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*, 601–610. © 2016, The Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), Hong Kong.

ARCHITECTS AND DIGITAL DESIGNING TECHNIQUES FRONTIERS

WAJDY QATTAN AND STEPHEN HARFIELD
University of Technology Sydney, Sydney, Australia
wajdy.s.qattan@student.uts.edu.au, Steve.Harfield@uts.edu.au

Abstract. Recently digital-design techniques have influenced the way architects think and design. This extends to impact architectural education by drawing new boundary lines. Therefore, it is desirable for architectural educators and students to consider these lines and to know how to establish them within these technological trends. This will be through raising their knowledge and skills in three aspects, which are algorithms and geometry characteristics, authorship, and fabrication in digital architecture.

Keywords. Algorithms; authorship; fabrication; education; digital.

1. Introduction

With the emergence of the developing digital-design techniques, architectural educators and students are facing some difficulties and complexity. These difficulties are manifested in algorithms, authorship, and fabrication in digital architecture design. While utilizing the significant role of computers and fabrication machines, architectural designers need to manage using algorithms and, at the same time, remain the central operators and thinkers in the design processes. This paper argues that the frontiers between architects and digital-design techniques come in four aspects. First, architectural educators and students need to be knowledgeable, qualified and skilled to access the computational capacity to get the most out of it. Second, they are still the decision makers who are developing the design process. Third, they are the authors of the design, and are thus controlling the design most of the time or, potentially, co-authors at some stages. Fourth, they need to be aware and updated in terms of the new and prevalent fabrication techniques and machines.

In terms of using algorithms in architectural design, the paper argues that architectural educators and students need to appreciate the new relationship between design, computation, and algorithms, and also to use them within particular aspects of the digital. Using algorithms in architectural design means formalizing the design process as procedures and instructions.

To use algorithms, architectural educators and students need to understand, write and design them. Designers therefore need to control and utilize the algorithms to undertake the design process. Designing the design is a procedural issue controlled by the individual designer. Hence, there are steps to (re)define the design process as integrated design systems. It thus poses design itself as a significant problem where architects engage with broad cultural and technological discussions between scripting as both an ‘open’ and a ‘closed’ system.

Furthermore, the paper also argues that despite the fundamental shift in design processes, architects remain the author(s) of the design and/or co-authors some times. They are playing crucial roles in almost all the design processes, even if there is a need some times for other disciplines (e.g. programming, mechanical, environmental, electrical, etc.). They are also in control in all design stages from setting up the initial ideas, writing the code, judging, evaluating and decision making, and fabricating materials and techniques. However, they have less or no control in the stage where computers run the code.

Finally, the paper suggests that gaining knowledge in relation to fabrication and its materials and machines is one of these frontiers that architectural educators and students should consider. It is important to know the current digital fabrication techniques and machines to be able to transform a digital model to reality, right from the initial design idea. This imposes a shift in the design process to consider fabrication materials, structure, and generated forms.

2. Algorithms and geometric characteristics

In terms of algorithms and geometric characteristics, architectural educators and students now need to understand the relationship between architectural design, computation and algorithms, and to be able to use them to serve their design goals and objectives. An algorithm, according to Burry and Burry:

> “is a very specific set of instructions for carrying out a procedure that generally includes an instruction to stop. It may be long or very short”. (2010, p. 252)

In computing, algorithms are procedures to transform inputs into outputs. In addition, in architecture algorithms are used to formalize the design process as procedures and instructions to produce geometries. Thus, the importance of algorithms manifests itself in its power to create functions and instructions; it is also a very crucial part of the geometric traits. When architects want to create a computational geometry it is better for them to know its algorithms, even if it is ready to use as a function on the software interface. As a result, sufficient knowledge is desirable in fields such as catenary models, cellular automation, curvature, developable surfaces, dynamic relaxation, elliptical geometry, emergence, fluid dynamic, fractals, hyperbolic geometry, immersion, inversion, Lindenmayer systems, minimal surfaces, non-Euclidean geometry, non-linearity, Non-Uniform Rational B-splines NURBS, recursion, system dynamics, topological transformation, topology models, and a voronoi diagram. To understand the algorithms and geometry architectural educators and students need to look at their mathematical and logical composition before using them.

