Taylor & Francis
Taylor & Francis Group

Spatial Cognition & Computation
An Interdisciplinary Journal

ISSN: 1387-5868 (Print) 1542-7633 (Online) Journal homepage: http://www.tandfonline.com/loi/hscc20

Spatial specification and reasoning using grammars: from theory to application

Yufeng Liu, Kang Zhang, Jun Kong, Yang Zou & Xiaoqin Zeng

To cite this article: Yufeng Liu, Kang Zhang, Jun Kong, Yang Zou & Xiaoqin Zeng (2018) Spatial specification and reasoning using grammars: from theory to application, Spatial Cognition & Computation, 18:4, 315-340, DOI: 10.1080/13875868.2018.1490290

To link to this article: https://doi.org/10.1080/13875868.2018.1490290

SPATIAL COGNITION & COMPUTATION
2018, VOL. 18, NO. 4, 315–340
https://doi.org/10.1080/13875868.2018.1490290

Taylor & Francis
Taylor & Francis Group

Spatial specification and reasoning using grammars: from theory to application

Yufeng Liu[a], Kang Zhang[b], Jun Kong[c], Yang Zou[a], and Xiaoqin Zeng[a]

[a]Institute of Intelligence Science and Technology, Hohai University, Nanjing, China; [b]Department of Computer Science, The University of Texas at Dallas, Richardson, TX USA; [c]Department of Computer Science, North Dakota State University, Fargo, ND, USA

ABSTRACT

This article reviews grammatical formalisms that are capable of supporting spatial specification and reasoning, or spatial-enabled grammars, and their wide range of applications. The review takes two typical grammars, i.e., shape grammar and spatial graph grammar, as concrete representatives to consider connectivity and spatial relations in the parsing and generating processes. This article proposes four aspects as a set of criteria to compare the commonality and differences among spatial-enabled grammars, i.e., parsing and generation, granularity of spatial specification, form of spatial specification, and 2D and 3D modeling. Finally, further developments related to spatial-enabled grammars are discussed.

KEYWORDS

Grammar formalism; spatial specification; shape grammar; spatial graph grammar

1. Introduction

In computer science, formal grammars are used to describe various programming languages as the underlying foundation for supporting translators or compiler generators. Spatial properties are an essential feature in a 2D or 3D space and play an important role in the representation and reasoning of spatial relationships. Therefore, researchers have investigated the definition of spatial relations in formal grammars. Some approaches explore spatial relationships from the layout perspective. For example, layout graph grammar (Brandenburg, 1994), first proposed for syntax-directed layouts, is able to generate a desirable layout according to the rewriting rules drawn on a plane. Marriott (1994) proposed a constraint multiset grammar (CMG), which encodes spatial layout and relationships. As a context free grammar formalism, CMG introduces extra spatial constraints to determine whether a production can be applied. In addition, Marriott and Meyer (1997) developed a hierarchical classification for visual languages using CMG and discussed the expressive power and the complexity of parsing for each class. Based on CMG, Wittenburg

CONTACT Xiaoqin Zeng xzeng@hhu.edu.cn

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/iceh.

and Weitzman (1996) proposed a relational grammar (RG), which embeds relational constraints, e.g., spatial layout information, into productions.

From the perspective of shapes defined as a finite arrangement of spatial elements (Krishnamurti & Stouffs, 1993), e.g., line segments, points, and rectangles, shape grammar (Stiny, 1975; Stiny & Gips, 1971) is widely applied in the fields of architecture and mechanical engineering, such as layout design and parts design. The shape grammar is fundamentally a graphic generative formalism, which applies rules to replace one shape with a different subset of shapes iteratively.

Researchers have attempted to abstract spatial information to several concrete relations. In an interactive visual query interface on spatial/temporal data, Li and Chang defined orientation, distance, and five other spatial relations (i.e., Disjoin, Meet, Overlap, CoveredBy, and Inside) between two objects (Li & Chang, 2004). Based on these spatial relations, iconic visual languages where each object is displayed as an icon with a precise position have found different applications, such as visual queries to databases (Chang, 1990). Later, a linear positional grammar is extended with spatial relations to formally specify visual languages in a 2D space, and accordingly a 2D interactive parser was developed (Orefice et al., 1992). There are also many spatial-enabled grammars proposed for specific application domains. For example, Mayall and Hall (2005) defined a spatial landscape grammar to describe a landscape's character and generated landscape scenes, which formally draws parallels between the structure of linguistics and the character of real-world landscape. Hoisl and Shea (2011) presented an interactive 3D spatial grammar for CAD, which primarily focuses on the generation and manipulation of solids. Qi (2015) defined a spatial grammar to explain observed data (an 2D image). Based on Bayesian framework, the approach describes the underlying arrangement and structure of a scene, according to a parse tree that maximizes the posterior probability.

The aforementioned traditional spatial grammars represent abstract spatial relations implicitly as constraints over objects, which do not make a strict distinction between objects and relationships. They, however, could be used to support visual languages with formal mechanisms for visual expressions, possibly with precise constraints over graphic elements.

Graph grammars with their well-established theoretical background can be used as natural and powerful syntax-definition formalism (Rozenberg, 1997) for visual languages, which model structures and concepts in a 2D fashion. Various graph grammar formalisms (Rekers & Schürr, 1997; Costagliola & Polese, 2000; Zhang et al., 2001a; Kong, Zhang & Zeng, 2006) have been proposed for different purposes. Applying a production to an application graph, usually called a *host graph*, is referred to as graph transformation, which can be classified as an L-application (in a forward direction) or R-application (in a backward direction). A *redex* is a sub-graph in the host

graph that is isomorphic to the right graph in an R-application or to the left graph in an L-application. A production's L-application to a host graph is to find in the host graph a redex of the left graph of the production and replace the redex with the right graph. A graphical language defines a set of all possible graphs that have only terminal objects and can be only derived through L-applications (i.e., a generating process) from an initial graph. On the other hand, an R-application is the reverse replacement (i.e., from the right graph to the left graph) used to parse a host graph. The parsing process validates the syntactical correctness and interprets the certain semantics of a given host graph. With a visual yet formal specification, graph grammars find many applications, such as automatic generation of visual programming environments (Costagliola, Lucia, Orefice & Tortora, 1997), defining the semantics of UML diagrams (Kong, Zhang, Dong & Xu, 2009), verifying the program behavior (Zhao, Kong & Zhang, 2010), inter-model consistency checking (Leblebici, Anjorin & Schürr, 2017), and software architecture verification and reconfiguration (Li, Huang, Chen & Yu, 2013).

To enhance the representational power of graph grammars and extend their application scope, SGG (Kong et al., 2006) introduces a spatial mechanism into the graph grammar formalism of RGG (Reserved Graph Grammar) (Zhang et al., 2001a). SGG provides an explicit way to specify spatial syntax and semantics, including six topological relationships, four direction relationships, ordered distance distinctions, and seven alignment relationships.

With strict mathematical definition and operation, formal grammars provide a solid theoretical foundation for defining and specifying various languages, including programming languages, visual languages, graph modeling languages, unified modeling languages, business process execution languages, etc. On the other hand, spatial relationships and semantics are crucial in many applications, such as graphical user interfaces and geospatial systems. It is therefore necessary to investigate both the structural and spatial specification mechanisms in a formal grammatical setting.

This article reviews grammars that are suitable for supporting the spatial specification and their applications, in which shape grammar and SGG as two typical representatives are discussed in detail, and so are their applications and related research. From the perspective of grammar formalism and spatial semantics, four criteria are proposed to compare spatial-enabled grammars, with the discussion on their merits and drawbacks. The comparison could provide clear distinctions that are needed when selecting a formalism for a specific application. Moreover, several related or extended aspects are discussed for further developments of spatial-enabled grammars.

The reminder of the article is organized as follows. Section 2 reviews two representative grammar formalisms with the capacity of spatial specification and their applications. Section 3 presents a set of criteria for comparison of

spatial-enabled grammars. Section 4 proposes future developments, followed by conclusion in Section 5.

