Evolutionary Development of Mathematically Defined Forms

Robert J, Krawczyk
College of Architecture
Illinois Institute of Technology
Chicago, IL 60616 USA
E-mail: krawczyk@iit.edu

Abstract

With the increase use of algorithms to develop images, as well as, three-dimensional sculptural and architectural forms, additional focus should be placed on methods to teach students how to approach such a design technique. This paper reviews one such method. Architectural students are looking for inspiration for new forms to better realize their own designs and computer methods are offering them another tool to expand their investigations. This paper outlines one method used as part of a programming course developed for architectural students. The method highlights a very sequential approach to form investigation in using a common starting geometry. This approach stresses the development of rules and evaluating their results as a method to determine the next step to investigate. Equal importance is placed on the anticipated, as well as, the unexpected.

1. Introduction

As commercial Computer-Aided Drafting and Design systems have grown in power to handle both two-dimensional and three-dimensional representations, many of the basic functions for creating drawings and models have been adequately addressed. A variety of primitive shapes are available. Operations to either join or subtract them in addition to options to modify them at the vertex level are also readily available. Primitive shapes range from lines and planes to surfaces of revolution and free-form blobs. These systems rarely allow for an organized and repeatable method to create models. They all create “hand-crafted models”. Some are parametrically based, which allow for dimensional relationships, such as: “dimension A changes by 10% if dimension B changes by 5%”. In these systems all changes are highly regulated, and it is difficult to control a entire series of parameters or components.

Other methods to develop forms have been mathematically based. These include fractals (Yessios [10]), space curves (Krawczyk [3]), and complex surfaces (Sequin [8,9]). Still others have investigated agent based autonomous systems which are capable of collaborating, building, degenerating, and transforming, using genetic algorithms, Krause [2]. In these types of form building methods, rarely is the process explained in a sequential fashion on how to arrive at the final results. The process is very important for students to see. The drawback is that these approaches require a great amount of research and extensive mathematical knowledge, plus advanced methods in programming. The typical architectural student would find these difficult in an introductory course.

One aspect of these systems I wanted to develop was how a commercial CAD system could be used to investigate a geometrically based design concept and how a CAD programming language could be used to establish rules and procedures for such an investigation. The best way to demonstrate this idea was to have the students follow a step-by-step approach that could show them how a design concept could be developed.

These exercises were designed as "programs as pencils"; every concept, every variation was implemented by an individual program modification. The programs were meant to hold all the rules needed for the

form being generated. Manual modifications were not allowed to the model itself. All ideas were to be implemented by actual program modifications. When students asked if another variation would be interesting, the usual response was "*I don't know, try it*".

Mitchell [7] took a similar approach in introducing programming to architectural students using graphic exercises. Even though the focus of his book was programming, it is a good source for how to organize work to be presented one concept at a time. Maeda [6] more recently took a similar approach in demonstrating the creation of drawings and images for those starting graphics and using programming as a method.

The incremental process of developing a computer program is a very powerful. It allows one to concentrate on a single concept at a time, make small manageable changes to the program, and most importantly, always work from success. One always begins the next step from a previously working program. This method has been shown to work with students whose primary interest is not computer science. It also introduces the idea that a program evolves and is not a static statement of a concept, but a starting point to further investigation.

Working within an established CAD environment may have some limitations, but having easy access to basic modeling primitives, operations, and visualization aids enables the student to concentrate on the design aspects of the forms.

The following section describes such a series of exercises. In developing these exercises I did not know exactly where they were going to end. I tried to incorporate in the steps some of the questions I asked myself as I developed each one. The importance of these exercises becomes the steps and decisions at each step rather than the resulting form. I wanted to show the students the process of program development and how it could be used to investigate basic geometric forms.