Burry & Burry (2010, p. 7) argue that the mathematical processing of the algorithms is usually overlooked or deliberately concealed within the commercial design software. For example, using this software is actually a series of mouse clicks and keyboard strokes; but behind that there are very fast complex mathematical operations undertaken. Using algorithms and computation in architecture allows the producing of more options (forms) resulting from a range of constraints. Within this process the algorithms are driven to meet the designer intentions to eventually find the final form. At the same time, it follows the geometric constraints that require strong understandings of the relationship between design, computation and algorithms. This relation may be shown in the examples presented in Burry & Burry's book, such as mathematical surfaces and seriality, chaos, complexity, emergence, packing, tiling, and topology. This is to show the ways to understand and use these mathematical and geometrical concepts through computation. Consequently, this will help finding new and unique solutions.

Architectural educators and students are required to know that using these techniques are more than simply a set of formal software ready to use commands/icons. They need also to know that there are a new range of emerging terms of algorithms and geometry that are in use now within the digital architecture circles. Moreover, they need to know that the design processes and techniques are shifted towards generative, self-organization, and optimization, which are different from the traditional concepts and techniques (but, this does not mean omitting the traditional techniques). This shift is defined when the architects add the possibilities of scripting (generative techniques such as multi-agent systems or genetic algorithms). This imposes a significant shift from the traditional top-down forms of control to more fluid possibilities of a bottom-up approach. However, this shift is usually permeated with hardship, mistakes, errors, and frustration especially at the beginning.

As we all know, using algorithms and geometric properties in scripting is a new knowledge that architectural educators and students need to obtain. To break it down, they need to be skilled in using algorithms, geometric calculations, and scripting languages and logic. According to S. Nassir (2015, per comms., 3rd Sep) learning programming skills requires different knowledge compared to what architects are doing. This will move architects from an architectural design environment to a programming environment, and this will be very hard. It is like doing two disciplines at the same time. Thus, architects need to learn how to master using the commercial software plus using their programming languages. For example, they need to learn how to use Rhino and Python or Maya and Maya Script.

One of the challenges of using algorithms and geometries is in combining designing and scripting abilities. To overcome that, architects need to look at, understand and learn from some famous buildings which are designed using these digital techniques. In other words, they need to look at and learn from famous precedence. Moreover, the other challenge may be the algorithmic activities, based on parameters and rules that allow design strategies which are different from the conventional design strategies (Lee, Gu & Williams 2014).

3. Are architects the authors or the co-authors

What do we need to know to answer this question? Architectural educators and architecture students should know that using digital-design techniques will permit them to be either the authors or the co-authors of the design. In the 1970s the role of computers changed towards creating an intelligent assistance system. But, today, computers are significantly involved in the design process, from drafting and modelling to intelligent systems and processing architectural information. Now it is important to ask: who designs? (Terzidis & Vakalo 1992, p. 5). It is important to know that digital-design techniques are no longer threatening the architects' authorial role. They are driven by the designer decisions, and their outcomes are also subjected to the designer's interpretation and understanding. This contradicts what Burry is arguing for. He suggests (2014, p. 388) that scripting is threatening design authorship richness, e.g. dilution of ideas, too much external agency, or just being straitjacketed through code.

Architectural educators and students need to know the difference between the design process when either using or not using computers to generate design. They need to know the difference between the explicit and implicit design process. In the explicit design the designer inputs affect directly the resulting output, whereas in the implicit design the designer decisions are embedded in the logic of the script, which form an algorithmic layer between the designer input and the resulting output (Marble 2012, p. 44). This layer is where the designer has less control over the running script and as a result he/she becomes a co-author.

According to Oxman (2006, p. 242), architects now interact with generative, performative processes and mechanisms, using information as a new material. The designer becomes a tool builder, which means designers need to improve their computational skills to be able to deal with this new architectural trend. Moreover, Oxman (2006, pp. 242-243) puts forward four components of digital design, namely: representation, generation, evaluation, and performance. Generation includes the process where algorithms are involved to create geometry. This imposes a shift – and restrictions – in the authorial role of the designer. In other words, the historical architecture designer's role has been shifted; he is no longer the sole author of the design. According to Marble (2012, p. 8), the author or creator role of architects is replaced with semi-autonomous, algorithmically-driven design workflows within a collective digital communication infrastructure.

