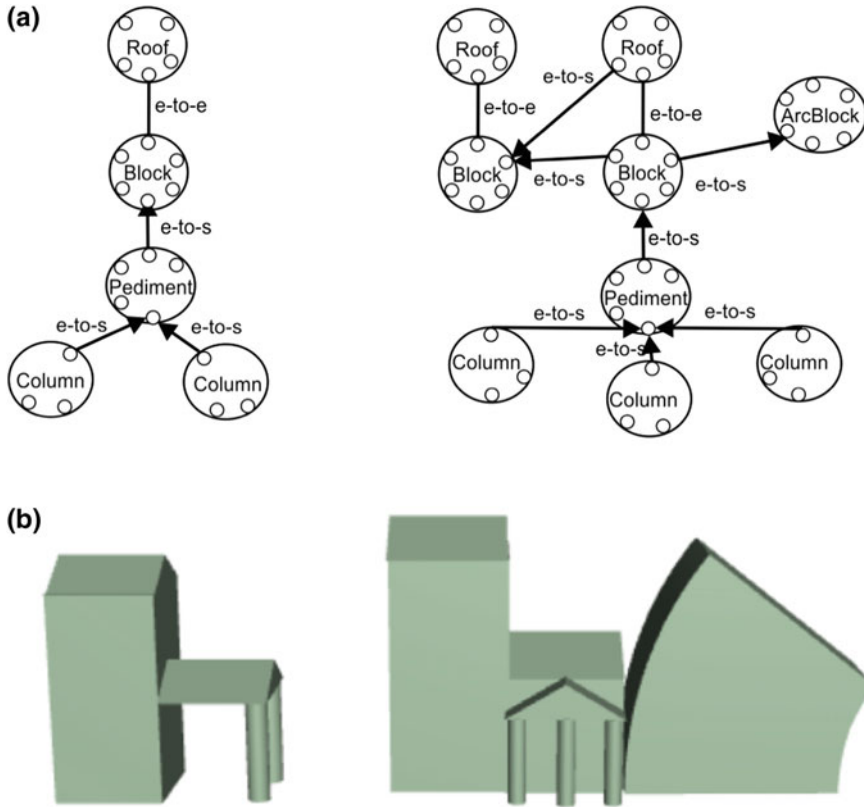


Fig. 7 **a** Two composition graphs obtained after the crossover operation on graphs from Fig. 6 and **b** their corresponding building phenotypes

The proposed algorithm starts with a population of individuals in the Neoclassical style. After that, the evaluation is performed by the fitness function and individuals with highest scores are chosen for reproduction. The chosen individuals are randomly paired. Each pair produces two children with the use of the crossover operator. In random cases, the mutation operator modifies the child. The new population is created from the reproducing pairs and their children, and the whole process starts again. The number of iterations, the size of the initial population and the number of selected individuals are defined by the user.

Building phenotypes represented by composition graphs obtained after the crossover operation on composition graphs shown in Fig. 6a are presented in Fig. 7.

In order to introduce new features to the population, the evolutionary algorithm uses a mutation operator. In this paper, the mutation operator can modify the graph structure by deleting and adding nodes, the relation type or the value of the attribute. Genotypes of individuals are slightly modified by changing the value of a random



Fig. 8 Examples of mutation in one of the buildings from Fig. 7

attribute or a random relation type, or by adding a random node. Beneficial mutations have a chance to be copied into the next generations.

Two examples of buildings obtained as the result of the mutation operator applied to the building shown on the right-hand side of Fig. 7 are shown in Fig. 8. In the first case, one cuboid and one prism have been removed, while in the second one the shape of a cuboid has been modified by changing the axis attribute from straight to a curved one.

Fitness Function

The best-fitted individuals are chosen for reproduction during the selection process. In this paper, the fitness function encodes a sense of aesthetics. Moreover, this function takes into account adherence of design objects to the specified style. As we have sets of patterns associated with newly generated graphs we can easily verify if each of the sets contains all required patterns. Thus we are able to evaluate the degree in which style requirements are fulfilled by calculating the percentage of patterns present. The individuals which do not have any style characteristic features are removed from the population.

Fitness function evaluates also to what degree each phenotype fulfills aesthetic criteria of an architectural object related to the Neoclassical style. This evaluation is performed on the basis of Birkhoff aesthetic measure for polygons adapted for 3D solids [24]. Human sense of aesthetics correlates with our urge for gathering information about environment. Therefore, presence of some kind of order increases aesthetic quality of an object, however, a highly ordered structure may not deliver enough information, as it can be too predictable. It is essential then to ensure optimal balance between the new and the ordered.

To obtain this result we construct a fitness function that rewards the following:

1. equilibrium,
2. elements of the Neoclassical style,
3. every relation of order, i.e., symmetry and alignment to the same plane,

4. every geon in the relation of order,
5. every geon type provided that the number of geon types does not exceed a critical value.

The first condition concerns both aesthetic and functional requirement of an architectural object and enables us to obtain a prototype that is possible to be built. The second condition ensures that the building prototype refers to the Neoclassical style. In result of the third condition, objects with more different relations of order are preferred, which enables novelty, as not every geon of the solid is arranged in the same way as the others. The fourth condition values relations containing high number of geons, which decreases chaos. Finally, the fifth condition ensures diversity of components and at the same time prevents confusion, inevitable when an object consists of too many different elements.

The algorithm is designed to generate phenotypes which are well balanced and have some relations of order (like symmetry or alignment to the same plane).

Let us define the following components of the fitness function:

- equilibrium $E \in \{0, 1\}$,
- number of Neoclassical patterns $S \in N$,
- alignment $A = \sum_{i=1}^{n_A} k_i^2$ and $k_i, 2 \leq i \leq n_A$, is a number of geons aligned to one plane or line, where n_A is a number of relations of alignment between object's geons (i.e., a number of different planes and lines geons are aligned to),
- vertical symmetry $V = \sum_{i=1}^{n_V} k_i^2$ and $k_i, 1 \leq i \leq n_V$, is a number of geons composing one vertically symmetrical part of an object, where n_V is a number of relations of vertical symmetry between object's geons (i.e., a number of different vertical axes of symmetry in an object),
- rotational symmetry $R = \sum_{i=1}^{n_R} k_i^2$, and $k_i, 1 \leq i \leq n_R$ is a number of geons composing one rotationally symmetrical part of an object, where n_R is a number of relations of rotational symmetry between object's geons (i.e., a number of different axes of rotational symmetry in an object),
- geon types ratio $T = \begin{cases} t, & t \leq p \\ p - t, & t > p \end{cases}$, where t is a number of geon types and p is a number of allowed geon types,
- number of geons $G=n$, where n is a number of geons an object is built of.

