Paperless Grammars

Athanassios Economou and Thomas Grasl

Abstract A workshop in formal composition using machine-based specifications of parametric shape rules is presented. The workshop is structured along two different trajectories: one starting from existing grammars and one starting from scratch, and both in a rising complexity in the specification of the rules and the ways they affect design. Rules, productions and designs in corresponding languages illustrate the findings. A speculation on a new design workflow whereas the designers seamlessly design and test their rules within their design processes is briefly discussed in the end.

Keywords Shape grammars · Shape computation · Computer implementation
Formal composition · Rule-based design · Symmetry

1 Introduction

Designing grammars is not different from designing any other artefact. The formidable shape grammar computations published at the early 70s and 80s showcased an intellectual edifice that was meant to be appreciated and used as a complete, flawless project—in point of fact, a visual machine that could exemplify logical rigor and stylistic consistency. The manual execution of the sleek and carefully crafted rules of the ice-ray grammars [1], the Palladian grammar [2], the Mughal garden grammar [3], the Japanese tearoom grammar [4], the Hepplewhite chairs grammar [5], the Terragni façade grammar [6], and so many more, whether in paper and pencil, or painstakingly in the 90s and 00s simulated in AutoCAD or a similar software always provided a clear window of the precision, clarity and insight that each grammar respectively brought in the constructive understanding of the class of artefacts that the grammars modelled.

A. Economou (✉)
College of Design, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: economou@gatech.edu

T. Grasl
SWAP Architekten, Vienna, Austria
e-mail: tg@swap-zt.com

Still, there has always been a great part of the design process that the grammars themselves always left polemically out of the question, namely, the design process of the design of the grammar itself. This is not the same with the ability of a grammar to capture the design process that characterizes the design of a class of artefacts; a good example would be the relation of the ice-ray grammar and the ice-ray lattice Chinese windows [1]. Clearly, the grammar captures the design processes that most probably have been followed by the medieval Chinese artisans. Here instead, the question shifts in the design of the rules themselves: Is it important to know how the author (the shape grammarist, not the Chinese artisan) came up with these rules and not some others? Did she try some other approach that did not work? Are there better ways to deal with specific kind of problems than others? Are these rules applicable only to the specific problem they address in that particular grammar or can they be applicable to other sort of problems in other domains? And such questions do not remain on the analysis of the thought processes of the author side and role; even more questions arise when the questions shift to the user role and the student at large who studies these rules to generatively understand the language at hand. If a computation fails, it is clear who is to blame? Did the user of the grammar followed rightly all the rules when she attempted to make a production of her own? Did he erase the labels properly? Did he apply the rules in all possible parts before he would move one to the next stage? If a mistake happens, is it in the designer's side? Is it in the publisher's side? Is it on the reader's side? And so on. A most satisfying effort, and a great lesson to learn for everyone involved, but error prone too.

And yet, all such reservations are immaterial; the most important point in this process of seeing, understanding and doing design is that one never learns a rule by looking at a page but by actively drawing the rule and seeing with her own eyes the possibilities that this rule opens up, both in the parts of the design that the rule can apply to and in the effects that this rule infers on the design once it is applied.

The machine-based specification of shape rules and parametric shape rules in rule schemata has been the subject of a growing field in computational design and computer-aided design discourse [7–16]. Among these shape grammar interpreters, the parametric shape grammar interpreter GRAPE [16] has emerged as a robust computational framework that allows the itinerant decomposition of existing designs in various non-anticipated ways and the visual specification of labelled shape rules and labelled parametric shape rules. The technical specifications of the application and its ongoing development is given in [16–18]. The ways that these types of rules could begin to work in design practice and research is the subject of this work.

The setting for the testing of the GRAPE parametric shape grammar modeller involved a successive series of workshops in an academic setting having students trying out rules, known and new ones, and exploring on their own the expressiveness of the rules and the software itself. The workshops were structured along two different trajectories: one starting from existing grammars and one starting from scratch, and both in a rising complexity in the specification of the rules and the ways they affect design. The key idea in the first series of studies was the implementation of known rules in the literature—and there are many to admire—to produce the designs that have manually been produced in the original papers, additional ones that are in principle possible by the original grammars, and new ones by purposefully playing with the rules, i.e. the transformations under which the rules apply, the assignment of the values of the parametric shape, the shapes themselves in the schemata rules, and so on. The key idea in the second series of studies was the implementation of new rules designed from scratch to produce the designs that were selected to provide the corpus for the grammar, additional ones that are in principle possible by the grammars, and new ones by purposefully playing with the rules. Both types of studies relied extensively in a critical comparison between shape rules and rule schemata in terms of their expressive and productive features in design inquiry [19]. A brief description of both experiments follows below along with some examples from each case.

2 From Rules to Rules

The first series of studies foregrounds a hands-on constructive understanding of existing grammars in the literature. The students are encouraged to copy rules from the literature, see on their own how the rules were supposed to apply in the original design setting and what they can make, and once they get a command of their expressiveness and generative power, start editing them in a variety of ways to make them their own. The range of techniques used to edit the rules is open-ended; once the copy and implementation of the rule has been considered successful, the students are encouraged to alter the rules in some way to accommodate design ideas and insights that might have emerged through the computations with the existing rules. A sample of three studies is given below to showcase some of the findings of the workshop in GRAPE. Each grammar is briefly discussed in terms of some of its key rules, one or more productions and a list of designs automatically produced in GRAPE. The authors of the grammars are specified in parentheses within the brief account of each project.

2.1 Pinnacle Grammar

The nested square rule [20] is one of the most popular rules in the shape grammar discourse because it shows a nice example of visual recursion and an effortless visual transition from squares to triangles, pentagons, hexagons and all sorts of other more exotic shapes [20, 21]. The workshop builds up on the initial rule in Fig. 1 cast in the rule schema $x \rightarrow x + t(x)$, for t a similarity transformation consisting of a product of a rotation and a scale transformation, to a series of identity rules cast in the rule schema $x \rightarrow x$ to facilitate the understanding of the seamless shifting between different decompositions in the design, and ends with the design of a new series of rules cast in the rule schemata $x \rightarrow x + y$ and $x \rightarrow y$ whereas the spatial relation between the two squares is considered now as a starting point for the design of a new

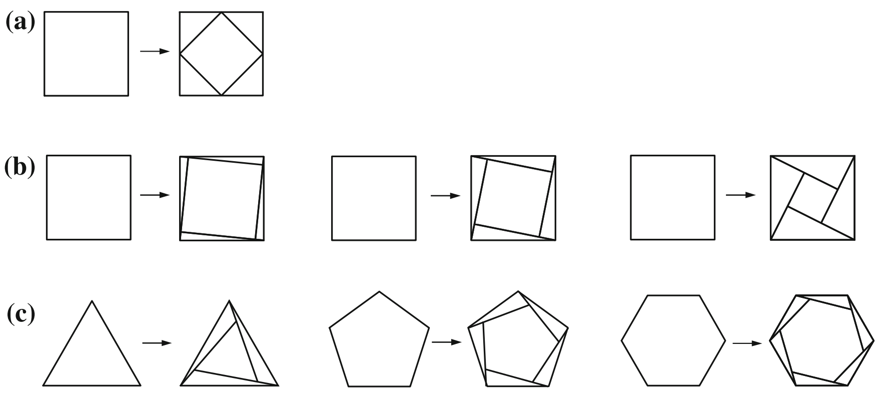

Fig. 1 The pinnacle grammar (Hong Tzu-Chieh) **a** Original nested square rule; **b** Substitution of the inside smaller square by a series of whirling squares with rotational handles; **c** Extension of the parametric rule to the class of regular polygons

design idea. The pinnacle grammar takes on the lessons found in the recursive point symmetry of the initial shape rule in Fig. 1 and extends them to explore natural growth exhibiting 3-fold, 4-fold, 5-fold, 6-fold point symmetry, and so forth. The grammar is not meant to be exhaustive for the generative description of the form of specific class of biological formations but instead it uses these initial symmetrical configurations as a departure point for the exploration of designs that all have a point symmetry in a nested configuration one within the other. The rules of the grammar are extremely simple and their visual interest lies in their itinerant recursive applications, a front that computers—and GRAPE, excel. Some of the rules of the pinnacle grammar are shown in Fig. 1; clearly many more are possible and a more definitive treatment still remains to be undertaken.

