Option Explicit – Scripting as Design Media

Tim Schork
Spatial Information Architecture Laboratory
School of Architecture and Design
Royal Melbourne Institute of Technology University
GPO Box 2476V, Melbourne, VIC 3001, Australia
tim.schork@rmit.edu.a

Abstract

In practice, the domains of architecture and computation have traditionally been perceived as distinct. Computation and its associated technologies, such as computers and software applications, have primarily only been applied to the domain of architecture.
The aim of this paper is to reconsider the relationship between these domains. In moving away from separate entities towards a synthesis of architecture and computation, this paper explores the potential and the challenges of this rich creative space that opens up for architectural design through a series of case studies.

1. Introduction

"*Architects tend to draw what they can build, and build what they can draw*"[1]. In this statement William J. Mitchell describes to a great extend the predominant use of computer technology and software applications in architecture. They are still to a great extent only used as 2 dimensional drawing tools that mimic conventional modes of production in order to maximize efficiency. While this uptake of computer technology has caused a skill change in the profession, the attitude towards it as an applied tool has to a great extend remained the same, keeping the domain of computation hiding within the black box of the 'tool'. Architecture stays passive in the use of technology and merely consumes whatever tools are provided.

In contrast to this, I suggest an alternative attitude, one that questions the status quo use of computer technology and explores its capacity for extension and customization to ones own needs. In this scenario computer technology can bring with it not just a skill change, but more importantly an attitude change in the use and role of computers. Here the designer takes ownership of the generic tool by opening up the black box and inserting his domain knowledge into it, making it a specific tool that caters to his design intentions. This approach considers concepts and theories that are integral to the domain of computation and have useful applications in the domain of architecture.

An indicator of this synthesis is the currently increasing engagement of architects with scripting, a subset of computer programming, in order to design their own set of tools for design, analysis, visualisation and documentation. This use of scripting as a design media marks a different way of thinking and working architecturally, while drawing on a thirty year history for instance to the work of others such as John Frazer.[2] Thus the computer can become more than a 'drawing tool', and instead becomes a 'design tool'.

This synthesis needs also to be considered in conjunction with a series of wider cultural shifts. Outside the discipline of architecture, technology affects nearly every aspect of our social and cultural lives. We are generally comfortable with technology and regard it as an integral part of our everyday life and work. We are accustomed to the idea that technology not just helps and serves us, but that we can modify it to meet our needs. Examples of this are the programming of DVD recorders and mass cultural phenomena such as Google™, Blogs, Wikis or MySpace™ websites. As Mitchell states, "*computers have become consumer items; and computing is now a mass medium*"[3].

Within the discipline, architecture's re-incorporation of aspects from theories of mathematics and science has expanded the disciplinary boundaries. This knowledge has led to a shift of architectural design from form to process, extending the existing architectural repertoire. This is characterised by the transformation from 3D representational models of architectural form to more performative models of processes. The difference between the two can be illustrated by a simple example. In a representational model the geometry is something explicit that is based on implicit rules. In procedural models this relationship is inverted, meaning that implicit geometry is based on a network of interconnected entities which are described by a system of explicit design rules. These design rules define the relationships between individual parts and enable a constant negotiation of the geometry with its context.

2. Case Study – Screenresolution

Sceenresolution is a case study that investigated how scripting can be a useful design media exploring the creative design space that exists at the overlap of architecture and computation. The project was undertaken by MESNE[4] in collaboration with seven interior design students at the School of Architecture and Design at RMIT University. The intention of this seminar was to produce a 1:1 prototype of an architectural screen made from non standard components that was collectively designed by the entire group of students. By developing a series of strategies to converge the abstract model (the code) and the physical model (the materialisation) the project further explored the synthesis of generative design processes and fabrication techniques. Throughout the entire semester the students were required to post their scripts to an open source script library, so that they can be shared among all the design members. The final 'master script' used to generate the final screen is compiled from this library, giving the project not just a single author