2. Grammatical formalisms with spatial specification and its applications

Spatial specification extends the expressive power of grammatical formalisms, providing a high-level language for explicitly expressing spatial relations, such as topology, direction, and distance between objects. This characteristic makes the grammatical formalisms with spatial specification exhibit a remarkable diversity in their application areas. This section reviews two important grammars from which applications targeting a variety of subject areas are derived.

2.1. *Shape grammar*

Shape grammar (Stiny, 1975; Stiny & Gips, 1971) was first proposed for the general specification of architectures, floor plans, and non-representational arts, by modeling abstract spatial relationships of geometries. Shape grammars allow calculation in algebras of shapes, not just as a method of analysis (Stiny, 2006). To date, it has been explored to various domains, such as pattern recognition, mechanical design, and industrial design. Most of the geometries in shape grammars are in the form of labeled rectangles.

As defined by Stiny and Gips (1971), a shape grammar is a 4-tuple: SG = (V_T, V_M, R, I), where V_T is a set of terminal shapes. V_T^* is formed by the finite arrangement of an element or elements of V_T in which any element of V_T may be used a multiple number of times with any scale or orientation. V_M is a set of markers, which contain no duplicate shapes with V_T, to control how rules are applied to the left-hand side (LHS) shapes. R represents a finite set rules (u, v), such that u is the LHS shape consisting of possible combination of initial shapes in V_T with markers in V_M and v is the right-hand side (RHS) shape after transformation. I is the initial shape consisting of elements of V_T^* and V_M.

Defining rules for a shape grammar is a difficult task for designers, due to the requirement of unambiguity (Grasl and Economou, 2013). Strobbe, De Meyer and Van Campenhout (2015) proposed a semi-automatic approach to generate new rules during derivation, enabling a flexible exploration of the design space. It should be noted that the shape grammar differs from traditional graph grammars in the definition of graphical elements, where each basic operational object of shape grammars is a shape consisting of maximal lines (Stiny, 1980), while traditional graph grammars take nodes and edges as their basic elements, as shown in Fig. 1. Furthermore, the distinctions of the graphical elements between the shape grammar and traditional graph grammars lead to differences in graph transformation processes, e.g., subgraph

search, etc. In some cases, the information described by a shape grammar can be abstracted into a graph grammar to enable easy computation. For example, Grasl and Economou (2010) proposed a graph grammar to simulate the shape grammar productions for Palladian villas.

A shape grammar can operate rectangles (or other shapes) to generate various kinds of shapes in a 2D Cartesian coordinate system through an iterative induction process based on predefined production rules. As a generative shaping rule, it was first used in the application of building image parse, and was successfully applied to scene reconstruction, and graph, tree, and artwork design during the previous decades.

2.1.1. *Parsing building images*

Recognizing the semantic meanings of objects within an image has been a core problem in computer vision for more than 30 years (Mahfoud & Willis, 2013). It has numerous applications, such as interactive multimedia games, online map service, and reconstruction of a building model from images. Several attempts have been made to use the shape grammar as a modeling language to solve the building image parsing problems (Muller, Zeng, Wonka & Gool, 2007). Teboul et al. (2010a, 2010b, Teboul, Kokkinos, Simon, Koutsourakis & Paragios, 2013) have published a sequence of articles that use the shape grammar to parse rectified facade images of urban style buildings. However, these approaches are limited to 2D grammars and could only analyze rectified orthographic images. Mahfoud and Willis (2013) used a 3D shape grammar to segment and estimate rectilinear shapes in no-rectified images, thereby increasing the application scope.

In summary, the majority of these methods use scene façade images as input, and apply shape grammars to divide the input image into semantically meaningful rectangles, such as windows, walls, floors, etc. Figure 2 shows an example of parsing a building façade image, in which four kinds of terminal

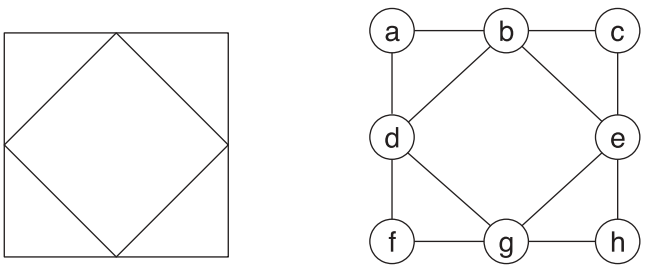
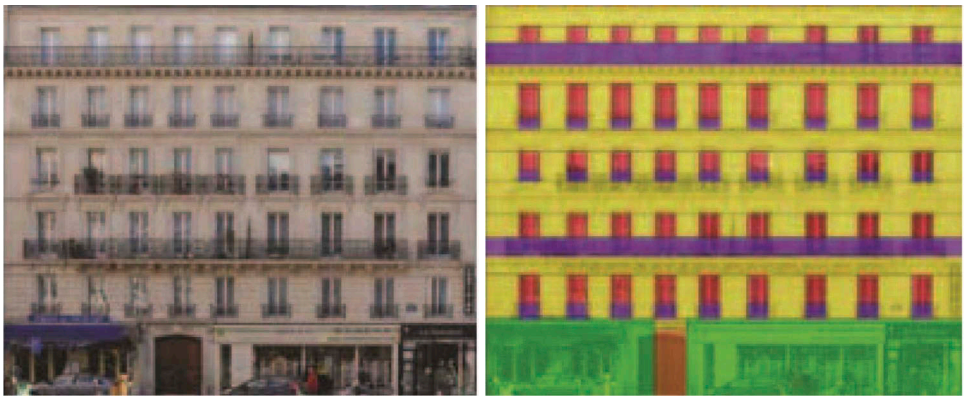


Figure 1. Graphical elements in shape parsing and traditional graph grammars. (a) Shapes consisting of maximal lines in shape grammar. (b) Nodes and edges in traditional graph grammars.



(a) (b)

Figure 2. Parsing building facades (Teboul et al., 2013). (a) Original image. (b) Segmentation with different colors indicating different recognized objects of the input image.

elements, i.e., wall, window, floor, and shop, are recognized and colored differently.

2.1.2. *Scene reconstruction*

As a generative tool, shape grammars are often used for building reconstruction and building model generation. Wonka, Wimmer, Sillion and Ribarsky (2003) proposed the split grammar, a variation of shape grammar, which splits the facade hierarchically in a top-down manner. Furthermore, to enable automatic and effective rule derivation, CGA shape grammar (Muller, Wonka, Haegler, Ulmer & Gool, 2006), extending the split grammar by introducing the notation of mass model and context-sensitive grammar, was proposed to generate massive urban models.

Muller et al. (2007) used CGA to design an algorithm to automatically derive high quality 3D models from a single façade image of arbitrary resolutions. Their method takes an image of a real building facade as input and is able to reconstruct a detailed 3D facade model, combining imaging and shape grammar generation techniques. Chen, Kang, Xu, Dorsey and Shum (2008) also proposed a method for creating building facades from images, but using hand sketches as input. Hou, Qi and Qin (2012) proposed a modeling framework to build 3D models of Chinese architectures by automatically extracting production rules from elevation drawings. Grzesiak-Kopec and Ogorzalek (2013) combined the graph grammar with computational intelligence methods to solve the 3D layout problem, which can be applied to interior floor plan of buildings. Koutsourakis, Simon, Teboul, Tziritas and Paragios (2009) introduced an approach to single view reconstruction using shape grammars. Different from the above woks that generate single parts of buildings, Tutenel, Smelik,

Lopes, Kraker and Bidarra (2010) proposed a framework to integrate multiple procedural techniques to generate consistent buildings from exterior to interior.

City environment generation is another application in this category. Liu, Xu, Pan and Pan (2004) used an extended shape grammar to generate city models, and the generated models contain streets, housing blocks, roads, and houses with components such as gates, windows, walls, and roofs. Weber, Muller, Wonka and Gross (2009) designed a simulation system that could simulate a 3D urban model using a shape grammar. Kelly and McCabe (2007) also developed a grammar-based system, called Citygen, aiming to rapidly create an urban geometry.