The itinerant applications of the rules of the pinnacle grammar in different versions and settings is rewarding. A series of productions and final designs in the language is shown in Fig. 2 for the regular triangle, square, pentagon and hexagon. Note that all the designs shown are produced by versions of the rules that erase the circumscribing square every time they apply. The elimination of the bounding exterior polygons foregrounds the emergent spatial relations in the centre of the shape and the spiral growth of the pattern.

Significantly all the derivations in Fig. 2 are produced by a singular shape rule applying in an identical manner every time. A much greater variety can be achieved by applying the same parametric shape rule under different assignments and the applications of these rules can be orchestrated and altered in significant ways. For example, the rules in Fig. 1b show three different instantiations of the same parametric shape rule for different sizes of inserted squares and different degrees of rotation within the original circumscribing one. The different sizes and rotations produce distinct series of spiral growth and even more their rhythmical interchange within each size and among all sizes produce interesting visual counterpoints not entirely anticipated till the moment the rules are executed by the machine. A series of designs

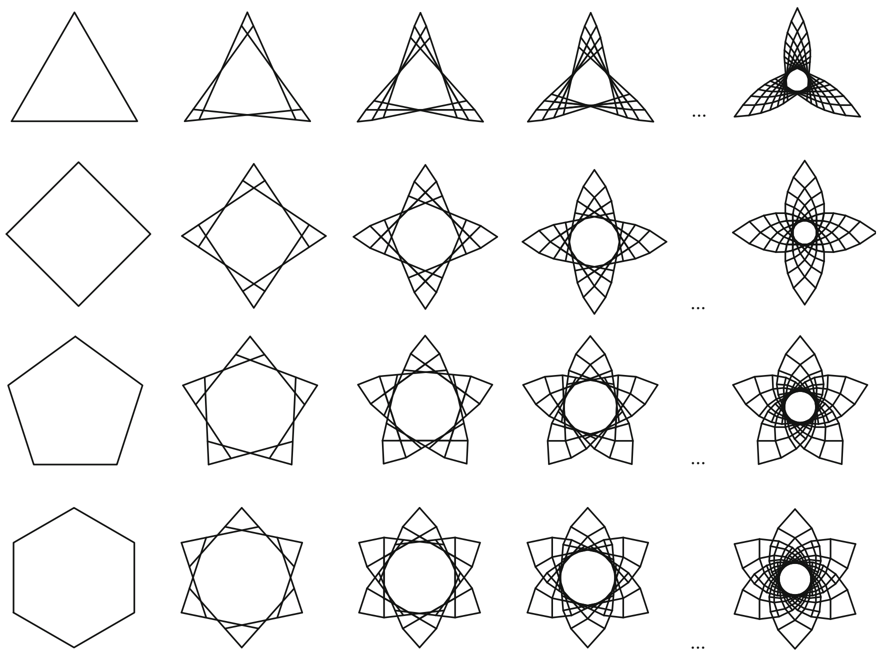

Fig. 2 The pinnacle grammar (Hong Tzu-Chieh). Four productions showcasing different parametric shapes inserted as predicates in the same flower schema rule

in the language of the pinnacle grammar featuring four-fold symmetry is shown in Fig. 3.

2.2 *Checkerboard Lattice Grammar*

The checkerboard lattice designs, a specific subset of the Chinese lattice designs that fill window frames [1] provide a great initial framework to discuss key ideas in formal composition including repetition, recursion, modularity, grid, frame, boundary, proportion, symmetry, and many more. The additional constraint of a labelled grid that gets filled by different modules whose combinations make produce predictable or unpredictable results provides a rewarding visual context for taking on in a constructive way the ideas of formal analysis and synthesis and the specification of a whole series of diverse designs that all share a common framework. The two schemata typically used for the casting of the labelled shaped rules are the schema $x \rightarrow x + t(x)$ for the generation of the underlying rectangular lattice, and the schema $x \rightarrow y$ for the generation of the different substitute motifs upon the square or the rectangular module. Figure 4 shows three of the shape rules of the ice-ray lattice grammar redrawn in GRAPE. Figure 4a shows two of the shape rules of the original grammar that define the underlying structure of the design: the first adds a second labelled

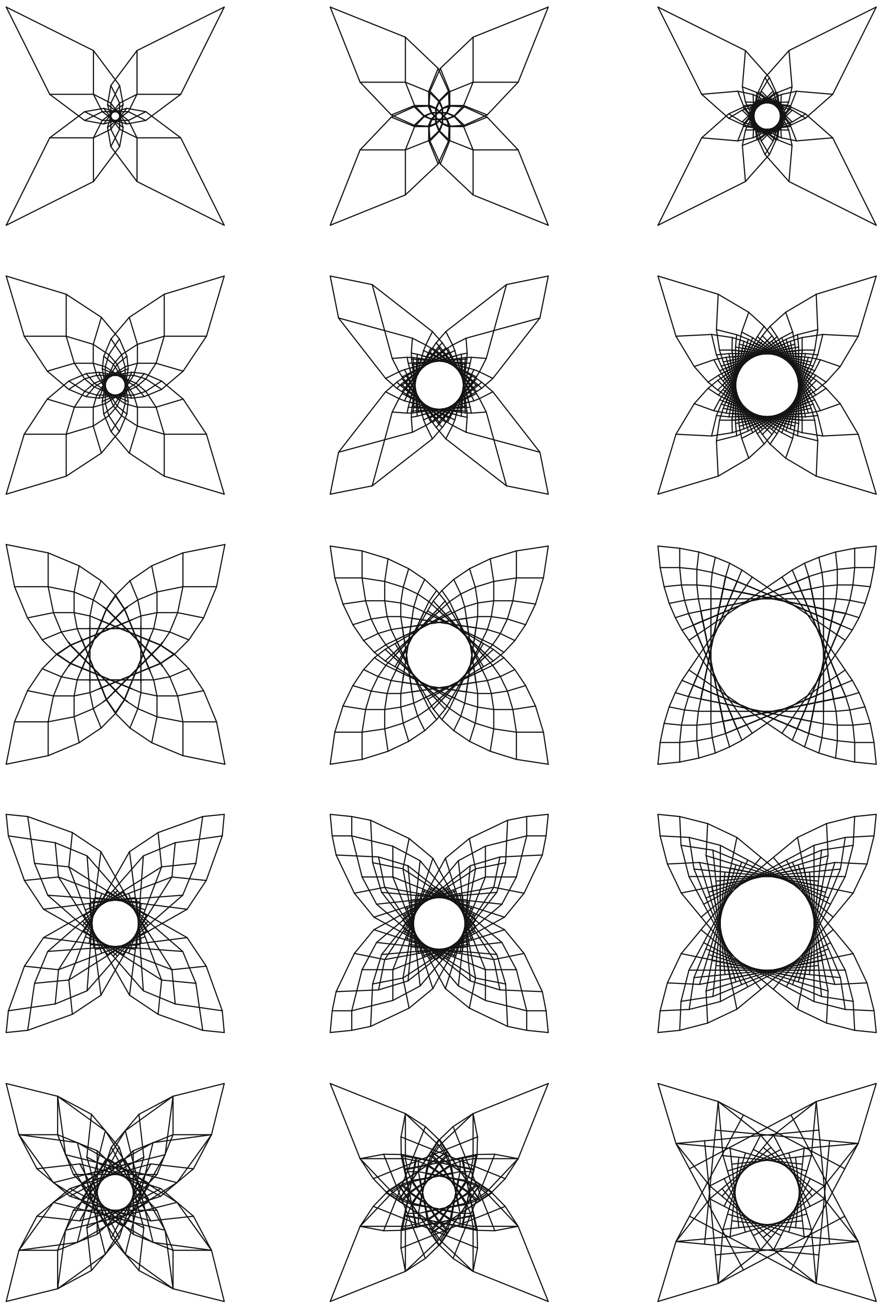

Fig. 3 The pinnacle grammar (Hong Tzu-Chieh). Some designs in the language of the pinnacle grammar featuring four-fold symmetry