2.1. Decomposition and recomposition

At the beginning of the seminar students were introduced to rule-based design systems and were asked to investigate natural occurring surface tessellations for their underlying 'generative code'. The task was not merely to produce '*bonsai architectures*'[5], small scale representational models of something already existing, but aimed to discover the geometric rules and the internal logic of their structuring processes. This required the mental task of decomposition and recomposition, which is something inherent to both design and scripting. As with good design, scripting generally improves through a process of drafting and redrafting and requires clear thinking. When working with the technique of scripting one first decomposes the design intent into a series of rules and parameters, then transcribes them into a series of functions and recomposes them by bringing these entities back into relationship with one another when running the script. This approach reflects Burry's description about generative design. He notes "*With a generative approach to creative*

production it is possible for there to be a clear distinction between what is generated and what generates, between the code and the resultant objects."[6]
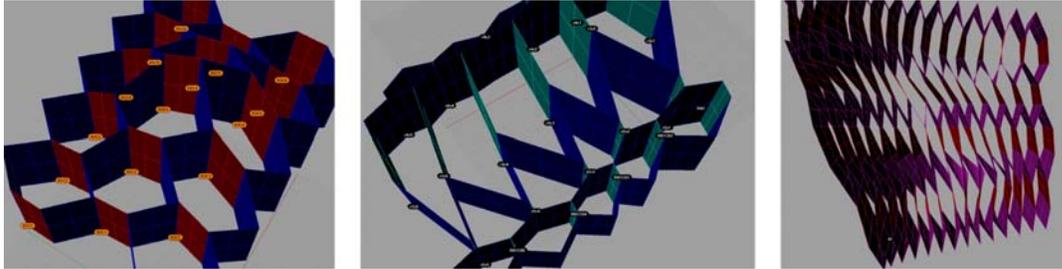

Figure 1. Geometric studies generated by scripting

After creating a design schema and establishing a common and generic component based assemblage, each student investigated a particular area of individual design interest, such as colour, porosity and light transmission. All these design intentions were encoded as a set of instructions into each individual component in order to define how the component can adapt to its environment. The environment setup for this generative system was a manually created simple 3D model in Rhino™ that consisted of two variable NURBS surfaces. These surfaces defined the position of the screen in the exhibition space, between which the components were instantiated.

The screen also needed to fulfill a several programmatic requirements, which were located by a series of points in the 3D digital model. The proximity of a component to the field of influence of each locator was used as an input for the generative system that would trigger a response in the component geometry. For example if a screen component was within the range of influence of a locator for multimedia projections, a closed component type was generated, while the proximity to a locator that defined the area for a proposed lounge caused a colouration of the component. The final configuration of each component is an accumulative response that synthesis the influence of all the different locators within the exhibition space.
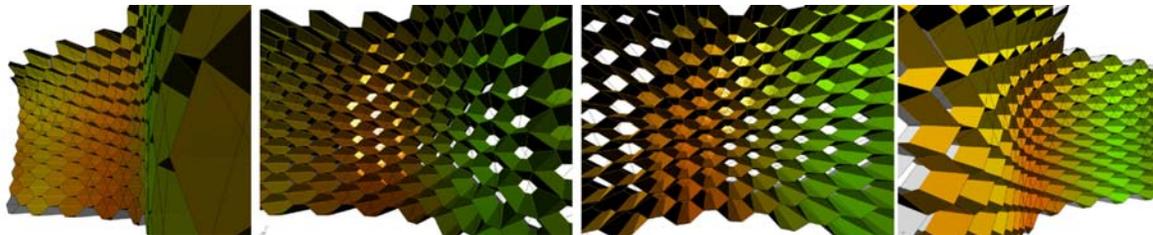

Figure 2. Colouration of components based on proximity to lounge locators

2.2. Component based authorship

The technique used in this project is a combination of RhinoScript™ and Visual Basic for Applications™ (VBA) in order to create a design tool that is able to generate a series of different design versions. This design tool is then used to iteratively explore possible configurations of the proposed screen within the exhibition space.

