Design Patterns from Empirical Studies in Computer-Aided Design

Rongrong Yu[1(✉)] and John Gero[2,3]

[1] The University of Newcastle, New South Wales, NSW, Australia
rongrong.yu@uon.edu.au
[2] George Mason University, Fairfax, VA, USA
john@johngero.com
[3] University of North Carolina at Charlotte, Charlotte, NC, USA

Abstract. This paper presents the results from studying the effect of the use of computational tools on designers' behavior in terms of using design patterns in the conceptual development stage of designing. The results are based on a protocol study in which architectural designers were asked to complete two architectural design tasks with similar complexity, one in a parametric design environment and one in a geometric modeling environment. To explore the development of design patterns during the design process, the technique of 2nd order Markov model was used. The results suggest that there were more design patterns adopted in the parametric design environment than in the geometric modeling environment. Also, there are more design patterns related to structure in the parametric design environment than in the geometric modeling environment.

Keywords: Design pattern · Markov model · Protocol studies

1 Introduction

In computational design environments, designers often adopt existing design patterns based on their experience of using their design knowledge and their experience in using computational tools. Design patterns have been studied in architectural design: The re-use of existing problem-solution pairs extracted from a designer's own or others' professional experience makes the design process more efficient [1]. This idea has been widely applied in the software design domain. However, this phenomenon has not been adequately studied and evaluated in computer-aided architectural design environments (CAAD).

To improve our understanding of the possible use of design patterns while designing in CAAD, the results of a cognitive study in which designers were asked to complete two architectural design tasks with similar complexity in a parametric design environment (PDE) and a geometric modeling environment (GME) are presented. Protocol analysis was employed to study the designers' behavior. The technique of Markov model analysis is used to analyze the protocol data collected. From the Markov model analysis results describing the occurrence design patterns in computational design environments are derived and discussed.

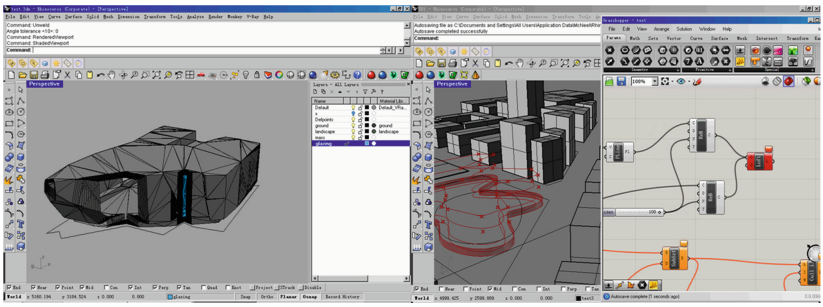
G. Celani et al. (Eds.): CAAD Futures 2015, CCIS 527, pp. 493–506, 2015.
DOI: 10.1007/978-3-662-47386-3_27

2 Background

2.1 Selected Computational Design Environments – PDE and GME

Design media have significant effects on designers' thinking processes [2, 3]. Past research has suggested that sketching can assist design thinking as an effective design medium [4, 5]. In a similar way, with the increasing application of digital design tools, researchers have started to study the influence of computational design tools on design processes [6–9]. Oxman [10] argues that design media are knowledge-intensive computational environments. Designers share the design knowledge that can be represented and employed in computational environments.

Geometry modeling tools have largely replaced 2D drafting tools in many design practices. GMEs as 3D digital design tools are more effective in assisting in the design process than 2D drafting tools in various ways [6], for example, better visual representations including the ability to produce perspective and walk-through animations, and better coordination of documentation. 3D geometry modeling tools applied in architecture include ArchiCAD, AutoCAD, Microstation, Sketchup, 3ds Max, Maya, Rhino, and many others. In this study, for comparison, we chose Rhino as an example of a GME, Fig. 1(a). In the late 1990s, with the growth in importance of 3D digital tools in the design industry, architects began to identify a range of ways where these were superior to previous 2D computer-aided design systems [6, 11]. More recently another shift has begun to occur, with BIM and parametric software tools beginning to challenge the role played by 3D geometry modeling software in the AEC industry.