The fitness function of the proposed algorithm can then use aesthetic measure defined by the formula:

$$M = E \cdot S \cdot \frac{A + V + R + T}{G}$$

Figure 9 presents some results of computing the fitness function for exemplary buildings. The number of allowed geon types has been set to 3. Figures 9a, b present buildings lacking equilibrium or Neoclassical style elements, respectively, which results in 0 value of the aesthetic measure ($M=0$). The building in Fig. 9d contains fewer geons with the vertical symmetry relation and exceeds the number of allowed

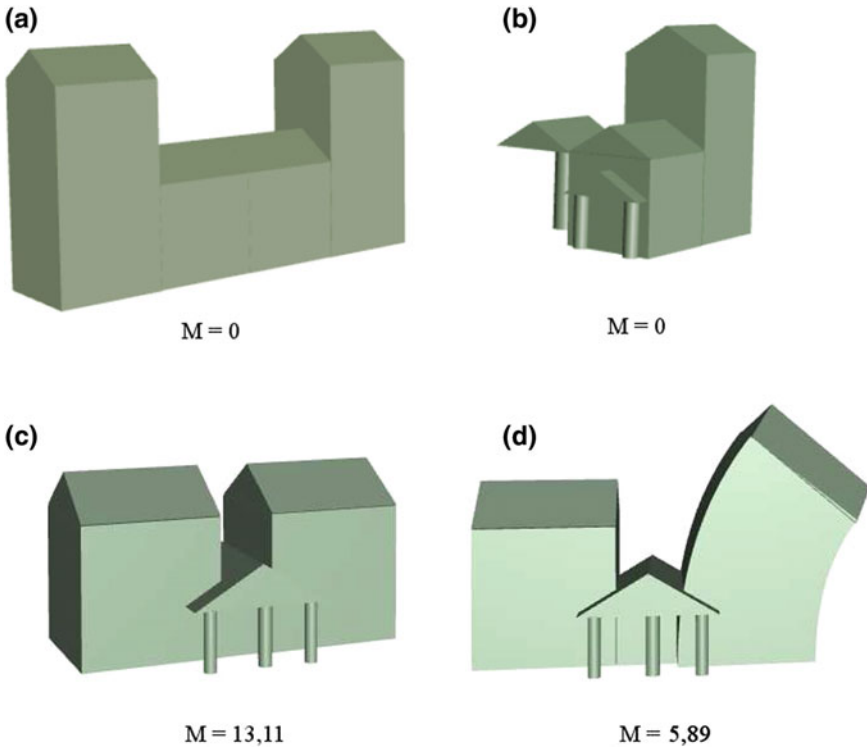


Fig. 9 Exemplary buildings evaluated by the fitness function

geon types, so its grade ($M=5.89$) is lower than the grade of the building presented in Fig. 9c ($M=13.11$).

Conclusion

The aim of this paper is to present a new approach to style-oriented creative design evaluated according to aesthetic criteria. This approach is based on the combination of three theories—recognition, generation and evaluation. The Biederman's recognition-by-components theory is used to recognize the design structure. An evolutionary algorithm is proposed as a generative tool, which enables obtaining the variability of design structures enhancing creativity of the design process. The design evaluation is defined by means of the fitness function of this algorithm and based on Birkhoff's aesthetic measure.

Aesthetic evaluation is strictly connected with visual perception. Therefore, it seems reasonable to combine aesthetic measure with an object recognition model in a generative tool that complies with aesthetic rules. RBC theory gives an opportunity

to examine components and relations between them. Birkhoff's aesthetic measure adapted for 3D objects matches well with such an approach, as it is based on similar examination, although it obviously does not cover every aspect of human aesthetic sense.

An evolutionary process has been used to stimulate the creativity of the designer and to suggest an aesthetic evaluation mechanism for architectural object prototypes encoded in the fitness function. A design structure of an architectonic object has been determined on the basis of the Biedermann's structural skeleton of the object. This kind of higher level structural analysis is a way of making clear genotype representations in the form of graphs.

As in the proposed approach, a composition graph is used as a genotype, equivalents of standard genetic operators are defined on graphs. These operators are more complex than standard binary ones but they provide us with benefits like the possibility of coding relationships between components of an artefact and ability to introduce structural changes which compensate for it. The strongest point of a graph-based representation is its ability to represent in a uniform way all types of relations and objects and to preserve some required characteristics of the design.

In this paper, such characteristics are related to the style of the object but in future, we plan to investigate the possibility of applying such an approach to other features. It could be used to preserve some parts of the design that is considered optimal and allow only for the improvement of the remaining parts of the design. It is also possible to use this approach to assure the presence of a predefined number of some components within a designed object.

Another direction for future research is related to defining the strength of unbreakability of patterns. In this paper, none of the patterns designated as unbreakable can be broken. Yet, it can be observed that in some situations it is possible that a given pattern is present in a graph multiple times thus breaking one of the occurrences would still allow for the required style to be preserved but at the same time give possibly more freedom for creative results.

It should be noted that the proposed method can be used also to create architectural objects in different styles, as the proposed evolutionary operations can be easily modified to include other design requirements. Moreover, it can be used to support creative engineering design of machines or products, represented as compositions of geons, as an evolutionary process generates variability of design structures which can be evaluated according to aesthetic criteria.

References

1. Bentley PJ (1999) Evolutionary design by computers. Morgan Kaufmann
2. Jupp J, Gero JS (2010) Let's look at style: visual and spatial representation and reasoning in design. In: Argamon S, Burns K, Dubnov S (eds) The structure of style. Springer, New York, pp 159–195
3. Schnier T, Gero JS (1996) Learning representations for creative design using evolution. *AIEDAM* 10:175–177

4. Ding L, Gero JS (2001) The emergence of the representation of style in design. *Env Plann B Urban Analytics City Sci* 28:707–731
5. Wallendorf M, Zinkhan G, Zinkhan LS (1981) Cognitive complexity and aesthetic preference. In: Hirschman EC, Holbrook MB (eds) *SV—symbolic consumer behaviour*. Association for Consumer Research, New York, pp 52–59
6. Biederman I (1987) Recognition-by-components: a theory of human image understanding. *Psychol Rev* 94:115–147
7. Marr D (1982) *Vision*. Freeman, San Francisco
8. Ware C (2008) *Visual thinking for design*. Morgan Kaufmann, Elsevier
9. Alexander C, Ishikawa S, Silverstein M, Jacobson M, Fiksdahl-King I, Angel S (1977) *A pattern language: towns, buildings, construction*. Oxford University Press
10. Simon H (1975) *Style in design*. In: Eastman C (ed) *Spatial synthesis in computer-aided building design*. Applied Science, London, pp 287–309
11. Park SH, Gero JS (2000) Categorisation of shapes using shape features. In: Gero JS (ed) *Artificial intelligence in design'00*. Kluwer, Dordrecht, pp 203–223
12. Davies J, Goel AK (2001) Visual analogy in problem solving. *Proceedings of international joint conference on artificial intelligence*. Morgan Kaufmann, Seattle, pp 377–382
13. Gero JS, Kazakov V (2001) Entropic similarity and complexity measures for architectural drawings. *Visual and spatial reasoning in design II*. Key Cent Des Comput Cogn Sydney:147–161
14. Forbus K, Tomai E, Usher J (2003) Qualitative spatial reasoning for visual grouping in sketches. In: *Proceedings of 16th international workshop on qualitative reasoning*, Brasilia, pp 79–86
15. Dowling E (2004) *New classicism: the rebirth of traditional architecture*. Rizzoli International Publications, New York
16. Dunlop C, Associates (2003) *Architectural styles*. Dearborn Real Estate
17. Goldberg DE (1989) *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, MA
18. Holland JH (1975) *Adaptation in natural and artificial systems*. Ann Arbor
19. Grabska E, Borkowski A (1996) Assisting creativity by composite representation. In: *Artificial intelligence in design '96*, Kluwer Academic Publishers, pp 743–759
20. Ślusarczyk G, Strug B, Stasiak K (2016) An ontology-based graph approach to support buildings design conformity with a given style. *Appl Ontol* 11:279–300
21. Strug B, Ślusarczyk G, Grabska E (2017) Design patterns in generation of artefacts in required styles. In: *Generative art 2016: proceedings of XIX generative art conference, 2016*, pp 71–78
22. Rozenberg G (1997) *Handbook of graph grammars and computing by graph transformations*, vol 1. Foundations, World Scientific London
23. Strug B, Grabska E, Ślusarczyk G (2014) Supporting the design process with hypergraph genetic operators. *Adv Eng Inform* 28:11–27
24. Mars A, Grabska E (2015) Towards an implementable aesthetic measure for collaborative architecture design. *LNCS* 9320:72–75

