# Shape Grammars: A Key Generative Design Algorithm

Ning Gu and Peiman Amini Behbahani

## Contents



**Abstract**

Shape grammars are one of the main generative design algorithms. The theories and practices of shape grammars have developed and evolved for over four decades and showed significant impact on design computation and contemporary

N. Gu (✉) · P. Amini Behbahani
School of Art, Architecture and Design, University of South Australia, Adelaide, South Australia, Australia
e-mail: Ning.Gu@unisa.edu.au; Peiman.AminiBehbahani@unisa.edu.au



B. Sriraman (ed.), *Handbook of the Mathematics of the Arts and Sciences*,
https://doi.org/10.1007/978-3-319-70658-0_7-1

architecture. The formal computational approach to generative design as specified in shape grammars, and the novel descriptions and applications of shapes and shape rules for representing and composing a design, has become the foundation and inspiration for many contemporary computational design methods and tools, especially parametric design, which is a current leading computational design method. This chapter gives an overview of the historical developments and applications of shape grammars. The algorithm is introduced by highlighting the background, key components, and procedures for design generation, methods, and issues for authoring shape grammars, shape grammar evolution and extension, purposes of shape grammar application, as well as implementation of shape grammars. The characteristics of shape grammars are presented and discussed by comparing them to other key generative design algorithms, some of which have been applied in conjunction with shape grammars. This chapter shows that shape grammars have significant potentials in design generation, analysis, and optimization, as seen in many of the grammar studies. The future directions should focus on further research, improved pedagogy, as well as validation in design practice, to further advance the field.

**Keywords**
Shape grammars · Shape rule · Generative design algorithm · Design generation · Design analysis

## Introduction

Generative design algorithms as an important part of computational design research are used to formulate or automate parts of the design process. They are intended to assist the human designer to explore the design space more formally and effectively (Singh and Gu 2011). They add to the traditional design philosophy and methodology by enabling designers to systematically manipulate, engage with, and interact with design (McCormack et al. 2004).

Shape grammars have been one of the main computational approaches to generative design for about four decades. Emerging during the epoch of Noam Chomsky's structuralist ideas in the 1960s and 1970s, they have since been developed for various design applications, including stylistic analysis, design generation, and customization, and have also been used in conjunction with other generative algorithms for more optimal results. Shape grammars and their extensions have been used in various creative fields, ranging from architecture – where the earliest examples were demonstrated – to industrial design, visual art such as paintings and textile design, and even expanding to music, animation, and computer languages.

This chapter aims to provide an overview of shape grammars, highlighting various aspects of this key generative design algorithm. The chapter begins with the background to grammar development. It proceeds by introducing the main components and procedures of the algorithm. Further, it introduces the extensions of shape grammars. A number of noteworthy examples of shape grammar applications are

then presented to demonstrate the algorithm. The chapter concludes by summarizing the current developments and future directions of shape grammars.

## Background

After World War II, a movement started that criticized Modernism for an excessive functionalism in modernist architecture (Argan 1996). One of the main arguments of this movement, called *typology*, valued the continuity and integration of pre-modernist architecture and urbanism. With typology, designers started to develop and apply the abstraction of classic shapes, forms, and their combinations, which were labeled as *types* (Colquhoun 1996). Although typology was efficient in describing features of a style, it fell short of providing a mechanism for generating instances under the style (Duarte 2001).

In parallel and within the structuralist school of linguistics research, Chomsky (1965) proposed that humans possess an innate mechanism (called *Language Acquisition Device* or *LAD*) which includes basic grammatical features and enables infants to quickly learn the grammatical syntax of their language. He called these hard-wired features *universal grammars* (UG). Chomsky hypothesized that UG provide a set of rules for generating different syntactic possibilities in all human languages. The generative capability of UG allows a language to produce an infinite number of grammatically correct sentences and phrases out of a limited vocabulary, relative to each language. One of the basic rules of UG is *transformation*: the components of the sentence (or their positions) are transformed to generate new sentences while preserving the correct grammatical structure of the sentence. The major innovation of LAD and UG was that they opposed the behaviorist view that a human learns the language through interaction and inference by trial and error (Chomsky 1965). While there have since been rigorous debates (Everett 1993; Tomasello 2009) on the extent or even existence of UG within linguistics and cognitive sciences, the UG hypothesis has created the prospect of structuring the generative process or algorithm. One of the areas where UG has been adopted is design, and this concept has proved to be extremely useful in furthering design studies on typology.

Under the above context, Stiny and Gips (1972) first applied the core concept of UG in design to develop *shape grammars*. A generative system with transformation rules was demonstrated, which could generate geometrical shapes subject to the application of the rules. The premise of such design grammars was that, similar to language, there are abstract transformation rules which can generate numerous design instances out of a finite set of components. Each set of rules can capture a particular style or a language of similar designs by formally representing and reproducing them. Stiny first demonstrated a shape grammar for generating patterns for acrylics on canvas, as reported by Stiny and Gips (1972). This was followed by more complex shape grammars for generating lattice patterns for Chinese traditional windows (Stiny 1977) and for generating the ground plans of Palladian villas (Stiny and Mitchell 1978) (Fig. 1).

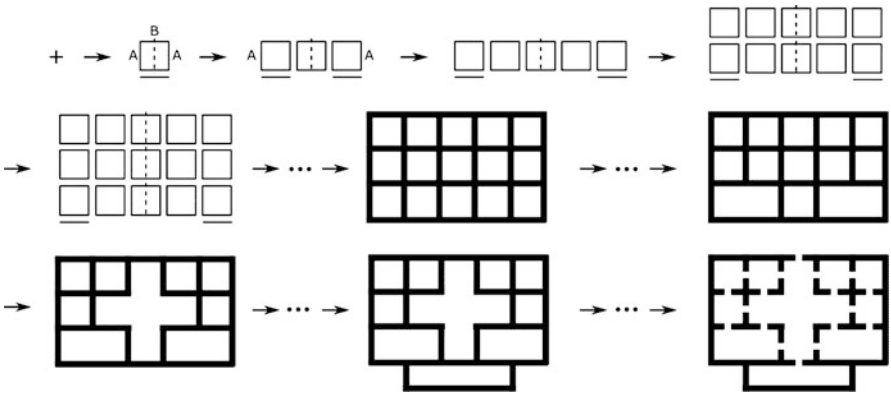
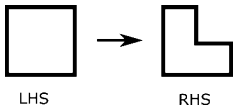



**Fig. 1** Selected processes of the Palladian grammar application for generating the ground floor plan of La Malcontenta

**Fig. 2** A common shape rule containing the LHS and RHS shapes




## Basic Shape Grammars

A shape grammar is generative in nature, which provides a formal and recursive framework to rapidly produce designs within a certain style (Chakrabarti et al. 2011). For this purpose, grammars encode design knowledge within a computational structure as rules. Because the design knowledge can be embedded into the grammar, users of the grammar do not need to have disciplinary expertise in order to generate a design. The application of the grammar is supported by the transformation rules that each replaces one shape with another through a set of geometrical operations. This section briefly reviews the basic components of a shape grammar and its application.