2.1.3. *Graph, tree, and artwork design*

The main purpose of graph drawing is to develop an effective layout algorithm that can generate a readable graph, tree or art representation. Huang, Dudek, Sharman and Szabo (2005) combined manually defined productions with visualization tools built into Mathematica to generate two types of highly symmetric arts. Li, Bao, Zhang, Kobayashi and Wonka (2011) introduced the concept of field-guided shape grammars, and used it to create 3D geometry or texture on surface. Li, Zhang and Li (2017) used a shape grammar to specify various logo design requirements for semi-automatic generation of logo designs that meet the specified requirements. Table 1 summarizes the aforementioned representative applications using shape grammars.

2.2. *Spatial graph grammar*

Different from other graph grammar formalisms, the spatial graph grammar (SGG) introduces spatial notions to the abstract syntax. In SGG, nodes and edges together with spatial relations construct the pre-condition of a production application. Using spatial information to directly model relationships in the abstract syntax is coherent with the concrete representation, and avoids converting spatial information to explicit edges. Allowing designers to specify design knowledge in both structure and spatial layout, SGG is particularly suited for specifying the derivation of the content organization underlying a concrete layout.

A SGG is a 4-tuple: gg=(A, P, T, N), where A is an initial graph. P is a production set. T and N are the terminal and non-terminal node sets respectively. A node has a two-level structure: the node itself and the small rectangles called vertices embedded in the node, as shown in Figure 3. All vertices in a node should be uniquely labeled. In order to identify any graph elements that should be preserved during the transformation process, each isomorphic vertex in a production graph is marked by prefixing its label with

Table 1. Representative Applications for Shape Grammars.

Application Domain	Works	Description
Building Image Parse	(Teboul et al., 2013)	Define a subclass of the shape grammar, called Binary Split Grammar to describe various façade layouts and produce a parsing algorithm that formulates the façade parsing problem as a hierarchical decision process
	(Mahfoud & Willis, 2013)	Use 3D shape grammars to segment and estimate rectilinear shapes in non-rectified images
	(Muller et al., 2007)	Automatically infer shape grammar rules from façade images to extract high-level façade structure
	(Teboul, Simon, Koutsourakis & Paragios, 2010)	Implement a southeast China vernacular urban modeling system. This system converts the basic modeling components of geometry units into the semantic components. An improved production based grammar system is used to control the process of urban generation
	(Teboul, Kokkinos, et al., 2010b)	Address shape grammar parsing issues for façade segmentation using reinforcement learning
Scene Reconstruction	(Muller et al., 2006)	Propose a shape grammar (namely CGA shape) for the procedural modeling of CG architectures; produce building shells of high visual quality and geometric details
	(Tutenel et al., 2010)	Propose an approach that integrates multiple existing procedural techniques to generate building models
	(Grzesiak-Kopec & Ogorzalek, 2013)	Apply the shape grammar to conduct intelligent 3D layout design
	(Koutsourakis et al., 2009)	Use the shape grammar with Markov Random Field framework to reconstruct a 3D model from a single view
	(Hou et al., 2012)	Extract rules and construct semantic models for Chinese architectures from elevation drawings, to generate rule-driven Chinese architectures
Graph, Tree, and Artwork Design	(Huang et al., 2005)	Combine manually defined productions with visualization tools built into Mathematica to generate two types of highly symmetric arts
	(Li et al., 2011)	Generate a geometric design using the shape grammar that reveals aspects of the underlying geometric contents of an image
	(Li et al., 2017)	Generate logo designs based on the requirement specified in shape grammars

an integer unique in the node. Figure 4 shows a SGG production that has two pairs of vertexes marked "1" and "2" respectively. The purpose of marking a vertex is to establish a connection between the surrounding of a redex and its replacing graph to preserve the context. Based on the marking mechanism, an embedding rule: If a vertex in the right graph of the production is unmarked and has an isomorphic vertex "v" in the redex of the host graph, then all edges connected to "v" should be completely inside the redex. This rule elegantly avoids ambiguity and dangling edges.

SGG productions can be depicted with a user-friendly graphical interface, such as the VEGGIE interface (Ates & Zhang, 2007), or with a grid representation of spatial relationships (Zhang, Kong, Qiu & Song, 2005). Each SGG production also includes a spatial specification to model the spatial layout among different objects. SGG also supports syntax-directed

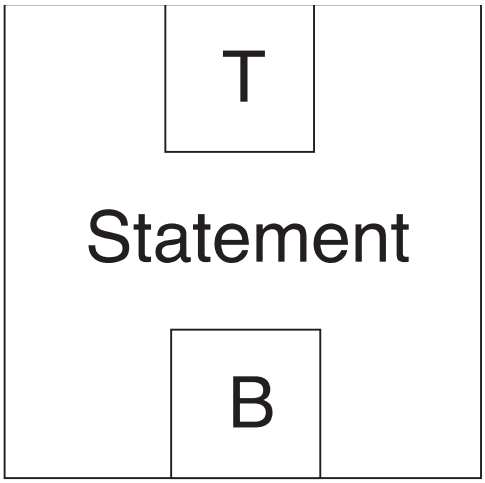


Figure 3. Two levels of a node in SGG.

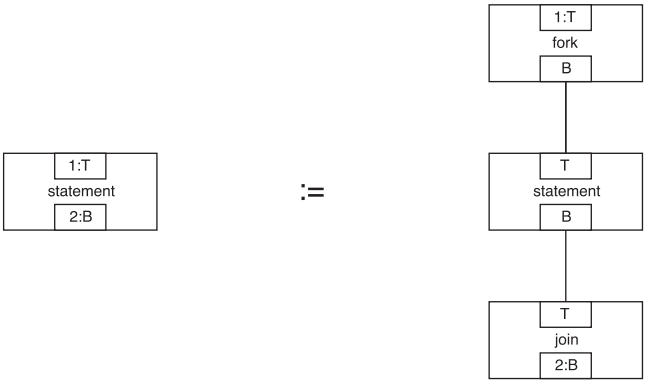


Figure 4. A production of SGG.

computation through action code. An action code is associated with a production, and executed when the production is applied. Writing an action code is like writing a standard event handler in Java.

Every graphical interface has an instinctive internal semantic structure. For example, related information may be enclosed in a table, and topic names could be placed on top of the detailed contents. Such a semantic structure can be represented by constructing a graph, in which each node encodes an element on the interface and an edge connecting two nodes represent the semantic relationship between the nodes. Extracting the semantic structure of the graph is useful for numerous applications. SGG has been widely used in this aspect. For example, Kong et al., Ates, Zhang and Gu (2008) proposed a grammar-induction based approach to partition a web page into several small pages according to the semantic relationships among them, by which the web

(a) (b)

Figure 5. An example of interface object recognition and pattern interpretation (Kong et al., 2012). (a) Page segmentation. All the recognized atomic objects are highlighted in a red rectangle. (b) Page interpretation with a hierarchical structure, in which leaf nodes represent atomic objects while intermediate nodes indicate composite objects.

page can be flexibly browsed smartphone screens. SGG has also been used to analyze the semantics of different types of websites (Kong et al., 2012; Zhang & Kong, 2010). All of these applications interpret a web page bottom up, where a well-established image processing technology is first used to recognize atomic interface objects in an interface image (see Figure 5a). The output of the image processing is a spatial graph, which records significant spatial relations among the objects. Based on the spatial graph, the SGG parser can obtain the hierarchical relations among the interface objects and thus provides semantic interpretation (see Figure 5b). In order to reduce the effort of manual grammar design, Roudaki, Kong and Zhang (2016) took the advantage of spatial information to implement an efficient grammar induction algorithm, which could automatically induce part of a graph grammar to discover web design patterns.

In addition to the interface pattern analysis, graph grammars with spatial specification have also been used to help designers to better perform interface design tasks. A representative work is to design a compatible interface display mechanism suitable for various types of display devices (Zhang et al., 2005). Table 2 summarizes several representative applications for SGG.

The above grammars and their applications are neither exclusive nor exhaustive. Other grammars exist for these applications, just as these grammars may likewise be applied to other areas. For example, attribute graph grammar formulated by Knuth to assign semantics to context-free languages (Tsai & Fu, 1980) has been used in parsing image objects (Han & Zhu, 2005) and tree drawing (Kirishima, Goto, Yaku & Tsuchida, 2010). The graph

Table 2. Representative Applications for Spatial Graph Grammar.

