S. Chien, S. Choo, M. A. Schnabel, W. Nakapan, M. J. Kim, S. Roudavski (eds.), *Living Systems and Micro-Utopias: Towards Continuous Designing, Proceedings of the 21st International Conference of the Association for Computer-Aided Architectural Design Research in Asia CAADRIA 2016,* 435–444. © 2016, The Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), Hong Kong.

THE STG PATTERN

Application of a "Semantic-Topological-Geometric" information conversion pattern to knowledge modelling in architectural conceptual design

CHIEH-JEN LIN
Tainan University of Technology, Tainan, Taiwan
T60011@mail.tut.edu.tw

Abstract. Generative modelling tools have become a popular means of composing algorithms to generate complex building forms at the conceptual design stage. However, composing algorithms in order to meet the requirements of general design criteria, and communicating those criteria with other disciplines by means of generative algorithms still faces technical challenges. This paper proposes the use of a "Semantic-Topological-Geometric (STG)" pattern to guide architects in composing algorithms for representing, modelling, and validating design knowledge and criteria. The STG pattern aims to help architects for converting semantic information concerning the situations of a project into design criteria, which are usually composed of topological relations among design elements, in order to explore the geometric properties of building components by means of generated 3D models.

Keywords. Generative modelling; design criteria; design pattern; semantic ontology; BIM.

1. 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 (Yu et al, 2015) 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 the constructability issues involving complex geometry, however, one of the major issues affecting application of generative modeling 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 design criteria, and how to communicate those criteria and algorithms with other disciplines still faces many technical challenges.

The Alexander's "Pattern Language" concluded the good practices of endemic buildings that serve as a paradigm for acquiring the knowledge to solve common problems (Alexander et al, 1977). However, while few architects actually employ Alexander's language, more 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 based on object-oriented programming. With generative modelling and parametric design becoming more popular and important in architectural design, how to translate design criteria into algorithms has become a widespread problem when applying generative modelling tools. However, there is still no pattern to guide architects in composing algorithms to implement their design criteria.

2. Computational patterns of digital architectural design

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 modelling tools such as Grasshopper, a designer needs more knowledge of mathematics and data processing than basic architecture knowledge and aesthetics skills (Woodbury, 2010). However, the need for additional skill and knowledge 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 (Yu et al, 2015).

Cognitive studies have revealed that parametric design mainly supports designers' geometric intentions (Yu et al, 2015). Beyond geometric intentions, designers tend to select existing solutions instead of developing new solutions for known problems, which meets the ideas of Alexander's pattern (Yu and Gero, 2015). Obviously, generative algorithms should potentially be able to go beyond geometric intentions (Terzidis, 2006). For example, the use of generative algorithms in optimization of spatial planning (Bazalo and

Moleta, 2015) and building performance (Chang and Shih, 2014) during early design stages has been explored. Although those attempts had implemented 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 design criteria than just novel geometric intentions.

While the impact of generative modelling and parametric design on thinking and methodology during the early design stages, BIM have been used to improve workflow during the later design stages. Since users can manipulate more semantic and topological information of building components than 2D CAD and 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 (Lin, 2015), and this pattern aims to enable designers compose algorithms for modelling general design criteria at the early design stages.

2.1. SEMANTIC ONTOLOGY AS INFORMATION MODELS OF DESIGN CRITERIA

The first component of the STG pattern is "Semantic Ontology," which consists of a semantic information model of design criteria. In a MVC-based system, a "model" module is used to capture behaviour and rules in the problem domain. To associate generative modelling 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 into a computable format, semantic ontology was incorporated into the STG pattern.

Protégé is one of popular tools for encoding domain knowledge of AI. Protégé is based on the Ontology Web Language (OWL), which is an XML-based language, and was originally used to develop semantic networks. Based on the semantic web rule language (SWRL), which can used to express rules as well as logic, the logic reasoner in Protégé can validate and infer implicit knowledge within an ontology. If architectural design criteria can be expressed in OWL, then a SWRL reasoner can help architects and stakeholders of a building project to validate criteria, infer implicit criteria, and keep those criteria consistent throughout the design process.