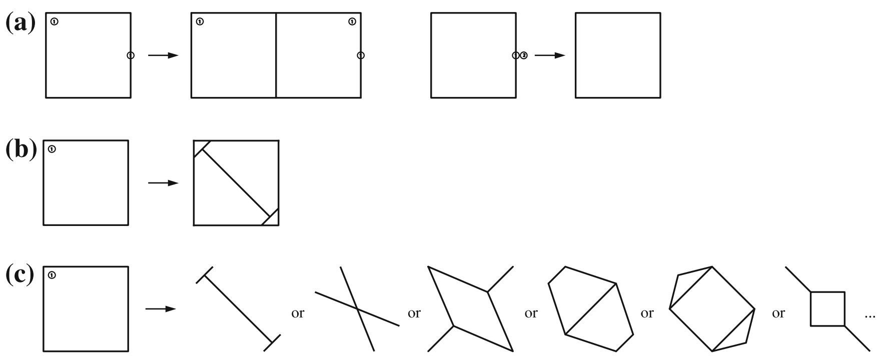

Fig. 4 The checkerboard lattice grammar (Stephanie Douthitt): **a** Shape rules 1 and 5 in [1]; **b** Termination rule (Rule 6 in [1]); **c** Some possible new termination rules. The RHS of the shape rules are given in a single array for brevity

square to an initial square and removes the label from the right edge of the square of the left-hand side (LHS) to the right edge of the second square at the right-hand side (RHS) (Rule 1 in the original grammar); and the second removes both labels from the square of the LHS of the rule (Rule 5 in the original grammar). Figure 4b shows the terminal rule that substitutes the labelled square in the LHS of the rule with the square with the diagonal pattern in the RHS of the rule (Rule 6 of the original grammar). Clearly, the differences of the checkerboard lattice grammar here with the original are few and immaterial: the line shapes of the original grammar have been substituted by squares; the circular black labels have been substituted by white labels with the symbol 1 and the triangular black labels by white labels with the symbol 2; and the label deletion rule has been substituted by the erasure labelled shape rule. In other words, what is emphatically insisted in this detailed account is to show that the visual specification of the labelled shape rules in the original publication, can be copied and drawn in a straightforward way in the GRAPE environment.

Figure 5 provides a complete derivation of the lattice described in the original ice-ray grammar. The derivation here is based on a version of a grammar that does not use the initial shape of the frame but uses instead the rules 3 and 4 of the original grammar to achieve the boustrophedon deployment of the grammar, that is, the bidirectional production unfolded from left-to-right and left-to-right in alternate lines. The application of the termination rule in the complete lattice takes place in one square at a time in a random manner. Still, what is more interesting is the different treatment of the sixth rule in the grammar: The series of designs in Fig. 6 showcase nicely the substitution of this rule with the shape rules in Fig. 4c that create very diverse designs. In this series, the shape in the RHS of the rule was drawn and tested on the fly in GRAPE to test the design, see how the symmetries of the RHS partake of the overall symmetry of the lattice, and explore different versions of the rules in different schemata too, for example, eliminate the frame of the square module to foreground and enable other relations in the overall design including quadrilaterals, pentagons, hexagons, octagons and so forth.

2.3 Mughal Gardens Grammar

The Mughal gardens grammar is one of the earliest labelled parametric shape grammars in print [3] and one of the most didactic showcasing in an exemplary way the layering of formal analysis in shape grammar discourse starting from a concept, here the idea of the paradise, to the history of its design, its geometry, and finally the postulation of a formal grammar that can capture its salient features. The implementation of the Mughal garden is by no means a simple feat as it requires the implementation of several rules and relies extensively on parametric definitions of rules in GRAPE. The most challenging aspect of this implementation has been the implementation of the rules 19, 20 and 21 in the original grammar that are offered there as three samples of treating the corner of the inner squares of the garden to allow for inflections in three distinct ways. The resolution of these conditions has repercussions for the rest of the rules in the grammar. A sample of the parametric shape rules drawn explicitly in GRAPE to account for the instantiation of these conditions are shown in Fig. 7.

Most rules in the grammar are cast in the schema rule $\sum t(x) \rightarrow \sum t(x+y)$ whereas $t(x)$ is one of the transformations of the symmetry group of the square, the underlying framework of the garden and $\sum t(x)$ the complete symmetry group of the square. In this sense, a rule that may apply, say, in the upper left quadrant of the design has to apply to other corresponding parts of the design and depending on the exact location of the part, the rule may apply in seven more locations (for a total of 8) or three more (for a total of 4), or one more (for a total of 2) if the rule has already some symmetry built in that aligns with the overall symmetry of the part. A sample of a complete production in the grammar is given in Fig. 8.

The implementation of the complex grammar in GRAPE required some restructuring work in the underlying engine so that the software would be able to execute all the possible matchings of a rule in a design and expedite the design derivation. Figure 9 show instances of nine Mughal gardens in the language that represent instances of the nine configurational unique possibilities produced by the grammar. Clearly, architectural elements such as the ornamentation of the reservoir of water at the centre of the canals in a square or an octagonal form, the ornamentation of the endings of the water canals, and the parameterization of the dimensions of the borders and the canals, can provide a rich palette for an expressive language with unique characteristics. Still, while this first implementation resulted in a series of designs that indeed captured the initial grammar, there were indeed great difficulties with the implementation and the running time of execution of each rule, and certainly the project remains to be redone in a way that can conclusively demonstrate the visual subtlety of the original parametric shape grammar.