Creative Sketching Apprentice: Supporting Conceptual Shifts in Sketch Ideation



Pegah Karimi, Kazjon Grace, Nicholas Davis
and Mary Lou Maher

Sketching in design is typically a part of the ideation process. A common occurrence in sketching creativity is the *conceptual shift*, or when a drawn object is reinterpreted as belonging to a different object category. Conceptual shifts are known to be an instrumental component of some creative processes. This paper presents the Creative Sketching Apprentice (CSA) that reinterprets sketches drawn by a user and proposes conceptual shifts, and the paper describes recent work in the field of augmented sketching, including the Drawing Apprentice, an earlier version of the CSA. The contribution of this paper is a computational approach using a convolutional neural network model for classifying sketched objects to identify potential conceptual shifts. This approach is described as the Creative Sketching Apprentice and demonstrated through a case study that illustrates conceptual shifts in a design context. The paper concludes with a discussion of the implications of this approach for a co-creativity system in design sketching.

Introduction

Sketching is a critical part of the early stages of the design process, facilitating ideation and the exploration of conceptual designs [1]. Digital sketching tools have been introduced as a method for augmenting and supporting the sketching process. These tools offer a variety of functions that allow designers to rapidly explore alternatives, share their digital sketches, and even suggest new and related ideas to facilitate creativity [2].

P. Karimi (✉) · N. Davis · M. L. Maher
University of North Carolina at Charlotte, Charlotte, USA
e-mail: pkarimi@uncc.edu

K. Grace
The University of Sydney, Sydney, Australia

© Springer Nature Switzerland AG 2019
J. S. Gero (ed.), *Design Computing and Cognition '18*,
https://doi.org/10.1007/978-3-030-05363-5_39

Intelligent systems that collaborate with designers on creative tasks are referred to as computational co-creative systems, or often just “co-creativity”. These systems contribute to a shared creative artifact with users in a process that can support and inspire user creativity. In this, they share many qualities with human-to-human collaboration. During co-creation, ideas and contributions originate both from the human designer as well as the co-creative system itself. These ideas can mix, combine, and synthesize in unexpected and serendipitous ways, which can change the overall trajectory of the creative artifact and produce novel variations that help users discover previously unexplored aspects of the creative space.

Co-creative systems have been developed for a number of creative domains, in both art and design domains, including music [3, 4], dance [5], poetry [6], and drawing [7, 8]. In this paper, we develop a prototype of a component of a co-creative system based on conceptual shifts. We hypothesize that our conceptual shift-based approach will be beneficial to co-creative systems in design sketching, and are developing a prototype system called the Creative Sketching Apprentice (CSA) to explore this. We intend for the CSA to a co-creative sketching partner that collaborates with the user in real time to create a sketch. The system described in this paper is a prototype of the CSA’s capacity to diversely contribute to the user’s current sketches by identifying potential conceptual shifts. Future work will explore how these shifts can be pursued by the CSA, and how that act can be embedded in an interactive system.

The work reported here is an interdisciplinary effort nestled between the fields of computational creativity and creativity support tools. Computational creativity is a field of artificial intelligence focused on developing agents that generate creative products autonomously [9–12]. Creativity support tools, on the other hand, are technologies designed to enhance and augment the user’s creativity, typically aiming to improve the quality of the final product [13–16]. Computers can adopt a variety of roles when supporting human creativity, including acting as a nanny, coach, pen pal, and colleague [17]. The CSA can be considered a computational colleague that designs alongside users.

This paper introduces the idea of identifying potential conceptual shifts into the framework of human–computer co-creation. Conceptual shifts occur in the creative process when elements on the canvas that were originally interpreted as one concept are perceived as another concept. Identifying conceptual shifts involves viewing what has been drawn through a new conceptual lens. Identifying and capitalizing on conceptual shifts is an important part of the creative process as they involve reinterpreting input in a new context, category, or domain. In this paper, we describe the ability of the CSA to create conceptual shifts by reinterpreting objects sketched by the user into another object category. We hypothesize that, embedded into an appropriate co-creative system, this ability could increase the fluency and flexibility of creative ideation for the user. We envisage that this would occur by prompting them to reframe the emerging design, and by encouraging multiple interpretations. We also explore the implications of this new type of co-creative system for creativity and collaboration. The CSA’s precursor, the Drawing Apprentice will be described to provide context for our current contribution, but see [18] for a full technical description of it.

The structure of the paper is as follows. We will first consider related work in three areas: creativity support tools, co-creative systems, and conceptual shifts in designing. Then, we describe the Drawing Apprentice system and its user experience. Next, we describe the new conceptual shift algorithms of the CSA. Finally, we explore the creative and collaborative implications of utilizing conceptual shifts in a co-creative drawing context.

The Role of Conceptual Shifts in Design

Design is fundamentally nonlinear. It is comprised of iterative framing and reframing, not a linear sequence leading from framing to concept to details [19–22]. A critical component of this process is *reflection-in-action*, which Schön [23] uses to describe in-the-moment reinterpretation and metacognition. Reflection-in-action has been studied in the context of design sketching, where it has been shown to lead to a cyclical process of externalizing (sketching) and internalizing (perceiving) that yields unintended yet valuable consequences [24]. Computational models of how and why this process occurs remain an area of active research, but it is known that this reframing can be triggered by surprise. It is proposed that an unexpected discovery about the emerging design solution violates the designer’s expectations, triggering reframing [25, 26].

In this paper, we explore how a computational sketching aid might trigger reflection-in-action. We hypothesize that this might cause designers to productively reframe or reinterpret their design concepts. Our strategy for doing so is to algorithmically induce what we call *conceptual shifts*: reinterpreting an object belonging to one category to be an object belonging to another. Ambiguity in sketching is known to aid exploration [27], and it has been argued that design activity can be represented as a sequence of dynamically constructed situations, with each adopting a different perspective on ambiguous or incomplete information [28].

We hypothesize that presenting users with potential conceptual shifts based on the visual structure of their sketches will lead to more frequent reflection-in-action, facilitate serendipitous and surprising discoveries, and lessen the occurrence of design fixation. This paper presents a first step toward supporting that hypothesis: a computational model for identifying conceptual shifts. For the purposes of development and validation, we frame these conceptual shifts as undirected turn-based sketching between designer and algorithm.

To explain conceptual shifts, we first define two kinds of similarity: *conceptual* (i.e., *semantic*) and *structural* (i.e., *syntactic*). Two sketches can express a similar concept while being structurally very distinct. An idea can be explored in multiple perspectives, yielding structurally distinct representations of a single concept. One may generate multiple sketches exploring different versions of the same concept, such as many different styles of chairs. The features of those chairs change in each design, but the concept of ‘chair’ is common across all of them. Conversely, two

structurally similar sketches can embody different concepts. For example, a sketch of a chair may visually, i.e., structurally, resemble a ladder.

These two dimensions of similarity represent distinct methods of introducing novelty into a sketching session. Co-creative systems can utilize these dimensions of similarity to generate novel sketches that are either conceptually or structurally relevant to the user's current sketching activity. The original Drawing Apprentice performed mimicry by generating sketches of the same concept with structurally distinct qualities. It also introduced new objects that had a semantic relationship to the input object, changing both the concept and structure from the user's input (e.g., introducing a flower if a tree was drawn). These types of contributions helped generate new or related ideas, respectively.