Controlling the design process is desirable for architects when using digital-design techniques. Computers are not always aware of their environment, therefore architects believe that the mental processes of design are conceived, envisioned, and processed in the human mind. Thus computers are just calculation, organization, productivity, and presentation tools (Terzidis 2006, pp. 20-21). Because of that the algorithmic outcome must be referred to human minds, i.e. the programmer or the designer. Therefore, critiquing the algorithms' outcome is always directed to the designer who uses the algorithms. In other words, the designer is responsible and will be blamed for the final outcome, not the algorithms that help him/her to produce it (Terzidis 2006, p. 23).

Traditionally, the designer maintains full control and intellectual property over their design; therefore he/she owns the idea. With algorithms and computation the outcome is not always under the control of their author. This suggests: (a) that the author of the script may not necessarily be the only author of the final outcome, and may not determine all the final characteristics of it (Carpo 2011, p. 5); (b) that if a discrepancy (between the original idea and the actual result) has happened, therefore the designer is not the author of the generated design (Terzidis 2006, p. 20); and (c) that the algorithms and computation imposes randomness, probability, unpredictability, and emergence, which are external and foreign to the human mind (Terzidis 2006, pp. 27, 55). However, all of these factors may be pre-considered by the designer and want them to happen while computers are running the given script. This happens if the designer has used computer to perform tasks that cannot be done manually (in a conventional way). At this time (when the computer is processing the given script) he/she will be at the co-author position until the computer finish processing. Then he/she will be again at the author position.

To be the author or the co-author of the design, architects need to be connected to the predictability of these techniques' outcomes. Thus, the absence of the designer control shifts the authorship of the algorithmic process to computers (Terzidis 2006, p. 20). Here the authorship will be shared between the author and the computer. To maintain this authorship, architects need to develop an input logic and behaviour of generative process through scripting and algorithms, and evaluating and dispensing among the resulting options (Marble 2012, p. 203). They need also to be the author of the design rules. This includes establishing the logic, guidance, and setting up of the design problem, and being the manager of design data. This includes setting up clear and well-designed organizational strategies (Marble 2012, p. 249).

According to Burry (2014, p. 405) the majority views among scholars show designers as the driver of the design process, regardless of the tools in their hands. That could mean that algorithms should be written, designed and controlled by an architect. This is what (Terzidis 2006, p. 20) claiming, architect programmer must compose the algorithms which then generate forms and he/she must be in control of the outcome and the idea. Finally, the designer who wants to be in control of the result should be in control of the process. To be in control of the process the designer must be in control of the tools. The tools are computational, therefore a designer who wants to be in control must also be a scripter (Burry 2014, p. 405). This will be one of these frontiers that architectural educators and students need to consider.

4. Digital fabrication and its materials and machines

As one of these frontiers, fabrication plays an important role in digital design with deep connection to industries, technologies and materials. While fabrication materials constitute the idea of 'building', fabrication machines are devices that can automatically transfer digital objects from the design world into material realization (Mitchell 2004, p. 78). With fabrication technologies, architects shift design processes from form-making to form development, together with material and structure. According to Oxman and Oxman (2014, p. 302) with fabrication, the design sequence has changed from form-structure-material to material-structure-form. However, what are the most popular fabrication technologies?

Hensel, Menges and Architectural Association (2006, pp. 37-38) declare that in the 1950s the US military introduced numerical control (NC) as a machine of metalwork to break the limitation of mass production. In the following decades, the computer numerical control (CNC) was introduced to produce a wider range of material and scales. This happened with the increase usage of CAD applications. With that, Hensel, Menges and Architectural Association (2006, p. 38) uphold that once the potential of CNC is understood as a key aspect, the integration of materialization and form-generation becomes clear and essential. This suggests merging constructing constraints with the use of materials and the assembly logics to allow more exploration.