During the early design phase the group was given two requirements that needed to be considered and implemented into the setup of the generative code. The first requirement was that the screen needed to be produced out a flat sheet material, which imposed a number of constraints on the geometry of each individual component. The second requirement was that each component had to be fabricated with an available numerically controlled (NC) flatbed cutter. This meant that no component within the entire assemblage could be bigger than the maximum bed size of the cutter. This incorporation of material and fabrication constraints into the generative script and its generated geometric outcomes facilitates an interface between the design concepts and the material practice of architecture and makes it possible to use the information contained within these models directly for fabrication.
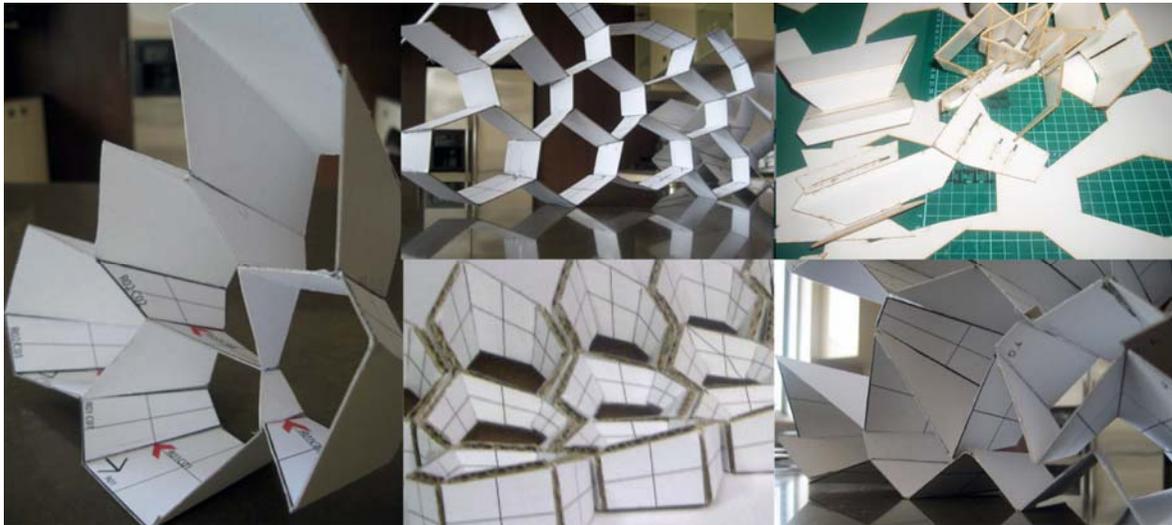

Figure 3. Series of small scale prototypes

2.2 Converging modes of representation

With this knowledge of material and fabrication constraints in mind, the development of the generative code of the screen was constantly informed by the simultaneous making of physical prototypes that for example either tested the connection details between individual components in the assemblage, the properties of the used cardboard material or the fabrication constraints of the flatbed cutter. These physical prototypes, incorporating as well the design intentions, as well as the fabrication constraints, created a feedback loop between all the different modes of representations of the design. It is through this constant moving back and forth between the different modes of representations of a design, the scripted text based, the geometrical visual and the material physical, that enriches our understanding and the quality of the designed object.