Application Domain	Examples	Description
Interface Generation and Pattern Verification	(Roudaki et al., 2016)	Use spatial information to convert 2D grammar induction problem to 1D string induction and then apply the induced SGG grammar to discover web design patterns
	(Kong et al., 2012)	Recognize atomic objects from the screenshot of an interface through a graph parsing technique and then use SGG to analyze the spatial relations among those objects to perform semantic grouping and recover interface semantics
	(Kong et al., 2008)	Generate a graph representing web page structures, and parse the graph to obtain its hierarchical structure, which is then used to partition the web page into multiple small pages for mobile display.
	(Zhang & Kong, 2010)	Generate a graph from an interface image using an image processing technique, and then use SGG to discover the relations among interface objects and provide semantic interpretation
	(Zhang et al., 2005)	Add spatial notations to RGG and extract the spatial relationships among the objects of online multimedia contents. Adapt the layout of multimedia objects to mobile screens according to the extracted spatial relationships

Table 3. Representative Applications for Other Grammars with Spatial Specifications.

Application Domain	Examples	Description
Building Image Parse	(Han & Zhu, 2005)	A simple attribute graph grammar as a generative image representation with an effective top-down/bottom-up inference algorithm for parsing images
Graph, Tree, and Artwork Design	(Kirishima et al., 2010)	An attribute graph grammar that enables trees of minimum width to be drawn
Expression Recognition	(Costagliola et al., 2006)	A parsing grammar-based strategy for the recognition of hand-drawn diagrams
Creative Processes	(Tching et al., 2013)	Assisting architects to add possible solutions and make decision in architectural design by using shape grammars

grammars with spatial specifications can also be used in other areas, such as expression recognition (Costagliola, Deufemia & Risi, 2006) and creative processes (Tching, Reis & Paio, 2013), as in Table 3.

Shape grammar and SGG could complement each other. First, a shape grammar could be used as a generative engine for applications such as artistic design. On the other hand, SGG is often used as a parsing tool. Second, the theoretical framework of shape grammars is based on quantitative analysis, whereas SGG abstracts spatial relations to several concrete relations based on qualitative analysis. The former is precise while the latter is intuitive. Third, the basic elements of a shape grammar are shapes, e.g., line segments, while SGG treats objects and the relationships as abstract nodes and edges according to graph theory, hence the two grammar formalisms differ in their roles but complementary.

3. Criteria

To evaluate the spatial-enabled grammars' capabilities of specifying and manipulating spatial information, this section presents four criteria for comparing the grammars. The criteria cover the most aspects of the spatial-enabled grammars and can be used to analyze and compare the grammars systematically. The criteria include: a) parsing and generation; b) granularity of spatial specification; c) form of spatial specification; d) 2D and 3D modeling. Here parsing and generation builds the main workflow of a grammar system; the granularity and form of spatial specification are two essential aspects in spatial mechanism; 2D and 3D modeling represents the dimensional extend to which a spatial-enabled grammar may be applied. Together, the set of criteria help to provide a systematic classification of the grammars reviewed.

3.1. *Parsing and generation*

Based on the definition of a grammar, the workflow of spatial-enabled grammars can be divided into *parsing* and *generation* processes. The parsing process is useful to interpret and validate the internal structure of an input graph while the generation process produces a visual sentence that observes the properties defined by a grammar. Figure 6b shows an example of the parsing process, where two SGG productions in Figure 6a are used to parse a simple flowchart. In particular, parsing helps analyzing the spatial layout patterns among objects, such as web interface interpretation (Kong et al., 2012). Since an analyzed object is in general presented as an image (such as the screen shot of a web page), image processing techniques are commonly integrated with a graph parser. The image processing technique can recognize essential objects and generate a node-edge graph that abstract the original image. The generated graph keeps important spatial and logical information in the original graph while eliminating unnecessary details to facilitate the parsing process. The generated graph is analyzed by iteratively applying graph transformation in a bottom-top manner.

On the other hand, the generation process can be viewed as an inverse process of graph parsing. Prior to generation, design constraints and layout requirements are formally defined as a graph grammar. Based on the defined graph grammar, various kinds of graphs satisfying the productions can be generated. The generation process is particularly useful for shape grammars, which were designed primarily for automatically generating designs. Li et al. (2017) proposes a design framework that is able to semi-automatically generate logo designs based on a set of shape grammar specifications.

In summary, the parsing process is able to reveal and validate the internal structure of an existing graph while the generation process serves to

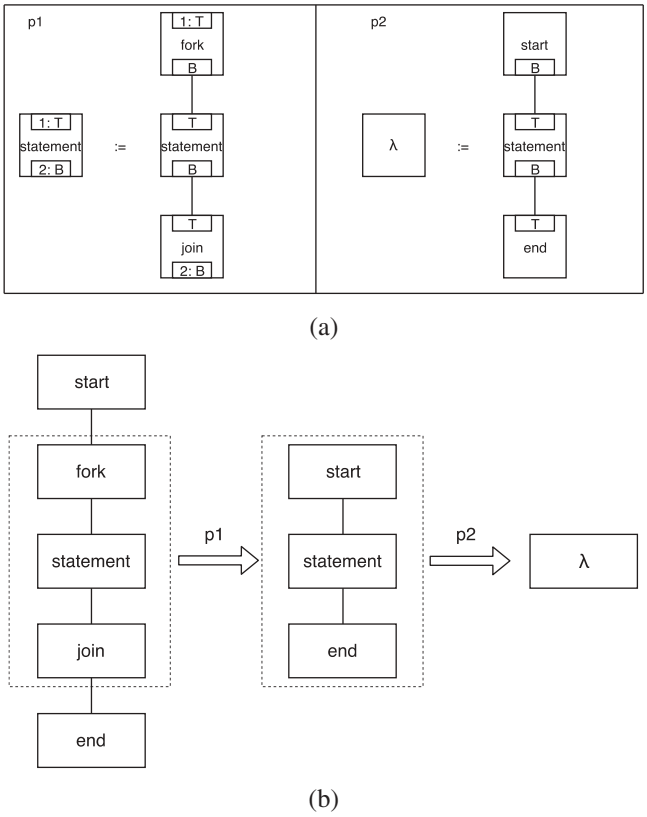


Figure 6. An example of the parsing process. (a) Two SGG productions. (b) A parsing process using the productions.

automate the design process by generating an unlimited set of layouts or designs that observe the given requirements.

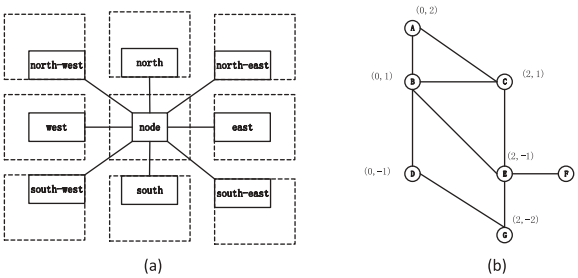


Figure 7. Granularity of spatial specification. (a) a qualitative direction specification in SGG. (b) a quantitative specification on coordinates.

3.2. *Granularity of spatial specification*

In a spatial-enabled grammar, spatial relations can be defined by discrete values (such as north or south) or continuous values (such as coordinates). For example, the SGG (Kong et al., 2006) abstracts directions into eight values, i.e., north, north-east, east, south-east, south, south-west, west, and east-west (see Figure 7a.). On the other hand, the shape grammar directly uses coordinates to indicate the spatial relation between objects. A qualitative spatial specification (i.e., discrete values) captures the essential spatial relations, abstracted from concrete coordinates, and allows variation among graphs defined by a spatial-enabled grammar. Moreover, an abstract spatial specification facilitates grammar designers to define spatial relations at a high level, making qualitative analysis relatively easy and friendly to use. On the other hand, a quantitative specification on coordinates integrities all the spatial relations among objects as a whole in the Cartesian coordinate system with ultimate precision (see Figure 7.b).