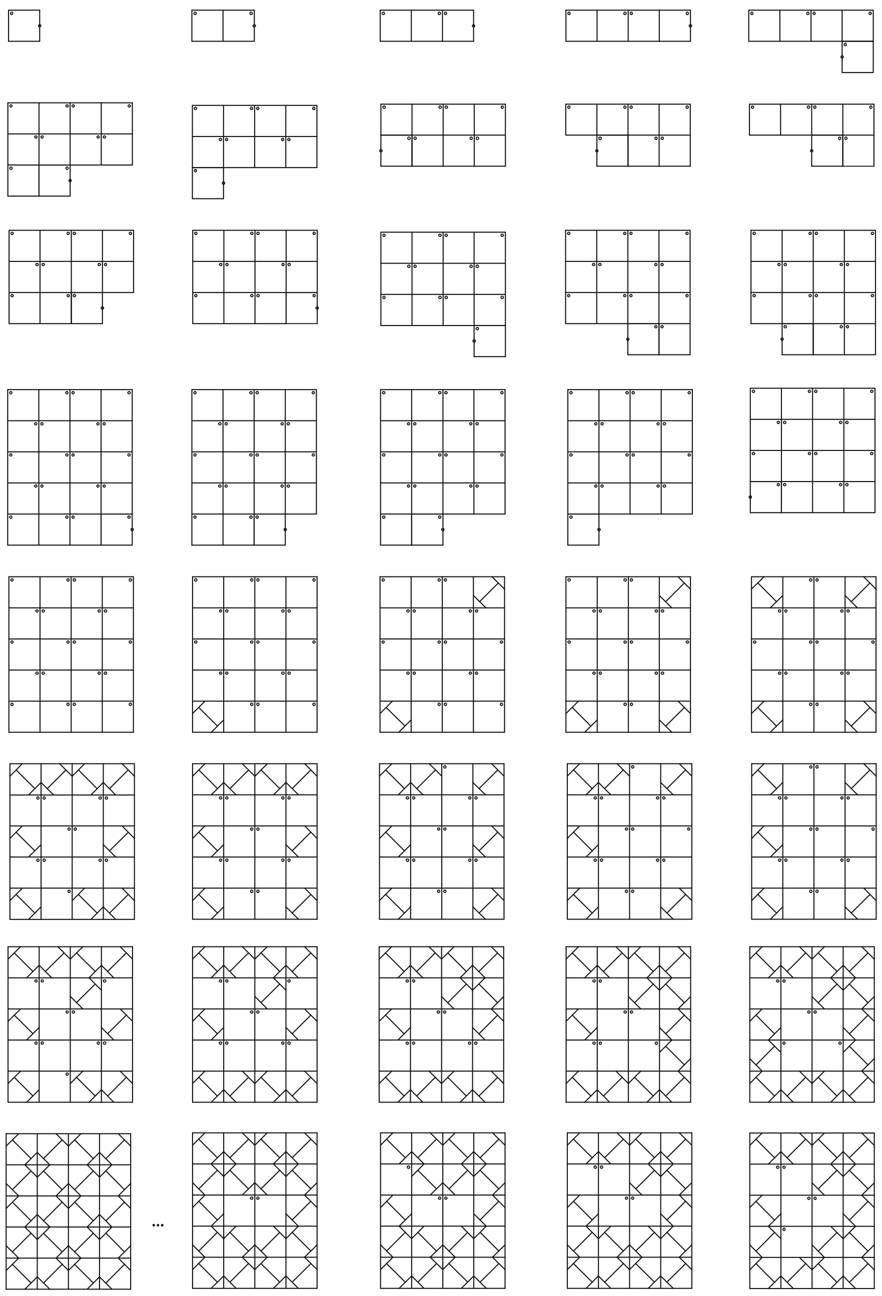

Fig. 5 An automated derivation of the original ice-ray checkerboard lattice grammar (Stephanie Douthitt). The productions are laid out automatically in a boustrophedon manner

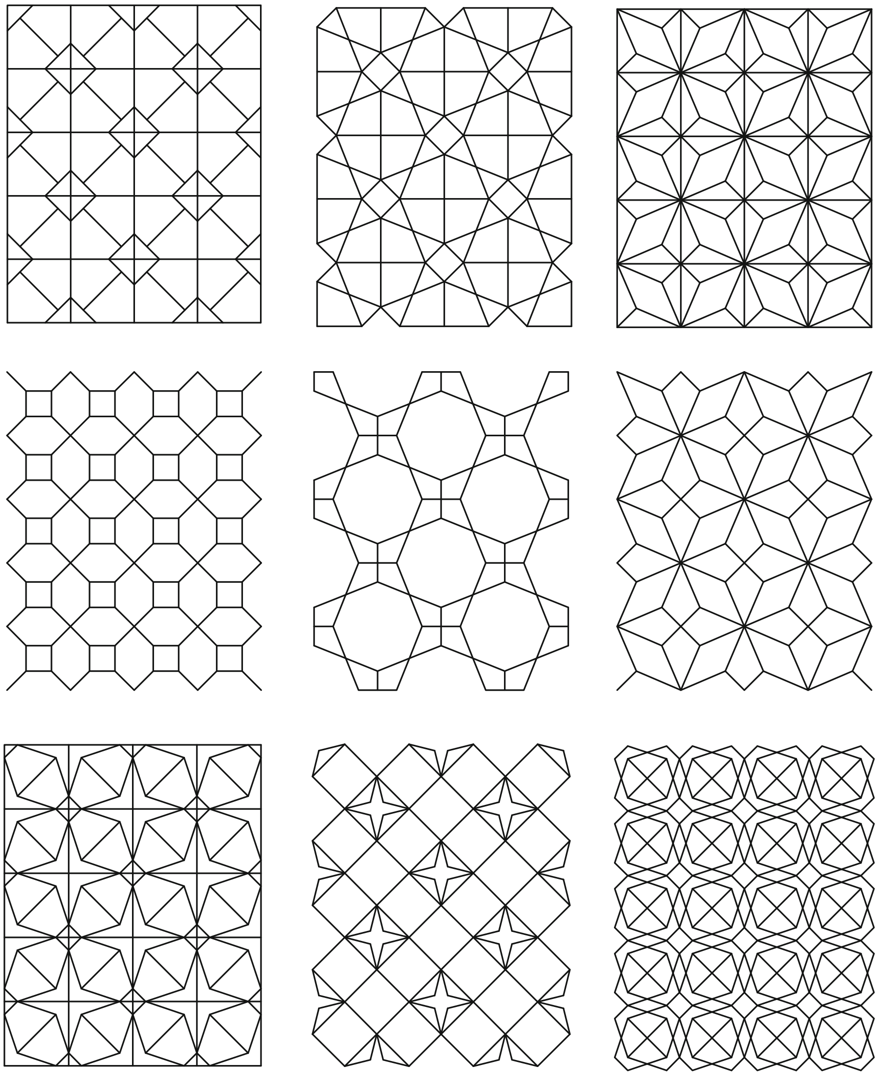
Fig. 6 Some designs of the checkerboard lattice grammar (Stephanie Douthitt)

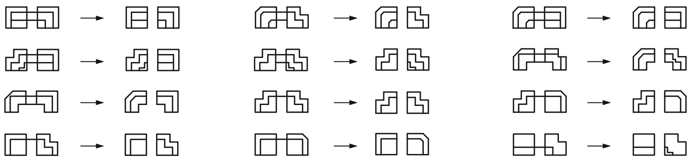
Fig. 7 Some of the parametric rules of the Mughal garden grammar (Nirvik Saha).

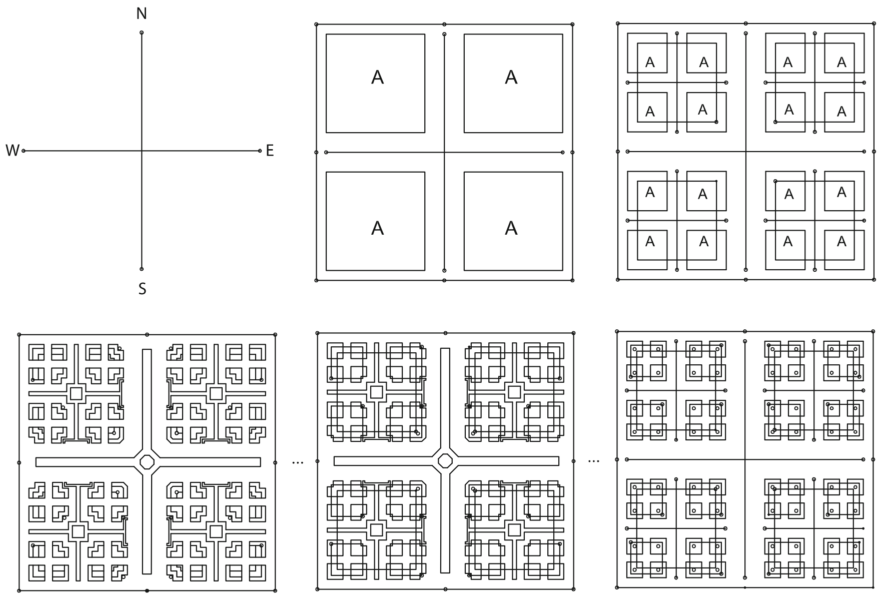

Fig. 8 The Mughal garden implemented in GRAPE: An automated production depicted in a boustrophedon manner (Nirvik Saha)

3 From Designs to Rules

The second series of studies foregrounds a hands-on constructive inquiry on new grammars in the literature. The students are encouraged to come up with a list of existing artefacts or buildings they want to explore, or alternatively, a brief for the design of some new artefact or building. In either case, the rules for the generative description of the existing designs or the new ones, do not exist and the students have to think them through, design and test them in GRAPE. The goal in this exercise is not the complete formal specification of a set of artefacts or buildings, existing or new; rather, it is the testing of how existing artefacts or briefs can be used constructively in the design of new rules. Again, the range of techniques used to design rules is open-ended; once the design of a rule has been deemed successful, the students are encouraged to take advantage of the rule in some way to accommodate additional design requirements and constraints that might have been observed in the corpus or given explicitly in the design brief. A sample of three studies is given below to showcase some of the findings of the workshop in GRAPE. Each grammar is briefly discussed in terms of some key rules, one or more productions, and a list of designs produced in GRAPE. The authors of the grammars are specified in parentheses within the brief account of each project.