The final screen is a unique object and consists of an array of non-standard components. As a field, these components act collectively to express properties of porosity, colour, and the interplay of light and shadow. This collection of properties generates a moment in a continuous state of change.
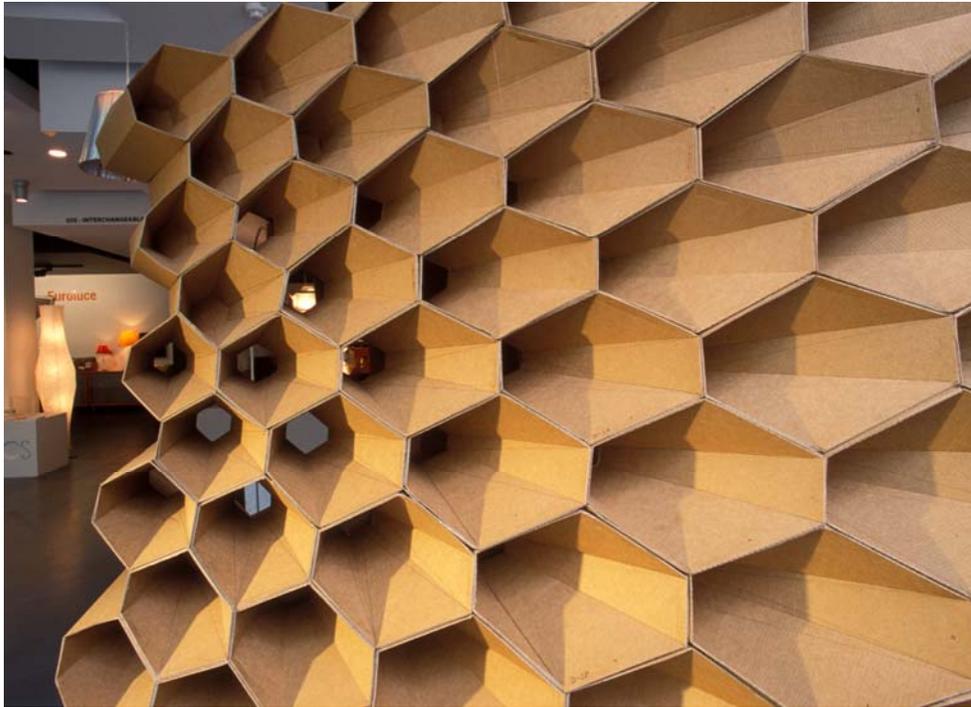
Figure 4. View of 1:1 prototype showing variation of individual components

Conclusion

This paper has presented how a group of students collectively designed a screen made out of non standard components and why the use of scripting as a design media marks a different way of thinking and working architecturally. On reflection, we can draw several conclusions from this specific case study.

Firstly, the overlap created through the synthesis of aspects of the domains of architecture and computation supports design thinking. Once the scripted design tool was compiled, the group of students explored the design space that opened up through this overlap and continuously generated a vast number of possible designs. This allows to "*ask 'What if?' more often, and about more kinds of assumptions*"[7] creating the opportunity to investigate more design alternatives.

Secondly, it illustrates how scripting can foster collaboration and act as lingua franca among the members of a design team, therefore becoming a catalyst for innovation within design. The project challenges the proverb that '*too many cooks spoil the broth'*.

Lastly, the paper shows how scripting creates an interface between design and fabrication processes. This interface this bridges the representational gap between the abstract (virtual) and the concrete (materialization) leading to more buildable design outcomes.

Can scripting overcome Mitchell's statement about the production of architecture from the beginning of this paper?

[1] see Mitchell, W. J., "Roll over Euclid: How Frank Gehry Designs and Builds", in *Frank Gehry Architect*, J. Fiona Ragheb (ed.), New York: NY: Solomon R. Guggenheim Foundation, 2001, pp.352-63.

[2] see Frazer, J., *An Evolutionary Architecture*, London: Architectural Association, 1995.

[3] see Mitchell, W. J., Liggett, R. S., Kvan, T., *The Art of Computer Graphics Programming*, New York: Van Nostrand Reinhold Company, Inc., 1987.

[4] MESNE is an architecture practice, founded in 2005 by the author and Paul Nicholas, www.mesne.net

[5] Herzog & de Meuron: Natural History, Ursprung, P., Basel, Lars Müller Publishers, 2006

[6] Burry, M., Burry, J., Dunlop, G., Maher, A., "Paramorphs: An Alternative Generative System", in Proceedings of Second International Conference on Generative Systems in the Electronic Arts, CEMA, Melbourne, 2001.

[7] See McCullough, M., 20 Years of Scripted Space, in *Architectural Design 76*, no.4, 2006, pp.12-15