Reflections: The Creative Process of Generative Design in Architecture

Jeffrey Krause, BArch USC, SMArchS MIT
Blacksquare LLC
jeffrey@blacksquare.com - http://arch.blacksquare.com

Abstract

This paper explores the connections between cognitive design processes and the composition of software code which generates architectural forms autonomously. Similarities are drawn between the pedagogic process (teacher – student) and the crafting of applications to generate forms (programmer/designer – computer). Through an interactive iterative process of generative design and programmatic intervention, architectural forms evolve and are refined.

Background

One can argue that the architectural design process—pen/paper or CAAD—is driven by a process of communication which is part of an iterative loop where feedback, interventions, and reflection occur. The process is framed within the realm of multiple goals or challenges: the communication between the goals and the physical manifestation of the design allow the design and its intentions to emerge, fortified and refined. Creativity within the design process is a complex play of conversation, variation of shapes, and ideological notions within certain determined limits [1]. Within design education, an instructor (architecture studio professor) works under similar circumstances with design students. While not advocating any position on design education or the principles taught within, the position this paper takes describes the instructor’s role as instantiator of a working framework. Within this framework, the instructor influences and challenges the student in an environment open to the questioning of all ideas, encouraging the student to explore their own design sensibilities, methodologies, and craft [2]. Similar open environments can be designed within the realm of generative design and the interaction between a designer and a computer program. This paper explores the potential of mapping instructor based feedback into an iterative generative design process.

Within the realm of architecture and design, technology interactions are becoming more sophisticated. Traditionally, computer/architect interaction centered on models of efficiency and documentation. Recently, this interaction has shifted away from productivity tools and moved towards design exploration and experimentation. One of these fields of exploration is generative design. Within architecture generative design can be defined as the approach of developing applications, or systems which can develop, evolve, or design architectural structures, objects, or spaces more or less autonomously depending on the circumstance. This paper shows that generative design has great potential for uncovering new avenues of research within creativity and computation.
Within this paper, all forms (and by-products of forms, images, rendering, etc.) are generated entirely by programmatic output. The experiment is set up to use the process of studio exploration or a studio charette as an analogy for the generative design process. The two processes were explored in parallel from concept, exploration, refinement, to final design. Ultimately, the experiment is an exploration of creativity, aesthetics, and the creative process. It embraces the expansion of a problem’s solution space rather than the traditional computational approach of optimizing a solution.

**Inspiration and Process – Development of the Framework**

![Okay Coal Co., Tremont, Schuylkill County, Pennsylvania, USA 1975](image)

**Start with a Goal**

In this experiment, the challenge is to generate structures informed and inspired by Bernd and Hilla Becher’s black and white photographs of Mineheads. The underlying experiment directive is the investigation of the aesthetic cohesiveness of the Minehead forms as opposed to the physical or functional attributes of the structures. Looking more closely at the photographs from an aesthetic viewpoint, one notices a haphazard yet purposeful interaction of the structural elements: a minimal subset of items form a bricolage of structural members whose sole function is to support coal extraction and rock movement.

With the definition of the project challenge and internal conceptualization of the Minehead aesthetics, the process of developing a framework initiates a series of traditional representational sketches. These sketches become conversations with one’s self—in-depth investigations of forms, repetitions, patterns, emerging and diverging relationships, similarities, and differences.
**Describe Consistencies**

To determine the generalized code framework, one must first study the Minehead structures, looking for descriptors: potential classifications, hierarchies, general material interactions, and member placement. These descriptors are possible routines or procedures which may be formalized and transformed into generative algorithms. One notices while looking at the structures that they are made from multiple nominal structural elements. One also notices that the elements are not cut to size, but are cut to an approximate size. The elements, some longer than their joining counterparts, slip past intersection endpoints. The general compositional shape of the system, principally driven to accomplish a functional task, is also influenced by the imprecision of construction resulting in slight shifted angles which make up smaller orthogonal units, or triangulated shapes.