(a) (b)

Fig. 1. (a) Design environment – GME, (b) design environment – PDE

Parametric design is a dynamic, rule-based process controlled by variations and parameters, in which multiple design solutions can be developed in parallel. According to Woodbury [12], it supports the creation, management and organization of complex digital design models. By changing the parameters of an object, particular instances can be altered or created from a potentially infinite range of possibilities [13]. The term "parameters" means factors which determine a series of variations. In architecture, parameters are usually defined as building parameters or environmental factors. In the

architectural design industry, parametric design tools are utilized mainly on complex building form generation, multiple design solution optimization, as well as structural and sustainability control. Parametric design, in a computational form, is a new way of thinking about architectural design. Its impact on designers' processes has not been adequately explored. Currently, typical parametric design software includes Generative Components from Bentley Corporation, Digital Project from Gehry Technology, Grasshopper from McNeel. Scripting tools include Processing based on the Java language, Rhino script and Python script, based on the VB language from McNeel. In this study, Grasshopper was chosen as the parametric design environment, Fig. 1(b). Grasshopper is both an advanced environment for facilitating conceptual design and is in relatively wide-spread use in the architectural profession. Each software classified as a GME or a PDE has its unique features for designing. The selection of Rhino and Grasshopper as the GME and PDE in this study is due to their features, which are representative of the main characteristics of GMEs and PDEs.

2.2 Basis for Protocol Coding Scheme – FBS Ontology

As one of the main design ontologies, Gero's FBS ontology [14] has been applied in numerous cognitive studies [9, 15, 16]. Researchers argue that it is potentially capable of capturing most of the meaningful design issues and design processes [14] with the transitions between design issues clearly classified into eight design processes. The FBS ontology, Fig. 2 shows the three classes of ontological variables: Function (F), Behavior (B) and Structure (S). Function (F) represents the design intentions or purposes; behavior (B) represents the object's derived behavior (Bs) or expected behavior from the structure (Be); and structure (S) represents the components that make up an artifact and their relationships. The ontology as the basis of a coding scheme includes two additional design issues that can be expressed in terms of FBS and therefore do not require an extension of the ontology. These are requirements (R) and descriptions (D). The first of these represents requirements from outside design and the second, descriptions, mean the documentation of the design. Figure 2 shows the FBS ontology indicating the eight design processes—formulation, analysis, evaluation, synthesis, and reformulation I, II and III. Formulation defines the process that generates functions and then expected behaviors, i.e. sets up expected goals from the requirement, while synthesis generates a structure as a candidate solution. Analysis produces a behavior from the existing structure and evaluation compares Bs and Be to determine the success or failure of the candidate solution. Reformulation is the process from the structure back to itself, to the behavior or to the function, which is a reconstruction or reframing process. Among the eight design processes, the three types of reformulation processes are considered to be the dominant processes that potentially capture creative aspects of designing by introducing new variables or new directions [17]. The FBS ontology is claimed to be a universal coding scheme for various design environments [15]. By calculating the transitions between design issues from empirical data, various analyses can be conducted. In this study, the FBS ontology is utilized as the basis of the coding scheme in the protocol analysis.

In the present study, which explores designers' behavior in both a PDE and a GME, an instantiation of the coding scheme is required. Gero's FBS ontology has been applied

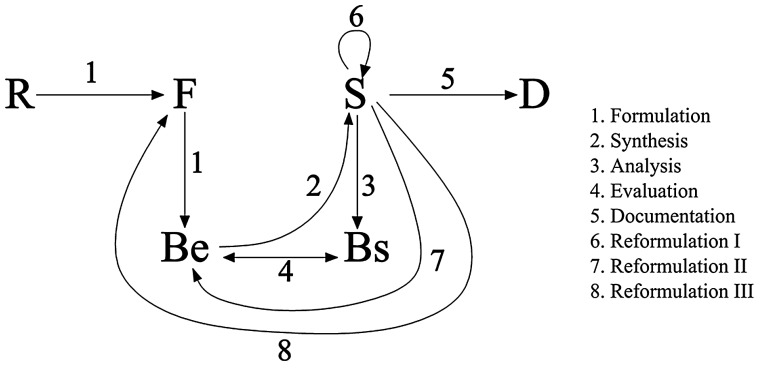


Fig. 2. The FBS ontology (after [14]).

