Title:
The STG Pattern: Application of a "Semantic-Topological-Geometric" Information Conversion Pattern to Knowledge-Based Algorithmic Modeling for Architectural Design

Authors:
Chieh-Jen Lin, t60011@mail.tut.edu.tw, Tainan University of Technology

Keywords:
Parametric Modeling, Generative Algorithm, Design Criteria, Design Pattern, Semantic Ontology

DOI: 10.14733/cadconfP.2016.65-69

Introduction:
Generative modeling tools like Grasshopper have become a popular means of composing algorithms for generating complex building forms, optimizing multiple design objectives, and structural and sustainability control[12] at the conceptual design stage. The visual programming interfaces of Grasshopper are easier to learn and understand than textual programming tools. With the help of immediate feedback of visualized 3D models in Rhino, generative modeling tools allow architects to freely explore creative ideas expressing geometric intentions. Except in the case of constructability issues involving complex geometric forms, however, one of the major issues affecting application of generative modeling in architectural design is how to associate generative algorithms with known design criteria in order to evaluate whether the generated forms are acceptable or not. How to compose algorithms in order to meet the requirements of general design criteria, and how to communicate those criteria with other disciplines by means of generative algorithms still faces many technical challenges.

The "Pattern Language" proposed by Alexander contains conclusions concerning the good practices of endemic buildings that serve as design paradigms for acquiring the knowledge needed to solve common problems[1]. However, while few architects actually employ Alexander's language, a relatively large number of software engineers apply "design patterns" in identifying and reusing the best practices in known situations. In the software engineering domain, "design patterns" do not determine the final design of software, but instead specify methodologies for solving commonly occurring problems within a given context. Some design patterns have been accepted as best practices, such as the model-view-controller (MVC) pattern for implementing user interfaces based on object-oriented programming. With generative modeling and parametric design becoming more popular and important in architectural design, how to translate design criteria into computational procedures has become a widespread problem when applying generative modeling tools. However, there is still no pattern to guide architects in composing generative algorithms in order to implement their design knowledge and criteria.

Main Idea:
With the widespread penetration of digital tools into almost all areas of architectural design, including both education and practice, digital tool application skills and knowledge have become more important than in the past. To apply generative modeling tools such as Grasshopper, a designer needs more knowledge of mathematical formulas than basic architecture knowledge[10] , and more data processing skills than aesthetics skills involving geometric forms. However, the need for additional skill and knowledge often causes the results of parametric design to be disconnected from architectural contexts, such as material, user, and usage requirements, and causes designers to expend more effort on programming/scripting of algorithms than on architectural design[12].

Grasshopper is a graphic algorithm editor integrated with Rhino that can be used to explore novel geometric shapes, and cognitive studies have revealed that parametric design mainly supports designers' geometric intentions[12]. Beyond geometric intentions, designers also tend to select existing solutions instead of developing new solutions for known problems, which meets the definitions of Alexander's pattern[11]. Obviously, generative algorithms should potentially be able to go beyond geometric intentions[8]. For example, the use of generative algorithms in optimization of spatial planning[2] and building performance[3] during early design stages has been explored. Although those attempts chiefly consisted of implementations of existing algorithms for specific design issues, such as spatial syntax for space planning, and genetic algorithms for optimizing multiple objectives concerning building performance, the results demonstrated potential application to more general design criteria than just the realization of novel geometric intentions.

While the impact of generative modeling and parametric design on thinking and methodology during the early and developing design stages, BIM applications have been used to extensively improve workflow during the later and detailed design stages. Since they can manipulate more semantic and topological information concerning building components than 2D CAD or 3D models, BIM applications provide a convenient platform for visually communicating with different disciplines, especially concerning the mechanical, electrical, and plumbing (MEP) engineering of a project. Based on the design information schema of BIM, which consist of semantic, topological, and geometric information, and referring to the MVC pattern in software engineering, therefore this paper proposes an algorithmic design pattern based on a "semantic-topological-geometric (STG)" framework[7], and this pattern can enable designers compose algorithms for modeling general design criteria at the early design stages (Fig. 1).

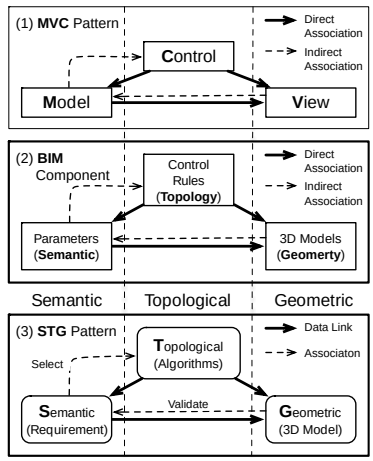

Fig. 1: Mapping information processing from MVC, BIM, to STG pattern[7].