**Formalize Code Parts**

Once descriptors have been created, they need to be developed into generalized code (functions and routines) to produce visual output. Generalized code within this experiment is defined as a structured system for output which is generated entirely from random input. For example, if a routine is set up to make 2 points, those points can be in an entirely random (X,Y,Z) coordinate position.

Below are two different outputs from the same set of basic functions in the system; both outputs use four random coordinate points to set up a frame structure. A few things to note about the output: the members (single structural units) are different random sizes within a visually constrained system; the members extend beyond the initial four-point system; and the points are not coplanar, but close to existing within the same plane (once again, constrained by visual interaction). All three of these examples express the aesthetic characteristics of the Minehead structures.
Four-point structures. Two different frame-like outputs from the same set of basic functions.

**Set Range Potential**

Generative code is typically executed (or called) many times during the generative process. To create a meaningful output range, one needs to develop systems with range potential. These systems are built in accordance with the challenge aesthetic from the initial form analysis. The ranges relate to the general tolerances accepted for conditions like slipped members or modified angles. The systems are also driven by random number generation within certain tolerances. In addition to set range potentials, occasionally it may be necessary to break out of preset ranges to allow for unexpected, and potentially illuminating, outputs. Therefore, it is important to code scenarios where unpredictable outcomes can occur, even if they are statistically unlikely.

**Evaluate Output**

As the code begins to generate visual forms, one needs to study the output of multiple iterations. Through multiple iterations one begins to see patterns, potentials, and unexpected conditions. These discoveries and learnings are then used to modify the code iteratively to generate forms fitting within general (visual) aesthetic expectations. This is a significant point to reflect upon: the most common misconception about generative architecture is that the system is entirely random. In an experiment such as this there are an infinite number of potential solutions, but these solutions will fall within certain groupings of visual and contextual appearance because the range of randomness is carefully coded into a variety of algorithms and managed by subsequent generations.

**Add Complexity**

Once generalized code has been developed to work at a fundamental level, it can be augmented to accept additional directions. For example, the structure below extends the development of the simple “frame” by combining it with another “frame” which shares a common structural member with the original as well as formal extensions at the joints.
Increase Tectonic Potential

Once the fundamental algorithms have been created, the next stage is to develop and refine the concept by applying architectural and formal conditions. This is accomplished with the introduction of other tectonic elements, planes, structural systems, and surfaces. This phase of the process becomes more influential as one departs further from the initial design inspiration.

Iterate

As one learns more about the underlying code, and the aesthetics which emerge, the entire process and its stages are usually repeated in order to refine the code, the visual outputs, and the system capabilities. The user-computer/teacher-student interactions grow more complex as the iterations of the proceeding process steps increase. Throughout the process, one needs to look for new design opportunities or unexpected inspirations to explore. These are the critical junctures in the teaching/coding process.

Departure from the Initial Inspiration

Generative design within a multi-output space has the potential to promote a greater degree of novel creativity. Boden suggests the transformation of conceptual or generative space allows for new mental structures to arise which simply could not have been generated from
the initial set of constraints [4]. Within this generative experiment, viewing a variety of
distinct forms prompts one to ponder and explore future possibilities and developments.

While working with the frame-like forms one could imagine alternate architectural
possibilities. What if the frame was a roof structure? What if a floor could be generated from
the frame, but be related at the same time? What about other structural elements? These
spatial musings lead the experiment through unexpected turns.

Framework Experimentation

The entire experimental process outlined in this paper is comparable to the interaction
between architecture studio instructor and design student. Mapping a simplified version of
the instructor/student interaction onto the generative design process, we see the computer
outputs as student works in progress and the generative designer in the instructor role
investigating the output, looking for potential or conditions which allow the design to
progress further. Often the output of this mentor relationship yields surprises and refreshing
new paths of inquiry—conditions and interactions of form not previously explored while
working solely with the Mineheads aesthetic. These “aha” fragments, appearing throughout
the process, allow the design to progress and the solution space to broaden, encompassing
new threads of thought. In this experiment, the breakthrough in understanding occurred upon
realization of the architectural and spatial potential of the Mineheads code aesthetic.