in many cognitive studies where it has been demonstrated as potentially capturing most of the meaningful design processes [15] and recording clear transitions between design issues. The FBS ontology is founded on the requirements of coverage and uniqueness: the categorical concepts that make up the ontology need to cover all the attributes of a design and there can be no overlap of categorical concepts. A major outcome of the FBS ontology is that design processes are a consequence of the transitions between ontological elements and do not require a separately produced ontology of processes. The FBS ontology has been used widely in the domains of mechanical engineering, architecture, software engineering, civil engineering, cognitive psychology, manufacturing, management and creativity research. The behavior of designers, using the FBS ontology as the basis, can be measured from empirically derived data from protocol analysis. With this ontology it becomes possible to compare designing independent of researcher, independent of domain, independent of education, independent of whether an individual or a team is designing, independent of location or co-location, independent of the use of tools, independent of design experience and independent of design task. Prior to this such empirically derived data from different researchers was generally not comparable and it was difficult to build directly on the research of others. Kan and Gero [18] applied the FBS ontology to a study of software designers' behavior, suggesting that the method is effective for encoding programming or rule-based activities across different design disciplines. Given that PDEs enable scripting and programming activities, similarly the FBS scheme will be able to encode both geometric modeling and rule-based algorithmic activities effectively. Therefore, in this study it is introduced as a conceptual foundation for developing the coding scheme for the protocol analysis.

3 Research Method

3.1 Protocol Analysis

Protocol analysis is a method for turning qualitative verbal and gestural utterances into data [19, 20]. It has been used extensively in design research to develop an understanding of design cognition [17, 21, 22]. According to Akin [23], a protocol is the record of

behaviors of designers using sketches, notes, videos or audio. After collecting the protocol data, a coding scheme is applied to categorize the collected data, enabling detailed study of the design process in the chosen design environments. As Gero and Tang [24] state, protocol analysis has become the prevailing experimental technique for exploring the understanding of design.

Usually in protocol analysis, concurrent and retrospective protocol collection methods can be applied in design experiments [19, 25]. A concurrent protocol involves participants in an experiment verbalizing their thoughts when working on a specific task – also called the "think aloud" method – whereas a retrospective protocol explores what designers were thinking while designing, a process which is applied as soon as they have finished the design task. Some studies have compared these two protocol collection methods. For instance, Kuusela and Pallab [26] argue that concurrent protocols are more suitable for examining the design process and can generate larger numbers of segments, while retrospective protocols are more suitable for examining design outcomes. Another example of this comparison is Gero and Tang's [24] study exploring design processes. Their results show that concurrent and retrospective protocols lead to very similar outcomes in terms of exploring designers' intentions during design processes. But they also conclude that concurrent protocols are an efficient and applicable method by which to understand design processes. Retrospective protocols are commonly believed to be less intrusive to the design processes.

Importantly, protocol analysis of this type deals with a relatively small number of samples, but it enables an in-depth exploration of the samples. Thus, a study of the cognitive behavior of eight designers is both acceptable and in keeping with past research in this field because of the quality and depth of information that is recorded and analyzed. However, for this reason we also cannot generalize the results of this research to describe the actions or behaviors of a much larger population of designers. Nevertheless, from such studies important patterns, which are repeated by designers can be used to provide an increased level of understanding of the design process.

3.2 Markov Model Analysis

A Markov model describes the probabilities of moving from one state to another [27, 28], it demonstrates the tendency of future design moves. Kan and Gero adopt the Markov chain model using the FBS ontology to describe cognitive design processes [15, 29]. Within the context of the FBS ontology, the Markov matrix can be applied as a quantitative tool to study design activities based on the transition probabilities between design issues or between design processes. Within the FBS context, two types of Markov models have been found to be useful: the 1st order Markov model and the 2nd order Markov model. The 1st order Markov model presents the probability of moving to a future state depending only on a knowledge of the current state, without considering the past states, Fig. 3.

The 2nd order Markov model includes the memory of the past state. That means a future movement is dependent on both the current state and its preceding state. For example, if the current design activity is Be, and the previous one was F, which is a formulation design process, then we use the 2nd order Markov model to calculate the probability of next state being S if the previous design process was formulation, Fig. 4.

