Abstract

In recent years, digital design tools have become prevalent in the design community and their capabilities to manipulate geometry have grown into a trend among architects to generate complex forms. Working as computational design consultant in an engineering firm, between architecture and engineering we often come across the problems generated by a superficial use of digital tools in both disciplines and the incapacity of the current system to cope with their byproducts. Here we will discuss the problems we see with the current system and the opportunities opened by digital design tools. Two guiding concepts are simplicity [the desire to fine tune and build a system that yields a solution to a specific design problem by collapsing its inherent complexity] and defamiliarization [a side effect of having to represent things as numbers]. They can both affect the designer as an individual who chooses to engage with digital media as well as the production system in which he/she is embedded since he/she will have to find new channels of communication with other parties. To demonstrate our strategy and the obstacles faced we will examine our involvement in the development of a computational design solution for a small house designed by Future Systems architects.

1. Introduction

During the past two decades we have witnessed an ever increasing invasion of information technologies and digital media in the field of architecture production. Because of hard socioeconomic reasons digital design tools have become ubiquitous in the field.

As Jane Jacobs writes, “The conflict between the process of adding new work to old and the guilds’ categories of work was a constant source of wrangling in medieval European cities.” The same struggle exists today, and will certainly continue into the future.” 1

Hannah Arendt, The Human Condition: “if the human condition consists in man’s being a conditioned being for whom everything, given or man-made, immediately becomes a condition of his further existence, then man “adjusted” himself to an environment of machines the moment he designed them.” 2

Therefore, debating over the question “Why – digital technology should be used?” becomes insignificant, as it comes from a mere resistance of the current established structure. What should be debated is “How” architecture should re-adopt to the diffusion of the digital technology by adding to the knowledge accumulated till now. Today, most of the parties related to architecture production are utilizing various software tools in order to perform some of their tasks. Though software tools generally try to facilitate the production within each party, they do share a common underlying representation of information in digital form. The fact that a consistent representation of the
problems, intentions and solutions enclosed within each domain exists is significant since it sets the ground for a rearrangement of the production system to occur. As architects, focusing on computational design, standing between the two domains of architecture and engineering, we will present our observation and effort in trying to pierce the complex considerations posed by the new media.

2. Programming Simplexity

2.1. Complexity and the emergence of the cliché
One of the side effects of the rapid adoption of the digital design tools at least in the domain of architecture was a fascination with geometry and the possibilities of producing prolific forms. There lies the misconception that advanced tools give advanced results and the confusion between the logically possible on one side and the technically, technologically and economically possible on the other. This turn to the complex was justified as a way to create “novel” forms. Does the use of digital design tools guarantee the reduction in the use of clichés or formal biases in architecture? The answer is probably not, as attested by the endless series of organic and biomorphic things that constitute the bulk of production of many architecture schools. There is a trend in the architecture community to visually manipulate “complex” forms without intellectually engaging with the concept underneath digital computation which yields problematic conditions especially at the interfaces with other disciplines. One reason for this is that current software prioritizes visual information and is based on metaphors of previous processes like drafting and mechanical assembly. Another reason is the adoption of easy algorithms that produce “spectacular” results with minimum effort and intellectual investment. In this respect the concept of complexity has acquired an emblematic function becoming the master signifier for a whole school of computational design. These ill effects are partly the result of an attempt to use programming in order to construct a pseudo science of design and hence justify and present as rational inherently irrational choices. We will try to show that programming cannot be the basis for a design science since programming itself is a craft. Moreover there is a complementary concept to complexity borrowed by system theory which can help us to reframe the aims, possibilities, aesthetics and ethics of computational design.

2.2. Simplicity

Figure 3. Simplicity in a particle system.

Simplicity is the complementary term to complexity we are going to employ in order to show a different path that one can follow when using digital media in design. We saw how the use of simplistic rule systems with only stylistic visual considerations gives rise to the clichés of complexity. There is however, a whole class of algorithms that deal with simplification instead of proliferation. These are usually more complex and require stronger engagement with the concept lying underneath the computation. Simplification in a way that produces meaningful results and renders the complex system more accessible to human thought or more efficient is harder to achieve. This is because filtering, reduction, selection and abstraction are procedures that require intelligent and responsible choices as well as some way to refer and operate on the totality of a system. The emergent simplicity out of such intricate and complex set of rules is called simplexity. Many of these algorithms require from the designer a deeper understanding of the system. Simplicity is the aesthetics of a programming craft that results from thorough fine tuning of the design system and expresses the will to understand and make decisions rather than to proliferate and decorate.