For example, because an enclosed and quiet space may make children nervous, the Japanese architect Tezuka asserted that the classrooms in a good kindergarten should not have boundaries between inside/outside (Tezuka, 2014). Tezuka therefore designed the "Fuji Kindergarten," an award-winning kindergarten in Tokyo, which is without walls, and enclosed only by sliding patio doors (Figure 1). In this context, design criteria may be presented as a simplified set of semantic ontologies as follows: (1) "Classroom" with the "hasBoundary" property will prompt the inference of a "NotGoodClassroom" and "FeelNervous" property, (2) "Classroom" with "hasNoBoundary" property will prompt the inference of a "GoodClassroom" and "FeelRelaxed" property, (3) "Wall" is a kind of "Boundary," and (4) "Roof," "Floor," and "PatioDoor" with the "CanFullyBeOpened" property are not a kind of "Boundary." Although this ontology cannot directly generate a result of "GoodClassroom," the logic reasoner can infer that an instance of "Classroom" without "Wall" but enclosed by "PatioDoor" with "CanFullyBeOpened" property is an instance of "GoodClassroom."

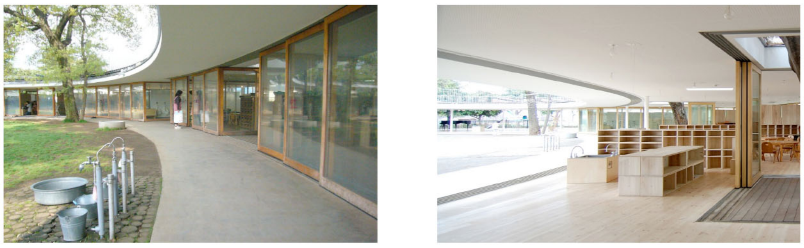

Figure 1. The classrooms of the Fuji Kindergarten designed by Takaharu Tezuka: (a) the classrooms are enclosed by patio doors but without walls (left), and (b) the patio doors can be fully opened to remove the boundaries of classrooms (right) (OpenBuildings, 2014).

2.2. TOPOLOGICAL RELATIONS AS CONTROLLING ALGORITHMS OF DESIGN CRITERIA

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 (Eastman et al, 2011). 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 (Ho and Wang, 2009). Except for some geometric rules, such as the constraints in Autodesk Revit, abstract topologies for early design stages are absent from most BIM applications, however.

In an MVC-based system, a "Controller" module is used to accept operations from users to modify the data of models. "Controller" can therefore control the interactive behaviour among different “models” in a system. Topological relations among design criteria can thus be regarded as the “Controller” of design criteria. In the case of the “Fuji Kindergarten,” for example, the architect’s design concept was that all “Classrooms” must be “Connective” and “Enclose” the “Playground,” but “Avoid” existing “Trees” (Figure 2). The topological relations, such as “Connective,” “Enclose,” and “Avoid” relative to the “Classroom,” “Playground,” and “Tree,” become the “Topological” components in a STG pattern waiting for implementation.

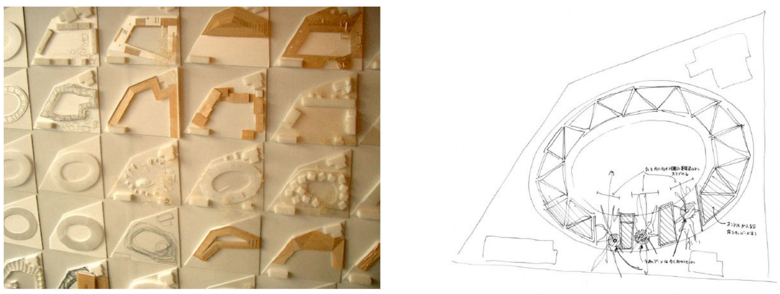

Figure 2. The topological concepts of the Fuji Kindergarten: (a) conceptual models for exploring the similar concepts of topological relations (left), and (b) a diagram of the design criterion for avoiding existing trees in the site (right) (OpenBuildings, 2014).