The first component of the STG pattern is "Semantic Ontology," which consists of an information model of design criteria. In the MVC pattern, a "model" module is the central component used to capture behavior and logical rules in the problem domain. To associate algorithmic generative modeling with design criteria, it must first represent design criteria in a computable format. In early design stages, architectural design criteria are usually abstract and textual descriptions of various requirements. In order to represent semantic criteria in architectural design into a computable format, semantic ontology was incorporated into the STG pattern.

The second component of the STG pattern is the "Topological Relation," which is a controlling algorithm for design criteria. Eastman declared that topologies are the fundamental definitions of parametric models in BIM[4]. In an early design stage, the topological relations of design criteria are usually abstract, and may consist of enclosure, extension, and concentration of indoor/outdoor spaces and building masses[5]. In MVC pattern software architecture, a "Controller" module is used to accept operations from users to modify the data within models. "Controller" can therefore control the

interactive behavior among different “models” in a system. Topological relations among design criteria can thus be regarded as the “Controller” of design criteria.

One of the major obstacles to applying algorithmic generative modeling is that stakeholders cannot understand the algorithms, especially in the case of those algorithms too complex to be explained by designers who are actually implementing those algorithms. The “Model” of semantic ontology and the “Controller” of topological relations of design criteria can therefore help to associate algorithmic generative modeling tools like Grasshopper with the architectural design knowledge that was applied within those algorithms.

The final component of the STG pattern is the “Geometric Feature,” which can validate views of design criteria. In a MVC pattern, a “View” module is used to display information of a retrieved “Model” and the results of “Controller.” In architectural design, architects always need visual feedback to validate the content of semantic ontologies or computing behaviors of algorithms. Immediate visual feedback of generative algorithms is one of the most attractive features of Grasshopper for designers. However, designers mainly apply Grasshopper to the exploration of geometric intentions. The visual validation of other design criteria, rather than geometric intentions, is usually ignored, especially in the case of those that are invisible or non-obvious.

Unlike Alexander’s pattern language, which can package instances of solutions with relevant design problems, a programming design pattern should not only provide direct solutions to known problems, but also meta-knowledge for identifying design situations, and then select appropriate methods for addressing those problems. In the wake of a “geometric-topological-geometric” information conversion framework[7], this paper proposes a algorithmic pattern for modeling general design criteria that go beyond the geometric intentions of a building's form.

For example, because an enclosed and quiet space may make children nervous, the architect Tezuka asserted that the classrooms in a good kindergarten should not have boundaries between inside/outside[9]. Tezuka therefore designed the “Fuji Kindergarten,” an award-winning kindergarten in Tokyo, which is without walls, and enclosed only by sliding patio doors (Fig. 2). In semantic hierarchy of BIM, doors and windows are usually parts or sub-classes of a wall. In a BIM application, it usually must model a wall firstly then to insert patio doors and to enlarge those doors enough make the wall disappear. Therefore, it sometime is difficult to model a building like “Fuji Kindergarten” in a BIM application.

Fig. 2: The classrooms of the Fuji Kindergarten designed by Takaharu Tezuka: (a) classrooms are enclosed by patio doors but without walls; and (b) the patio doors can be fully opened to remove the boundaries of classrooms.

In previous studies, a prototype Python scripts, which was entitled “Design Criteria Modeling (DCM),” which sought to help designers to model and validate semantic ontology within Grasshopper by hooking with OWL files and the SWRL reasoner of Protégé, was implemented[7]. The DCM prototype was provided to students for modeling their semantic design criteria. They were asked to rapidly design a “Community-Friendly Primary School,” which was a topic on Taiwan's architect qualification exam Taiwan in 2015 (Fig. 3a). The design context consisted of two sites located on north and south sides of a primary school, and the building’s purpose was for the learning and leisure usage of seniors in the community. Except for such basic issues as the building code, traffic, and climate response, the

existing site context, including existing trees, classrooms, exercise yard, and green areas, served as the predominant element for the development of design criteria (Fig. 3b).