2.3. Programming as a craft
The fact that programming is a craft becomes apparent if one considers that a good computer scientist is not necessarily a good programmer, and a good programmer is not necessarily a good designer. The first is about theoretical knowledge, the second is associated with experience and technical artisanship, and third includes the ability to make aesthetic judgment, ethical considerations and tectonics. Programming is never taught directly but is rather mastered through practice while theoretical principles of programming, mathematics and formal languages can be taught.

2.3. Defamiliarization

One aspect of the programming craft is the necessity to choose a numerical base representation of objects. This forces the designer/programmer to select properties or aspects of objects relevant to the problem considered and decide how to best represent them numerically and retroactively to reframe the problem he/she has to solve. This can have profound effects on how one perceives and operates on objects. Karatani in Architecture as metaphor notes that:

"The flaw in Euclid's work lies in his reliance on perception, or natural language, and in his inference of the straight line and point." 5
"From the moment Descartes defined points as coordinates of numbers, the point and line segment in geometry became an issue of numbers." 6

We see here that the use of a numerical representation for points was the decisive move that led to modern differential geometry and the break with intuition. We recognize an affinity between this side effect of choosing a numerical representation and the concept of defamiliarization.

Defamiliarization is a technique widely used in art that was described by Viktor Shklovsky in the beginning of the 20th century.

"Tolstoy makes the familiar seem strange by not naming the familiar object. "
"In describing something he avoids the accepted names of its parts and instead names corresponding parts of other objects. For example, in "Shame" Tolstoy "defamiliarizes" the idea of flogging in this way: "to strip people who have broken the law, to hurl them to the floor, and "to rap on their bottoms with switches," and, after a few lines, "to lash about on the naked buttocks."" 7

The aim is to achieve a specific effect of slowing down perception and halting the automatic, habitual interpretation of things. Here we are not considering defamiliarization as a technique but as a necessity implicit in having to choose a numerical representation. Being forced to think of an object as a set of quantities8 means that one has to momentarily abandon the habitual perception of a problem and the automated solutions one developed through training in an architectural schools. 9

Defamiliarization is not exclusive to architects, engineers have to defamiliarize with many well established concepts when dealing with finite element models for example and when they have to build some intelligence to these models. Probably similar processes can be described in each one of the involved disciplines.

Hence the pervasiveness of digital media is a contingency that at the moment affects the whole established chain of production of the built environment. Each discipline has a chance
to influence the system by the way it chooses to make use of the digital media by reconfiguring itself and subsequently the ways of communicating with others.

3. Simplicity in the system of Production

Hence, the idea of simplicity can be discussed in a larger context where the design community is embedded in, that is the architecture production system. It seems to us that the current system is not adequate in adapting to the current context shaped by recent technological developments, yet as individuals in the system, it is impossible to design or set the rules to the whole system. Considering the current socio-economic condition as Complexity, Simplicity of the architecture production system should emerge as the result of the partial pressures of the subjects to the system in order for the system to move to a direction rather than remain static but rigid or dissolve in incoherence. Therefore, Simplicity in this context is a responsible will and strategic decision making that is firmly located with the individuals. Yet, as a community, we see the importance in re-examining the architect’s role as well as the existing production system in order for the practice to engage with the current context.

3.1. Architecture as contingent planning

Karatani in his “Architecture as Metaphor” discusses how the term architecture has been conceived in philosophy as a realization of design qua idea. He however, in order to deconstruct this idea, refers to architecture as a practice being an event par excellence exposed to contingencies.

"Nothing is less relevant to the reality of architecture than the idea that it is the realization of design qua idea. Far more critical factors are involved, such as the collaboration with other staff members and the dialogue with and persuasion of the client. The design as initially conceived, is invariably destined to be transformed during the course of its execution." 10

The current architecture production system inherits its structure form the time of industrial revolution where the division of lobar in manufacturing was developed in order to adapt to the technological development at that time. The system is often conceptualized as tree structure where complex entities composing the production are divided into units [architect, engineer, fabricator, etc] and linear construction phases are used to measure the progress of the projects [Design Development, Tender documentation, etc.] Partly due to the effect of the relative isolation of architecture design process in the highly refined production chain, it seems that today many architects themselves have forgotten about the contingent character of architectural production, and instead embraced the misconception that Karatani attempts to dispel. We see architecture practice as a contingency planning aiming to realize the design not as an idea that becomes tainted by the realization process but as a problematic field that becomes enriched by contingencies. Each contingency then creates new rules in the system and fine tunes the previous ones.
3.2. Rigidity of the system

