Explorations in Teaching Design Students to Think and Produce Computationally

Birgul Colakoglu
Yildiz Technical University, Turkey
http://www.bot.yildiz.edu.tr

This paper describes a dynamic teaching model developed to explore new intellectual and theoretical directions of design generation. It utilizes shape grammars as a pedagogical tool for introducing the concept of design computing to graduate students of architecture without a background in computation theory. The teaching model has developed in two parallel run courses: 1-Introduction to Computational Logic in Design: a visual generative design tool shape grammar is used as a tool to teach design students computational logic, 2-Designing the Design: The design students are introduced with the new language and method of designing. Examples from the Computational Design graduate curriculum at Yildiz Technical University are given to illustrate explorations in this new teaching of design..

Keywords: Computational design; shape grammars; design education.

Introduction

Digital technologies today are currently challenging both architects and educators to formulate a new understanding of the design realm. In the last decade, important changes have been made to architecture curricula worldwide to accommodate new demands, opportunities, processes and potentials provided by digital media as discussed in Kvan et al. (2004). As a result, a number of additional subjects have been introduced in architectural curricula (Knight, 1999), (Terzidis, 2006) Also, in practice, concepts of digital media, mathematical and computer algorithms have played a key role in design conception and development as described in Balmond (2002) and in Kolarevic (2003).

This paper is motivated by the challenge of new emerging educational paradigms of digital media and explores the possible new intellectual and theoretical directions in design education.

Computational Design – Shape Grammars

In the realm of architecture, computational design (CD) has emerged as a sub-discipline, which is multidisciplinary in nature, and uses advanced computing capabilities to understand and solve complex problems in architectural design. It provides methods for the architect/designer in harnessing more deliberate and conscious thought processes in design.

Computation involves algorithmic logic that is deterministic rational, consistent and systematized. Most algorithms are symbolic and are used to automate manual methods by means of formal languages. Designers use visual algorithms in their
design process. Therefore, in architecture and design it would be more natural to compute with shapes directly rather than symbols. This idea prompted the creation of shape grammars (Gibs & Stiny, 1980) and they have been since applied in architecture to produce languages of form often to re-create the style of particular architect (Duarte, 2002) or a new style of particular architectural language through transformation of an exiting language. (Colakoglu, 2005)

Shape Grammars form a bases for visual computation. Their mechanism is formal however; the primitives in shape grammars are shapes rather than symbols. Their relations and operations are spatial rather symbolic. Shape grammar itself is a concept expressed as an algorithm which describes the computational process of design. It is an example of a production system defined as a string, consisting of symbols of specified vocabulary. It was expanded into generative grammars by Chomsky, which are now integral to computer languages (Gibs & Stiny, 1980). Shape grammar uses the mechanism of production system however; it differs from it by use of objects. The objects in production systems are strings where in shape grammars they are shapes. Also, shape grammar computations differ from general understanding of computational design in which computation is considered as counting where in shape grammars computation is seeing. Computation in shape grammars works in terms of embedding. It presents things in terms of maximal elements and allows shape to be presented in infinitely different ways. In shape grammars, designs are not made of fixed elements, they are made of shapes and sub-shapes. They do not parse shapes in fixed sub-shapes. Spatial ambiguities are allowed. Designs can be recombined and decomposed in different ways. The power of shape computation is that according where one is in computation ones reading of the shape differs (Stiny, 1999).

Shape grammars have been used in education as pedagogical tool for teaching computational design for more then a decade (Knight, 1999; Fleming, 1987). Considerable number of design (architecture, engineering, art) schools has included shape computation and grammars as part of their curriculum. The experiences of teaching have shown that to understand and deploy full capabilities of shape grammars, a unique combination of technical, spatial and intellectual abilities and interests are required (Knight, 1999). This raises the question for teaching strategies of shape grammar computation.

Computation in a mathematical sense is the root of shape grammar theory and many people misleadingly associate shape grammars with “computers”. As it is implied by Knight (1999) it is important to distinguish computational aspects of shape grammars and computer programming aspects. “Shape grammars are spatial algorithms that may be developed and implemented on paper independent from computers. ….The kind of computation underlying shape grammars derives not from CAD, but from traditional design practice and the unique perceptual, spatial and intellectual abilities of human designers”

**Motivation**

The generative and creative potential of digital media is opening up new dimensions in the realm of architecture. It challenges not only what we design but also how we design. Recent theories of form in architecture focused on computational explorations and expressions challenged how we teach design. Barts Lootsma in “Hybrid Space” (Zelner, 1999) speaks of new direction in architecture, “instead of trying to validate conventional architectural thinking in a different realm, our strategy today should be to infiltrate architecture with other media and disciplines to produce a new cross bread.”