The conceptual shift algorithms introduced in this paper reinterpret an object by changing its category while holding its structure mostly constant. We hypothesize that incorporating this type of conceptual change in a co-creative sketching environment could encourage the designer to explore related but distinct notions while sketching. Our proposed mechanism for this is that conceptual shifts may create associations between disparate concepts which may encourage analogical reasoning.

The Drawing Apprentice System

The Drawing Apprentice (DA) is a co-creative drawing partner that analyzes the user's sketch input and responds with contributions of its own in real time [18]. This section describes the user experience and basic technical details of the Drawing Apprentice functions as it will be extended using the algorithms proposed later in this paper. The extended version of the Drawing Apprentice is referred to as the Creative Sketching Apprentice.

To explain the functionality of the DA, we begin by describing the user interface and its features. The interface has three main components: a palette of functions to control the agent, a palette of drawing tools, and a shared canvas. The agent palette contains buttons for controlling the five drawing modes (top section of Fig. 1), voting buttons for providing feedback (top middle of Fig. 1), and the "home base" of the character icon representing the co-creative agent (top right in Fig. 1). The agent's interpretations of the user's drawing activities appear as a speech bubble in this home-base. The drawing palette consists of functions traditionally associated with drawing applications, such as selecting a color, line thickness, saving the image, and starting a new canvas.

After the agent finishes its turn, the user can provide feedback via voting buttons to inform the agent whether the user liked its contribution. This voting information is used to learn the aesthetic preferences of users and fine tune what types of contributions the system would make by using a Q-Learning algorithm described in our previous paper [18]. From the perspective of optimizing the machine learning algorithms, the user should be required to provide feedback every time the system acts. However, to maintain the flow of the creative experience, users are never required

Fig. 1 Drawing apprentice interface



to provide feedback, but instead given the choice to vote whenever it occurs to them. While more naturalistic, this scheme makes the voting process highly variable between users. Early versions of the DA system had individual user accounts, but in practice, it was more effective to maintain one account that had votes from multiple users to train the system to understand what types of objects are appropriate together.

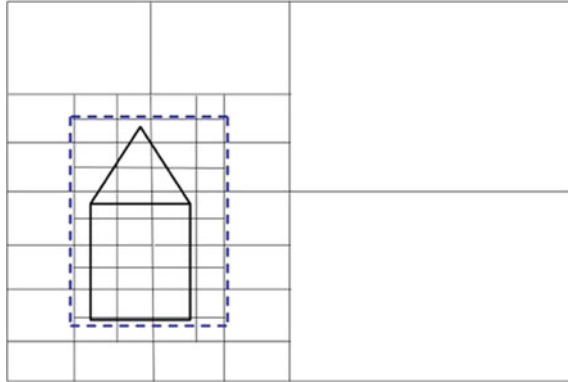
Turn-taking was designed to facilitate emergent interaction dynamics, meaning the number and length of the agent's lines are dependent upon the user's recent contributions. As soon as the user ends their current line, a timer begins. If this timer passes the arbitrary value of 2 s before the user begins their next line, their turn ends, and the system starts to draw. The system's turn will be approximately the same number of lines as the user's turn to mirror the interaction. However, the user may begin drawing at any point, which can lead to synchronous collaboration.

Pilot studies revealed the importance of having a character represent the Drawing Apprentice on the canvas. To simulate the dynamism and embodied nature of real-time human collaboration, the Drawing Apprentice character draws lines dynamically, meaning lines do not appear at once in full, but are gradually animated through until their completion. Dynamic line drawing is meant to provide a sense that the system is going through the embodied act of creating a line.

The drawing modes can be divided into two functional categories: responding to individual lines, and responding to groups of lines. When a drawing mode is selected, the image on the button animates to provide a prototypical demonstration of what the drawing mode entails to help the user understand what to expect from the system. The first three drawing modes all respond to individual lines by either (1) *tracing* the user input, (2) *transforming* user input (e.g., scaling, rotating, translating), or (3) *mimicking* user input by stretching and skewing it. This category of drawing modes was designed primarily for use in abstract (i.e., nonrealistic and (potentially) nonrepresentational) drawing to provide users with novel input to stimulate ideas. Findings from user studies [18] indicated an expectation that the system would understand and respond to common objects on a conceptual level: it needed object representations.

To enable the system to comprehend representational contributions, a second category of drawing modes was created that attempts to classify the type of object the user is drawing. First, the system must determine which lines to group, then

Fig. 2 Quadtree representation of canvas for the line grouping algorithm. Area inside blue dotted line is an “area of interest” as it has a higher line density than other areas



an image is formed from those lines and sent to a pretrained convolutional neural network model for classification. Finally, the system employs one of the grouped-line drawing algorithms and outputs the results to the shared canvas.

One of the significant challenges for implementing object recognition in an open-ended drawing application is determining which lines to group together to send to the sketch recognition module. To address this challenge, a three-tiered solution was implemented for grouping sketched lines: (1) time-based implicit grouping, (2) space-based implicit grouping, and (3) explicitly assigned grouping through user input.

In the *time-based implicit grouping* method, the system starts a timer every time the user lifts their pen from the sketch canvas. If a prespecified period of time passes between strokes, the system assumes the user has completed a full ‘turn’ to fully express their idea, and it will mark the last stroke as an “end stroke”. Based on our observations, we set this interval to 3 s. After the time is up, the system groups all of the strokes between the previous “end stroke” and the current “end stroke” as one turn. These strokes are rendered as a small temporary image isolated from the other strokes on the canvas, and fed into the sketch classification procedure to classify the sketch.

In the *space-based implicit grouping method*, the system constructs a quadtree data structure that includes all the points from strokes collected from the human users and AI agent. In this quadtree, the data in the regions that have a higher density of lines will be contained in the nodes with higher depths, as shown in Fig. 2. Once one particular node is four levels deeper than the average depth of the tree, it returns the area surrounding the node as an “area of interest” Then, the system draws an image from the selected strokes in this area similar to the time-based method for sketch classifications.

Users can also manually group sketches in the canvas using a Lasso tool in the UI for sketch classification. The user can choose to manually label the object themselves to serve as another “ground truth” example to help improve the sketch recognition model.

Once the lines have been grouped and made into an image, that image is passed to the sketch classification module. We employed convolutional neural networks to classify sketch input due to their state-of-the-art performance in the image and sketch recognition, greatly surpassing methods like bag-of-visual features and SVM [29–31]. Our sketch classification model is based on the VGG neural network [32] due to its recent success in large-scale image recognition. We modified a pretrained model of the VGG-19 architecture to suit our task.

After the sketched object is recognized, the system must determine a target location to draw an object. We established two main criteria when finding a location for the agent to draw a new object: (1) an empty region where the target-drawing object would minimally intercept with existing objects; and (2) being close to recently drawn objects. We employ the same quadtree data structure mentioned before to determine object placement in real time.

Once the system detects a turn, it uses the bounding rectangle formed by the users' sketches to find an area to draw. The system iterates through the surrounding locations starting from the top-left corner of to the sketch from the user's current turn first, and then queries the quadtree to get a candidate bounding rectangle containing the least points for a drawing area. This approach ensures the target object is drawn as close to the user's previous input as possible without drawing on top of existing elements. With the results of turn detection, sketch classification, and placement, the system utilizes the following two modes for generating the new sketch objects.