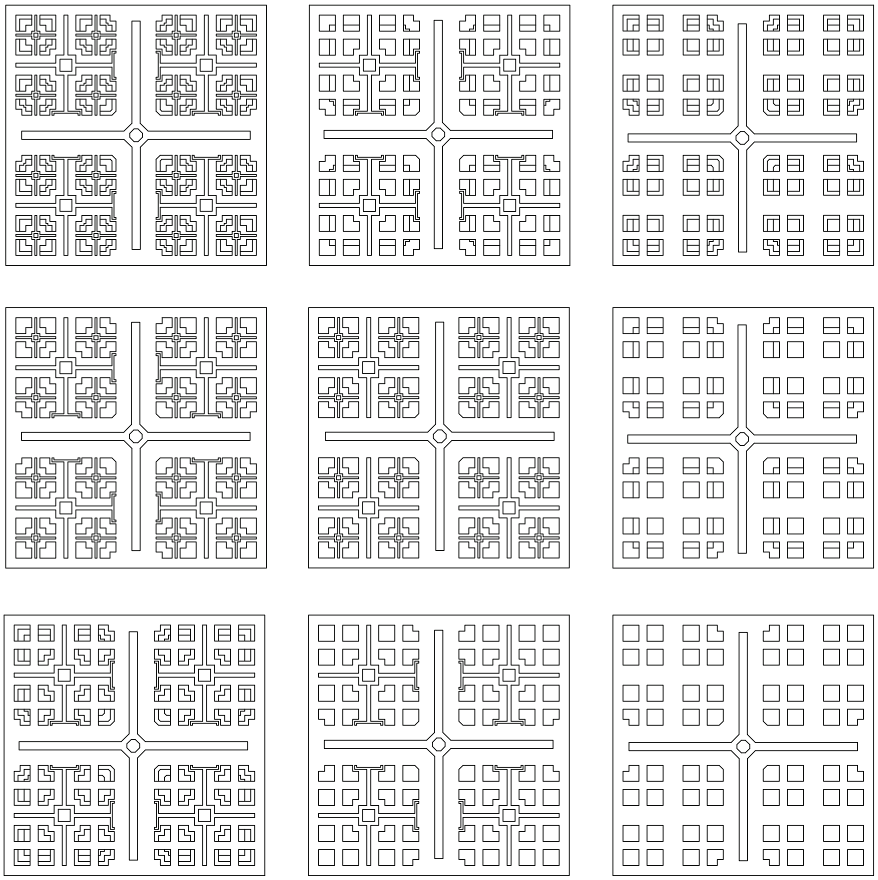

Fig. 9 The Mughal garden implemented in GRAPE: The nine basic configurations of borders and canals in the Mughal garden language (Nirvik Saha)

3.1 Ad Quadrant Grammar

The ad quadrant grammar takes on the ad quadrant division schema, a basic compositional schema used in the design of artefacts, buildings and town plans across many cultures and times as diverse as Hellenistic and Roman ones to twentieth century and contemporary one. The ad quadrant grammar is based on March's Speculation 8 [22] and its range of nucleated and linear distributions from 10% to 90% coverage within an abstract square configuration. Clearly any subdivision of a square by $n \times n$ squares creates a simple division based on the square number sequence and any combination of such divisions nicely illustrates the ideas of recursion and emergence too, the two

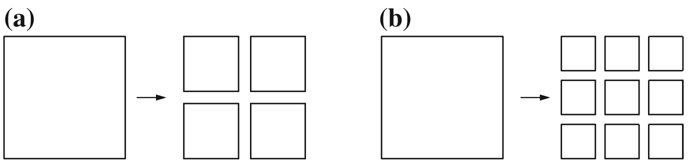

Fig. 10 Two rules in the ad quadrant grammar (Abigail Smith): **a** Division to 4 squares; **b** Division to 9 squares

paramount characteristics of shape grammars, and the ability of GRAPE in capturing both in the application of the division algorithms upon the nested smaller copies of the square. Two rules of the grammar are given in Fig. 10 for a division of a square to four and nine squares respectively.

The simplicity of these two rules hide the delightful complexity they can generate once they apply recursively and at different scales within the design production. The straightforward application of each rule creates the familiar series of configurations 1, 4, 16, 64, 256, …, and 1, 9, 81, 729, 6561, …, respectively. A production in the grammar is given in Fig. 11. The first rule that instantiates a square is encoded in the software but omitted here for brevity. The production here applies for the total matches of the LHS rule in the design simulating a fractal derivation: Because the symmetries of the LHS and the RHS are the same and embedded one-to-one the rule applications match the number of discrete squares present at each stage. The production is generated in a boustrophedon manner as an output at the end of the derivation and the double arrows between each step are omitted.

More interestingly, the combinations of the rules a and b in the Fig. 10 in various sequences, say, ababab…, bababa…, aabbaa…, bbaabb…, aaabbb…, bbbaaa…, and so forth, and in other non-regular series too, immediately shows the inexhaustible possibilities that emerge out of the different sequence of rule applications and the corresponding designs all with their own visual characteristics. And clearly, the incorporation of specific ratios between the diminishing squares produces interesting densities and frit effects in the overall configuration. More importantly, the definition of the rules in the schema $x \rightarrow x + t(x)$ rather than the $x \rightarrow t(x)$ retain the original circumscribing square from the LHS to the RHS of the rule and produce a whole new range of designs that show interesting and complex patterns of emergent spatial relations between squares at various scales. A sample of designs in the language of the ad quadrant grammar is shown in Fig. 12. Additional designs utilizing square numbers for prime numbers using any of the techniques outlined above are readily available too.

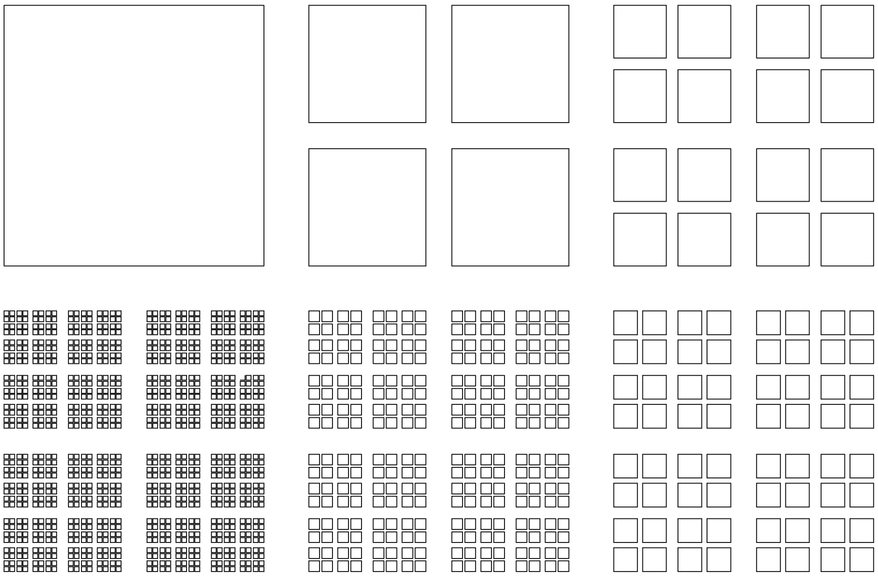

Fig. 11 A production in the ad quadrant grammar (Abigail Smith)

3.2 MCE Grammar

The MCE grammar takes its inspiration from M. C. Escher's recursive tilings [23]. The grammar is not meant to be exhaustive for some given corpus of the Dutch graphic artist but playfully uses his idea of substitution regular shapes or the division of the plane with some spatial idea or motif that typically recalls a biomorphic association. The complete grammar consisting of 6 rules is given in Fig. 13.