2.3. GEOMETRIC FEATURES AS VALIDATING VIEWS OF DESIGN CRITERIA

The final component of the STG pattern is the “Geometric Feature,” which can validate geometric views of design criteria. In an MVC-based system, a “View” module is used to display a retrieved “Model” and the results of “Controller.” In architectural design, architects always need visual feedback to validate the semantic ontologies or computing behaviours of algorithms. Immediate visual feedback of generative algorithms is one of the most attractive features of Grasshopper for designers. 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.

In the case of the “Fuji Kindergarten,” for example, the concept of building form consists of a simple circle around the playground. However, the architect must carefully arrange the geometric features of the roof into two ellipses in order to fit the irregular shape of the available land within the site, and avoid existing trees and buildings (Figure 3.a). Even though there may be some controlling algorithms concerning the shapes of the building and the site (Figure 3.b), those design criteria sometimes cannot be clearly observed in the design results. With the help of algorithmic generative modelling tools like Grasshopper, which can take complex geometric features consisting of 2D shapes or 3D models as input parameters for algorithms, a designer can easily explore appropriate combinations of algorithms assign means of representing geometric intentions. The "Model" of semantic ontology and the "Controller" of topological relations of design criteria can help to associate algorithmic generative modelling tools like Grasshopper with the architectural design knowledge that was applied within those algorithms.

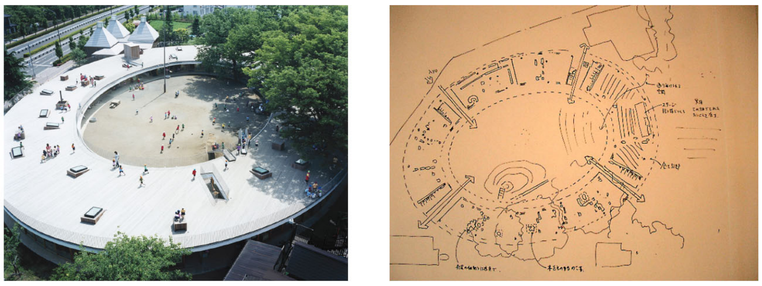
Figure 3. Geometric features of roof of the Fuji Kindergarten: (a) the geometric features of the roof consist of two simple ellipses (left), and (b) a diagram of invisible design criteria concerning the building and the site (right) (OpenBuildings, 2014) .

3. Implementation and initial testing of STG

In previous studies, a prototype Python scripts entitled "Design Criteria Modelling (DCM)" was implemented to help designers to model and validate semantic ontology within Grasshopper by hooking with OWL files and the SWRL reasoner of Protégé (Lin, 2015). The DCM prototype was provided to students for modelling their semantic criteria of a "Community-Friendly Primary School" design, which was a topic on Taiwan's architect qualification exam in 2015. The design context consisted of two sites located on north and south sides of a primary school (Figure 4.a), 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, the existing site contexts served as the predominant element for the development of design criteria.

In this test, some 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. Those 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).

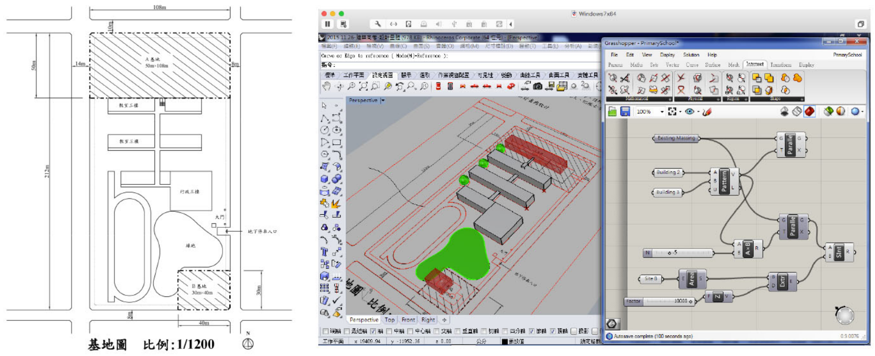

Figure 4. The rapidly design exam of a "Community-Friendly Primary School": (a) the context and content of the site (left), and (b) test modelling of design criteria (right).