Drawing Similar Objects Mode

In this drawing mode, the system recognizes the user's drawn object and then responds with a different representation of that same object. Figure 3-left shows an example where the user drew a chair in the perspective view. The system responded with another chair similar to the original chair. The system uses the t-SNE algorithm [33] on the visual features extracted by the convolutional neural network to compute the nearest neighbor image in the 2-dimensional embedding of the features. This method provides the ability to draw visually similar or dissimilar objects (relative to the user input) of the target category.

Drawing Complimentary Objects Mode

In this mode, rather than drawing an object from the same classification, the system selects a semantically related category and then randomly picks an object from that category. The right side of Fig. 3 shows that the system recognized a tree has been drawn by the user, then responded with a message in a speech bubble stating its interpretation and planned contribution, and finally drew a mushroom on the canvas. To pick a category, we manually created a dictionary that categorizes the sample



Fig. 3 Drawing modes using sketch recognition to draw similar (left) and complimentary (right) objects next to the user’s most recently drawn object. The agent explicitly expresses what it recognizes and plans to draw (middle)

sketches into 15 high-level categories (with several subcategories) based on their semantic meanings. For instance, we group all the animals as one category with marine, bird, and land animals as subcategories.

The Creative Sketching Apprentice

We are developing the CSA to support creativity and concept exploration in design ideation. We present an algorithm for generating conceptual shifts that will be integrated into a full-featured CSA system with many of the same features of its predecessor, the Drawing Apprentice, to enable an intelligent co-creative sketching tool. This paper lays the conceptual groundwork for our approach for identifying and generating conceptual shifts based on sketched user input.

In this section, we describe our process for analyzing the dataset of sketched objects, finding structurally similar objects from other categories, and selecting objects to be placed on the shared canvas. We then provide a case study demonstrating the results of the algorithm, including individual conceptual shifts, how multiple conceptual shifts can be generated from the same input category and even from the same individual input image.

A Computational Model of Conceptual Shifts

In order to represent sketches, we used the same neural network architecture that was used in the Drawing Apprentice for sketch classification, as it had been shown to provide usefully discriminative features in this domain. The model was pretrained on ImageNet [34], a massive dataset of natural images that has been shown to produce useful representations for a wide variety of tasks—even those not based on natural images, such as sketches. We further fine-tuned the features of this pretrained model

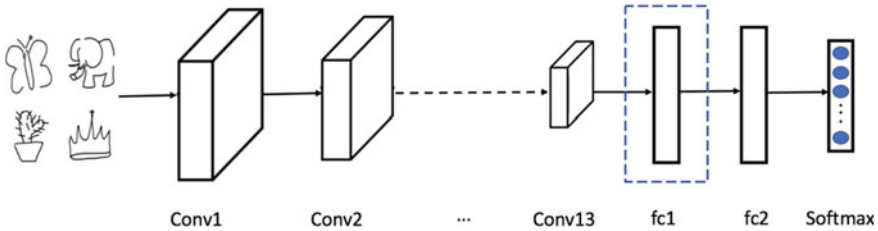


Fig. 4 The neural representation of sketches, drawn from the activation of the first fully connected layer (fc1) in the VGG-16 convolutional neural network architecture [32]

on our sketch dataset. We use the features generated by these networks as a representation on which we can measure the similarity between images and perform the conceptual shift. Preparing our model of conceptual shifts has two steps: learning the visual representations of sketches by fine-tuning the ImageNet classification network, and then performing clustering on the resulting representations of the sketches within each object category.

Learning the Visual Representation of Sketches

We use the Google “Quick, Draw!” (QD) Dataset [35] to train our system on sketched object categories. For this prototype, we used a subset of 65 of the 345 categories in the QD dataset, with up to 110,000 images in each category. Each sketch is represented as a high-level feature vector obtained using VGG-16 [32], a state-of-the-art Convolutional Neural Network (CNN) architecture. We started with a model that was pretrained on the ImageNet dataset [34] and then fine-tuned the weights by training on the sketches (Fig. 4).

The model contains 13 convolutional layers, 2 fully connected layers, and an output layer with Softmax activation of 1000 nodes. We extract the features from the first fully connected layer which produces 4096 features per image, and use these features to represent each sketch. This layer captures the spatially invariant presence of complex visual features, such as an arm, window, or wheel.

Clustering Within Categories of Sketches

The objects of each category have high variance, being depicted with different shapes, viewpoints and orientations. Exploring the data reveals that this variance is not uniform or continuous: there are clearly distinct “clusters” of sketches within each category, representing the set of common ways people sketch that object. Our approach to modeling conceptual shifts operates at this “way of sketching” level—we aim to

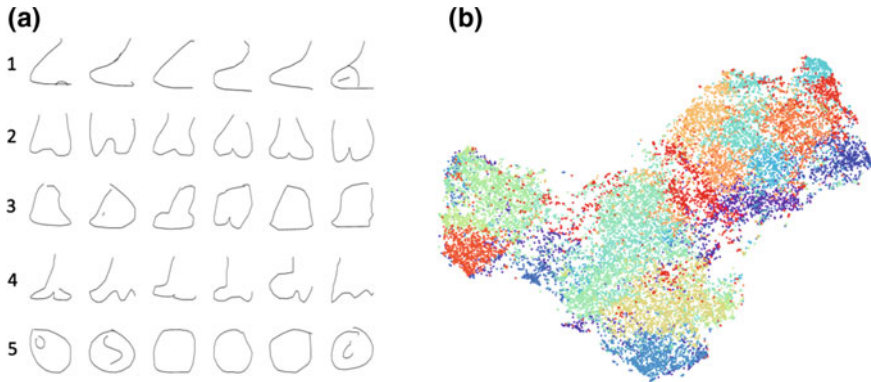


Fig. 5 **a** Five subcategories (“clusters”) within the “nose” category and **b** 2D embedding of the “nose” category using largeVis [36]

identify to the designer that their object of category A is structurally similar to objects of category B *that were drawn a particular way*.

We identify these conceptual subcategories by clustering the QD dataset using our VGG-16 representation. This identifies structurally similar subcategories, which we refer to as “clusters” that we use for identifying potential conceptual shifts. Figure 5a shows a few samples from five selected clusters for the “nose” category, and Fig. 5b shows the embedding visualization of the clusters in a 2D scatter plot.

After the images have been clustered, the next step is to determine structurally similar objects between clusters of different categories.

Determining Structurally Similar Objects

In the CSA, our proposed co-creative drawing system, the designer and an agent will take turns to sketch. The prototype perceives the designer’s input and identifies opportunities for conceptual shifts. Our prototype has no “designer”, nor is it currently embedded in an interactive co-creative system: it is a collection of algorithms which take an input drawing (intended to be the designer’s drawing) and identifies potential conceptual shifts. To identify potential conceptual shifts, it matches objects from the category of object the designer is drawing to structurally similar objects in another category. We compare the Euclidean distances between the centroids of the cluster (i.e., subcategory) to which the sketch belongs and clusters from other categories. We identify the cluster with the minimum distance to that of the current object and use this to propose a conceptual shift. Our approach involves three steps.

- **Recognizing:** This step represents the system’s perceptual phase, where the agent takes the input sketch, extracts its feature vector, and determines its cluster. In

our current approach, we select a random sketch from the database as a proxy for drawn input.

- **Matching:** The input sketch is matched to the most similar cluster that is not in the same object category as the current sketch.
- **Contributing:** The agent contributes a sketch from the matched cluster. This currently occurs by random selection from the dataset, but in future, it will involve generating a new drawing.