The first rule in Fig. 13a captures the initialization of the grammar and the instantiation of the initial shape. The core of the grammar is the rule in Fig. 13b that showcases the replacement of an isosceles Root2 (R2) triangle with three copies of itself, an identity R2 triangle and two smaller R2 triangles whose sum measures exactly the original one to produce a square. Interestingly, the smaller triangles within the original R2 triangle function as labels to guide the computation. The rule in Fig. 13f produces the visual stylistic effect of the Escher language by substituting the original R2 triangle with a complex shape featuring a half-turn line along the hypotenuse of the R2 triangle and a meandering right-angle line in the right angle of the R2 triangle. Clearly other shapes are possible and several were explored interactively in GRAPE before the designer committed to this final substitution. A production in the grammar is given in Fig. 14. The production is generated in a boustrophedon manner directly in the software and the double arrows between each step are omitted.

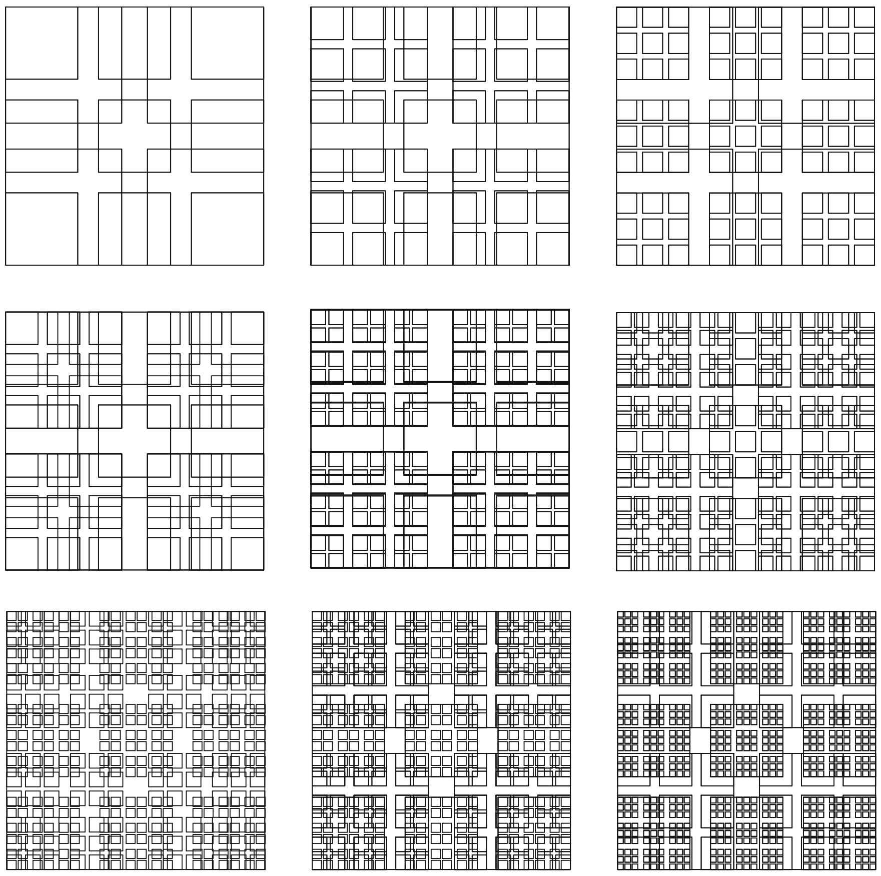

Fig. 12 Some designs in the ad quadrant language (Abigail Smith). All designs use the rules in Fig. 10 defined in the schema rule $x \to x + \sum t(x)$

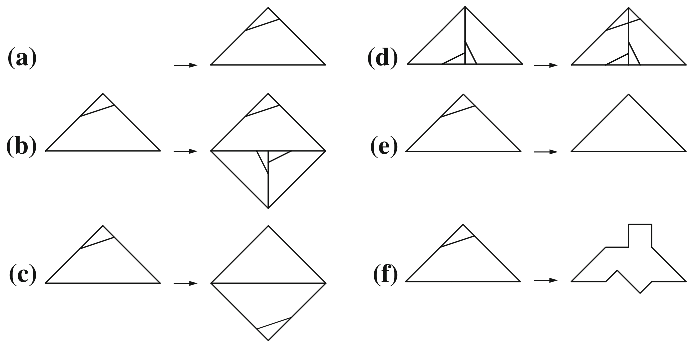

Fig. 13 The six rules of the MCE grammar (Kelsey Kurzeja)

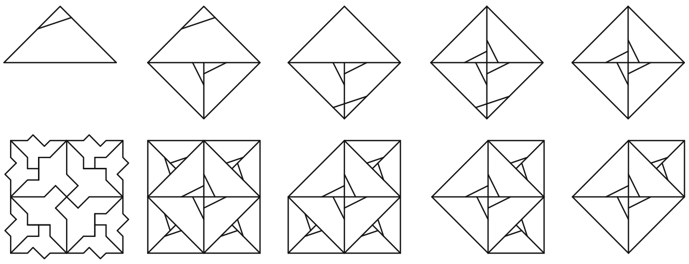

Fig. 14 A production in the grammar MCE grammar (Kelsey Kurzeja)

The derivation of the design in Fig. 14 shows nicely the reasons of the design of the other rules in the grammar and their role in guiding the generation of the design by reflecting or erasing the smaller triangles that function as labels in diverse ways. A catalogue of some possible designs in the language is given in Fig. 15. These designs showcase alternative ways of using the grammar either by applying shape rules manually to specific parts in the design, or by applying a shape rule simultaneously to all possible matches within the design.

3.3 The Dirksen Grammar

The Dirksen grammar provides a generative specification of Mies Van Der Rohe's Everett McKinley Dirksen United States Courthouse in Chicago, US [24]. The grammar postulates its basic rules from a set of 135 sketches produced by the office of Mies during the design process of the courthouse and documented in the Mies van der Rohe Archive at the Museum of Modern Art. Significantly, the rules are thought, represented and implemented in GRAPE all in a truly three-dimensional form. The project is quite ambitious in the sense that it attempts to take on several fronts in the formal analysis of the work, and more specifically, cast light in the design process of Mies' office, articulate the compositional aspects of this project that embodied the architect's mature view on architecture and law, showcase the ways that Mies' architectural language is deployed for this building type, and foreground the sectional principles in the arrangement of the program. A sample of the three-dimensional parametric labelled shape rules of the grammar is shown in Fig. 16. The complete grammar consists of 68 parametric rules and is currently under preparation for a formal publication.

The Dirksen grammar is so far the most ambitious project in GRAPE taking on the complexities of writing new rules from scratch to specify a given corpus, parameterizing them fully to be able to capture specific proportional ideas, model everything in three-dimensions in a uniform way and implement all directly in a computer-aided design system. Clearly these ambitions do not come without challenges. All rules here are implemented in the GRAPE for Rhino version and are encoded directly