4. Discussion

Because of more requirements for prior knowledge apart from basic architectural knowledge, many misunderstandings of digital architectural design have also arrived with the increased availability of digital tools (Peters, 2013). Confusion about methods, thinking, and modelling among generative, parametric, and algorithmic approaches has emerged with more new digital tools. Kotaik therefore proposed a theoretical framework that addressed the differences among the generative, parametric, and algorithmic approaches in terms of computability (Kotnik, 2010). Kotaik remapped Oxman's five class of digital design, which are CAD, formation, performance, generative, and integrated compound model (Oxman, 2008), into three levels of design computability, which are representational, parametric, and algorithmic from low to high computability. A formal description of digital architectural design was proposed as a computable function for generating "variables" of architectural properties by inputting "parameters" of the architectural factors of design situations. The STG pattern is discussed as follows based on Kotaik's computability outlook of computable functions.

4.1. SEMANTIC ONTOLOGY AS THE INPUT PARAMETERS OF COMPUTABLE FUNCTIONS

Woodbury defined parametric design thinking as thinking with abstraction, mathematically, and algorithmically (Woodbury, 2010). Algorithmic thinking means to understand the interpretive correlations between algorithms and architectural design criteria (Bazalo and Moleta, 2015). However, most of abstract design criteria at early design stages are too abstract to input as parameters, and it is not easy to find an algorithm for generating or validating them. For instance, designers cannot input a “FeelRelaxed” property into a function for generating the “hasNoBoundary” property of a classroom.

Kotnik indicated the output “Variable” should be a building property, such as a building’s form or components. However, the possible “Parameters” are left with open definitions. If the input parameters cannot be associated with any known architectural design criteria, in view of the principle of “garbage-in, garbage-out,” it is not surprising that parametric design is often criticized as being disconnected from basic architectural design knowledge. In order to convert semantic design criteria into computable parameters to facilitate machine processing, it is necessary to represent them in a formal and computable format. With help from semantic ontology based on OWL, it is easy to infer implicit design criteria and validate given design criteria by employing the SWRL reasoner of Protégé. Semantic criteria at an early design stages can therefore be regarded as input parameters into the algorithms of generative modelling tools like Grasshopper.

4.2. TOPOLOGICAL RELATIONS AS GENERATIVE ALGORITHMS OF COMPUTABLE FUNCTION

Generative algorithms are the cores of a computable function. However, a parametric model implies the algorithms are fixed, and the correlations between parameters and variables are predictable. Cognitive studies has found that designers spend more effort for finding appropriate algorithms fitting their intentions (Yu et al, 2015), rather than testing possible output variables by modifying the input parameters. The extra costs of applying parametric modelling usually do not mean those problems are so novel that the solving algorithms have not been invented, but usually only mean that the designer does not know the solution or has not mastered the algorithms.

Although Grasshopper requires no scripting skills, when designers’ intentions become complex, mathematics knowledge and programming skill become critical. Researchers claim that parametric design imposes new challenges by asking architects to play the roles of programmers (Yu et al, 2015). However, algorithms of some complex geometry have already been developed, and some even have been packaged as plugins, such as the “LunchBox” plugin of Grasshopper. The population of “LunchBox” plugin reveals that extra effort is usually spent on implementing algorithms by scripts, rather than converting design knowledge into algorithms. In the case of Fuji Kindergarten, for example, the architect can only identify the topological relations, such as connective and surrounding as the controller, and then identify classrooms and the playground as the input parameters. Thus, scripting tasks can be handed over to professional programmers/scripters.