Putting the Framework to Use

Using the fundamental algorithms described in the “Inspiration and Process” section above
and applying, combining, and modifying these over multiple generations of interaction,
structures begin to emerge from the system. In general, the visual density of the output makes
it difficult to evaluate looking at a single rendering. To view the entire 3D object, the forms
are visually analyzed and interpreted through animations (turntable to see all sides), or
through OPENGL visualizations in environments where the form can be rotated and
investigated manually. The images represented in this paper are snapshots from simple
animations; these animations and larger images can be viewed in more detail at the website.
First Output

The complexity and richness of the structures which emerge derive from the formal interaction of multiple tectonic elements generated simultaneously. Running the simulation several times reveals compelling spatial arrangements grouped within a similar aesthetic family. After viewing several generations, the next logical step of the process is to introduce more architectonic textures and features.

Over subsequent generations, the form takes on different characteristics with the introduction of other elements and through progressive refinements. Once the basic algorithms for generation have been established, the next phases reflect the concept of real-time design. Forms are generated, assumptions are made from the output, and the code is influenced once again with more architectural modifications and new directions.

Moving Towards Architecture

The next generation reflects the first significant architectonic intervention with the introduction of horizontal planes and the addition of solid roof-like membranes. The horizontal planes beneath the frames immediately convey a semblance of scale. Vertical planes are also introduced as transparent walls to resemble glass.
After the introduction of planes, the next generation attempts to introduce variability into the fundamental structural elements such as the generation of frame systems without related planes. Below one notices “roof” structures that contain planes and “roof” structures that are open. Floor structures are also unpredictable both in terms of the number of floors and in their relative distances.

Output of struct03, introduction of variability, 2 views [7]

In addition to variability in the “roof” structure; options are selected for vertical wall structures, allowing the potential for both transparent and opaque vertical surfaces.

Output of struct04, variable planes, 2 views [8]

**Adding Refinements**

Generations are examined to explore refinements such as structural elements which connect roof and floor members. Other refinements include architectonic features such as mullions within the transparent planes to break up the homogenous surfaces.
Alternative spatial arrangements are introduced with stacking of elements near or on top of each other, creating distinct “pod-like” units, or “clusters”. Each arrangement technique is investigated, exploring the spatial richness of the open space between forms as well as the complexity of interactions within the intersecting forms.

Below is an example of a more horizontal or linear progression of the frame forms. Positional variation was superficially explored in this experiment and explored in more detail in other experiments.
Another generation applies refinements of more delicate architectural elements such as railings along planes or lighter structural elements in planes and in the transparent objects.
Output of struct17, addition of lighter elements, 2 views [13]

Output of struct20, railings and more refinement, 2 views [14]
Inside Out

The final experiment positions the viewer within the generated structures, evoking the feeling of being inside the virtual spaces and forms.
The examples shown within this paper are only a few of the thousands of images which were generated, based on hundreds of generations, and 20 code versions. More examples, images, and animations can be found on the website.

The goal within experiments of this type is to gain insight into computer and human design interaction and allow us to explore the cognitive processes involved in acts of creativity. The generated objects are not meant to be inhabitable or functional but are explorations of creativity between a designer and interactive computer code. One needs to be facile in both design comprehension and in computer programming in order to see the process as an improvisatory, inspired sketch rather than a tedious programming experiment.

**Conclusion**

Human and computer interaction within the design realm exists on many levels, occupying coexisting methodologies and working processes. This paper describes one experiment which uses generative design as a conversation, a sketching tool, and a working relationship akin to a professor and student. In the process, communication and understanding exist, and Knuth’s statement that “a person doesn’t really understand something until he teaches it to someone else. Actually a person doesn’t really understand something until he can teach it to a computer” [17] is explored. Ultimately, the experiment itself represents a personal approach to visual form generation in purely interactive and aesthetic terms, as a visualization tool for “sketching” formal occurrences within the realm of controlled random possibilities.
References

[5] Krause, Jeffrey, struct01; seed: {DC81BB46-8BAF-40e4-95B4-81D1EC84582E}, 2003