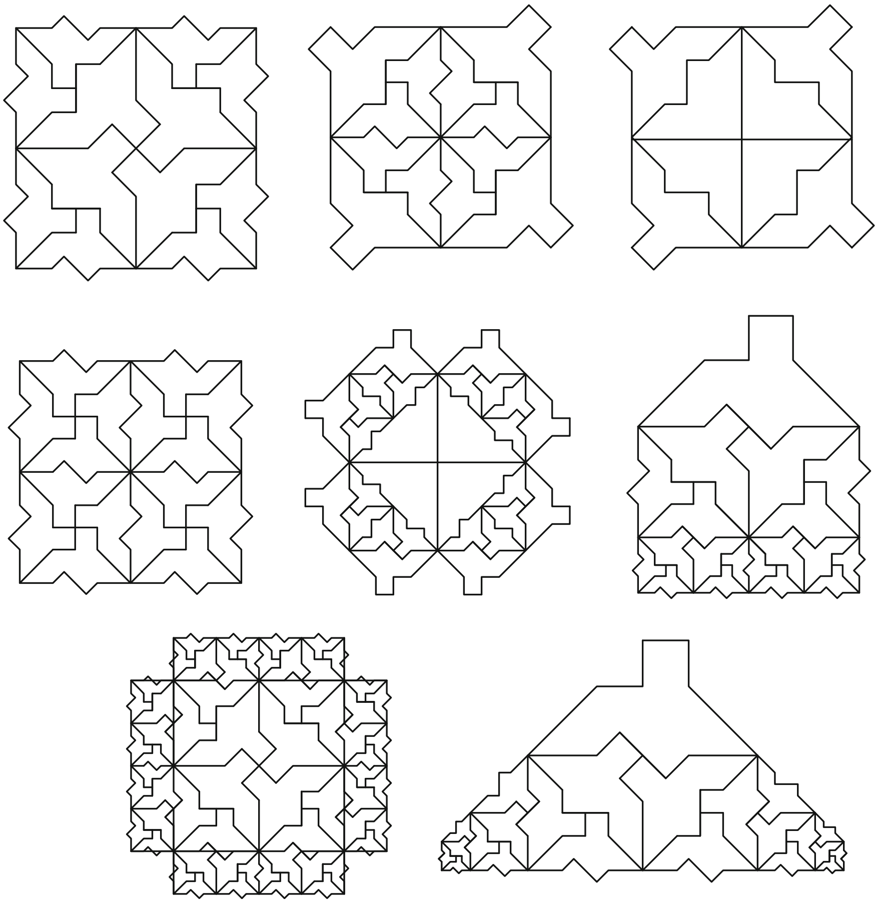

Fig. 15 Some designs in the MCE language (Kelsey Kurzeja)

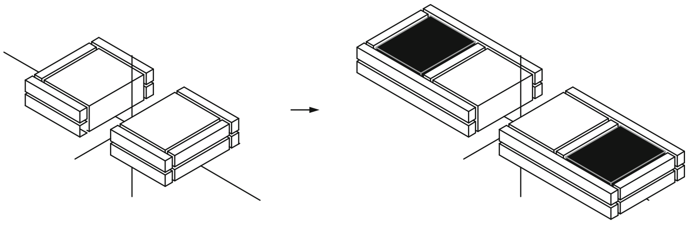

Fig. 16 The Dirksen grammar (James Park). A parametric 3D labelled shape rule that specifies the addition of two courtrooms (in black) in a bilaterally symmetrical manner around the dual service core of the floor plate (Rule 17 in original grammar)

in the scripting environment, an interface quite distinct from the visual one in the web version of the software. Still the current state of the work is very promising in that it manages to produce models that can be exported and used in a variety of other applications for reviewing, rendering, slicing, printing and so forth. Interestingly enough, the majority of the rules in the grammar follow the schema rule $x + R(x) \rightarrow (x + y) + R(x + y)$, for R a reflection, that is, a similar schema rule to the one used in Palladian grammar capturing Mies' rule for bilateral symmetry along the short axis of the courthouse. And interestingly enough too, the Dirksen grammar provides a three-dimensional analog for the Palladian one by modelling the empty spaces of the building and having the walls emerge as poché walls in a specific state of the production. Figure 17 shows a complete production in the grammar starting from the initialization of the grammar to the generation of the complete envelope.

A set of six three-dimensional courthouse models produced by the Dirksen grammar in GRAPE for Rhino are shown in Fig. 18. All models feature a core of 24 courtrooms in various configurations per floor. The model in Fig. 18a shows a 4 courtroom per floor plan in a 1-2-1 variation and 6 double floors. The model in Fig. 18b shows a 3 courtroom per floor plan in a 1-1-1 variation and 8 double floors. The model in Fig. 18c shows a 2 courtroom per floor plan in a 1-1 variation and 9 double floors. The models in Figure (d–f) show corresponding configurations of courtrooms per floor but in different spatial relations. Interestingly, the first three models (a–c) are models that are found in the archives of Mies' office. The last three models (d–f) are theoretically possible designs in Mies' language. All six models are derived by manual applications of the rules of the Dirksen grammar within the GRAPE for Rhino environment and are shown here in sectional axonometric projections manually prepared in Rhino for illustrative purposes.

4 Discussion

A key motivation underlying the work described is the speculation on a new design workflow whereas the designers seamlessly design and test their rules within their design processes. While shape grammars applications have been developed for over 40 years and especially over the last 10 years integrating early pioneering work on shape representation and computation and recent advances in emerging applied technologies including parametric design tools, generative design tools, procedural modelling tools, information modelling applications and rule-based design systems, still no parametric shape grammar interpreter has emerged to address comprehensively the range of issues that have emerged in the discourse over the years. It is precisely what this model is all about: to propose a single engine that takes on effectively the issue of maximal element representation of shape and to manage to technically encode it in a software application that allows the community to freely write, exchange, inspect, and use parametric shape rules in any conceivable way.

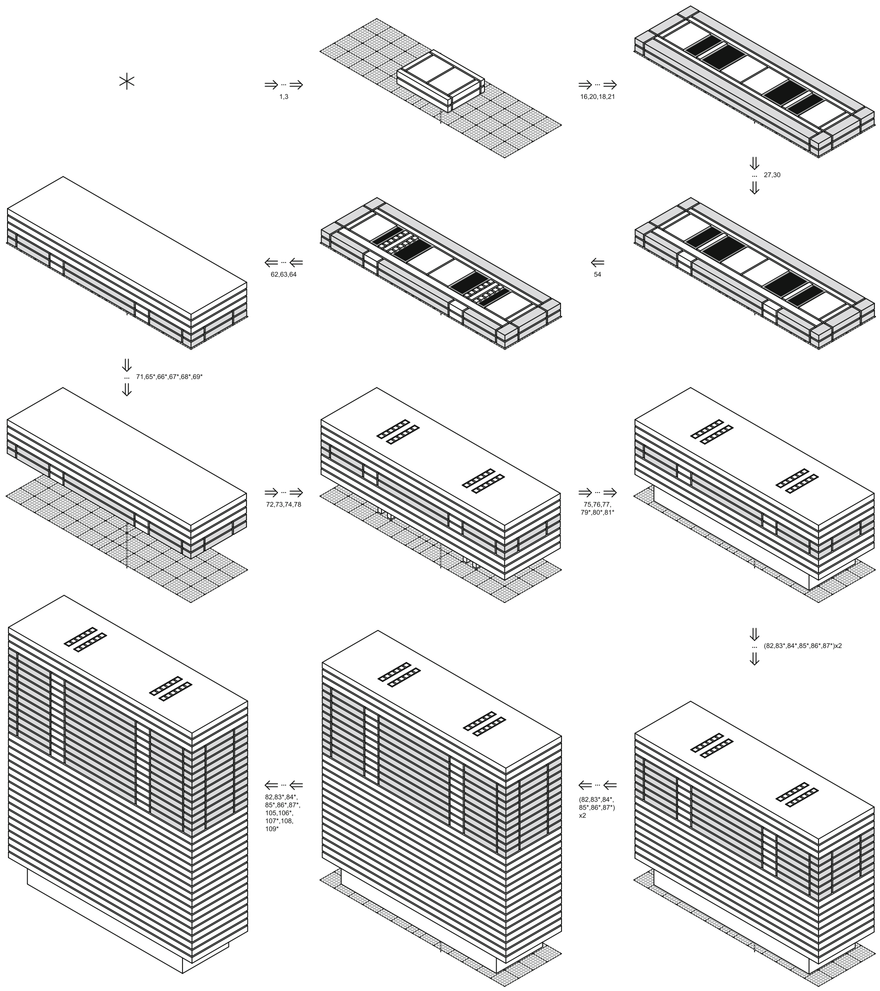

Fig. 17 The Dirksen grammar (James Park). A production in the grammar showcasing a complete derivation of three-dimensional model in Mies' courthouse language