3.3. *Form of spatial specification*

A graph grammar or shape grammar formalism provides a visual yet formal tool to model spatial structures. In fact, spatial properties are 2D or 3D in nature. Consequently, a textual description of spatial relations may limit the visibility of a grammar specification. It is therefore desirable to visualize the specification of spatial relations. For example, in the SGG based on qualitative analysis, a 3*3 grid was proposed to visually define spatial relations (Qiu, Song, Kong & Zhang, 2003). While in quantitative analysis, such as shape grammars, spatial relations among objects are expressed by coordinate values of graphical elements. A visual specification of spatial relations allows designers to capture a grammar quickly and may improve the design efficiency.

3.4. *2D and 3D modeling*

Most existing applications are conducted in a 2D space. These applications take images as input. Using image processing techniques, the geometrical objects in an image can be extracted without any shape deviation. However, to adapt to more application domains, numerous applications attempt to extend the definition of shapes in grammars from a 2D to a 3D Euclidean space. Those applications that support 3D are mainly in the architecture areas (Chen et al., 2008; Hou et al., 2012; Stiny, 1980), in which generating 3D models is a necessity and preprocessing images is difficult especially when the scenes to be modeled contain numerous complex 3D objects. Figure 8 shows a 3D model of Chinese architectures that is generated according to

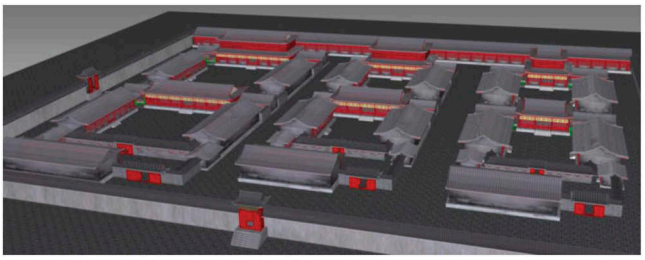


Figure 8. A 3D model of Chinese architectures (Hou et al., 2012).

Table 4. Comparison between Shape Grammar and Spatial Graph Grammar.

Grammar Formalism	Generating and Parsing	Granularity of Spatial Specification	Form of Spatial Specification	3D and 2D Modeling
Shape Grammar	Generation	Continuous and Fine	Quantitative	2D and 3D
Spatial Graph Grammar	Parsing and Generation	Discrete and Coarse	Qualitative	2D

shape grammar rules, where the rules are used to encode a semantic shape tree for the generation (Hou et al., 2012).

According to the aforementioned four criteria, a comparison between shape grammar (SG) and SGG is shown in Table 4. It supports the argument that SG and SGG complement each other. Based on the graph theory, a SGG provides a formal yet intuitive framework for spatial semantics. A shape grammar, on the other hand, is more generic in handling the definition of the elusive character of spatial components and relations, i.e., 3D objects (Krishnamurti & Earl, 1992) and parametric curves (Jowers & Earl, 2010). Therefore, shape grammars and SGGs excel in different application areas. As summarized in Section 2, the applications of spatial-enabled grammars are classified into six domains, i.e., *Building Image Parse, Scene Reconstruction, Interface Generation and Pattern Verification, Expression Recognition, Creative Processes*, and *Graph, Tree, and Artwork Design*. Shape grammars are mainly used in architectural design and graph drawing in the generation process, while SGGs facilitate parsing-directed analysis. This is due to the differences in their analytical approaches. Quantitative approach is applicable to automatic generation of accurate layouts that meet the spatial specifications defined in grammars. Qualitative approach is suitable for analyzing concrete images, which in general is used with a parsing process to analyze the semantics of an image.

4. Further development

4.1. *Grammar induction*

Though graph grammars provide a solid foundation to define structures in a 2D space, it is time consuming to design a graph grammar. Grammar induction is an automatic process of generating a grammar from given example graphs, saving the effort of manually designing the grammar. Ates, Kukluk, Holder, Cook and Zhang (2006); Ates & Zhang (2007)) developed an induction system, called VEGGIE, which extends the SUBDUE Grammar Learner (Jonyer, 2003), with the support for the graphical notions of the SGG. The SUBDUE induction process is based primarily on the idea of graph compression. Graphs can be compressed in a similar manner to file compression. Like popular file compression techniques, the compression process looks for common substructures within the graph, and compresses the graph by recording the substructure and replacing all instances by a marker. As substructures are captured, this process models a simple context-free grammar induction process. Graph-based data compression relies on a substructure matching technique to find instances within the graph. However, unlike string-based data compression, for exact sub-graph matching, graph compression requires an exponential runtime. The induction process can generate a base or partial grammar that is then edited manually to complete the specific context for an application domain. A context-sensitive extension of VEGGIE has also been investigated and prototyped that is able to induce context-sensitive graph grammars based on overlapping substructures within a graph (Ates et al., 2006).

Little research has been conducted to use spatial information to speed up the induction process. The parser of the SGG uses the spatial information to sequence objects and provides an efficient parsing algorithm. Recently, Roudaki et al. (2016) proposed a new grammar induction algorithm to sequence objects and to convert the 2D graph induction to 1D string induction. This work only considers one type of spatial relations. We expect that the spatial information can play a more important role in grammar induction.

4.2. *Shape grammar interpreter*

Since Stiny introduced the shape grammar formalism, scientists have come up with various shape grammars to represent designs in different areas. In the early days, the capability of automatically generating shape grammars is limited, due to the complexity of the shape grammar and the ability of the computer hardware. A program enabling the generation process of the shape grammar is called a *shape grammar interpreter* (Gips, 1999). According to Yue and Krishnamurti (2012), shape grammars could be classified by shape

dimensions. Here, we divide the shape grammar interpreters to be reviewed into two categories: 2D and 3D.

2D: The first shape grammar interpreter (Gips, 1975) accomplishes basic functionalities of shape grammars, allowing users to input sample rules and generating shapes according to the given rules. This simple interpreter ignores one of the essential concepts of shape grammars, i.e., sub-shape detection or shape emergence. Then, Krishnamurti (1982) designed a sub-shape detection algorithm and implemented to another shape grammar interpreter. The shape grammar interpreter developed by Krishnamurti is the first that follows the shape grammar concept originated by Stiny and Gips (1971). Recently developed interpreters, e.g., Shape Grammar Interpreter (SGI) (Trescak, Rodríguez & Esteva, 2009) provides interactive platforms for users. Wang and Zhang (2018) have enhanced SGI with more powerful functionality, such as image and shape import and color specification. Grasl and Economou (2013b) develop a plug-in named Grape (GRaphs and shAPEs), where a graph grammar is used to implement the shape grammar mechanism in a graph-based underlying engine. Moreover, Grape allows the decomposition of existing designs in various non-anticipated ways (Economou & Grasl, 2017). Economou and Grasl (2017) demonstrate a set of designs produced in Grape. Designers generate initial shapes and transformation rules of their own choices. The interpreters focus on translating abstract design rules and plans to final design products according to these given rules.

3D: To support 3-dimensional designs, many interpreters focus on specific design task. For example, Agarwal and Cangan (1998) proposed Coffee Maker Grammar for coffee maker designs, which specify a 3D product from three views – top, side, and front – using a 2D shape grammar. Ertelt and Shea (2009) represented a shape grammar-based approach for automatically creating fabrication plans for CNC matching from a given part geometry, which uses a CAD kernel to support the shapes of different dimensionality. However, these interpreters embed grammars into the source code, making it hard for designers to use. The designers have to change the source code in order to update a certain function, which is an impractical challenge for designers, since they typically have little coding experience. To provide an interactive platform for designers, Hoisl and Shea (2011) proposed an Interactive 3D Spatial Grammar System based on a set of parameterized primitives, making it possible to visually define 3D rules.

Table 5 lists most of the shape grammar interpreters, with the information of implementation languages, whether they are able to detect sub-shapes, and whether they support 2D or 3D shapes.

Table 5. Features in Different Shape Grammar Interpreters.