### Main Components of a Shape Grammar

The main components of a shape grammar are shape rules that specify a set of spatial transformations of shapes. A shape is a generic term for any set of geometrical entities containing finite sets of vertices connected by maximal lines (Stiny 1980). A shape rule is a couplet of left-hand side (LHS) and right-hand side (RHS) shapes representing the states before and after the rule application, respectively, where the RHS shape is the result of the spatial transformations of the LHS shape. For illustrating a common shape rule, the shape couplet is ordered from left to right usually separated by an arrow symbol (Fig. 2).

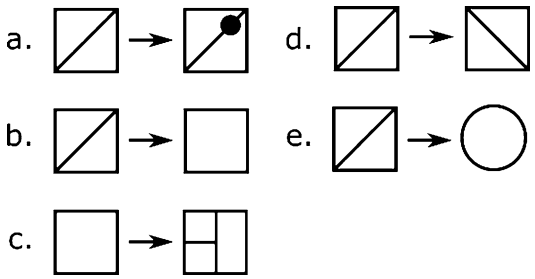



**Fig. 3** Shape rule examples based on different types of transformation: (**a**) addition, (**b**) subtraction, (**c**) division, (**d**) modification, and (**e**) substitution

A shape grammar contains a finite set of shapes, a finite set of symbols, as well as a finite set of rules that can be applied to transform the shapes. The final key component of a shape grammar is the *initial* shape that specifies the starting point of the grammar application (Stiny 1980). While the initial shape can be any shape (including void), a part or the whole of the shape should match the LHS shape of at least one shape rule so that the grammar can be initiated. After the initiation, the grammar recursively applies the rules by transforming LHS shapes that are generated in each earlier step into RHS shapes. Generally, for a grammar application to proceed, a part or the whole of the generated shapes will continue to match the LHS shape(s) of the shape rule(s); otherwise the application will be terminated. Stiny (1980) emphasizes the use of labels in specifying shapes and rules in a shape grammar, and this will be discussed in section "Shape Grammar Application."

The rules can transform shapes in various ways. The literature often refers to five common types of transformation (also called replacement format), which are *addition*, *subtraction*, *division* or *split*, *modification*, and *substitution* (Knight 1999; Trescak et al. 2009; Wonka et al. 2003). Addition rules retain the LHS shape while adding a new part to form the RHS shape. In contrast, subtraction rules remove a part from the LHS shape. In division rules, the LHS shape is divided into two or more parts while its outline remains intact. Modification rules preserve the geometrical elements of the LHS shape but modify its properties, such as proportion, orientation, or direction. Finally, in substitution rules, the RHS shape replaces the entire LHS shape, and the two shapes can differ significantly, for example, a square replaces a circle. See Fig. 3 for examples.

## Shape Grammar Application

A shape grammar is essentially a production system specifying a formal procedure for design generation that starts from the initial shape and proceeds by recursively transforming that shape through the shape rules. For each step of the application, it is critical to detect and match possible LHS shapes and apply the corresponding rules until there are no matching LHS shapes from the generated design. To apply a shape grammar would also require further consideration of the following three issues:

- In the case where more than one LHS shape is detected, should one or some be prioritized? If so, which shape(s) and rule(s)?
- Further, how are these prioritization and selection processes facilitated, and by whom?
- During shape detection, shape grammars allow designers to freely decompose and recompose shapes. This feature is called *emergence*, which signifies shapes that are not predefined but emerged. How should emergence be handled appropriately in a shape grammar application?

The further development of shape grammars flexibly addresses these three key concerns in terms of varying *decidability*. For example, shape grammars can be completely *unrestricted*, allowing multiple rules to be applied simultaneously in a flexible order; or they can be *deterministic*, with strict controls for both rule selection and application (Knight 1999).

To manage decidability, a number of control mechanisms are used during shape grammar application to determine what rules to apply and in what order they will be applied, as well as what degree of emergence is to be enabled. This chapter categorizes the control mechanisms into three general levels: *internal*, *external*, and *parallel*. The internal mechanisms are essentially geometric. The shapes in the rules are defined in a way that enables or does not enable the application of multiple rules, to determine order of rule application and to anticipate possible shape emergence. We called this level internal, as it only pertains to shapes that are the core of shape grammars.

The second category is external, allowing an external agent – human or computer – to decide altogether. The criteria for rule selection, application, and termination can be the fulfillment of certain design criteria and preferences or time, process, and scale constraints.

The third category of control is parallel, which can be described as the incorporation of selected factors influencing the external agent's decision-making within the shapes and rules. For this purpose, shapes are accompanied by nongeometrical descriptions or symbols. This is where the earlier mentioned "labels" become applicable. See Fig. 4 for examples.

In a simpler form, the descriptions can be represented as various markers (symbols, labels, lateral geometrical features, etc.), which are only used for controlling the detection of LHS shapes for rule application. This approach has been most common in shape grammar development (Stiny 1980). In a more elaborate form, the descriptions can contain semantic indicators related to function or shape. An important role of these descriptions is to facilitate the detection of LHS shapes (Stiny 1981). In other words, they control shape emergence in order to effectively detect and match LHS shapes from the resultant design for grammar application. Descriptions are especially important in more complex shapes where visual scanning of the design becomes difficult or unexpected shape emergence cannot be effectively managed. Figure 4 illustrates some example descriptions and demonstrates their uses.