4.3. GEOMETRIC FEATURES AS VISUAL VALIDATIONS OF COMPUTABLE FUNCTIONS

The original purpose of Grasshopper was to automate and accelerate modelling tasks. However, the computability of generative algorithms does not be restricted to generating complex geometries. According to Kotnik's opinions, a "threshold" between non-digital and digital design is located between representational and parametric models of design thinking and methods, and the representational applications of digital tools are only an alternative of paper drawings and hand-made models like traditional 2D/3D CAD software.

From the view of computability, the application of "LunchBox" to generate complex geometries actually is closer to representational than parametric. The critical design knowledge is not located in the selection of algorithms, but rather in the selection of input parameters and how to filter out generated forms. In the case of the Fuji Kindergarten, before the architect took an ellipse as the predominant geometry, he had to develop more criteria concerning geometric features of the playground and the building, such as minimal required spaces, and the necessary retreats along the trees and the site. Even though these criteria may not directly generate forms, the generated geometries still can be used to visually validate those criteria. For example, a designer can model existing trees into "subtraction" objects for generating the open spaces. Therefore the geometric features of generative forms can also visually validate those invisible design criteria of architectural design.

5. Conclusions

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. 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 modelling design criteria knowledge in parametric modelling.

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 has supported this paper under grant number MOST 103-2221-E-165-001-MY2.

References

Alexander, C., Ishikawa, S. and Silverstein, M.: 1977, *A Pattern Language: Towns, Buildings, Construction*, Oxford University Press, New York.

Bazalo, F. and Moleta, T. J.: 2015, Responsive Algorithms - An investigation of computational processes in early stage design, *Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia* (CAADRIA 2015), Daegu, 209–218.

Chang, M.-C. and Shih, S.-G.: 2014, A Hybrid Approach of Dynamic Programming and Genetic Algorithm for Multi-criteria Optimization on Sustainable Architecture Design, *Computer-Aided Design and Applications*, **12**(3), 310–319.

Eastman, C., Teicholz, P., Sacks, R. and Liston, K.: 2011, *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors*, 2nd ed., John Wiley & Sons Inc., Hoboken, N.J.

Ho, H.-Y. and Wang, M.-H.: 2009, Meta Form as a Parametric Design Language, *eCAADe 2009,* Istanbul, Turkey, 713–718.

Kotnik, T.: 2010, Digital Architectural Design as Exploration of Computable Functions, *International Journal of Architectural Computing*, **8**(1), 1–16.

Lin, C.-J.: 2015, Design Criteria Modeling, *Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia* (CAADRIA 2015), Daegu 479–488.

OpenBuildings: 2014, "Fuji Kindergarten". Available from: <http://openbuildings.com/buildings/fuji-kindergarten-profile-2425> (accessed 20 November 2015).

Oxman, R.: 2008, Digital Architecture as a Challenge for Design Pedagogy: Theory, Knowledge, Models and Medium, *Design Studies*, **29**(2), 99–120.

Peters, B.: 2013, Computation Works: The Building of Algorithmic Thought, *Architectural Design*, **83**(2), 8–15.

Terzidis, K.: 2006, *Algorithmic Architecture*, Architectural Press, Burlington, MA.

Tezuka, T.: 2014, "The Best Kindergarten You've Ever Seen". Available from: <https://www.ted.com/talks/takaharu_tezuka_the_best_kindergarten_you_ve_ever_seen> (accessed 20 November 2015).

Woodbury, R.: 2010, *Elements of Parametric Design*, Routledge, New York.

Yu, R. and Gero, J.: 2015, An Empirical Foundation for Design Patterns in Parametric Design, *Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia* (CAADRIA 2015), Daegu, 551–560.

Yu, R., Gero, J. and Gu, N.: 2015, Architects' Cognitive Behaviour in Parametric Design, *International Journal of Architectural Computing*, **13**(1), 83–102.