ID	Author	Language used	Subshape detection	Dimension
1	Gips, 1975	SALT	No	2D
2	Krishnamurti, 1982	Conventional language	Yes	2D
3	Chase 1989	Prolog	Yes	2D
4	Heisseman 1991	C	No	3D
5	Heisseman 1994	C++	No	2D/3D
6	Agarwal & Cangan, 1998	Java	No	2D/3D
7	Piazzalunga 1998	C++	Yes	3D
8	Wang & Duarte 1998	Java	No	3D
9	Tapia, 1999	Unknown	Unknown	2D
10	McGill 2001	Java	No	2D
11	Chau 2002	Perl	No	3D
12	Jowers 2006	PROLOG	Yes	2D
13	Trescak et al., 2009	Java	Yes	2D
14	Li et al. 2009	Perk	No	3D
15	Ertelt & Shea, 2009	C++	No	3D
16	Jowers & Earl, 2010	Unkown	Yes	2D
17	Jowers, Hogg, Mckay, Chau & Pennington, 2010	Unkown	Yes	2D
18	Correia, Duarte & Leitão, 2010	Java	Yes	3D
19	Hoisl & Shea 2010	Python	No	3D
20	Trescak, Esteva & Rodriguez, 2010	Java	Yes	3D
21	Grasl & Economou 2013b	C#	Yes	2D
22	Wang & Zhang, 2018	Java	Yes	2D

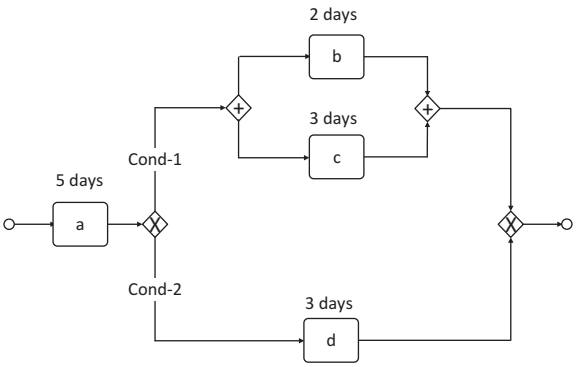


Figure 9. A business process.

4.3. *Integration of time and space*

Spatial-enabled grammars could be used for multimedia applications, which involve a combination of different content formats. Similarly, temporal features are also essential to display multimedia contents and in many real-time applications. For instance, various tasks need to handle time or time sequence, such as project progress schedules, business process sequences, etc. Consider Business Process Modeling Notation (BPMN) (Mazanek & Hanus, 2011) as an example. In Figure 9., each rounded rectangle indicates an

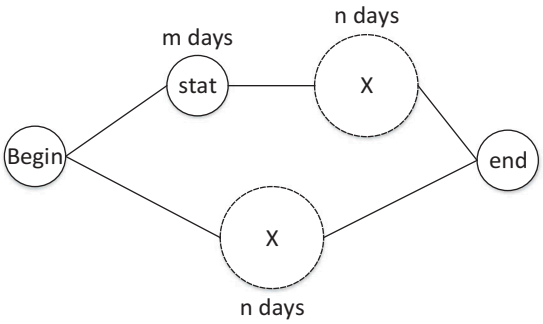


Figure 10. A graph with two redexes.

activity and each node with a "+" or "×" indicates a parallel or exclusive gateway. The graph in Figure 9 implies many temporal requirements, such as searching for one possible path or all possible paths, computing time durations for a path, finding the shortest time to completing the process (e.g., $a \rightarrow b$ is the shortest path), and determining the start time of a given activity.

However, existing approaches only consider the spatial specification while the temporal relation was rarely explored in graph grammars. Traditional graph grammars are also incapable of handling time semantics. For example, Figure 10 shows a flow-process diagram where each node specifies an activity and two subgraphs indicated by "X" in dashed boxes are the redexes of the right graph of a production. Traditional graph grammars treat those two subgraphs as completely identical redexes, leading to inadequate parsing result since two redexes may vary in their start times. Inspired by the concepts of time-environment relationship (TER) nets, Gyapa, Heckel and Varró (2002) developed a time model in attributed graph transformation systems. Focused on time semantics, the model uses node attributes as time stamps to represent the "age" of nodes, and updates those time stamps when a rule is applied. However, merely using time stamps, the model is awkward in handling complex time-related problems.

The capability of specifying both space and time could potentially support significantly more real-world applications. It is therefore desirable to integrate both the temporal and spatial mechanisms into graph grammars to improve their expressive power. Moreover, new restrictions brought by the application conditions for temporal and spatial semantics are useful for narrowing down the search space during parsing. Designing a parsing algorithm with low time complexity is an important requirement for the parser's practical application. However, adding a temporal dimension to graph grammar could raise several challenging issues, such as how to visualize the temporal relation in a 2D or even 3D space, how to analyze time semantics in a graph, how to make full use of temporal semantics to reduce the parsing

complexity, how to efficiently design graph grammars in a higher dimensional space (2D or 3D plus an extra time dimension).

There are mainly two ways to parse a graph that carries time semantics. One is to design a new set of productions concerning only with time specification, separated from the productions with structural specification. The other is to integrate the process of parsing the structural syntax and temporal semantics as one set of productions. The former keeps two different parsing processes apart, which is intuitive for users to design each group of productions separately. However, two parsing processes would be time consuming since the matching computations are duplicated. The latter handles both temporal semantics and structural syntax simultaneously within a single set of productions. The downside would be that complex productions lead to inefficient parsing, especially for those cases involving a large amount of computations on time.

5. Discussion and conclusions

In this article, we have examined recent works on spatial-enabled grammars with spatial specifications from theory to application. We have reviewed the works that combine the grammar formalism and spatial mechanism. As the representative formalisms, shape grammar and SGG are discussed in details, together with their applications. In addition, we propose four evaluation criteria for systematically comparing spatial-enabled grammar formalisms.

Furthermore, using this set of criteria, we compare the shape grammar and SGG formalisms and conclude that the two formalisms complement each other, also providing a guideline for selecting a formalism to suit concrete applications. Moreover, we discuss further developments in several aspects, including grammar induction, shape grammar interpreter, and integration of space and time.

With respect to the criteria, the following observations are outlined for discussion and further exploration.

- There are two major processes in any spatial-enabled grammar formalism in reverse directions: parsing and generation. The parsing is to check whether an input graph or shape is valid according a given grammar, while generation is the reverse process that generates a set of unlimited graphs or shapes satisfying the grammar in a top-down manner. Any grammar formalisms supporting only one direction have limited application power. Would it be possible to extend such a grammar formalism to support the other direction of process?
- The granularity and form of spatial semantics determine qualitative and quantitative specifications for spatial syntax and semantics. Qualitative

descriptions are more intuitive while quantitative descriptions are more precise, depending on application requirements. Many applications with logical reasoning in spatial domains require only qualitative specifications. Other applications, such as graph layout and spatial positioning, may require precise quantitative specifications.
- The ability of specifying 3D models increases the expressive power of a spatial-enabled grammar and extends the application scope. However, spatial specifications for complex 3D scenarios significantly increase the complexity in both specification and processing (i.e., parsing). Also, how 3D views could be rendered effectively without occlusion has been a challenging for many years. Recent advances in virtual reality may help alleviate this issue to a certain extend.

The above discussion and further investigation into these issues may provide insights into further research in the theoretical framework and applications of spatial-enabled grammars to enhance existing spatial-enabled grammars. Here we give two suggestions of the improvement directions.

- Some spatial-enabled grammars are performed as unidirectional process, e.g., shape grammar only supports generation. Therefore, researchers could consider addressing the deficiency by introducing bidirectional mechanism, e.g., adding the parsing capability.
- Since either the qualitative or quantitative specification has its advantages and disadvantages, one may consider to combines two types of specification capabilities in one theoretical framework in a coherent fashion with the user's choice for one or the other. Integral use of both qualitative and quantitative specifications would inevitably increase the parsing complexity and would be an interesting direction to investigate.

We hope our critical analysis and discussion will motivate and facilitate continued research in the creative development of this constantly evolving domain, leading to new approaches and applications that better serve the challenges in spatial-enabled grammars.

Funding

This work was supported by the China Scholarship Council [201706710082]; National Natural Science Foundation of China [61170089,61572348]; National High Technology Research and Development Program of China [215AA020506].

References

Agarwal, A., & Cangan, J. (1998). A blend of different tastes: The language of coffee makers. *Enironment and Planning B: Planning and Design*, *25*(2), 205–226.