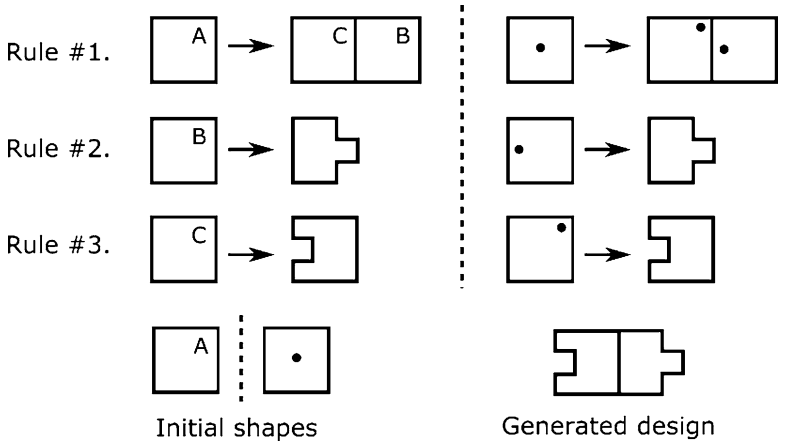



**Fig. 4** Two variants of descriptions using different labels: alphabets (left) and points (right)

In addition to the practical uses discussed above, descriptions can also have a further effect on the development and extension of shape grammars. With descriptions, it is possible to combine nongeometrical elements with shapes. Shapes have become symbolic objects rather than pure geometries. This notion is the main cause for the emergence of several shape grammar extensions, as outlined in section "Extensions of Basic Shape Grammars." With descriptions, the original *shape* grammars have also inspired the development and application of other grammars such as *spatial* grammar (McKay et al. 2012), *design* grammar (Pauwels et al. 2015), and *set* grammar (shapes are treated as members of a predefined set) (Garcia 2017).

## Designing a Shape Grammar

In this subsection, common methods and procedures for developing a shape grammar are discussed. The section presents three aspects of developing a grammar: the selection of design instances (*corpus*) to prepare for the development, the grammar development, and, finally, evaluation.

### Corpus Selection

The *corpus* of a shape grammar is a collection of design instances which are selected out of a particular style or class. Such a corpus is only necessary for *analytical* grammars (grammars that aim to represent or reproduce existing styles or classes), but not for *original* grammars (grammars that aim to develop original or new designs) (Pauwels et al. 2015). The corpus forms the basis for grammar development. Ideally, the grammar corpus contains all the available instances. However, this rarely happens due to the large and diverse collection of some styles.

The importance of the corpus selection is that different selections may result in different grammars (Stiny 2006).

The corpus selection is an inductive process (Johnson 2003) involving the extraction of common rules to generate the selected design instances. The goal of this process is to capture commonalities of the instances and to remove superficial characteristics in order to access deeper and more essential properties which underpin the generalized concepts of the style or class.

Seawright and Gerring (2008) define the following cases for selecting design instances:

- *Typical* cases form an exemplary selection of the sample population, which largely represent the average characteristics of the style or class (Flyvbjerg 2006). However, these cases are not necessarily the richest in information.
- *Diverse* cases retain maximum variation in different dimensions (Seawright and Gerring 2008), which enables information to be extracted from widely different situations (Flyvbjerg 2006).
- *Extreme* and *deviant* cases hold unusual or surprising values compared to other cases. They are usually used for enhancing and evaluating understandings about the main characteristics (Flyvbjerg 2006).
- *Influential* cases are those whose analysis may affect the overall understanding about the style or class, disregarding the rest of the cases.
- Most *similar* and *different* cases are those with the most similar or distinct features and values when compared to each other.

### Shape Grammar Development

Shape grammar development is a logical argumentation process based on shapes and their operations. It requires multiple rounds of axiom-based modeling/analysis, trials and errors, and so on. Duarte (2001) proposes a four-step process for developing a shape grammar based on an existing corpus. It includes *rule inference*, *prototyping*, *reverse grammar*, and *testing*:

- In rule inference, the corpus is analyzed from different perspectives to reveal the spatial or form-related structures of the individual instances.
- Prototyping accounts for finding common features of the individual structures identified from the previous step.
- Reverse grammar is about checking every instance in the corpus to test whether it is possible to trace back to the initial shape from an existing design.
- The testing step is to evaluate the success of the grammar applications in producing different designs in both the corpus and beyond. This step is further explained below.

### Shape Grammar Evaluation

The evaluation of a shape grammar examines both the feasibility and efficiency of the grammar. A well-designed grammar should be able to generate designs and alternatives that are consistent with the selected corpus. The grammar should also

be free from errors and provide controls over unexpected design emergence. For analytical grammars, three types of tests are considered for evaluation (Stiny and Mitchell 1978). The tests aim to evaluate the abilities of the grammar to:

- Regenerate the selected corpus.
- Generate other design instances of the style that are not in the corpus.
- Generate new designs based on the style (only if the grammar's goal is to also generate such new designs).

The evaluation of a grammar has subtle differences from the evaluation of its semantic outcomes (i.e., meeting specific design criteria). Shape grammars do not always have an internal mechanism to closely match their outcome against the design criteria (Singh and Gu 2011). Such evaluation and selection processes are usually performed after the generation, either by human designers or through other computational processes.

## Extensions of Basic Shape Grammars

Basic shape grammars are one of the earliest applications of Chomsky's grammars in the context of visual art and design. Since the 1980s, and on the basis of the shape grammar theorization, different developments have extended the basic shape grammars to address different needs of design generation or analysis. This section focuses on discussing three different types of extensions, including parallel grammars, parametric grammars, and graph grammars.

### Parallel Grammars

Additional descriptions of the shapes other than the geometries can also be specified in a shape grammar, enabling a parallel design transformation within the grammar. To facilitate this, an independent rule set can be added that corresponds and is synchronized with the respective shape grammar rules (Knight 2003). These types of shape grammars are called *parallel grammars* (prior to the 1990s, the term "parallel grammar" referred to an unrelated notion of a grammar being able to produce two or more shapes in each step during the application (see Stouffs 2016, "An algebraic approach to implementing a shape grammar interpreter," pp. 329–338)) where each rule set addresses a different aspect of the design, and together they are applied simultaneously (Knight 2003). The description rules work in the same way as the shape rules do, by replacing an LHS component with the RHS one. Descriptions in a basic shape grammar are realized using labels and symbols that attach directly to the shapes. In a parallel grammar, they can be an entirely separate rule set. The main advantage of parallel grammars is to allow a grammar to address other design requirements beyond shape or form. They can also effectively address some technical issues during shape grammar application, such