A few initial assumptions were made:

a. The exercises should concentrate on the development of forms and not engineering analysis.
b. The exercises should try to develop forms normally not available by assembling basic CAD entities.
c. The forms generated could be somewhat anticipated. Students should be able to easily visualize each form and determine variations to it.
d. The forms would be three-dimensional.
e. The exercises would investigate how to develop forms based on rules easily established and modified.
f. The primary objective was to create architecturally suggestive forms. Scale, proportion, and orientation will be considered.

To generate the forms the programming language used for the exercises was AutoLISP within Autodesk's AutoCAD R14. 3D Studio MAX was used for the renderings and animations.

2. Description of form development

Previous attempts to develop such a procedural approach to form development utilized a linear form (Krawczyk [4]). The linear form allowed for surface articulation and the development of both frame and surface models. The linear form had limitations for students to go beyond the set of exercises presented. In this series a circular form is used and then extended to the many other simple curved forms that are readily available.

Our initial class discussions included the basic geometric forms that architects have used to create built forms. Ching [1] developed an excellent description of how points, lines, planes, and volumes become the basic building blocks for form development. In addition to the basic geometric properties of each, many examples of existing architecture are shown. This discussion gave the class a context in which to work. We also covered basic computing concepts and programming language constructs.

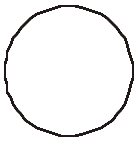

Figure 1: *Initial circular shape*

Figure 2: *3D extrusion*

1: The students were given an initial form to investigate, a circle. The program accepts a center point and radius for the circle. The edges of the circle are created by a series of line segments (Figure 1).

Discussion: Input functions, looping constructs, geometry for creating a circle, and computation of the line segments. This program duplicates the circle command found in most CAD systems.

2: Add a thickness to the lines to extrude the circle edges into three-dimensions (Figure 2)

Discussion: How 2D lines can be represented as 3D planes, how 3D planes are specified, and how generally 2D shapes can be represented in 3D?

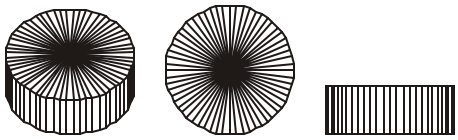

Figure 3: *Solid*

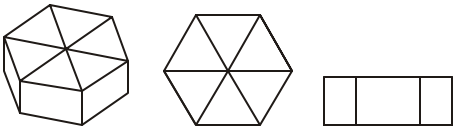

Figure 4: *Number of sides*

3: Add top and bottom surfaces to the edges to form a cylinder (Figure 3).

Discussion: Conversion of basic shapes to solids.

4: Add a parameter to modify the number of sides of the circle. Be able to create a series of polygons (Figure 4).

Discussion: How a circle is a multi-sided polygon.

Figure 5: *Change circle into an ellipse*

Figure 6: *Change center height*

5: Add option to change the values of the major and minor axis radius. Be able to turn a circle into an ellipse (Figure 5).

Discussion: The mathematical relationship between a circle and other related shapes.

6: Add a center vertex height in addition to the edge height. The center height can be less than or greater than the height of the edges (Figure 6).

Discussion: Beginning to investigate how the top surface can be articulated.

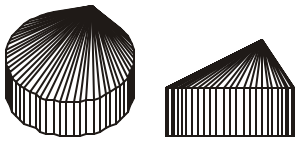

Figure 7: *Modify the center vertex offset*

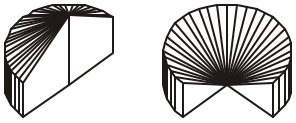

Figure 8: *Modify the included angle*

7: Add an offset distance for the center vertex so that it can be moved is the x, y, as well as, controlled in height (Figure 7).

Discussion: Continue to investigate how the top surface can be modified.

8: Modify the included angle of the circle to be able to sweep the profile through an angle less than 360 degrees (Figure 8).

Discussion: How to control the extent of the shape. How to handle the start and end of the shape.