The tree structure is a hierarchical structure that operates efficiently when fed with clear goal on what to execute. In this type of structure, any kind of unexpected scenario is a disturbance such as communication short circuit in the system. For example, we often “freeze” the architecture form so as for the next parties in the system to develop their solutions. In this case, it becomes difficult to integrate the concerns raised by the following parties to feed into the given formal solution. Moreover, when a unit fails to come to a solution with the given frozen form, the system needs to propagate up the tree structure where the cause of the failure may have occurred and re-trace that path hoping the following units will not do so in the second run. Since architecture by nature is exposed to such scenarios, at the end, the production system becomes tremendously complex and laborious process of trial and error. We can observe two ways in which the current practice is trying to improve the wastefulness of this trial and error. One approach tries to speed up the process of the trial-error loop using digital computers, and the other approach tries to stiffen the tree structure in a military manner by introducing a role such as project manager. However, both approaches seem devastating for a creative community. We believe the reconfigured partial subjects of the architectural project need to find new ways to connect with each other reconfiguring the well established communication patterns and transgressing boundaries on the way. Perhaps we should allow for the organizational structure of any project to pass through a phase of complexity and even incoherence before the parties involved can negotiate a new functional equilibrium.

4. Case Study: Clyde Lane House

4.1. Context

One of the ongoing projects we are involved in as computational design consultants, within a structural engineering office, is Clyde Lane, which is a small house in Dublin being designed by Future Systems. Our attempt here is to illustrate the aspects in which we have discussed thereof in a context of a real architectural project. The focus in the project is a roof that is designed by the architects as a doubly curved surface [a trimmed NURBS surface] dipping in the middle to form an atrium. For this roof design, the main communication has been taking place among the architects, engineers and ourselves, the computational design team, that develops a custom software tool in order to solve problems specific to the project. All software and interface was implemented in C++ using OpenGL and Intel math kernel.
4.2. Setting up the problem

The initial concern for the architects was to distribute the roof openings approximately following the lighting requirement derived from programmatic constraints. The first step of each project is to render such architectural concerns suitable for computational representation. In addition, at this stage, we try to define our problem as generic as possible such that other considerations and contingencies can be dealt with in the later stages. In this case, the main problem was not the opening shape itself but the density and directionality of the opening distribution pattern.

4.3. Reparameterization algorithm

Therefore, we opted for an algorithm that can generate a mapping / reparameterization of any surface topology that complies to given scaling and directional conditions expressed as a pair of orthogonal vector fields. A good candidate for the input vector field from an engineering perspective would be the two principal stress eigenvectors taken from a finite element analysis of the given surface. This field is chosen under the assumption that for a large enough surfaces with a fine material continuity, preservation of the material continuity along the principal stresses will result in some structural efficiency while at the same time it will reveal and imprint something of the structural behavior of its form. In order to be able to locally control the scaling of the pattern we employed an algorithm described by Ray N., Chiu Li W., Levy B., Sheffer A., and Alliez P. called Periodic Global Parameterization. This particular algorithm has the further advantage that is quasi-isometric which means that the resulting quads show small variation in sizes, a desirable trait of panelization solutions. This happens at the expense of introducing singularities [branching and T nodes]. These singularities however might have a positive impact in the way the designer understands and expresses the underlying geometry. The invariance of the algorithm to rotations of the field by \( \frac{\pi}{2} \) makes it suitable for eigenvector fields like principal stress and curvature where the sense of the vectors is irrelevant. In addition, because the result of the algorithm is a global reparameterization of the given surface we can map any pattern on it easily using the resulting coordinate systems.

![Figure 5. map of opening density.](image-url)
4.4. Integrating Contingencies

The project faced three major contingencies, up to now, that led to dramatic formal change yet, expressing the architectural requirement as density [lighting conditions] and the engineering one as directionality [material continuity along principal stress directions] were both preserved throughout the project.

Initially, the roof had Star of David pattern in which the triangles guarantee a planar solution for the strut problem. [Star of David pattern, Framing solution]
First event: Insurance issue related to the waterproofing
[Circular pattern, GRP solution]
For this event where the structural solution and opening geometry completely had to change, we didn’t have to intervene in the algorithm itself but just change the mapped pattern which was a trivial exercise. We simple had to fine tune form following principal stress directions by putting struts along integral lines to placing holes in between results in a similar material continuity condition.

Second event: Neighbors of the house appealing for a lower roof height
[Change in global geometry, ellipsoidal pattern]
This change in the global geometry had virtually no effect on us as the algorithm is capable of handling any input surface geometry. However, the architects, at this point, decided that the lower patterns will be elliptical openings. This meant that special consideration should be taken for some holes to align with the principal curvature directions so as the flat glass panels for the openings do not deviate too much from the outer surfaces.

Third event: limitation on available transportation means.
[Introduction of joints]
The incorporation of mechanical joints in the roof implied additional consideration because areas around the joints would appear as solid zones on the surface. First we tried to influence the position and shape of joint so that they fall in areas of minimum bending moments and in addition to disturb the pattern less. However this was not entirely possible due to the internal wall position, so in the end there was a clash between the pattern and the joints. To accommodate this we had to alter the control field of the patterning so that it blends smoothly around the joints. This was achieved by taking advantage of another field the distance map. This is a scalar field that has zero values at boundary edges and increases further away from them. The gradient of this field gave us the desired alignment vectors near the joints and the normalized value of this map gave us
the weight by which we blended the gradient field to the principal stresses field so that alignment to the edges fades with distance from them.