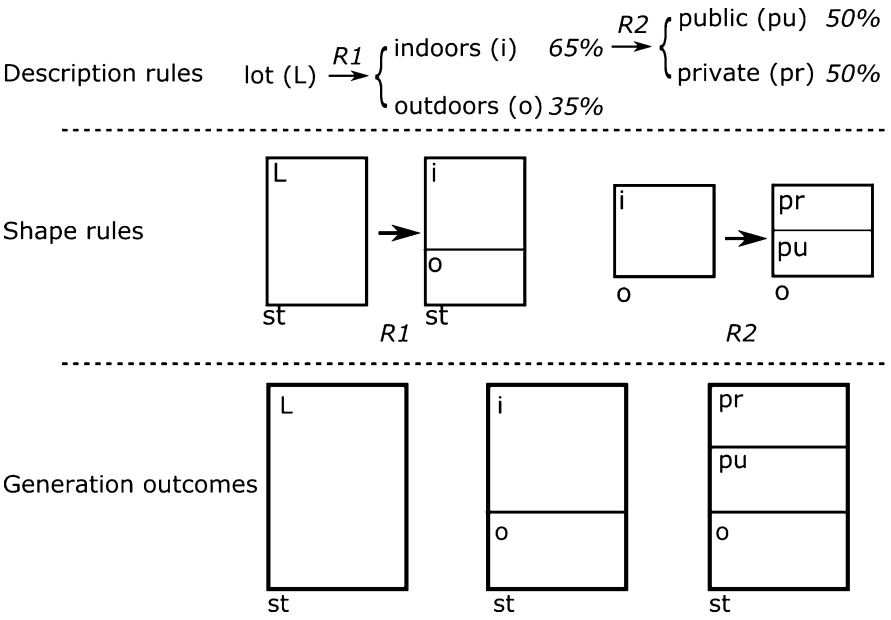



**Fig. 5** Selected rule sets from a parallel grammar, based on Duarte's (2001) Malagueira grammar

as shape matching and design emergence, due to the more structured and complex system of defining design descriptions.

Figure 5 demonstrates an example of parallel grammar developed for Siza's Malagueira houses (Duarte 2001). A set of semantic rules describe functional zones, dividing a general-purpose lot into indoor and outdoor spaces, then further splitting the indoor space into public and private areas. They are applied in parallel to the shape rules to generate the outcomes (Fig. 5).

## Parametric Grammars

Descriptions can add some levels of richness and complexity to the rigid geometries of shapes. To maximize such flexibility in a grammar, a selection of shapes and their transformations are generalized as one generic shape with parameterized features that can be adjusted according to a number of criteria and constraints (Stiny 1980). Such a grammar is called a *parametric grammar*.

Parameters can affect the grammar in different stages and forms. Parameters can be applied during the detection of LHS shapes, selection of the rules, or determining the features of the RHS outcome. For example, the LHS shape of rule can be a quadrilateral with a certain range of height-width ratio (i.e., the parameter). This ratio may in turn affect a corresponding ratio in the RHS shape. Parametric constraints are not necessarily numerical, as in the above example. They can be

geometrical and semantic or use any other form(s) a description may have. Because of their flexibility, parametric grammars possess more potential and have been applied more commonly in recent development (Abdesalam 2012). In addition, they may reduce the number of rules and therefore simplify the grammar (Stavrev 2011).

## Graph Grammars

Historically, graph grammars derive from string grammars, not Chomskyite generative grammars (Chakrabarti et al. 2011). More recently, they have been contextualized and applied within shape grammar research. Graph grammars contribute to shape grammar research by using two approaches, namely, *geometric* and *semantic* approaches. The geometric approach comes from the analogy between graph-topological and shape-geometrical vertices and edges, focusing on defining shapes or forms with graph elements. The shapes here are represented as a collection of vertices that are connected by edges. In the semantic approach, the graph's nodes are associated with the semantics of shapes. For example, in a building, the nodes of the graph represent rooms and their associated purposes, while the edges represent the opening or adjacency between them. Some examples of this approach are Eloy's transformation grammar (Eloy 2012) and Lee et al. (2016) graph grammar of Prairie-style houses.

The main incentive for incorporating graph theories into shape grammar is the simplicity and abstractness of graphs. On the one hand, graph grammars mitigate the complexities of visual shape detection. On the other hand, they represent shapes or forms in a topological manner, which may better reflect the interrelations within the design, both structurally and functionally.

## Further Discussion on the Extensions

The three extensions introduced above can be combined to create more powerful grammars. For example, it is possible to have a parallel parametric grammar that includes a rule set defined by graph elements (see Eloy 2012, the transformation grammar definition). These three extensions are only three typical formal grammars, contributing to aspects of the manageability, flexibility, and intelligibility of shape grammars. Other important extensions of shape grammars include:

- Color grammar (Knight 1989), which is a parallel grammar that uses colors as descriptive elements. A major difference between color and symbolic descriptions is that colors can be combined together to form a more visually rich design. They can also represent design attributes other than the actual colors, for example, materials, building elements, and so on.
- Sortal grammar (Stouffs and Krishnamurti 2001) abstracts shapes (or descriptions) as *sort*s to optimize the manageability and flexibility of the grammar.

- Discursive grammar (Duarte 2001) is a parallel parametric grammar with additional heuristics for managing rule selection. The heuristics are high-level non-procedural semantic descriptions (e.g., design goals or criteria), which can act as parameters for applying and assessing rules.

There are different terminologies regarding grammar developments and their application domains. In general, the most abstract levels of grammars are called *formal* or *structure* grammars, in which the vocabulary (LHS and RHS components) can be of any kind of structures. The next level pertains to the grammars with specific types of basic vocabularies. For example, *shape* grammars, of basic "shapes," have vocabularies consisting of pure geometrical elements, *graph* grammars involve graph representations, and the vocabulary of a *motion* grammar includes patterns of movement. Beyond visual art and design, developments such as motion or music grammars have extended the shape grammar logics into the respective new disciplines. Hence, shape grammars may be called design grammars in this regard. Figure 6 shows the chronology of the shape grammars and their related developments.

## Applications of Shape Grammars

Shape grammars are commonly known for being a type of generative design method. However, generation is not the only application of shape grammars. McKay et al. (2012) consider four major areas where shape grammars can be applied, based on a survey by Gips (1999): parsing, development, generation, and inference. Considering other studies (Huang et al. 2009; Özkar and Stiny 2009; Singh and Gu 2011), this section adapts the four areas as follows.

### Description and Analysis

Originally, shape grammars were developed for describing paintings and sculptures (Stiny and Gips 1972). Their aim was to understand and abstract the form-related structure of a certain style or from a particular era (Huang et al. 2009). This application was then extended to, and commonly applied in, architecture. Shape grammars have been used to study a wide range of architectural styles or related designs, such as Palladian villas (Stiny and Mitchell 1978), Turkish traditional houses (Çağdaş 1996), Safavid caravanserais (Andaroodi et al. 2006), and Glenn Murcutt's domestic architecture (Lee et al. 2016).