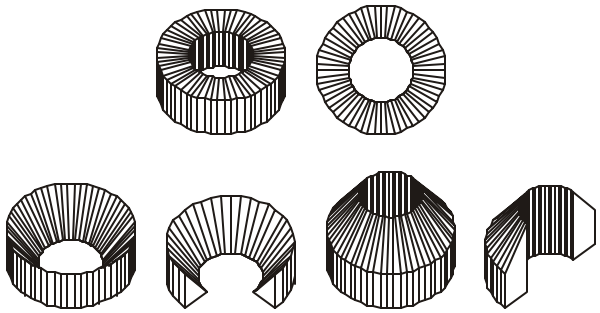

Figure 9: *Modify the inside radius*

9: Add a parameter to define an inside radius to give the circular shape an opening at the center (Figure 9).

Discussion: What are the variations in form when height and radius are considered?

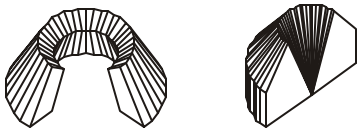

Figure 10: *Modify upper and lower radius*

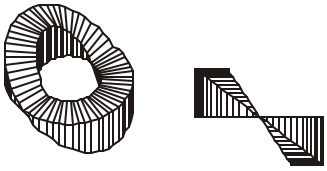

Figure 11: *Height generated by sine function*

10: Articulate the upper and lower surfaces by giving each an individual inside and outside radius (Figure 10).

Discussion: Begin to separate the different parts of the form and study how each could be further controlled.

11: Vary the edge height by multiplying the edge height by the sine of the circular angle (Figure 11). Assume that the angle covers the same included angle as a full circle, 360 degrees.

Discussion: What type of variations can be applied to edge height? Other height functions included stepping the top surface, constant ramping from 0 to the maximum height, or ramping by increasing to a mid-point then decreasing. Use of the sine curve seemed the most promising and interesting.

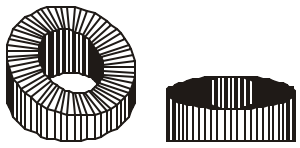

Figure 12: *Separate height for the sine function*

12: The problem with the sine function is that it generates zero and negative edge heights as seen in Figure 11. An additional height parameter is added to give the curve a positive offset, so that the edge always has a minimum height (Figure 12).

Discussion: What is the natural effect of the function selected and how can it be rectified?

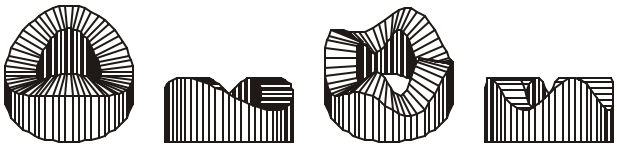

Figure 13: *Modify the angle for the sine function*

13: Add a parameter to vary the angle through which the sine function passes, which can now be different from the included angle of the circle (Figure 13). The circle will normally be drawn through 360 degrees; the top function could have any start and end angle value including multiples of 360 degrees.

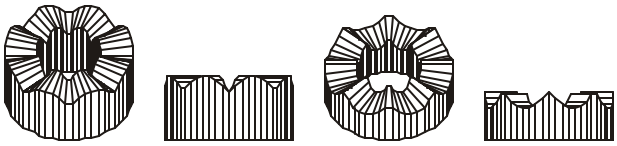

Figure 14: *Modify the sine function value*

14: Another method to modify the sine function is to compute its absolute value or its negative absolute value (Figure 14).

Discussion: How can basic functions like sine and cosine be modified to produce a variety of curves.