Ates, K., Kukluk, J., Holder, L., Cook, D., & Zhang, K. (2006). Graph grammar inference on structural data for visual programming.*Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, pp.232–239.

Ates, K., & Zhang, K. (2007). Constructing VEGGIE: Machine learning for context-sensitive graph grammars. *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence*, pp.456–463.

Brandenburg, F. J. (1994). Designing graph drawings by layout graph grammars. *Proceedings of Dimacs International Workshop on Graph Drawing*, pp.416–427.

Chang, S. K. (1990). A visual language compiler for information retrieval by visual reasoning. *IEEE Transactions on Software Engineering*, *16*(10), 1136–1149.

Chau, H. H., Chen, X., Mckay, A., & Pennington, A. D. (2004). Evaluation of a 3D Shape Grammar Implementation. In *Design computing and cognition'04*, pp. 357–376. Springer, Dordrecht.

Chen, X., Kang, S. B., Xu, Y. Q., Dorsey, J., & Shum, H. Y. (2008). Sketching reality: Realistic interpretation of architectural designs. *ACM Transactions on Graphics*, *27*(2), 1–15.

Chase, S. C. (1989). Shapes and shape grammars: from mathematical model to computer implementation. *Environment & Planning B Planning & Design*, *16*(2), 215–242.

Correia, R. C., Duarte, J. P., & Leitão, A. M. (2010). MALAG: A discursive grammar interpreter for the online generation of mass customized housing.*Proc. 4th Int. Conf. Design Computing and Cognition*, pp. 10–14.

Costagliola, G., Deufemia, V., & Risi, M. (2006). A multi-layer parsing strategy for on-line recognition of hand-drawn diagrams. *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'06)*, pp. 103–110.

Costagliola, G., Lucia, A. D., Orefice, S., & Tortora, G. (1997). A parsing methodology for the implementation of visual systems. *IEEE Transactions on Software Engineering*, *23*(12), 777–799.

Costagliola, G., & Polese, G. (2000). Extended positional grammars. *Proceedings of the 16th IEEE Symposium on Visual Languages*, pp. 103–110.

Economou, A., & Grasl, T. (2017). Paperless Grammars. In *Cultural DNA: Computational studies on the cultural variation and heredity*.

Ertelt, C., & Shea, K. (2009). Application of shape grammars to planning for CNC machining. *Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE*, pp. 651–660.

Gips, J. (1975). Shape grammars and their uses: Artificial perception, shape generation and computer aesthetics. *Birkhauser, Basel*, *10*(4), 230–236.

Gips, J. (1999). Computer implementation of shape grammars. *NSF/MIT Workshop on Shape Computation*, *55*, 56.

Grasl, T., & Economou, A. (2010). Palladian graphs: Using a graph grammar to automate the Palladian grammar. *Proceedings of the 28th eCAADe Conference*, pp. 275–283.

Grasl, T., & Economou, A. (2013a). Unambiguity: Difficulties in communicating shape grammar rules to a digital interpreter. *Proceedings of the 31st eCAADe Conference*, 2, pp. 617–620.

Grasl, T., & Economou, A. (2013b). From topologies to shapes: Parametric shape grammars implemented by graphs. *Environment & Planning B Planning & Design*, *40*(5), 905–922.

Grzesiak-Kopec, K., & Ogorzalek, M. (2013). Intelligent 3D layout design with shape grammars. *Proceedings of the 6th International Conference on Human System Interaction (HIS'13)*, pp.265–270.

Gyapa, S., Heckel, R., & Varró, D. (2002). Graph transformation with time: Causality and logical clocks. *Proceedings of the first International Conference on Graph Transformation*, pp.120–134.

Han, F., & Zhu, S. C. (2005). Bottom-up/top-down image parsing by attribute graph grammar. *Proceedings of the tenth International Conference on Computer Vision*, pp.1778–1785.

Heisserman, J. A. (1992). Generative geometric design and boundary solid grammars. Ph.D. Dissertation, Department of Architecture, Carnegie Mellon University.

Heisserman, L. (1994). Generative geometric design. *IEEE Computer Graphics & Applications*, *14*(2), 37–45.

Hoisl, F., & Shea, K. (2011). Interactive, visual 3D spatial grammar. In *Design Computing and Cognition '10*, Springer, Netherlands (pp. 643–662).

Hou, F., Qi, Y., & Qin, H. (2012). Drawing-based procedural modeling of Chinese architectures. *IEEE Transactions on Visualization and Computer Graphics*, *18*(1), 30–42.

Huang, X. W., Dudek, C. K., Sharman, L., & Szabo, F. E. (2005). From form to content: Using shape grammars for image visualization. *Proceedings of the Ninth International Conference on Information Visualisation*, pp.439-444.

Jonyer, I. (2003). Context-free graph grammar induction based on the minimum description length principle. *Ph.D. Dissertation*, The University of Texas at Arlington, August 2003.

Jowers, I. (2006). Computation with curved shapes: towards freeform shape generation in design, Ph.D. Dissertation, The Open University.

Jowers, I., & Earl, C. (2010). The construction of curved shapes. *Environment & Planning B Planning & Design*, *37*(1), 42–58.

Jowers, I., Hogg, D. C., Mckay, A., Chau, H. H., & Pennington, A. D. (2010). Shape detection with vision: Implementing shape grammars in conceptual design. *Research in Engineering Design*, *21*(4), 235–247.

Kelly, G., & McCabe, H. (2007). Citygen: An interactive system for procedural city generation. *Proceedings the Fifth Annual International Conference in Computer Game Design and Technology (GDTW'07)*, pp.8–16.

Kirishima, T., Goto, T., Yaku, T., & Tsuchida, K. (2010). An attribute graph grammar enabling narrower drawings of trees. *Proceeding of the 9th International Conference on Computer and Information Science (ICIS'10)*, pp.341–346.

Kong, J., Ates, K. L., Zhang, K., & Gu, Y. (2008). Adaptive mobile interfaces through grammar induction. *Proceedings of the 20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'08)*, pp.133–140.

Kong, J., Barkol, O., Bergman, R., Pnueli, A., Schein, S., Zhang, K., & Zhao, C. (2012). Web interface interpretation using graph grammars. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, *42*(4), 590–602.

Kong, J., Zhang, K., Dong, J., & Xu, D. (2009). Specifying behavioral semantics of UML diagrams through graph transformations. *Journal of Systems and Software*, *82*(2), 292–306.

Kong, J., Zhang, K., & Zeng, X. Q. (2006). Spatial graph grammar for graphic user interfaces. *ACM Transactions on Human-Computer Interaction*, *13*(2), 268–307.

Koutsourakis, P., Simon, L., Teboul, O., Tziritas, G., & Paragios, N. (2009). Single view reconstruction using shape grammars for urban environments. *Proceedings of the 12th International Conference on Computer Vision*, pp.1795–1802.

Krishnamurti, R. (1982). SGI: An interpreter for shape grammars. Reseach report, Centre for Configurational Studies, The Open University, Milton Keynes, England.

Krishnamurti, R., & Earl, C. (1992). Shape recognition in three dimensions. *Environment & Planning B*, *19*(5), 585–603.

Krishnamurti, R., & Stouffs, R. (1993). Spatial grammars: Motivation, comparison, and new results. In *International Conference on Computer-Aided Architectural Design Futures*, pp. 57–74.

Leblebici, E., Anjorin, A., & Schürr, A. (2017). Inter-model consistency checking using triple graph grammars and linear optimization techniques. *Proceedings of the 20th International Conference on Fundamental Approaches to Software Engineering*, pp.191–207.

Li, A. K., Chau, H. H., Chen, L., & Wang, Y. (2009). A prototype system for developing two- and three-dimensional shape grammars. In *Proceedings of the 14th In- ternational Conference on Computer Aided Architectural Design Research in Asia* (pp. 717–726).

Li, G., Huang, L., Chen, L., & Yu, C. (2013). Bgg: A graph grammar approach for software architecture verification and reconfiguration. *Proceedings of the 17th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp.291–298.