### Reproduction and Generation

As a generative design method, shape grammars can be used to automate parts, or all, of the design process. Specifically, shape grammars are useful when dealing with

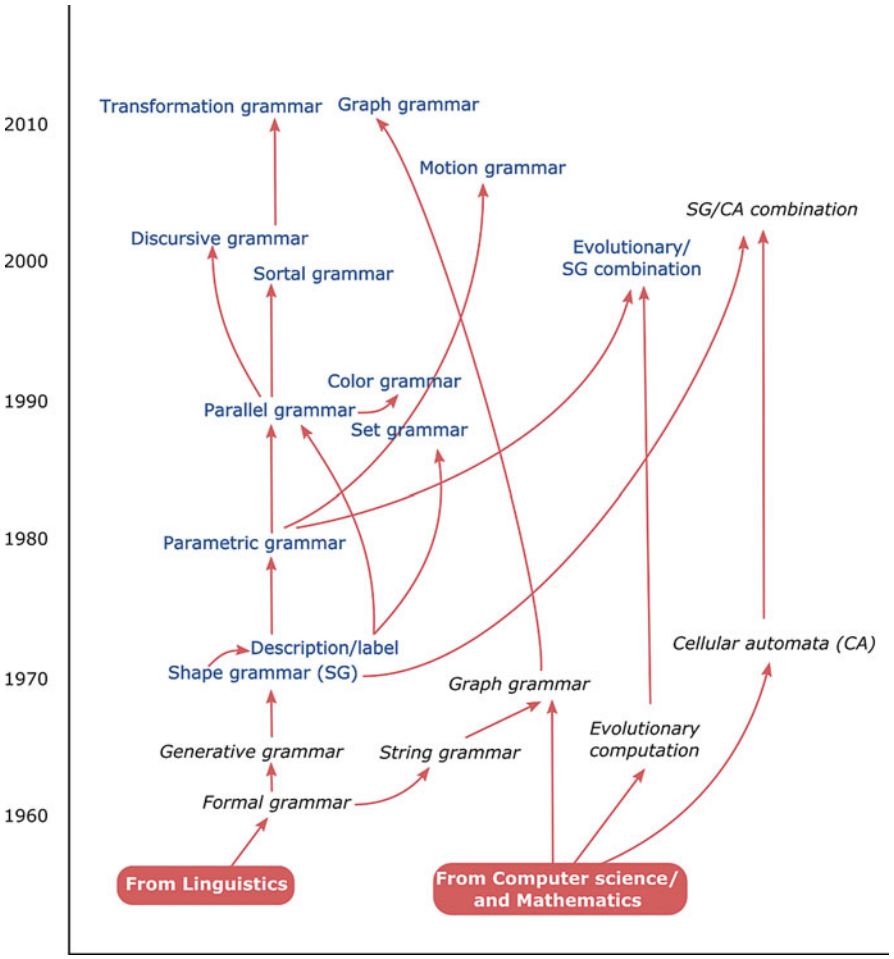



**Fig. 6** A diagram depicting the chronology of shape grammars and their related developments

modularity, due to its articulated nature (Özkar and Stiny 2009). It is significantly efficient for reducing costs in mass customization of design (Duarte 2001) because its embedded design knowledge reduces the need for more human expertise during the generation process. Through defined shapes and rules, shape grammars also retain the consistent compositions that grant a brand identity or design style (Özkar and Stiny 2009).

We have discussed the distinction between *analytical* and *original* grammars (Pauwels et al. 2015). Analytical grammars are developed based on a corpus of existing designs, while original grammars are developed from scratch to generate novel designs. So far, a significant number of shape grammar examples have been of the analytical type. Many of them have not been intended to create

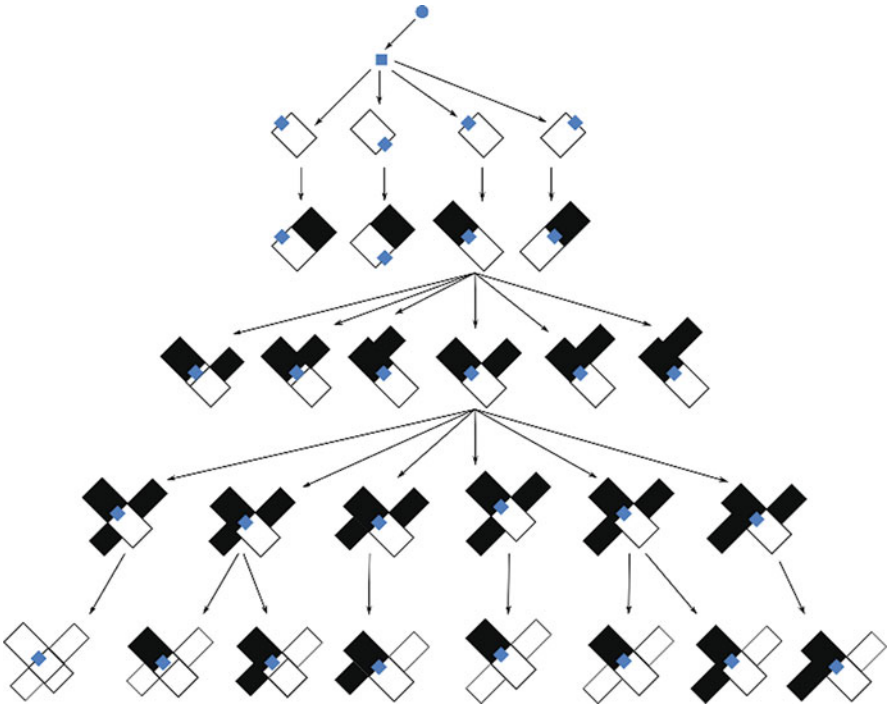


**Fig. 7** A schema of shape grammar application for Prairie-style houses, figure adapted by Amini-Behbahani (2016) based on Koning and Eizenberg's original (1981)

a complete or practical design, but to demonstrate the generative possibilities. Examples of analytical grammars are shape grammars for Wright's Prairie-style houses (Koning and Eizenberg 1981) (Fig. 7), coffee-making machine (Agarwal and Cagan 1998), Siza's Malagueira houses (Duarte 2001), and Harley-Davidson motorcycles (Pugliese and Cagan 2002).

Examples of original grammars are relatively rare. One reason may be that shape grammar research has focused more on design style, methodology, and philosophy rather than specific issues of practice (Krish 2011). Another possible reason is that, even for the purposes of generating original designs, there are always existing precedents that are useful as the foundation of the grammar and new design instances. There are, however, some examples of original grammars, including the planar truss grammar (Sass et al. 2005) (Fig. 8).