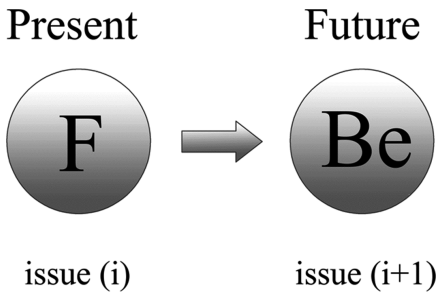


Fig. 3. An example of the foundation for a 1st order Markov model using the FBS ontology

Jiang [30] applies both the 1st order and 2nd order Markov model to study multidisciplinary designers' behavior. The result of his study shows that the main transition models match the original FBS ontological processes. Compared to the 1st order Markov model, the 2nd order Markov model presents a longer probability passage of transitions, which contains three sequential steps. This research utilizes the 2nd order Markov model to explore the utilization of design patterns in the CAAD environments.

An example of a 2nd order Markov model is shown in Fig. 4.

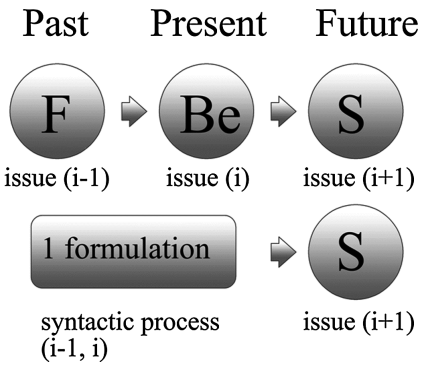


Fig. 4. An example of a 2nd order Markov model using the FBS ontology [30]: a 2nd order Markov chain, this is the interpreted as a transition from a design process to a design issue.

4 Experiment Setting

4.1 Selection of Subjects

In the any experiment, the selection of participants is important as it can influence the objectivity and reliability of the final results. The principle behind the selection is to reduce as much as possible individual differences and other subjective influences. The criteria of selection for the eight architects was that they should each have more than five years' architectural design experience and no less than two years' experience using

parametric design tools, to ensure that the participants are experienced both in architectural design and in operating parametric design software. The requirements of two years' experience using parametric design tools is based on the fact that Grasshopper, as a parametric design software, was developed in 2007 and gained wider adoption only during the 2010s. By the time this research was conducted, most parametric designers have only gained two to three years' experience with the tool. Previous protocol studies often selected subjects with experience levels ranging from five to ten years as expert designers [11, 31]. However most parametric designers tend to come from a younger generation. Therefore, architects with five years' architectural design experience are considered as sufficiently experienced designers amongst the younger generation and are suitable for the current study. Additionally, participants' abilities regarding creative design and manipulating software should be at a similar level so that individual differences would not greatly affect the final results. In the end, eight designers were found who could satisfy the selection criteria. Architects were recruited from architectural design companies, tutors of parametric design workshops, and lecturers (four practitioners and four academics). Among the participants were two female designers and six male designers, five of the participants are from Australia and three from outside Australia. The standard deviations of the measurements are the bases for determining whether these demographic variations are significant. If the standard deviation for a particular measurement is low then the effect of the demographic variation in the subjects is not statistically significant.

4.2 Design Brief

For applications of protocol analysis, it is suggested that the experiment normally be limited to around one hour in length, meaning that the design task should not be too complex. In real cases, an architectural design task usually takes weeks or months. A previous studies has shown that the design behavior of designers across long-term, multiple design sessions shows only minor variations [32]. In the current research due to the restriction of research method, the selection of a one hour design task with an appropriate complexity level in a simulated experiment environment is reasonable to explore designers' cognitive behavior during the conceptual design stage. Many previous studies used simple product design tasks, such as a computer mouse, packaging or even a symbol design. In the architectural field, design tasks are also simplified, such as rearranging furniture or producing a home office layout [33]. However, parametric design tools are appropriate for generating complex geometries. If the design task is too simple, the advantages of parametric design tools are difficult to express.

In the present experiment, each designer was required to complete two different design tasks with similar levels of complexity, one using Rhino (GME) and one using Grasshopper (PDE). Designers were given 40 min for each design session, but were allowed to continue for an extra 20 min, if required, in order to complete the task. Considering the time necessary for a conceptual design task, as well as the time involved for later data analysis of the results, 40 min is a reasonable time constraint. Task 1 is a conceptual design for a community center and Task 2 is a similar study of a shopping center, with both containing some specific functional requirements. These