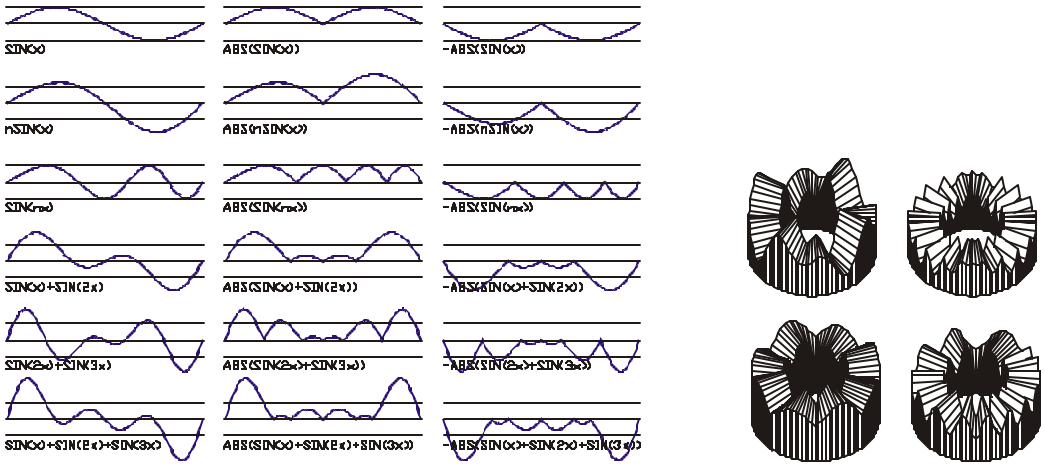

Figure 15: *Other curve functions*

15: In addition to the simple sine function, others are developed that include factors applied to it (Figure 15). In addition to these, an identical series was developed for the cosine function.

Figure 16: *Curve function on edge height*

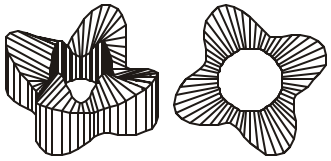

Figure 17: *Curve function on outside edge*

16: Apply a different curve function to the outside and the inside top edge. Each curve also has its own total angle, which it passes through (Figure 16).

Discussion: Given the 18 combinations of sine and cosine functions shown in Figure 15, what are the possible variations? How do we investigate each?

17: Apply a curve function to the outside edge as it extends around the circular form (Figure 17). Use the circular edges not as the edge of the form but as an axis to apply a curve function to.

Discussion: How the actual form evolved into a spine for further articulation development.

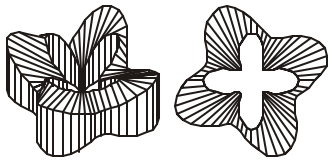

Figure 18: *Curve function on inside edge*

18: Apply a curve function to the inside edge as it extends around the circular form (Figure 18). Same as the step above using the inside edge as an axis.

Discussion: Figure 19 demonstrates some variations based on the developed parameters.

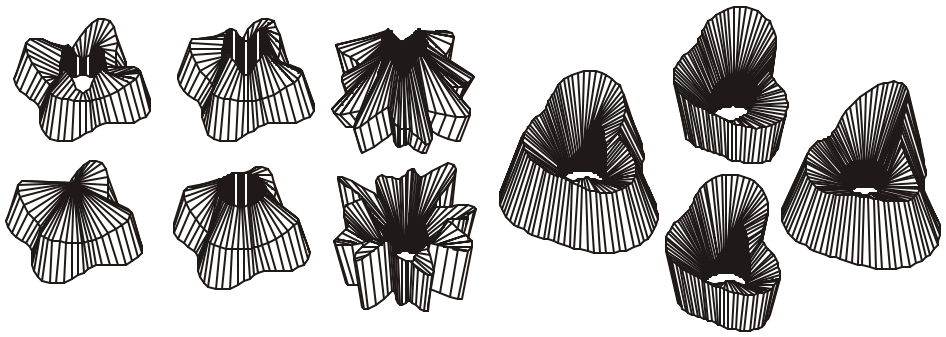

Figure 19: *Variations with all parameters*

3. Extending the basic concept

Once the students completed the previous set of exercises, the same concepts were applied to a set of more interesting curves. Each student selected a curve and produced a series of variations based on the incorporation of the parameters previously covered. The curves that were selected included the following: Bicorn, Piriform, one-half on an Eight Curve, one-half of a Leminscate of Bernoulli, Cardoid, Nephroid, Epicycloid of 3 and 4 cusps, Deltoid, Astroid, Lame Curve, and a Hippopede. These were selected for their continued use of simple combinations of sine and cosine functions and their potential to generate forms that could suggest architectural elements. Lawerence [5] was an excellent reference to develop these curves.