## Optimization and Customization

One of the generative purposes of shape grammars is to support designers to explore and optimize generated design solutions. This is especially beneficial in education

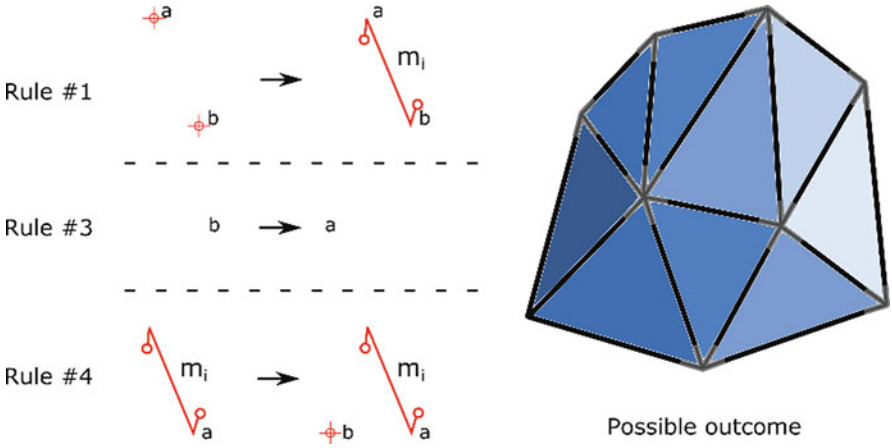



**Fig. 8** Selected rules and a possible design outcome from a shape grammar for generating space trusses (Sass et al. 2005), adapted by the authors

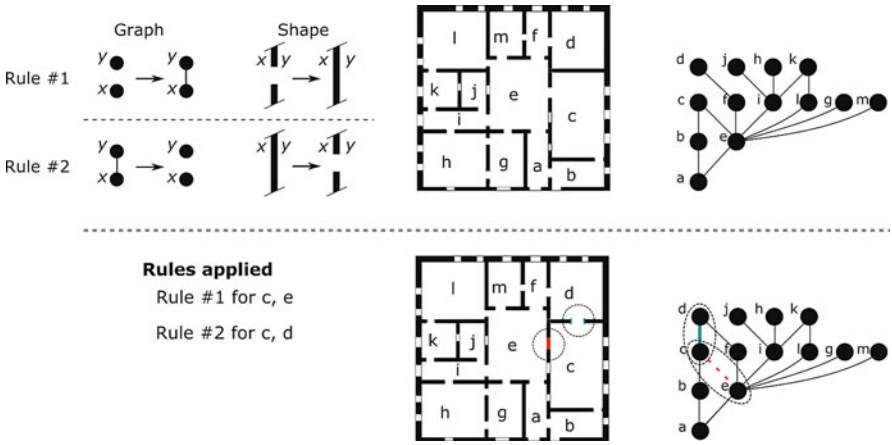



**Fig. 9** A demonstration of transformation grammar (on shapes and graphs in parallel): two selected rules and current configurations in plan and graph (above); optimized configurations by applying the rules (below), with changes highlighted in circles on both plan and graph (Eloy 2012), adapted by the authors

for understanding design alternatives and possibilities (Özkar and Stiny 2009). A typical example of such is the transformation grammar (Eloy 2012) to modify and upgrade apartments in Lisbon (Fig. 9).

### Combination with Other Methods

Referred to as inference, Gips (1999) suggests that a shape grammar can be used for extracting or developing another shape grammar. Moreover, shape grammars have been used in conjunction with other generative methods, such as cellular automata (Speller et al. 2007). In another attempt, Kielarova et al. (2015) demonstrate a combination of shape grammars with evolutionary processes for fine-tuning the grammar application progress of earring design, directed and controlled by evolutionary fitness functions.

## Implementation of Shape Grammars

The earlier grammars were implemented manually. However, most current grammars are applied through digital means. This is likely due to the increased complexity of more recent shape grammars and the digital nature of contemporary design practice. The implementation of shape grammars has always been feasible through *interpreters* (Gips 1999), which are software packages that enable the design or application of grammars. Interpreters are often plug-ins within CAD/BIM software or stand-alone programs with possible interactions with CAD/BIM software.

Earlier interpreters were stand-alone tools with a basic interface. Since the 1990s, selected interpreters have been developed as plug-ins or scripts in commercial CAD packages (e.g., AutoCAD and Rhino) (Gips 1999). These interpreters address various aspects of grammar implementation, including capabilities of 2D or 3D, bitmap or vector-based shape detection, parametrization, graph grammar incorporation, etc. The development of new interpreters, or new features of interpreters, is likely to continue into the future for more efficient and effective implementations.

There are limitations regarding shape grammar implementation that require further research. These limitations are both technical and, more importantly, behavioral. In one of the earliest reviews of interpreters, Gips (1999) argues that there is a gap between the developers and users of grammars. The developers are usually advanced in symbolic thinking and computer programming, while the users excel in visual thinking and designing. As a result, the main problem of most interpreters is that they do not adequately address the behavioral needs of both groups. Gips further correlates this issue to the fact that most interpreters are prototypes for demonstration and proof of concept and the available industrial and commercial tools are scarce. The core of this problem still exists today, despite dozens of grammar interpreters having since emerged for either general purposes or specific contexts.

The lack of current CAD integration is also limiting shape grammar implementation (Chakrabarti et al. 2011). With the rapid development of contemporary CAD programs, the development of new interpreters does not seem to pick up the same pace and correspond to the latest CAD tools or methods. The existing grammar

interpreters also do not closely resemble the interfaces and project workflows of the common CAD tools with which the designers are more familiar (Tching et al. 2016). Subsequently, because of the lack of commercialization, they can also lack the interactivity and appeal often expected by designers.

Another limitation regarding shape grammar implementation (and the grammar itself) is that it does not adequately facilitate the reframing of design alternatives (Pauwels et al. 2015). In practice, designers may drastically move from one idea to another, but such exploratory flexibility is not feasible in most interpreters and shape grammars.

## Shape Grammar and Other Generative Design Algorithms

*Generative design algorithm* (GDA) or simply *generative algorithm* (the acronym GA is also used for generative algorithms. This chapter uses GDA to avoid confusing it with a more common use of the acronym GA that refers to genetic algorithms.) refers to various computational algorithms that assist in exploring the design space, usually by outlining a formal framework with principles and processes for generating numerous alternatives. These algorithms vary significantly in their components, procedures, criteria, and efficiency. Shape grammars are one of these algorithms, and this section discusses several other main algorithms in relation to shape grammars.