functional requirements are the main differences between the two tasks. In all other ways the two design tasks are similar, including the site provided, the required building size, and the extent of the concept development. A pre-modeled site, shown in Fig. 5, was provided to the designers for each task. Because the present study is focused on exploring designers' behavior at the conceptual design stage, the designers were required to only consider concept generation, simple site planning and general functional zoning. No detailed plan layout was required. Both tasks focus on conceptual design in general to enable the design process to be completed in a relatively short time period, and therefore to be captured and analyzed using the protocol analysis method. The tasks were both open and general enough to provide designers with the freedom to enable various possible design strategies to be applied during parametric design. As a result, the designers were allowed to exhibit different ways of approaching parametric design, which are similar to the actual practices of parametric design and therefore useful in order to examine the findings about parametric design. The design sessions and tasks were randomly matched among different designers. During the experiment designers were not allowed to sketch, ensuring that almost all of their actions happened within the computer. This ensured that the design environment was purely within either the PDE or the GME.

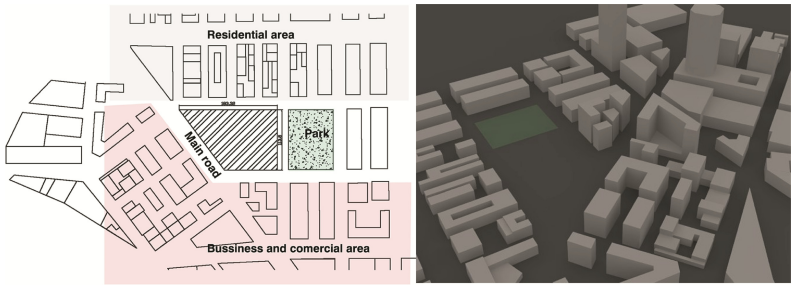


Fig. 5. Site model provided

4.3 Experiment Procedures

Before the data collection part of the experiment commenced a "warm-up" process was used to familiarize participants with the equipment. According to some current studies, the "think aloud" method for protocol data collection may influence participants' perception during design processes [19, 22]. As a consequence, designers may not be used to talking while they are designing, which could lead to incomplete data from such experiments. The purpose of the warm-up training is to explain to the participants the significance of the research and to provide training to practice the "think aloud" skills required [34] so that they can better verbalize their thoughts during the experiment.

The experiment is divided into two parts. In the first part, participants are required to speak aloud what they are thinking while designing. A screen capture program records both their words and actions. If there is not sufficient verbal data produced, in the second part the retrospective protocol method is used to produce complementary verbal data.

That means that, after finishing the design task, the videos are played back and participants are asked to make additional comments about what they were thinking while designing. The data collected, therefore include verbal information about participants' design intentions as well as visual information about their activities.

5 Analysis

5.1 General Analysis

This study employs an integrated segmentation and coding method. The segmentation and coding process are based on the "one segment one code" principle [35]. It means there is no overlapped code or multiple codes for one segment. If there are multiple codes for one segment, the segment will be further divided. Table 1 provides the general information of the coding coverage. The values shown in the table are the average of the eight protocols. The average overall numbers of segments are respectively 244 in the PDE and 224 in the GME. Designers, on average, spent more time in the PDE session (48 min) than in the GME (44 min). On average over 92.2 % of segments can be coded as FBS codes. Non-coded segments include communication and software management. The design speed is very similar between the two design environments, with means of 5.11 and 4.78 segments/min and low standard deviations. The individual speed of design varies between 3.06 and 6.86 segments/min, especially in the GME session. That indicates that designers have their own design habits or strategies or that some designers may think and act faster than others.

Table 1. General coding information of design sessions. Low standard deviations in these results indicate that the demographic variation in the subjects is not significant.

	Design environment	Time (min)	Number of segments	Coded percentage (%)	Speed (segments/min)
Mean	GME	44.0	224	92.2	5.11
	PDE	48.0	244	92.2	4.78
SD	GME	11.2	45.3	4.3	1.20
	PDE	7.4	29.7	3.5	0.53

After transcription, two rounds of segmentation (the division of protocols into individual segments based on their content) and coding were conducted. The coding was conducted by one researcher with a time interval of two weeks between the two rounds of coding. Following this an arbitration session (to make decisions on any disagreements between codes) was carried out to produce the final protocol. The agreement between the two rounds of coding is 84.8 % (GME) and 83.5 % (PDE). The final arbitrated results were 92.1 % (GME) and 91.5 % (PDE). The high level of agreement suggests the reliability of the coding results.