A Demonstration of our Conceptual Shift Algorithm

To demonstrate the utility of our approach to identifying conceptual shifts we provide three scenarios exploring how the system could respond to designers' sketches, particularly the different ways it could respond to the same input sketch. In our previous work, we explored examples related to drawing [37] while here we explore design applications. The first example explores a scenario where the designer creates a sketch and the system produces one potential conceptual shift as a response. The second example demonstrates how the system can produce many potential conceptual shift categories from one sketch. The third example demonstrates how multiple different sketches of the same object could result in different types of responses from the system. We then discuss the potential implications of this kind of support for ideation, analogical reasoning, and reinterpretation.

Call and Response with Conceptual Shifts

In this scenario, a designer's sketch is matched to a sketch from another category. The algorithm determines which category is most structurally similar to the user's input and then generates a response based on that calculation. Figure 6 shows the resulting proposed conceptual shifts from 6 categories of input sketches: *bridge*, *ceiling fan*, *bathtub*, *axe*, *toothbrush*, and *dumbbell*. The Euclidean distance metric is used to determine which category is the closest to each input sketch. Figure 6 shows two groups of input–response pairs, those that are very similar (i.e., have a low Euclidean distance, signifying a good potential conceptual shift), and those for which the closest match was less similar (i.e., a higher distance, perhaps signifying less potential for a conceptual shift).

Single Input and Multiple Responses with Conceptual Shifts

In the second scenario (Fig. 7), the (hypothetical) designer draws one input item and the system generates many potential conceptual shifts from different categories. In

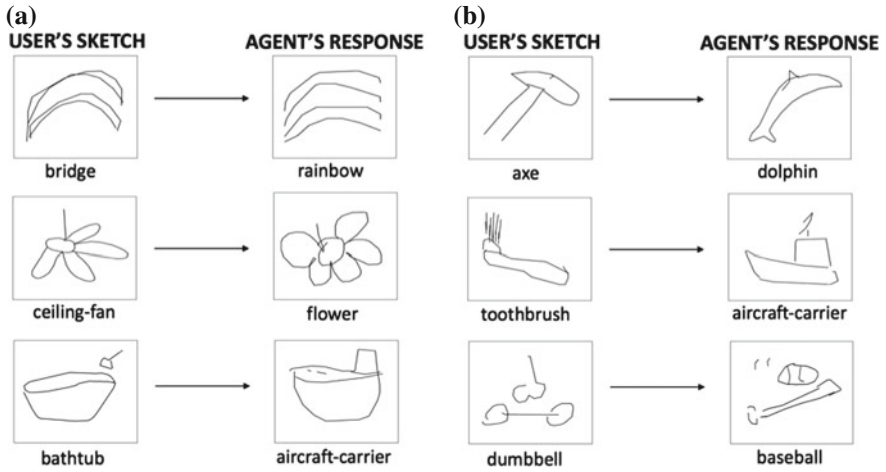


Fig. 6 Examples from three categories with **a** conceptual shifts that are very visually similar to the input (lower distance between matched clusters) and **b** conceptual shifts that are less visually similar (higher distance)

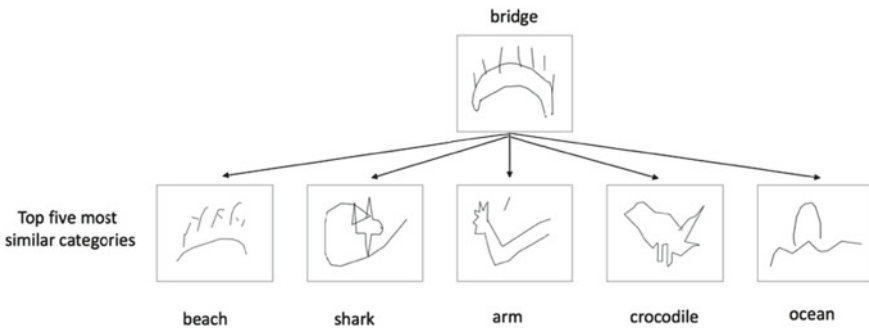


Fig. 7 Top 5 potential conceptual shift responses to a single sketch of a bridge

Fig. 7 assume the designer began by sketching a bridge. After the system extracted the visual features from the bridge sketch, it matched five potential conceptual shifts. These categories represent potential shifts that the designer might leverage in their further exploration of bridge designs.

Multiple Inputs and Multiple Responses with Conceptual Shifts

The final scenario, shown in Fig. 8, explores what might happen if a designer produces multiple iterations of an object in the same category. The system selects a different

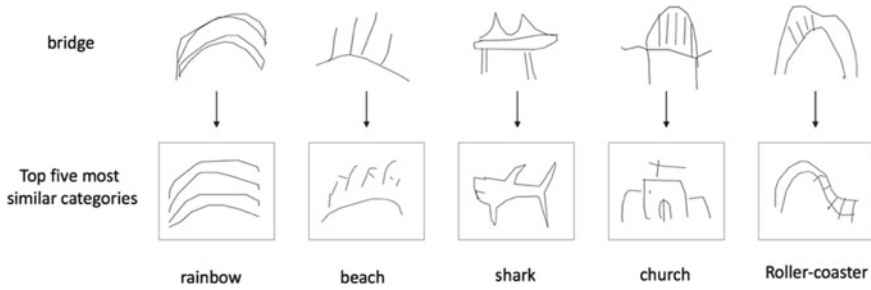


Fig. 8 The system's conceptual shift responses to multiple bridge sketches

conceptual shift category for each input. In this case, the user drew several types of bridges that each have unique structural characteristics. The algorithm finds different potential conceptual shifts in each case.

Supporting Design Creativity with Conceptual Shifts

Our prototype demonstrates the potential for conceptual shifts in a sketching context. We hypothesize that this capability, embedded into a co-creative system like our proposed CSA, could support creativity by promoting ideation, analogical reasoning, and conceptual reinterpretation during the design process. The next sections explain each of these hypotheses.

Ideation

The CSA could potentially facilitate creative ideation. Designers could engage with the CSA in a free association drawing game, where both designer and system contribute ideas to a shared canvas. Once the designer sketches an initial object, the system would generate a response by drawing a potential conceptual shift. The designer would then leverage the system's contribution to generate a new idea, to which the system would again respond, and so on. The system would provide structurally similar results from different categories to help designers see how their input relates to other categories. Revealing these categorical links can help designers find new connections between categories and spark new ideas for them to explore. This process could help designers overcome fixation.

While there are many ways to evaluate creativity and creative thinking, the Torrance Test of Creative Thinking (TTCT) is a well-established approach [38]. In the TTCT, users are evaluated with respect to fluency (amount of content generated), flexibility (number of categories covered in responses), originality (uniqueness), and

elaboration (detail). Playing the free association game with the CSA could help designers be more fluent due to the reduction of fixation on any one idea. The continual association with other categories could help the designer generate more diverse ideas, potentially also increasing the flexibility of their ideation process.

The multiple input to multiple responses use case shown in Fig. 8 could lead the designer to generate many versions of a target concept to see what types of conceptual shifts the system might generate. Seeing the different categories, the system generates might inspire the designer to explore a wide variety of concepts to see the system's responses.

Visual Analogy

Identifying conceptual shifts in visual domains can be considered a kind of visual analogy, a process known to be critical to design reasoning [36]. Visual analogy is characterized by mapping between structural elements of two dissimilar objects and then drawing inferences about the object one started with (the target) based on analogous one (the source).

The dominant theoretical model of analogy, Structure Mapping Theory (SMT) [39] says that the difference between analogy and the mere similarity is based on what match is made between the source and target. In Gentner's theory, analogy occurs when the equivalence between the two objects is based on the relationships within each object, not on each object's elements. For example, an analogy can be made between the solar system and an atom because each involves smaller objects orbiting a large, relatively stationary body. The analogy is not claiming any similarity between an atomic nucleus and the sun, nor between planets and electrons, merely that they share structural relationships.