GDAs can be approached and classified in various ways. In this chapter, we categorize them into three types, based on their identifiable approaches to design space exploration. These three types of GDAs are *replacement*, *evolution*, and *agent interaction*. In the replacement type, a part of design is replaced with another to generate new alternatives, usually based on certain rules. Shape grammars belong to this type of GDA. Another replacement type of GDA is *L-system* (or Lindenmayer system) which shares similar origins with shape grammars (namely, formal grammars from the 1960s) (Singh and Gu 2011). Unlike shape grammars, the components in L-systems are mainly symbols (or symbolic geometries). Another difference is that L-systems predominantly feature multiple rule selection in each step. These two differences make L-systems suitable for design situations with small components repeating in patterns over a relatively large extent. Examples of such situations include plant modeling and street design (Parish and Müller 2001).

Another GDA that can be loosely put under this type is *parametric combination* or *associative geometry* (Hernandez 2006). In parametric combination, the replacement affects features such as the proportions and dimensions of the design rather than the entire entity of shapes or components (as in basic shape grammars). Under this method, a design is defined with its components parametrically associated with each other. Once a parameter changes, the dependent components will also change automatically, resulting into new design alternatives.

The algorithms discussed above share a limitation – they do not have a native mechanism to evaluate the effectiveness of the outputs (Duarte 2001; Singh

and Gu 2011). This often necessitates a more elaborate approach in developing the rules by incorporating foreknowledge for quality control.

The second type of GDA, evolution, is theorized based on its biological inspirations. The essential feature here is that the design alternatives are matched against a set of fitness functions in each step; the next step continues from the "fittest" alternatives for maximal optimization. *Genetic algorithms* are the most typical example of this type of GDA, supported by an evolutionary mechanism.

For the third type of GDA, agent interaction, a society of agents navigate through the design space and generate design alternatives based on their interactions with the context and other related elements. Common examples of this type are *cellular automata*, *swarm intelligence* (Singh and Gu 2011), and *space colonization* (Runions et al. 2007). This type of GDA is usually a bottom-up process, where the useful outcomes emerge from the visual or spatial and functional congruence of different components. An obvious advantage of such a method is that the generated outcomes can be closely aligned to the requirement and context, minimizing the need for post-design evaluation. On the other hand, they can be visually or spatially restricted because their design space is geometrically limited (e.g., rigid grids for cellular automata).

The above categorizations are not mutually exclusive. In other words, it is possible to have a GDA with features from two or even three different types. For example, in the development of shape grammars, evolutionary processes have been incorporated to better direct and evaluate design generation. These examples include shape grammars for Coca-Cola bottles (Ang et al. 2006), earring design (Kielarova et al. 2015), Prairie house validation (Granadeiro et al. 2013), and steel section evaluation (Franco et al. 2014).

## Discussion and Conclusion

This chapter has presented shape grammars, including the key components, the procedures of design generation, the main applications and extensions, as well as their implementations. Starting in the 1970s, shape grammars have since evolved and extended into various forms and been applied in different disciplines. They have also demonstrated design capabilities in addressing nongeometrical and semantic issues. To further enhance the generative power and to better assist designers, research has also focused on providing flexibility and manageability during shape grammar application. The significant potentials of shape grammars in design generation, analysis, and optimization are evident, as seen in many of the grammar studies. A key to such success is perhaps due to the visual basis (shapes and their spatial transformations as specified in shape rules), which makes shape grammars more suitable and easily understood for visual thinkers, such as architects and product designers. In addition, their generative capabilities (especially empowered by computation) make them very suitable and useful for mass design tasks. Despite these promising potentials, shape grammars as a generative algorithm have limitations, as discussed throughout the chapter. Further, the theories and

applications of shape grammars have not yet been broadly utilized in practice. This will require further research as well as validation in design practice.

The last decade has especially seen a surge in studies that combine shape grammars with either other grammars or other computational methods, to address different design needs. The generative nature of shape grammars, and the formal approaches to design representation of generation as specified in shape grammar research, has become the foundation and inspiration for contemporary computational design methods and tools, most notably *parametric design*, which has been widely adopted in leading architectural practices. It is perhaps in this regard that shape grammar research has demonstrated its most significant impact on design computation and contemporary architecture. We expect these hybrid developments and applications will continue and further increase into the future.

Parallel to research and practice, pedagogy is another important future issue of shape grammars. Compared to research and practice, shape grammar pedagogy has often been overlooked and has rarely been recorded and debated in literature. To further develop shape grammars, and to effectively and broadly apply their generative power in the design industry, will rely on the future generation of designers and design students. Besides generation, the roles of shape grammars, especially in design analysis and optimization, have been largely unexplored in design education. They have the potential to provide very novel approaches for teaching design students about design history and science topics. To equally develop these three key pillars – research, practice, and pedagogy – is important for the future of shape grammars, and each will critically inform the other.

## References

Abdesalam M M (2012). The use of smart geometry in Islamic patterns. CAAD | INNOVATION | PRACTICE 6th international conference proceedings of the Arab Society for computer aided architectural design, pp 49–68

Agarwal M, Cagan J (1998) A blend of different tastes : the language of coffee makers. Environ Plan B: Urban Analytics City Sci 25(2):205–226

Amini-Behbahani P (2016) Spatial properties of frank lloyd wright's prairie style: A topological analysis. Dissertation. University of Newcastle, Australia

Ang M C, Chau H H, McKay A, De Pennnington A (2006) Combining evolutionary algorithms and shape grammars to generate branded product design. In: Gero J S (ed) Design computing and cognition'06, pp 521–540

Andaroodi E, Andres F, Einifar A, Lebirge P, Kando N (2006) Ontology-based shape-grammar schema for classification of caravanserais: a specific corpus of Iranian Safavid and Ghajar open, on-route samples. J Cult Herit 7:312–328

Argan GC (1996) Typology of architecture. In: Nesbitt K (ed) Theorizing a new agenda for architecture: an anthology of architectural theory 1965–1995. Princeton Architectural Press, New York, pp 240–247

Çağdaş G (1996) A shape grammar: the language of traditional Turkish houses. Environ Plan B, Plan Des 23:443–464

Chakrabarti A, Shea K, Stone R, Cagan J, Campbell M, Hernandez NV, Wood KL (2011) Computer-based design synthesis research: an overview. J Comput Inf Sci Eng 11:021003–021012