5.2 2nd Order Markov Analysis Results

The 2nd order Markov model analysis is presented in Table 2. It shows that all the transitions with higher probability are related to S. The 2nd order Markov model produces a larger pattern that includes two transitions. As shown in Table 2, the highest transition probability is from reformulation 3 to S, reformulation 3 refers to transition S to F, which means that the transition S-F-S is the most likely to occur pattern. The obvious difference between GME and PDE is the transition after reformulation 3 to S and the transition after reformulation 3 to Be. Reformulation 3 to S means that, after the designer has carried out the process reformulation 3, which is from S to F, the designers' consideration goes back to S. This transition, which is F to S is part of a larger pattern, which is S-F-S.

Table 2. The 2nd order Markov model analysis

	R		F		Be		Bs		S	
	GME	PDE	GME	PDE	GME	PDE	GME	PDE	GME	PDE
Formulation	0.04	0.02	0.22	0.20	0.24	0.19	0.24	0.22	0.26	0.37
Synthesis	0.00	0.00	0.04	0.05	0.14	0.24	0.36	0.28	0.45	0.43
Analysis	0.01	0.00	0.06	0.06	0.21	0.15	0.25	0.25	0.47	0.54
Evaluation	0.02	0.01	0.06	0.06	0.17	0.23	0.23	0.28	0.52	0.43
Reformulation 1	0.01	0.00	0.03	0.06	0.15	0.18	0.39	0.31	0.43	0.45
Reformulation 2	0.01	0.00	0.09	0.06	0.22	0.16	0.24	0.34	0.45	0.44
Reformulation 3	0.06	0.00	0.13	0.06	0.30	0.07	0.17	0.18	0.34	0.69

6 Design Patterns in Computational Design Environments

6.1 The Design Pattern S-F-S

A descriptive diagram of the 2nd order Markov model analysis in the GME and the PDE is presented in Fig. 6. The circles labeled with the FBS codes represent the design issues, and the size of circle represents the frequency of occurrence of the design issue. Each arrow shows the transition from one state to the other, and the thickness of the line represents the transition probability between design issues. To demonstrate the main activities of the designers, we select those transitions with the probability value larger than 0.4 and highlight them in Fig. 6. The value 0.4 is selected as threshold to abstract the model based on a transition probability that is 2 times that of the random transition probability. In the FBS model, each variable has 5 other states to go to, which means that the random probability is 0.2, therefore 0.4 is set as the threshold.

Fig. 6. (a) Primary transitions of the 2nd order Markov model in the GME, (b) primary transitions of the 2nd order Markov model in the PDE.

Applying 2nd order Markov model analysis, the results in Fig. 6 suggest that there are significantly more S-F-S transitions in PDE than in GME. During the F–S process, designers select an existing structure/solution for the particular design problem based on their experience or knowledge, which is a process of using an existing design pattern to the problem. The S-F-S transitions refer to a consideration to a structure issue (S) followed by the adoption of the design pattern (F-S). That is to say, design patterns usually based on the consideration of geometrical structure. From this result we can infer that when architects apply programming and scripting in their design, such as in a PDE, they exhibit the characteristic of using design patterns when building the structure of the geometry.

Within the context of the FBS ontology, this process of transitioning directly from function (F) to structure (S) is excluded from routine ways of design (excluded from the eight design processes expressed in FBS model). Previous research suggests from the study of software designers' behavior, that F to S is a typical design process that occurs frequently [18]. During the F–S process, designers select an existing structure/solution for the particular design problem based on their experience or knowledge, which is a process of selecting and applying an existing design pattern to the problem. This matches the concepts behind Alexander's "pattern language" [1]. Since software designers use design patterns when programming and scripting [36, 37], we can infer that when architects apply programming and scripting in their design, such as in a PDE, they exhibit the similar characteristic of using design patterns.

Design patterns are an important concept in both architectural design and software design. In software design, it assists software designers in working more efficiently and makes the programming and scripting process traceable. In the PDE, if we can generalize some useful design patterns, it would assist architects in their scripting process.

6.2 Design Patterns in Computational Design Environments

From the Markov model analysis results, we found design patterns are adopted in both GME and PDE, with more patterns in the parametric design environment. The idea of design patterns was first introduced by Christopher Alexander: "each pattern describes

a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice." [1, p. x]. That is, a pattern is a documentation of a solution suitable for certain kinds of design problems, which may occur frequently.