A three dimensional digital model of building can easily be cut by the CNC machine. This machine requires less control and it can create millions of copies of identical and non-identical elements. The functionalities of CNC machines can be grouped into three categories: cutting, subtractive, and additive (Kolarevic 2004, pp. 34-37; Krauel, Noden & George 2010, pp. 12-13). Usually, cutting happens with two-dimensional elements. A sheet of almost any material can be cut, and the common cutting technologies are laser, water-jet and plasma. Subtraction pertains to the removal of layers of the volume of the material to create a form. The removal could be mechanical or chemical, but it also depends on the freedom of the milling tool. For example, four or five axis tools make forms that are more complex. Additive, or as it is known by different names like 3D printing, is a way of adding modelling material layer-by-layer, and the only limitation of this technology was scale as it cannot create big objects. But now 3D printing is used to create large objects such as cars and houses.

Most of the recent studies and buildings have focused on the transformation and/or realization of digital models to physical prototypes. Add to that the consideration of function and materiality in relation to manufacturing and production. As a result, some fabrication and designing techniques appeared as a response to these technologies. These, for example, include – but are not limited to – the following: cross segmentation, accumulation, frameworks, loops, folding, twisting, lofting, triangulation, drilling, and knotting (Agkathidis 2012; Agkathidis et al. 2010). (Figure 1).

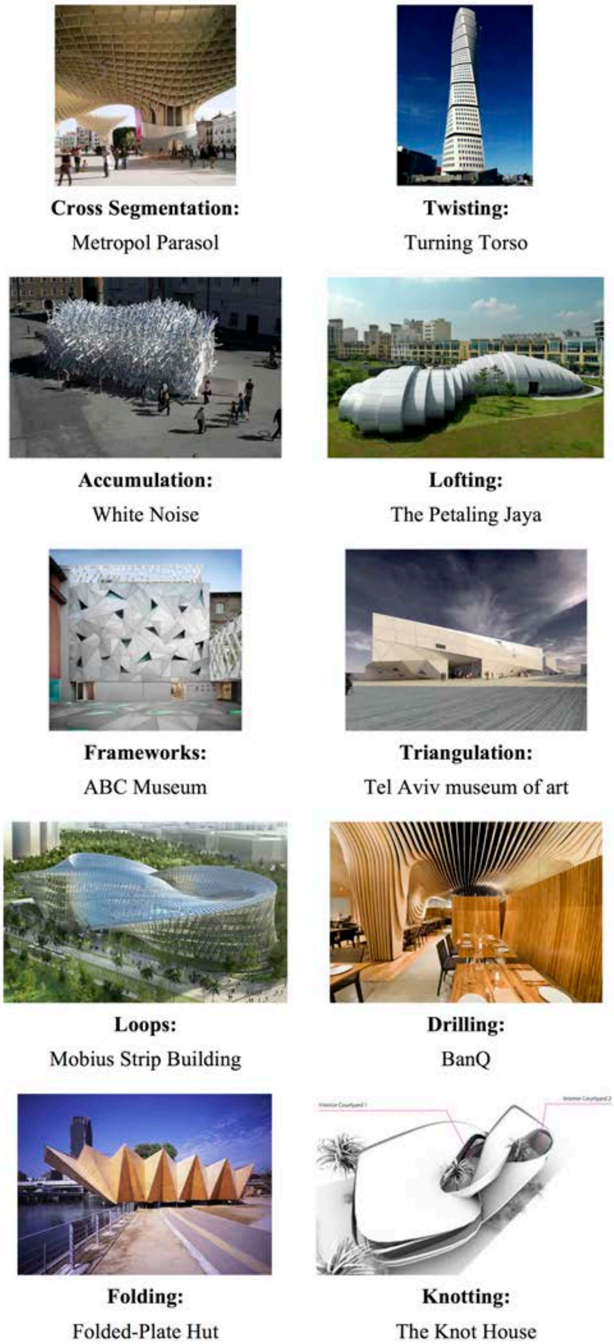

Figure 1. Architectural forms and techniques examples.

These techniques and others give an indication of their widespread use, for they now became available and affordable; they are no longer expensive, and are more efficient than traditional techniques. In fact, each time a new fabrication technology is invented and becomes available, a new architecture style/approach will appear and make its impact. Thus, it is hard to deny that the new designing and production technologies are making irreversible impact on the development of architecture practice today, and will continue to do so (Agkathidis 2012, p. 6). But what is interesting is how architects look at them? Architects, and especially students of architecture, are often fascinated and curious about what technology can do rather than architecture itself (Bettum & Frankfurt 2010, p. 122).