Chomsky N (1965) Aspects of the theory of syntax. MIT Press, Cambridge, MA
Colquhoun A (1996) Typology and design method. In: Nesbitt K (ed) Theorizing a new agenda for architecture: an anthology of architectural theory 1965–1995. Princeton Architectural Press, New York, pp 248–257
Duarte J (2001) Customizing mass housing: a discursive grammar for Siza's Malagueira houses. Dissertation, MIT
Everett DL (1993) Sapi, Reichenbach and the syntax of tense in Piraha. Pragmat Cogn 1(1):89–124
Eloy S (2012) A transformation grammar-based methodology for housing rehabilitation: meeting contemporary functional and ICT requirements. Dissertation, TU Lisbon
Flyvbjerg B (2006) Five misunderstandings about case-study research. Qual Inq 12:219–245
Franco JMS, Duarte J, Batista EM, Landesmann A (2014) Shape grammar of steel cold-formed sections based on manufacturing rules. Thin-Walled Struct 79:218–232
Garcia S (2017) Classifications of Shape Grammars. In: Gero JS (ed) Design computing and cognition '16. Springer, Cham, pp 229–248
Gips J (1999) Computer implementation of shape grammars. Workshop on shape computation, MIT 1999
Granadeiro V, Pina L, Duarte J, Correia JR, Leal VMS (2013) A general indirect representation for optimization of generative design systems by genetic algorithms: application to a shape grammar-based design system. Autom Constr 35:374–382
Hernandez CRB (2006) Thinking parametric design: introducing parametric Gaudi. Des Stud 27:309–324
Huang J, Pytel A, Zhang C, Mann S, Fourquet E, Hahn M, Cowan W (2009) An evaluation of shape/split grammars for architecture. Research Report CS-2009-23, David R. Cheriton School of Computer Science, University of Waterloo, Ontario
Johnson R (2003) Case study methodology. In: Proceedings of the international conference methodologies in housing research, pp 22–24
Kielarova SW, Pradujphongphet P, Bohez ELJ (2015) New interactive-generative design system: hybrid of shape grammar and evolutionary design – an application of jewelry design. In: Tan Y, Shi Y, Buarque F, Gelbukh A, Das S, Engelbrecht A (eds) Advances in swarm and computational intelligence. ICSI 2015. Lecture Notes in Computer Science, Springer, Cham, 9140:302–315
Knight T (1989) Color grammars: designing with lines and colors. Environ Plan B, Plan Des 16:417–449
Knight T (1999) Shape grammars: six types. Environ Plan B, Plan Des 26:15–32
Knight T (2003) Computing with emergence. Environ Plan B, Plan Des 30:125–156
Koning H, Eizenberg J (1981) The language of the prairie: Frank Lloyd Wright's prairie houses. Environ Plan B, Plan Des 8:295–323
Krish S (2011) A practical generative design method. Comput Aided Des 43:88–100
Lee JH, Ostwald M, Gu N (2016) A justified plan graph (JPG) grammar approach to identifying spatial design patterns in an architectural style. Environ Plan B: Urban Analytics City Sci 45(1): 67–89
McCormack J, Dorin A, Innocent T (2004) Generative design: a paradigm for design research. In: Richmond J, Durling D, de Bono A (eds) Proceedings of futureground. Design Research Society, Melbourne, pp 156–164
McKay A, Chase S, Shea K, Chau HH (2012) Spatial grammar implementation: from theory to useable software. Artif Intell Eng Des, Anal Manuf 26:143–159
Özkar M, Stiny G (2009) Shape grammars. In: Proceedings, SIGGRAPH 2009 (course)
Parish Y, Müller P (2001) Procedural modeling of cities. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques, pp 301–308
Pauwels P, Strobbe T, Eloy S, De Meyer R (2015) Shape grammars for architectural design: the need for reframing. In: Celani G, Sperling DM, Franco JMS (eds) Communications in computer and information science. Springer, Berlin, pp 507–526
Pugliese MJ, Cagan J (2002) Capturing a rebel: modeling the Harley-Davidson brand through a motorcycle shape grammar. Res Eng Des 13:139–156

Runions A, Lane B, Prusinkiewicz P (2007) Modeling trees with a space colonization algorithm. In: NPH'07 proceedings of the Third Eurographics conference on natural phenomena, pp 63–70
Sass L, Shea K, Powell M (2005) Design production: constructing freeform designs with rapid prototyping. In: Digital design: the quest for new paradigms – 23nd eCAADe conference proceedings, Lisbon (Portugal) 21–24 Sept 2005, pp 261–268
Seawright J, Gerring J (2008) Case selection techniques in case study research, a menu of qualitative and quantitative options. Polit Res Q 61:294–308
Singh V, Gu N (2011) Towards an integrated generative design framework. Des Stud 33:185–207
Speller TH, Whitney D, Crawley E (2007) Using shape grammar to derive cellular automata rule patterns. Complex Syst 17:343–351
Stavrev V (2011) A shape grammar for space architecture – part II. 3D graph grammar – an introduction. In: 41st international conference on environmental system, American Institute of Aeronautics and Astronautics
Stouffs R (2016) An algebraic approach to implementing a shape grammar interpreter. Conference: 34th eCAADe conference, At Oulu, Finland, vol 2, pp 329–338
Stouffs R, Krishnamurti R (2001) Sortal grammars as a framework for exploring grammar formalisms. Mathematics and Design '01, Deakin University July 2001
Stiny G (1977) Ice-ray: a note on the generation of Chinese lattice designs. Environ Plan B 4:89–98
Stiny G (1980) Introduction to shape and shape grammars. Environ Plan B 7:343–351
Stiny G (1981) A note on the description of designs. Environ Plan B, Plan Des 8
Stiny G (2006) Shape: talking about seeing and doing. The MIT Press, Cambridge, MA
Stiny G, Gips J (1972) Shape grammars and the generative specification of painting and sculpture. Inf Process 71:1460–1465
Stiny G, Mitchell WJ (1978) The Palladian grammar. Environ Plan B 5:5–18
Tching J, Reis J, Paio A (2016) A cognitive walkthrough towards an interface model for shape grammar implementations. Comput Sci Inf Technol 4:92–119
Tomasello M (2009) Universal grammar is dead. Behav Brain Sci 32:470–471
Trescak T, Rodriguez I, Esteva M (2009) General shape grammar interpreter for intelligent designs generations. Computer Graphics, Imaging and Visualization, 2009. CGIV'09, pp 235–240
Wonka P, Wimmer M, Sillion F, Ribarsky W (2003) Instant architecture. ACM Trans Graph 22:669–677