Li, X., & Chang, S. K. (2004). An interactive visual query interface on spatial/temporal data. *Proceedings of the Tenth International Conference on Distributed Multimedia Systems*, pp.257–262.

Li, Y., Bao, F., Zhang, E., Kobayashi, Y., & Wonka, P. (2011). Geometry synthesis on surfaces using field-guided shape grammars. *IEEE Transactions on Visualization and Computer Graphics*, *17*(2), 231–243.

Li, Y. N., Zhang, K., & Li, D. J. (2017). *Rule-based automatic generation of logo designs*. Leonardo: MIT Press.

Liu, Y., Xu, C. F., Pan, Z. G., & Pan, Y. H. (2004). Semantic modeling project: Building vernacular house of southeast China.*Proceedings of the 2004 ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry (VRCAI'04)*, pp.412–418.

Mahfoud, E., & Willis, A. (2013). Volumetric shape grammars for image segmentation and shape estimation. In *Proceedings of IEEE Southeastcon*, pp.1–6.

Marriott, K. (1994). Constraint multiset grammars. *Proceedings of IEEE Symposium on Visual Languages*, pp.118–125.

Marriott, K., & Meyer, B. (1997). On the classification of visual languages by grammar hierarchies. *Journal of Visual Languages & Computing*, *8*(4), 375–402.

Mayall, K., & Hall, G. B. (2005). Landscape grammar 1: Spatial grammar theory and landscape planning. *Environment and Planning B: Planning and Design*, *32*(6), 895–920.

Mazanek, S., & Hanus, M. (2011). Constructing a bidirectional transformation between BPMN and BPEL with a functional logic programming language. *Journal of Visual Languages and Computing*, *22*(1), 66–89.

McGill, M. C. (2002). Shaper2D: visual software for learning shape grammars. In *Proceedings of the 20th Conference on Education in Computer Aided Architectural Design in Europe*, eCAADe, Warsaw, pp. 148–151.

Muller, P., Wonka, P., Haegler, S., Ulmer, A., & Gool, L. V. (2006). Procedural modeling of buildings. *Proceedings of the 33th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '06)*, pp.614–623.

Muller, P., Zeng, G., Wonka, P., & Gool, L. V. (2007). Image-based procedural modeling of facades. In *Proceedings of ACM SIGGRAPH (SIGGRAPH '07)*, ACM New York, NY, USA (pp. 85).

Orefice, S., Polese, G., Tucci, M., Tortora, G., Costagliola, G., & Chang, S. K. (1992). A 2D interactive parser for iconic language. *Proceedings of IEEE Workshop on Visual Languages*, pp.207–213.

Piazzalunga, U., & Fitzhorn, P. (1998). Note on a three-dimensional shape grammar interpreter. *Environment and Planning B-Planning & Design*, *25*(1), 11–30.

Qi, S. (2015). *Analysis by synthesis: 3D image parsing using spatial grammar and markov chain monte carlo*. University of California Los Angeles, USA: M.S. Dissertation.

Qiu, M. K., Song, G. L., Kong, J., & Zhang, K. (2003). Spatial graph grammars for web information transformation. *Proceedings of IEEE Symposium on Visual/Multimedia Languages*, pp.84–91.

Rekers, J., & Schürr, A. (1997). Defining and parsing visual languages with layered graph grammars. *Journal of Visual Languages and Computing*, *8*(1), 27–55.

Roudaki, A., Kong, J., & Zhang, K. (2016). Specification and discovery of web patterns: A graph grammar approach. *Information Sciences*, *328*(C), 528–545.

Rozenberg, G. (1997). *Handbook on graph grammars and computing by graph transformation: Foundations*. Singapore: World Scientific

Stiny, G. (1975). *Pictorial and formal aspects of shape and shape grammars*. University of California, Los Angeles Birkhäuser: Basel.

Stiny, G. (1980). Introduction to shape and shape grammars. *Environment & Planning B Planning & Design*, *7*(3), 343–351.

Stiny, G. (2006). *Shape: Talking about seeing and doing* (pp. 58). Cambridge, Massachusetts, London, England: The MIT Press.

Stiny, G., & Gips, J. (1971). Shape grammars and the generative specification of painting and sculpture. In *Proceedings of the Workshop on Generalisation & Multiple Representation Leicester*, 71:1460–1465.

Strobbe, T., De Meyer, R., & Van Campenhout, J. (2015). A semi-automatic approach for the definition of shape grammar rules. *Journal of Materials Processing Technology*, *136*(s1–3), 174–178.

Tapia, M. (1999). A visual implementation of a shape grammar system. *Environment & Planning B Planning & Design*, *26*(1), 59–73.

Tching, J., Reis, J., & Paio, A. (2013). Shape grammars for creative decisions: In the architectural project. *Proceedings of the 8th Iberian Conference on Information Systems and Technologies (CISTI'13)*, pp.1–6.

Teboul, O., Kokkinos, I., Simon, L., Koutsourakis, P., & Paragios, N. (2010a). Shape grammar parsing via reinforcement learning. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*, pp.2273–2280.

Teboul, O., Kokkinos, I., Simon, L., Koutsourakis, P., & Paragios, N. (2013). Parsing facades with shape grammars and reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(7), 1744–1756.

Teboul, O., Simon, L., Koutsourakis, P., & Paragios, N. (2010b). Segmentation of building facades using procedural shape priors. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pp.3105–3112.

Trescak, T., Esteva, M., & Rodriguez, I. (2010). A virtual world grammar for automatic generation of virtual worlds. *The Visual Computer*, *26*(6–8), 521–531.

Trescak, T., Rodríguez, I., & Esteva, M. (2009). General shape grammar interpreter for intelligent designs generations. *Proceedings of the Sixth International Conference on Computer Graphics, Imaging and Visualization (CGIV'09)*, pp.235–240.

Tsai, W. H., & Fu, K. S. (1980). Attributed grammar-a tool for combining syntactic and statistical approaches to pattern recognition. *IEEE Transactions on Systems, Man and Cybernetics*, *10*(12), 873–885.

Tutenel, T., Smelik, R. M., Lopes, R., Kraker, K. J. D., & Bidarra, R. (2010). Generating consistent buildings: A semantic approach for integrating procedural techniques. *IEEE Transactions on Computational Intelligence and AI in Games*, *3*(3), 274–288.

Wang, X. Y., & Zhang, K. (2018). Enhancements to a shape grammar interpreter. In *Proceedings of 3rd International Workshop on Interactive and Spatial Computing (IWISC'18)*, pp. 8–14.

Wang, Y., & Duarte, J. P. (2002). Automatic generation and fabrication of designs. *Automation in construction*, *11*(3), 291–302.

Weber, B., Muller, P., Wonka, P., & Gross, M. (2009). Interactive geometric simulation of 4D cities. *Computer Graphics Forum*, *28*(2), 481–492.

Wittenburg, K., B., & Weitzman, L., M. (1996). Relational grammars: Theory and practice in a visual language interface for process modeling. *Proceedings of Workshop on Theory of Visual Languages*, May 30, Gubbio, Italy, pp.193–217. Springer New York.

Wonka, P., Wimmer, M., Sillion, F., & Ribarsky, W. (2003). Instant architecture. *Proceedings of the 30th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'03)*, pp.669–677.

Yue, K., & Krishnamurti, R. A. (2012). A paradigm for interpreting tractable shape grammars. *Environment & Planning B Planning & Design*, *41*(1), 110–137.

Zhang, D. Q., Zhang, K., & Cao, J. (2001a). A context-sensitive graph grammar formalism for the specification of visual languages. *The Computer Journal*, *44*(3), 187–200.

Zhang, K., & Kong, J. (2010). Exploring semantic roles of web interface components. *Proceedings of International Conference on Machine and Web Intelligence (ICMWI'10)*, pp.8–14.

Zhang, K., Kong, J., Qiu, M., & Song, G. L. (2005). Multimedia layout adaptation through grammatical specifications. *Multimedia Systems*, *10*(3), 245–260.

Zhao, C. Y., Kong, J., & Zhang, K. (2010). Program behavior discovery and verification: A graph grammar approach. *IEEE Transactions on Software Engineering*, 36, 431–448.