Patterns usually come from designer's experience [36], which can be seen as a "induction" process. Designers generalize from their own design experience or from observation of other designers, abstract the problem-solution pair, and formalize the "patterns" which can then be re-used. These generated patterns can be improved, and combined into a network of connections depending on the design purpose [38]. Woodbury writes that: "A pattern is a generic solution to a well-described problem. It includes both problem and solution, as well as other contextual information." [12, p. 185]. A design expert has accumulated a large number of examples of problems and solutions in a specific domain [39]. The pattern itself is an abstraction of that experience, when designers apply the patterns, they could revise them based on their own preference, or on the specific context of the current design task.

In the software design domain, educators found that Alexander's work on design patterns is suitable in software design pedagogy. For example, Gamma et al. [37] define patterns as a tool to describe compositional ideas in computer programming. This matches our analysis results that in parametric design, design patterns are developed and used.

7 Conclusion and Future Work

This paper has presented the results of a protocol study that explores the phenomenon of using design patterns in computational design environments. It compared the design patterns found in a parametric design environment (PDE) with those found in a geometry modeling environment (GME). The main finding is that: firstly, the adoption of design patterns is found in both computational design environments – PDE and GME. Secondly, significantly more design patterns are used in the PDE than in the GME. Since the main differences between the two design environments is that there is rule algorithm feature in the PDE, we can assume that the more rule algorithm features in the computational design environment, the more design patterns tend to be used during design process; Thirdly, the occurrence of design patterns is mainly based on the consideration of geometry. This claim is based on the higher number of S-F-S transitions in the PDE than in the GME. During this process designers' attention first focus on building the geometric pattern, followed with a design pattern. This is to say, when designers consider the structure of geometry, they tend to adopt design patterns based on their professional experience or knowledge.

The existence of design patterns implies that some aspects of the design processes in computational design are potentially generalizable and transferable, and can be learned by architectural designers and students. The design patterns identified from the current study can be potentially customized for different design scenarios and embedded as generic components in the system to allow designers to apply computer-aided design tools more effectively. These protocol analysis results suggest that some designers

currently define design patterns by themselves and repeatedly use them in a computational design process.

The future work based on this study will focus on exploring the development of design patterns in the PDE over time. This has pedagogical implications in terms of both teaching and learning.

Acknowledgements. This research has been supported in part by the National Science Foundation grant CMMI-1161715. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of National Science Foundation.

References

1. Alexander, C., Ishikawa, S., Silverstein, M.: A Pattern Language: Towns, Buildings, Construction. Oxford University Press, New York (1977)
2. Chen, S.-C.: The role of design creativity in computer media. In: Architectural information management, 30th eCAADe conference, Helsinki, Finland (2001)
3. Mitchell, W.J.: Beyond Productivity: Information Technology. Innovation and Creativity, Washington, DC (2003)
4. Black, A.: Visible planning on paper and on screen. Behav. Info. Technol. **9**(4), 283–296 (1990)
5. Schön, D.A.: The Reflective Practitioner: How Professionals Think in Action. Basic Books, New York (1983)
6. Bilda, Z., Demirkan, H.: An insight on designers' sketching activities in traditional versus digital media. Des. Stud. **24**(1), 27–50 (2003)
7. Fallman, D.: Design-oriented human-computer interaction. In: 21th ACM CHI Conference on Human Factors in Computing Systems, Florida, USA (2003)
8. Kim, M.J., Maher, M.L.: The impact of tangible user interfaces on spatial cognition during collaborative design. Des. Stud. **29**(3), 222–253 (2008)
9. Gero, J., Tang, H.-H.: concurrent and retrospective protocols and computer-aided architectural design. In: 4th CAADRIA conference, Shanghai (1999)
10. Oxman, R.: Design media for the cognitive designer. Autom. Constr. **9**(4), 337–346 (2000)
11. Kan, J.W.T., Gero, J.S.: The effect of computer mediation on collaborative designing, in between man and MACHINE? Integration, Intuition, Intelligence. In: Proceedings of 14th CAADRIA conference, Yunlin, Taiwan (2009)
12. Woodbury, R.: Elements of Parametric Design. Routledge, New York (2010)
13. Kolarevic, B.: Architecture in the Digital Age: Design And Manufacturing. Spon Press, New York (2003)
14. Gero, J.S.: Design prototypes: a knowledge representation schema for design. AI Mag. **11**(4), 26–36 (1990)
15. Kan, J.W.T., Gero, J.S.: Using the FBS ontology to capture semantic design information in design protocol studies. In: McDonnell, J., Lloyd, P. (eds.) About: Designing. Analysing Design Meetings, pp. 213–229. Taylor & Francis, New York (2009)
16. Kan, J.W.T., Gero, J.S.: Can entropy indicate the richness of idea generation in team designing? In: Digital Opportunities, 10th CAADRIA conference, New Delhi, India (2005)
17. Kan, J.W.T., Gero, J.S.: Acquiring information from linkography in protocol studies of designing. Des. Stud. **29**(4), 315–337 (2008)