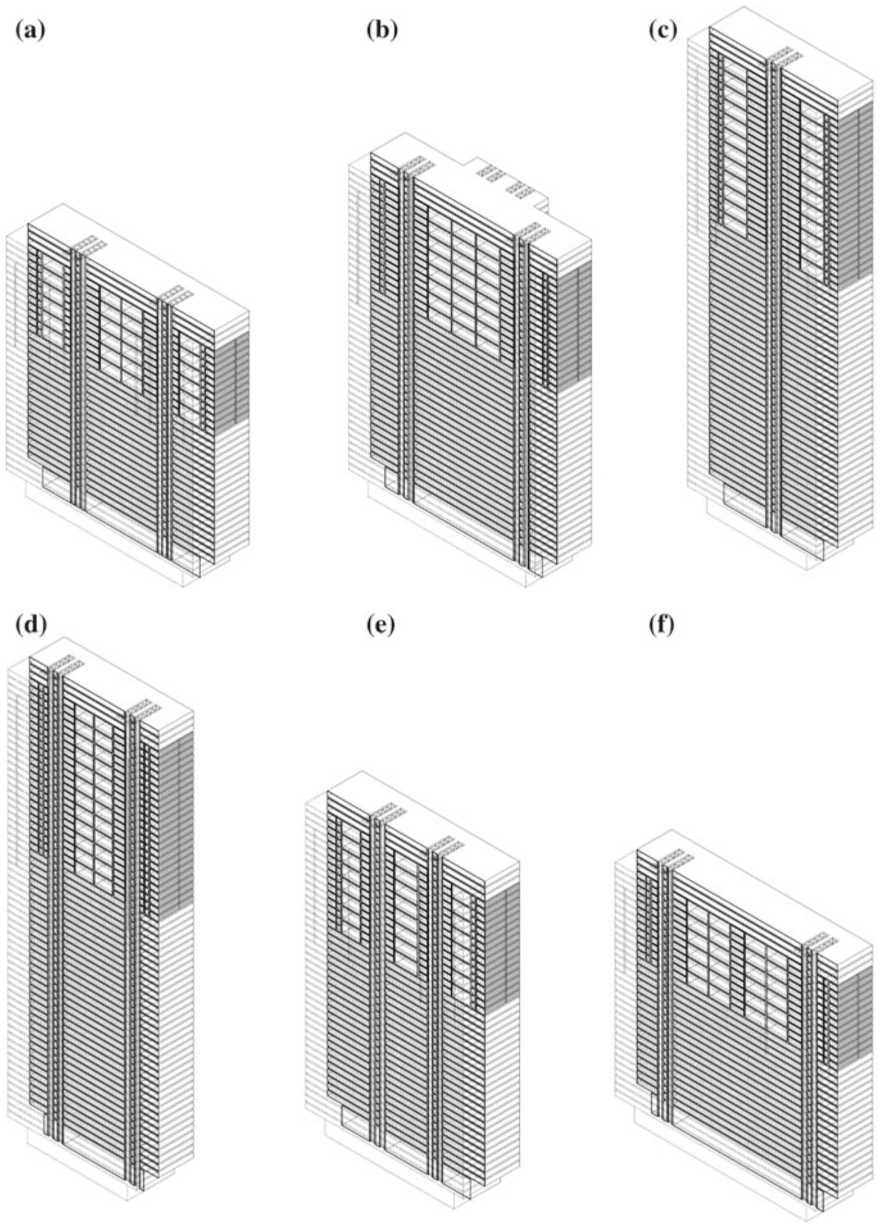

Fig. 18 A set of six sectional axonometric models of Miesian courthouse designs (James Park). All models feature a total of 24 courtrooms in various arrangements. Models (**a**–**c**) showcase designs that are found in the archives of the design process. Models (**d**–**f**) showcase possible hypothetical designs in the language

Acknowledgements We would like to thank the undergraduate and graduate students at the School of Architecture at the College of Design at Georgia Institute of Technology, for their great enthusiasm and hard work during the workshops on shape grammars and their implementation in GRAPE.

References

1. Stiny, G. (1977). Ice-ray: A note on Chinese lattice designs. *Environment and Planning B, 4,* 89–98.
2. Mitchell, W. (1978). The Palladian grammar. *Environment and Planning B, 5,* 5–18.
3. Stiny, G., & Mitchell, W. J. (1980). The grammar of paradise: On the generation of Mughul gardens. *Environment and Planning B, 7,* 209–226.
4. Knight, T. W. (1980). The generation of hepplewhite-style chair-back designs. *Environment and Planning B: Planning and Design, 7*(2), 227–238.
5. Knight, T. W. (1981). The forty-one steps. *Environment and Planning B: Planning and Design, 8*(1), 97–114.
6. Flemming, U. (1981). The secret of the Casa Giuliani Frigerio. *Environment and Planning B: Planning and Design, 8*(1), 87–96.
7. Krishnamurti, R. (1981). The construction of shapes. *Environment and Planning B, 8,* 5–40.
8. Piazzalunga, U., & Fitzhorn, P. (1998). Note on a three-dimensional shape grammar interpreter. *Environment and Planning B: Planning and Design, 25*(1), 11–30.
9. Tapia, M. (1999). A visual implementation of a shape grammar system. *Environment and Planning B: Planning and Design, 26,* 59–73.
10. Gips, J. (1999). Computer Implementation of Shape Grammars. In *NSF/MIT Workshop on Shape Computation*. http://www.shapegrammar.org/implementation.pdf.
11. Chau, H. H., Chen, X., McKay, A., & de Pennington, A. (2004). Evaluation of a 3D Shape Grammar Implementation. In J. S. Gero (Ed.), *Design Computing and Cognition'04* (pp 357–376). Dordrecht: Kluwer.
12. Trescak, T., Esteva, M., & Rodriguez, I. (2009). General shape grammar interpreter for intelligent designs generations. In B. Werner (Ed.), *Proceedings of the Computer Graphics, Imaging and Visualization* (pp 235–240). Washington, DC: IEEE.
13. Hoisl, F., & Shea, K. (2011). An interactive visual approach to developing and applying parametric 3D spatial grammars. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 25*(4), 1–64.
14. Jowers, I., & Earl, C. (2011). Implementation of curved shape grammars. *Environment and Planning B: Planning and Design, 38,* 616–635.
15. Yue, K., Krishnamurti, R., & Grobler, F. (2009). Computation-friendly shape grammars: Detailed by a sub-framework over parametric 2D rectangular shapes. In T. Tidafi & T. Dorta (Eds.), *Joining languages, cultures and visions: CAADFutures, University of Montreal, Montreal* (pp. 757–770).
16. Grasl, T., & Economou, A. (2013). From topologies to shapes: parametric shape grammars implemented by graphs. *Environment and Planning B: Planning and Design, 40*(5), 905–922.
17. Grasl, T., & Economou, A. (2013). Unambiguity: difficulties in communicating shape grammar rules to a digital interpreter. In R. Stouffs & S. Sariyildiz (Eds.), *Computation and Performance: Proceedings of the 31st eCAADe Conference—Volume 2, Delft University of Technology, Delft, The Netherlands* (pp. 617–620).
18. Grasl, T., & Economou, A. (2014) Towards controlled grammars: Approaches to automating rule selection for shape grammars. In E. Thompson (Ed.), *Fusion: Proceedings of the 32nd eCAADe Conference, Department of Architecture and Built Environment, Faculty of Engineering and Environment, Newcastle upon Tyne, England, UK* (Vol. 2, pp. 357–363).
19. Economou, A., & Kotsopoulos S. (2014). From shape rules to rule schemata and back. In J. S. Gero & S. Hanna (Eds.), *Design computing and cognition DCC'14* (pp. 419–438). Springer.

20. Stiny, G. (1980). Introduction to shape and shape grammars. *Environment and Planning B: Planning and Design, 7*(3), 343–351.
21. Mitchell, W. J. (2002). Vitruvius Redux. In E. K Antonsson & J Cagan (Eds.), *Formal engineering design synthesis* (pp. 93–125). Cambridge University Press.
22. March, L. (1972). Speculation 8. In L. Martin & L. March (Eds.), *Urban space and structures* (pp. 47–51). Cambridge Urban and Architectural Studies: Cambridge University Press.
23. Schattschneider, D. (1990) *M. C. Escher: Visions of symmetry*, Harry N. Abrams.
24. Schulze, F. (Ed.). (1992). *The Mies Van der Rohe archive*. New York: Garland Publishing.