The neuronal representations used in our model of conceptual shift are difficult to classify as "relations" or "elements". The upper layers of hierarchical convolutional neural networks identify visual features that correspond to invariant and discriminating elements of objects, such as eyes, noses, wheels, and wings [40]. These features combine characteristics of both elements and relationships between them—an eye is both a collection of lower level structures and a network of expected relationships between those structures. It is thus hard to say whether our system is performing what SMT would recognize as analogy-making.

Models of visual analogy posit that the visual representation acts as a kind of intermediary where semantically distinct objects may be depicted similarly, facilitating mapping [41]. Expert designers are known to more effectively employ visual analogy as a design strategy [42, 43]. There is some evidence to suggest that effectiveness is related to a tendency to reason about higher level, more abstract visual concepts, such as "repetitiveness", "density", or "compactness", while novices focus on more specific visual structures. The features identified by our model are definitely in the latter category—eyes occur frequently, but they are not at all abstract.

Advances in neural network architectures have begun exploring the representations that could capture spatial relationships between elements in sketches [44]. This representation could be used to make visual analogies with relational mappings. Vector embeddings of convolutional representations have been shown to capture more abstract concepts like age or gender from images [45]. Mappings between such features could be used to make analogies more like those constructed by expert designers. We hypothesize that these visual analogy-based extensions to our approach could lead to drawing systems are more effective at inspiring creativity.

Reinterpretation

One important factor in the creative process is the ability to reinterpret ideas from different perspectives in order to see the same input in new ways. Boden describes two types of creativity: exploratory and transformational [9]. Exploratory creativity refers to developing new ideas by traversing an established conceptual space. Transformational creativity refers to fundamentally changing that conceptual space through the act of developing a new idea, i.e., changing the rules of the game. The types of conceptual shifts introduced by the CSA have the capacity to result in transformational creativity. They help designers expand their conceptual space to include concepts that may not otherwise be considered. This type of reinterpretation could help users establish more fluid boundaries between categories.

The continual process of reinterpretation in the CSA turn-taking game may also train users to explore different ways of seeing their input. Suwa and Tversky [46] describe two distinct modes of perception used in design called *seeing-that* and *seeing-as*. *Seeing-that* describes a functional type of perception that looks at the concrete properties of a sketch and considers the role they play in the overall design. *Seeing-as* is a more interpretative process, where elements in a sketch can be viewed through different perspectives, such as seeing a collection of shapes in an architectural drawing as a face. Once the collection of objects is seen as a face, the designer might decide to add ears or a nose. Suwa and Tversky found that architects continually shift between *seeing-as* and *seeing-that* to help generate and refine their sketches. Playing the CSA could help users become accustomed to the *seeing-as* mode of perception by demonstrating how drawn structures could relate to a variety of highly different objects.

Figure 6 shows multiple instances where users might reinterpret their initial input and think about their idea from new perspectives. For example, when the designer sketches a ceiling fan, the system generates a flower. This might lead the designer to make an analogy and explore the idea of organic growth in their design task. This could lead them to consider creative and potentially transformative blade designs. Visual reinterpretation might be combined with further analogical reasoning in this case, such as if the designer considered designing the ceiling fan lights to look similar to the pollen in the center of a flower. Seeing one object as another object has the

potential to enable designers to generate novel and inspiring ideas to facilitate creative sketching.

Conclusions

This paper presents a computational model for reinterpreting a sketch into a structurally similar but semantically different category. We hypothesize that presenting this type of conceptual shift during the sketching process could support designers' creativity. We are developing a co-creative system embedding this capability, which we call the CSA, to test this hypothesis. We envisage that the CSA would promote ideation by presenting designers with diverse concepts to consider during their explorations. The conceptual shift algorithm selects concepts that have structural similarity to the emerging design but are semantically distinct. This may prompt analogical reasoning about the relationship between the user-generated sketch and the system's generated response. In this interaction, designers are prompted to see their design through a different perspective by exploring its similarity to other concepts.

The main contribution of this paper is our approach for identifying and generating conceptual shifts in the context of co-creative sketching. We have described the Drawing Apprentice to provide context for collaborative drawing. We then described our approach to identifying conceptual shifts, and provided a case study demonstrating how it could contribute to sketching in a design context.

References

1. Goel V (1991) Sketches of thought: a study of the role of sketching in design problem-solving and its implications for the computational theory of the mind. Ph.D. Dissertation, University of California, Berkeley
2. Johnson G, Gross MD, Hong J, Do EYL (2009) Computational support for sketching in design: a review. *Found Trends® Hum Comput Interact* 2(1):1–93
3. Biles JA (1994, September) GenJam: a genetic algorithm for generating jazz solos. In: *ICMC*, vol 94, pp 131–137
4. Hoffman G, Weinberg G (2010, April) Shimon: an interactive improvisational robotic marimba player. In: *CHI'10 extended abstracts on human factors in computing systems*, pp 3097–3102 (ACM)
5. Jacob M, Zook A, Magerko B (2013) Viewpoints AI: procedurally representing and reasoning about gestures. In: *DiGRA conference*
6. Kantosalo A, Toivanen JM, Xiao P, Toivonen H (2014, June) From isolation to involvement: adapting machine creativity software to support human-computer co-creation. In: *ICCC*, pp 1–7
7. Davis N, Hsiao CP, Singh KY, Li L, Moningi S, Magerko B (2015, June) Drawing apprentice: an enactive co-creative agent for artistic collaboration. In: *Proceedings of the 2015 ACM SIGCHI conference on creativity and cognition*, pp 185–186 (ACM)
8. Davis NM, Popova Y, Sysoev I, Hsiao CP, Zhang D, Magerko B (2014) Building artistic computer colleagues with an enactive model of creativity. In: *ICCC*, pp 38–45