18. Kan, J.W.T., Gero, J.S.: Studing software design cognition, In: Petre, M., Hoek, AVd (eds.) Software Designers in Action: A Human-Centric Look at Design Work. Chapman Hall, London (2009)
19. Ericsson, K.A., Simon, H.A.: Protocol Analysis: Verbal Reports as Data. MIT Press, Mass (1993)
20. Gero, J.S., Mc Neill, T.: An approach to the analysis of design protocols. Des. Stud. **19**(1), 21–61 (1998)
21. Atman, C.J., et al.: A comparison of freshman and senior engineering design processes. Des. Stud. **20**(2), 131–152 (1999)
22. Suwa, M., Tversky, B.: What do architects and students perceive in their design sketches? A protocol analysis. Des. Stud. **18**(4), 385–403 (1997)
23. Akin, O.: Psychology of Architectural Design. Pion, London (1986)
24. Gero, J., Tang, H.-H.: The differences between retrospective and concurrent protocols in revealing the process-oriented aspects of the design process. Des. Stud. **22**(3), 283–295 (2001)
25. Dorst, K., Dijkhuis, J.: Comparing paradigms for describing design activity. Des. Stud. **16**(2), 261–274 (1995)
26. Kuusela, H., Pallab, P.: A comparison of concurrent and retrospective verbal protocol analysis. Am. J. Psychol. **113**(3), 387–404 (2000)
27. Ching, W.K., Ng, M.K.: Markov Chains: Models, Algorithms and Applications. Springer, New York (2006)
28. Meyn, S.P., Tweedie, R.L.: Markov Chains and Stochastic Stability. Cambridge University Press, Cambridge (2009)
29. Kan, J.W.T., Gero, J.S.: Exploring quantitative methods to study design behavior in collaborative virtual workspaces. In: New Frontiers, Proceedings of the 15th International Conference on CAADRIA (2010)
30. Jiang, H.: Understanding senior design students' product conceptual design activities—a comparison between industrial and engineering design students. National University of Singapore, Singapore (2012)
31. Gero, J.S., Kannengiesser, U.: Commonalities across designing: empirical results. In: Proceedings of 5th International Conference on Design Computing and Cognition, College Station (2014)
32. Gero, J.S., Jiang, H., Vieira, S.: Exploring a multi-meeting engineering design project. In: Chakrabarti, A., Prakash, R.V. (eds.) ICoRD'13 conference, Springer, India (2013)
33. Kim, M.J.: The effects of tangible user interfaces on designers' spatial cognition key centre of design computing and cognition. Faculty of Architecture, Doctor of Philosophy (2006)
34. Nguyen, L., Shanks, G.: Using protocol analysis to explore the creative requirements engineering process. Information Systems Foundations Workshop, pp. 133–151. ANUE Press, Canberra (2006)
35. Pourmohamadi, M., Gero, J.S.: LINKOgrapher: An analysis tool to study design protocols based on FBS coding scheme. In: Culley, S., et al. (eds.) Design Theory and Methodology, pp. 294–303. Design Society, Glasgow (2011)
36. Fowler, M.: Patterns of Enterprise Application Architecture. Addison-Wesley, Reading (2003)
37. Gamma, E., et al.: Design patterns: abstraction and reuse of object-oriented design. In: Manfred, B., Ernst, D. (eds.) Software Pioneers, pp. 701–717. Springer, New York (2002)
38. Alexander, C.: The Timeless Way of Building. Oxford University Press, Oxford (1979)
39. Razzouk, R., Shute, V.: What Is Design Thinking and Why Is It Important? Rev. Educ. Res. **82**(3), 330–348 (2012)