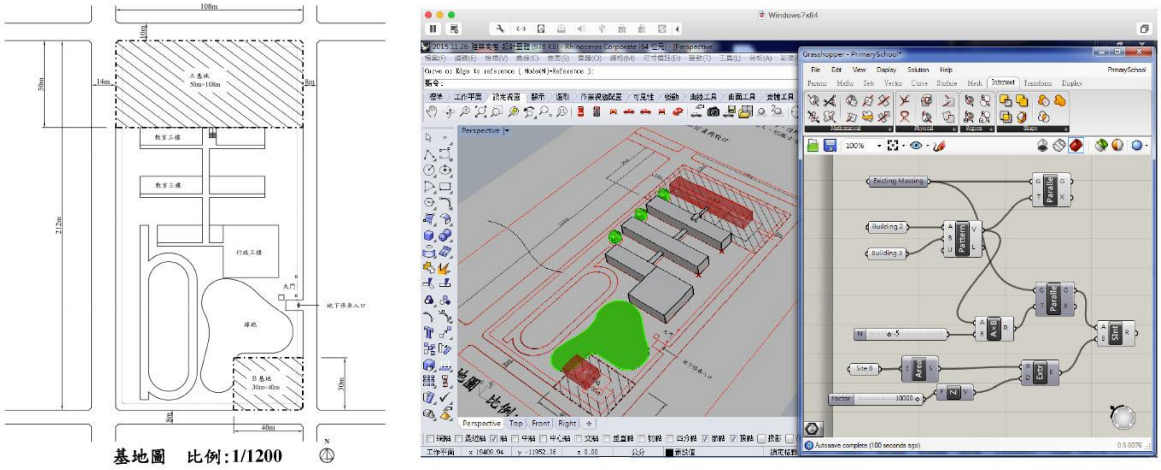

Fig. 3: The site context of a "Community-Friendly Primary School" exam: (a) the context and content of the site, and (b) test modeling of design criteria.

In this test, most of the students found the strong patterns of existing buildings, including an axis formed by the central corridor, and the rhythm formed by parallel building masses. They therefore tended to follow the axis by extending the central corridor to connect both sites, and then arranged the new building to be parallel with existing buildings. The students could consequently first code the "Parallel" and "Axis" topology, then sought to implement the algorithms of "Parallel" and "Axis," and finally select the input parameters, such as an existing building and its gaps as the generating rules (Figure 4.b).

Conclusions:

At present, parametric design mainly is applied to complex building form generation, multiple design solution optimization, and structural and sustainability control[12] . Parametric design is not only a novel tool of digital architectural design, but also a new methodology of architectural design thinking. Following the generative approach, Kotnik concluded that the exploration of computing functions is a critical feature of digital architectural design[6]. However, there may be insufficient clues for discovering the computing functions of general architectural design criteria, especially for those abstract concepts proposed by architects and emerging in the early design stages. For expanding the use of parametric design, this paper proposes an "STG" pattern based on the "semantic-topological-geometric" information-converting framework to guide designers in modeling design criteria knowledge in parametric modeling.

In view of the fact that one purpose of the MVC pattern is to divide programming tasks in complex systems into small, discrete, and independent objects, the STG pattern divides parametric design into three parts characterized by computable functions, which can implement generative algorithms by different designers. As building projects become more complex, instead of requiring architects wear many hats associated with other disciplines, it is better to hand over programming/scripting tasks of complex geometric generation and performance optimization to software and MEP engineers. It is therefore time to embed basic and traditional design knowledge back into the parameters and variables of generative architectural design.

Acknowledgements:

The Ministry of Science and Technology of Taiwan support this paper under grant number MOST 103-2221-E-165-001-MY2.

References:

[1] Alexander, C.; Ishikawa, S.; Silverstein, M.: A Pattern Language : Towns, Buildings, Construction, Oxford University Press, New York, 1977.
[2] Bazalo, F.; Moleta, T.J.: Responsive Algorithms - An investigation of computational processes in early stage design, in: Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA 2015), Daegu, 2015, 209-218.
[3] Chang, M.-C.; Shih, S.-G.: A Hybrid Approach of Dynamic Programming and Genetic Algorithm for Multi-criteria Optimization on Sustainable Architecture Design, Computer-Aided Design and Applications, 12(3), 2014, 310-319. http://dx.doi.org/10.1080/16864360.2014.981460
[4] Eastman, C.; Teicholz, P.; Sacks, R.; Liston, K.: BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors, 2nd ed., John Wiley & Sons Inc., Hoboken, N.J., 2011. http://dx.doi.org/10.1002/9780470261309
[5] Ho, H.-Y.; Wang, M.-H.: Meta Form as a Parametric Design Language, in: eCAADe 2009, Istanbul, Turkey, 2009, 713-718.
[6] Kotnik, T.: Digital Architectural Design as Exploration of Computable Functions, International Journal of Architectural Computing, 8(1), 2010, 1-16. http://dx.doi.org/10.1260/1478-0771.8.1.1
[7] Lin, C.-J.: Design Criteria Modeling – Use of Ontology-Based Algorithmic Modeling to Represent Architectural Design Criteria at the Conceptual Design Stage, in: Proceedings of the 13th annual International CAD Conference (CAD'15), London, UK, 2015, 337-341. http://dx.doi.org/10.14733/cadconfP.2015.337-341
[8] Terzidis, K.: Algorithmic Architecture, Architectural Press, Burlington, MA, 2006.
[9] Tezuka, T., The Best Kindergarten You've Ever Seen, https://http://www.ted.com/talks/takaharu_tezuka_the_best_kindergarten_you_ve_ever_seen.
[10] Woodbury, R.: Elements of Parametric Design, Routledge, New York, 2010.
[11] Yu, R.; Gero, J.: An Empirical Foundation for Design Patterns in Parametric Design, in: Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA 2015), Daegu, 2015, 551-560.
[12] Yu, R.; Gero, J.; Gu, N.: Architects' Cognitive Behaviour in Parametric Design, International Journal of Architectural Computing, 13(1), 2015, 83-102. http://dx.doi.org/10.1260/1478-0771.13.1.83