Figure 9. truck fitting of roof parts.

Figure 10. effect of joints on structural behaviour.

Figure 11. distance map and its gradient and tangent fields.
The way in which we coped with the changes did not have to do with trial and error, where you remodel everything from scratch. Instead, we added information gained at each stage to our program and developed functions that extract the desired solutions\(^\text{17}\). Because we opted for an algorithm where the solution is controlled through vector fields the design problem itself was transformed into the complementary problem of constructing a vector field that best negotiates project constraints.

---

**Figure 12. field construction and results.**

**Figure 13. optimized mapping.**
5. Conclusions
In this paper we discussed some of the effects of the introduction of digital design tools in the architecture building process. We tried to map out a potential way of taking advantage of the current situation not only in order to cope with the problems caused by the arbitrary use of these tools but also to increase the adaptability of the system as a whole and the quality of the solutions that it yields. To do these we had to examine the implications of programming as a craft and the effect of defamiliarization with given design problems. In addition the concept of simplexity operational in the level of the designer and in the system as a whole helped us to frame better our aims and decisions in dealing with such problems. The case study of the Clyde Lane house by Future Systems architects gave us a chance to apply some of these concepts and seek a deeper engagement of the parties involved in the design process with digital media. The biggest obstacles in the project did not come from the contingencies of the design itself and its realization. They came from rigidity in the current system of production as well as the ambiguous role of computational techniques in design and computational design consultancy as a practice. Computational design is still a signifier without content and for that reason it is not yet clear for other parties what is it for or how they can take advantage of it. Eventually it shouldn’t become a support discipline for architects but rather dissolve and affect all the involved parties in the same ways that IT technologies have disappeared by becoming ubiquitous in the design and building process. We hope that in the future this signifier will express a desire to develop a deeper understanding of digital representations and processes rather than a set of tools to automate and speed up existing processes.

6. Endnotes and References

3. Iterative function systems are a good example on point, since that which was intended to describe natural forms, has become a tool to reproduce their visual appearance even when the actual building process is going to be radically different to energy efficient organic growth. So while in nature the biomorphic is a necessity or an expression of biochemical and mechanical processes in architecture it becomes a liability.
6. Ibid.
8. Choosing a numerical representation is closely related to the problem considered. Let’s look for example at different ways of representing color by numbers.
   1. a computationally efficient linear system [RGB]
   2. a production system [CMYK]
   3. a perceptual system [LAB]
   4. a scientific system [Intensity, Frequency, Polarization]
   5. an intuitive system [HSV]
   6. a symbolic system [table of named colors]
The fact that translations between these numerical representations are not exact indicates that although they all refer to what we call color they do deal with different physical, social,
economical and cognitive aspects of it. For example, using a CMYK representation with its incapacity to represent the full range of human perceptible colors, one can efficiently use printing ink with economical and environmental benefits.

9 For example, in this context one is not asked to design a window but a distribution of light [interior condition] and visual contact [exterior condition] and a distribution of apertures that meets these criteria. By trying to represent the window numerically one has to eventually abandon the idea of a window altogether along with the established and habitual techniques of thinking in windows and designing windows.


12 Examples of defamiliarization and adaptable systems can be found in how people adopt to contingencies and how people conceptualize such phenomena. Masato Sasaki, who is a psychologist, documented a way in which a person who became motion-impaired below his shoulders by an accident developed a way to wear a pair of socks. After five month of trial and error, the time required for the subject to complete the task was reduced dramatically. In order to understand how the subject developed and mastered the skill, Sasaki decomposed the sock-wearing movement into smaller sub-movements and analysed them so as to conceptualize this complex phenomenon of adaptation to wearing a pair of socks. As a result, Sasaki, at the end, conceptualized the technique of the subject developing the system as coordination [Bernstein, Nichlai Aleksandrich] of movements, where two or more movements are combined and executed simultaneously. This coordination was developed by a combination of a conscious thinking about movement [intellectual engagement], the practise [skill development], and the body’s natural tendency to economize energy and find efficient paths between movements [athletic ability] of the subject. This example illustrates the concept of simplexity from both sides, the one that develop simplexity in his skill, and one that observe the complex process of mastering the skill and conceptually simplify them in order to understand such phenomena.


16 An algorithm for the computation of the distance map over meshes is described in : Wang C., Wang Y., Tang K., Yuen M.: *Reduce the stretch in surface flattening by finding cutting paths to the surface boundary*, Department of Mechanical Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