5. Conclusion

To crossover these three frontiers, architectural educators and students need to be aware of them. In terms of algorithms and geometry, the need to know first the algorithms' importance, the way they work, and their relation to architectural design and computation. Second, they need to be aware of and understand the new algorithmic terminology. Third, they need to know that algorithms impose a shift that moves architectural design toward generation, self-organization and optimization. Fourth, they need to know that using algorithms requires extra knowledge and skills such as programming languages. This suggests combining two skills or more, i.e. design, scripting and algorithmic activities.

In terms of the authorship rights, they need to know that computers are devices made by humans, but computers do not need humans to complete a given task. Usually tasks (script) are designed and written by humans, but then they pass them to computers to run them. After that computers produce an outcome that humans can judge, evaluate or re-process. Once the computer processing starts, the human role is dispensed. They cannot intervene until the computer finishes processing and here the architects will be in the co-author position. As a result, architects are not always in full control of the design, but they will be responsible and blamed for the outcome.

Moreover, in terms of fabrication they need to think about how to convert a design idea to a real building or physical model. That requires architects to link design with material, structure and fabrication from the very start. Digital-design techniques suggest an evolution and transformation of the current architecture. Therefore, architectural educators and students need to think about how architecture will be, what role they will play, and what extra skills and knowledge they need to be included.

Endnotes

1. This research is aiming to highlight three important frontiers that architects, architectural educators and students need to put in to account when intending to use the digital-design techniques.
2. This research could be an introductory lesson to those who do not know anything about the role of algorithms, geometry, computation, scripting and fabrication in architectural design.
3. Addressing these three frontiers does not mean ignoring the other aspects of digital design, but, because they are the most obvious and affective ones, we need to start with architectural education.
4. This research is reflecting the author's perspective. Thinking about introducing digital-design techniques to architectural educators and students, focusing on algorithms, geometry, authorship rights, and digital fabrication.

References

Agkathidis, A.: 2012, *Computational architecture*, BIS Publishers, Amsterdam.

Agkathidis, A., Bettum, J., Hudert, M. & Kloft, H.: 2010, *Digital manufacturing in design and architecture*, BIS publishers, Amsterdam.

Bettum, J. & Frankfurt, S.: 2010, On the Importance of Numbers and Roses, *in* A. Agkathidis, J. Bettum, M. Hudert & H. Kloft (eds), *Digital manufacturing in design and architecture*, BIS publishers, Amsterdam, 122–125.

Burry, J. & Burry, M.: 2010, *The new mathematics of architecture*, Thames & Hudson, London.

Burry, M.: 2014, Cultural Defence, *in* R. Oxman & R. Oxman (eds), *Theories of the digital in architecture*, Routledge, 483–407.

Carpo, M.: 2011, *The alphabet and the algorithm*, MIT Press.

Hensel, M., Menges, A. & Architectural Association: 2006, *Morpho-ecologies*, Architectural Association.

Kolarevic, B.: 2004, Digital Production, *in* B. Kolarevic (ed.), *Architecture in the digital age: design and manufacturing*, Spon Press, 30–54.

Krauel, J., Noden, J. & George, W.: 2010, *Contemporary digital architecture: design & techniques*, Links.

Lee, J., Gu, N. & Williams, A.P.: 2014, Parametric design strategies for the generation of creative designs, *International Journal of Architectural Computing*, **12**(3), 263–282.

Marble, S.: 2012, *Digital workflows in architecture: designing design, designing assembly, designing industry*, Birkhäuser, Basel.

Mitchell, W.: 2004, Design Worlds and Fabrication Machines, *in* B. Kolarevic (ed.), *Architecture in the digital age: design and manufacturing*, Spon Press, 55–79.

Nassir, S.: 2015, Inverviewed by W. Qatten, 3rd September.

Oxman, R.: 2006, Theory and design in the first digital age, *Design studies*, **27**(3), 229–265.

Oxman, R. & Oxman, R.: 2014, *Theories of the digital in architecture*, Routledge.

Terzidis, K.: 2006, *Algorithmic architecture*, Routledge.

Terzidis, K. & Vakalo, E.: 1992, *The Role Of Computers In Architectural Design*, IAPS Proceedings.