Following these concerns, new interdisciplinary curriculum for the computational design graduate program is developed reconsidering this relationship between design thinking, design computation and digital design in the light of current computational design theory and technology. The new curriculum of the graduate program is restructured towards the new theoretical framework in which digital media
is considered as environment for the exploration of new intellectual and theoretical directions in design. It aims to integrate computational design methods into the design culture as a whole. The program consists of three integrated sub areas: 1- Advance digital media, 2- Design computation, and 3- Design theory and computation. This paper describes teaching model developed under the design computation area.

Teaching Design Students to Think and Produce Computationally

In response to integrate the new formal possibilities into education of architecture, two courses; 1. Introduction to Computational Design, and 2. Designing the Design are developed.

Introduction to Computational Design

Is an introductory course for teaching computational design theory and methods to students of architecture without a background in computational theory. A unique visual kind of design computing system—shape grammar—is used as a pedagogical tool for teaching.

The course consists of four modules; Design Thinking, Algorithmic Thinking, Computational Design Model, Shape Grammars, and Shape Grammar Applications.

The first module attempts to make the students aware of different ways of thinking—analytical, systematic, and intuitive—applied in design process by asking them to externalize the design process of one of their school design project. They are not given any instructions but rather let free to create their own model of externalization of the knowledge. Then asked to describe types visual vs. symbol of thinking looking critically where and how these types are used in the design process and whether they can be described with formal languages.

The second module aims to describe that algorithmic thinking is a part of design thinking, by drawing examples from simple algorithmic inferences to complex ones in design process.

In the third module, shape grammars are introduced as a design tool that provide mechanism for computational representation of design. Here, shape grammars are used to illustrate that visual computing is actually a way of thought that is a cognitive process applied by the designers.

In the fourth module, the students are introduced with applications in education, possible practice and computer of shape grammars. The modules are supported with related readings. After each module the students are given homework that synthesize and apply what has been thought in that module. For the final work the students are left free to propose a project within the scope of “computational design” thought in parallel run two courses.

Designing the Design

Is a workshop aimed to introduce design students with the new algorithmic language of design. The course is developed in three modules; 1. Paradigm shift in design, 2. Algorithmic language of design, 3. Designer as casual programmer.

The first module discusses new computational methods of formal exploration and expression of design. It looks into design practices that use new computational techniques and formal systems in design process.

The second module introduces students with algorithmic design; a conceptual framework that explores forms, structures and processes of architectural design by the use of mathematical or logical methods. As pointed out by Terzidis (2003) the rule-based logic is inherent in architectural programs, typologies, building code and language itself however, has not been externalized, in other words not made operational. Grammars are used as a tool to externalize rule-based logic in architecture.

The third module teaches students how to use algorithmic logic and techniques in designing. It involves the codification of design intention through algorithmic scripts built on top of existing CAD sys-
tems. The workshop aims to make a designer literate in reading and writing in digital media.

**Algorithmic Design Explorations**

The following paragraphs present a series of educational design studies generated through the use of algorithmic processes thought in above described courses.

The study illustrated in Figure 1, explores algorithmic form generation. Shape grammar is utilized in manual form generation. Then, the grammar is codified in 3DStudioMax script language. Through random generation of rules and deformations done by adding simple lines into script, variations of 3D form compositions are created. The study compares manual versus computer generation of forms.

The study illustrated in Figure 2 and 3, explores algorithmic design generation in architecture practice by using hybrid applications of shape grammars. Analytical / synthetic grammars teach students about the particular style of design, their own work in progress, about ways of designing and about ways of developing their own work. In other words they teach multiple skills in a coordinated way.

Hybrid grammars are used for housing project in Mardin, a city located in south-east Turkey. First, types and principles of Mardin housing are extracted using informal shape grammars. Then these types
are interpreted in the context of contemporary housing. The principles of composing these types into housing settlements on sloping terrain are created. The study makes operational the rule-based logic inherited in architectural language.

The study in Figure 4, applies algorithmic thinking for shell structure design explorations for YTU technopark office complex. The composition of office buildings is generated by using CON-GEN (Colakoglu et al, 2005) a plug-in developed under 3D Studio Max. Shell structure design intentions are codified through algorithmic scripts in the existing modeling system. By changing parameters of the structure design, various structure compositions are explored.

**Concluding remarks**

There is a growing discussion that computational design teaching not be segregated to separate classes but be reflected holistically through the curriculum. Kavin at al. (2004) discusses Bauhaus strategy of integrating the technique (craft) and the content (art) or in today terms design and tools. He points out that rather than dividing the classes as digital skill teaching and as architecture, new strategies should be developed to demonstrate the impact of computation techniques in understanding the new ways of designing. The two courses presented in this study apply new strategies for computational design teaching. They serve as platforms where design methods integrated with the concept of computation and digital tools.

This paper demonstrates the value of grammars for understanding and teaching computational logic applied in design. It contends that a strategic use of grammars as a pedagogical tool may assist acquisition of both deep and surface knowledge about form-making and encourage the students to think about design as computation.

It suggest that grammars need to be considered as a continuously evolving resource with which design teaching can be supplemented especially in regard to new digital media.

**References**


Stiny, G.: 1999, personal conversation