In addition to still renderings and drawings, one of the students created an animation by developing a series of related designs and then morphing them from one to the other. This visualization technique allows for the development of the in-between forms and gives the student more ideas on the variations to attempt. It also suggests that architectural forms could actually change in time – an interesting idea.

4. Conclusion

The process of developing these forms was the most important aspect of these exercises. The step-by-step procedure, the evaluation of each as encountered, and the trial and error required were the basic concepts discussed. The students were amazed that they could generate such forms and control their variations so easily. As we progressed through the exercises the students began to imagine what some of these variations might be. The expectations readily enabled the student to think about how to approach each succeeding modification. The strict mathematical basis for these designs focused the students to investigate the possibilities in an orderly fashion.

The discussions at the end of the course clearly indicated that the students now consider the development of programs as their own personal expression of an idea and that CAD systems could be used to investigate ideas and not only document decisions already made. They began to understand the feedback their rules created and how it could be used to clarify concepts. Many also better understood the mathematical basis for many simple forms and were more comfortable about progressing to other related areas. The ability to establish fixed and variable parameters within the model enabled the students to focus on a single design decision. Being able to duplicate results, time-after-time, also allowed for changes to be made at an earlier previously created decision point.

The greatest problem encountered was the development of values for all the individual parameters to really be able to see the entire range of form possibilities. Since enumeration was not possible, changing individual values did at times generate unexpected results – exactly what we were looking for.

5. Acknowledgments

Special thanks to the following students who developed examples and encouraged me to develop these ideas: Peng-Chien Chang, Vasavi Duvvur, Ruben Gonzalez, Ying-Chun Hsu, Hsi-Hao Hsueh, Ekkachai Mahaek, Priyadarshini Naik, Faisal Navieed, Umnt Ongoren, Alfred Sanders, and Fareeda Zayyad. Additional thanks to Ying-Chun Hsu for collecting and organizing the student work and preparing the final renderings.

Additional thanks are extended to the reviewers for their helpful comments.

References

[1] Ching, Francis. *Architecture: Form, Space & Order*. Van Nostrand Reinhold Company, 1979

[2] Krause, Jeffrey. "Agent Based Architecture". Association of Computer-Aided Design. in Architecture Conference Proceedings, 1997

[3] Krawczyk, Robert. "Hilbert's Building Blocks". Mathematics & Design Conference Proceedings, 1998

[4] Krawczyk, Robert. "Programs as Pencils: Investigating Form Generation". Association of Computer-Aided Design. in Architecture Conference Proceedings, 1997

[5] Lawrence, J. Dennis. *A Catalog of Special Plane Curves*. Dover Publications, 1972

[6] Maeda, John. *Design by Numbers*. The MIT Press, 1999

[7] Mitchell, William, Liggett, Robin, Kvan, Thomas. *The Art of Computer Graphics Programming*. Van Nostrand Reinhold Company, 1987

[8] Sequin, Carlo. "Art Math, and Computers: New Ways of Creating Pleasing Shapes". Bridges Mathematical Connection in Art, Music, and Science Conference Proceedings, 1998

[9] Sequin, Carlo. "Analogies from 2D to 3D, Exercises in Disciplined Creativity". Bridges Mathematical Connection in Art, Music, and Science Conference Proceedings, 1999

[10] Yessios, Chris. "A Fractal Studio". Association of Computer-Aided Design. in Architecture Conference Proceedings, 1987