9. Boden M (1990) *The creative mind: myths and mechanisms*. Weidenfeld and Nicolson, London, UK
10. Colton S, Wiggins GA (2012, August) Computational creativity: the final frontier? In: *Proceedings of the 20th European conference on artificial intelligence*, pp 21–26 (IOS Press)
11. Wiggins GA (2006) Searching for computational creativity. *New Gener Comput* 24(3):209–222
12. Wiggins GA (2006) A preliminary framework for description, analysis and comparison of creative systems. *Knowl-Based Syst* 19(7):449–458
13. Edmonds EA, Candy L (2005) Computer support for creativity. *Int J Hum Comput Stud* 63(4–5):363–364
14. Shneiderman B (2007) Creativity support tools: accelerating discovery and innovation. *Commun ACM* 50(12):20–32
15. Shneiderman B, Fischer G, Czerwinski M, Resnick M, Myers B, Candy L, Edmonds E, Eisenberg M, Giaccardi E, Hewett T, Jennings P, Kules B, Nakakoji K, Nunamaker J, Pausch R, Selker T, Sylvan E, Terry M (2006) Creativity support tools: report from a US national science foundation sponsored workshop. *Int J Hum Comput Interact* 20(2):61–77
16. Voigt M, Niehaves B, Becker J (2012) Towards a unified design theory for creativity support systems. In: *Design science research in information systems. Advances in theory and practice*, pp 152–173
17. Lubart T (2005) How can computers be partners in the creative process: classification and commentary on the special issue. *Int J Hum Comput Stud* 63(4):365–369
18. Davis N, Hsiao CP, Singh KY, Magerko B (2016, September) Co-creative drawing agent with object recognition. In: *Twelfth artificial intelligence and interactive digital entertainment conference*
19. Getzels JW, Csikszentmihalyi M (1977) The creative vision: a longitudinal study of problem finding in art. *J Aesthetics Art Criticism* 36(1):96–98
20. Grace K, Maher ML (2015) Surprise and reformulation as meta-cognitive processes in creative design. In: *Proceedings of the third annual conference on advances in cognitive systems ACS*, p 8
21. Poon J, Maher ML (1997) Co-evolution and emergence in design. *Artif Intell Eng* 11(3):319–327
22. Schon DA (1984) *The reflective practitioner: how professionals think in action*, vol 5126 (Basic books)
23. Schön D (1983) *The reflective practitioner: how practitioners think in action*. London: Temple Smith
24. Schon DA, Wiggins G (1992) Kinds of seeing and their functions in designing. *Des Stud* 13(2):135–156
25. Grace K, Maher ML, Fisher D, Brady K (2015) Modeling expectation for evaluating surprise in design creativity. In: *Design computing and cognition'14*. Springer, Cham, pp 189–206
26. Suwa M, Gero J, Purcell T (2000) Unexpected discoveries and S-invention of design requirements: important vehicles for a design process. *Des Stud* 21(6):539–567
27. Prats M, Garner S (2006). Observations on ambiguity in design sketches. *Tracey Online J Contemp Drawing Res*, pp 1–7
28. Gero JS (1998) Conceptual designing as a sequence of situated acts. In: *Artificial intelligence in structural engineering*. Springer, Berlin, Heidelberg, pp 165–177
29. Eitz M, Hays J, Alexa M (2012) How do humans sketch objects? *ACM Trans Graph* 44:1–44:10
30. Yu Q, Yang Y, Song YZ, Xiang T, Hospedales T (2015) Sketch-a-net that beats humans. *arXiv preprint arXiv:1501.07873*
31. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. *Nature* 521(7553):436–444
32. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*
33. Maaten LVD, Hinton G (2008) Visualizing data using t-SNE. *J Mach Learn Res* 9(Nov):2579–2605
34. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009, June) Imagenet: a large-scale hierarchical image database. In: *IEEE conference on computer vision and pattern recognition, CVPR 2009, IEEE*, pp 248–255

35. Ha D, Eck D (2017) A neural representation of sketch drawings. arXiv preprint arXiv:1704.03477
36. Shneiderman B (2007) Creativity support tools: accelerating discovery and innovation. *Commun ACM* 50(12):20–32
37. Karimi P, Davis N, Grace K, Maher ML (2018) Deep learning for identifying potential conceptual shifts for Co-creative drawing. arXiv preprint arXiv: 1801.00723
38. Torrance EP (1962) Guiding creative talent. Prentice-Hall, Inc., Englewood Cliffs, NJ. <https://doi.org/10.1037/13134-000>
39. Gentner D (1983) Structure-mapping: a theoretical framework for analogy. *Cogn Sci* 7(2):155–170
40. Zeiler MD, Fergus R (2014, September) Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer, Cham, pp 818–833
41. Davies J, Goel AK, Nersessian NJ (2009) A computational model of visual analogies in design. *Cogn Syst Res* 10(3):204–215
42. Goldschmidt G (2001) Visual analogy: a strategy for design reasoning and learning. In: Design knowing and learning: cognition in design education, pp 199–220
43. Casakin H (2004) Visual analogy as a cognitive strategy in the design process: expert versus novice performance. *J Des Res* 4(2):1–18
44. Jaderberg M, Simonyan K, Zisserman A (2015) Spatial transformer networks. In: Advances in neural information processing systems, pp 2017–2025
45. Radford A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434
46. Suwa M, Tversky B (1997) What do architects and students perceive in their design sketches? A protocol analysis. *Des Stud* 18(4):385–403

Author Index

A

Abdellahi, Sarah, 55
Acartürk, Cengiz, 381
Andersen, Emil, 555

B

Bang, Hyunseung, 155, 195, 341
Beaudry Marchand, Emmanuel, 97
Burge, Janet E., 537

C

Cagan, Jonathan, 37, 613
Campbell, Matthew I., 75
Charidis, Alexandros, 285
Colaço, Canan Albayrak, 381

D

Davis, Nicholas, 721
Dhawan, Nikhil, 155
Dorta, Tomás, 97
Duarte, José P., 421, 499
DuPont, Bryony L., 75

E

Economou, Athanassios, 401
El-Khouly, Tamir, 115
Emanuel, Lia, 651

G

Gero, John S., 265, 361, 595
Geyer, Philipp, 21
Gopsill, James, 651
Goucher-Lambert, Kosa, 37
Grabska, Ewa, 705

Grace, Kazjon, 55, 721
Guerritore, Camilla, 499
Gyory, Joshua T., 613

H

Hicks, Ben, 651
Hoffman, Guy, 155, 195, 341
Hou, Dan, 439
Hoyle, Chris, 577
Huang, Jeffrey, 227
Hulse, Daniel, 577

J

Janssen, Patrick, 177
Ji, Guohua, 177
Jin, Yan, 303, 323
Joel-Edgar, Sian, 651
Jones, David, 651

K

Kannengiesser, Udo, 265
Karimi, Pegah, 721
Khokhlov, Matvey, 227
Kim, Jin Goog, 55, 135
Koh, Immanuel, 227
Koskela, Lauri, 247
Kotovsky, Kenneth, 613
Kozine, Igor, 555
Kroll, Ehud, 247
Krstic, Djordje, 479

L

Law, Matthew V., 155
Lee, Lina, 135

Lester, Miriam, [537](#)
 Ligler, Heather, [401](#)
 Liu, Xiongqing, [303](#)
 Lo, Tian Tian, [213](#)

M

MacNeil, Stephen, [55](#)
 Maher, Mary Lou, [55](#), [135](#), [721](#)
 Mahzoon, Mohammad, [55](#)
 Maier, Anja, [555](#)
 Mamoli, Myrsini, [459](#)
 Mao, Xiaoyang, [631](#)
 Mars, Agnieszka, [705](#)
 McCall, Raymond, [519](#)
 McComb, Christopher, [3](#)
 Milojevic, Hristina, [323](#)
 Mogles, Nataliya, [651](#)
 Moss, Jarrod, [37](#)

N

Nakakoji, Kumiyo, [687](#)
 Neramballi, Abhijna, [361](#)
 Newnes, Linda, [651](#)
 Ni, Ruichen, [421](#)

P

Perišić, Marija Majda, [595](#)
 Pierini, Davide, [97](#)

R

Robinson, Kevin, [651](#)

S

Sakao, Tomohiko, [361](#)
 Schnabel, Marc Aurel, [213](#)
 Selva, Daniel, [155](#), [195](#), [341](#)
 Sen, Chiradeep, [631](#)
 Shih, Yi-Teng, [669](#)
 Shi, Yuan Ling Zi, [195](#), [341](#)
 Short, Ada-Rhodes, [75](#)
 Singaravel, Sundaravelpandian, [21](#)
 Ślusarczyk, Grażyna, [705](#)
 Snider, Chris, [651](#)
 Štorga, Mario, [595](#)
 Stouffs, Rudi, [439](#)
 Strug, Barbara, [705](#)

T

Tedjosaputro, Mia A., [669](#)
 Tumer, İrem, [577](#)
 Tumer, Kagan, [577](#)
 Turner, Cameron, [631](#)

W

Wang, Likai, [177](#)

Y

Yamamoto, Yasuhiro, [687](#)
 Yoon, So-Yeon, [155](#), [195](#), [341](#)