Programs As Pencils: Investigating Form Generation

Robert J. Krawczyk
Illinois Institute of Technology,College of Architecture
krawczyk@iit.edu

ABSTRACT

This paper reviews two projects undertaken in a CAD programming course that demonstrate to the students how programs could be developed to investigate possible architectural forms. The projects highlight a very sequential approach to form investigation in using both common geometries and the introduction of randomness to control design rules. This approach stressed the development of rules and evaluating their results as a method to determine the next step to investigate. Equal importance was placed on the anticipated, as well as, the unexpected.

INTRODUCTION

The College of Architecture has already in place for a number of years a series of courses that cover 2D and 3D CAD, image processing, composition and multimedia, and animation. With the increased use of computers in the studio and the advanced level of CAD use by students who have taken the above series of courses, a number of students requested that a CAD programming course be developed.

The overall intention the course was to enable students to manipulate a CAD system, understand basic programming concepts, understand how CAD systems handle entities, and demonstrate the range of programs that could be written. Even though some of the initial students desired a CAD customization course, an attempt has been made to focus on design issues. Other traditional issues, such as, bill of quantities, space planning, entity properties and drafting aids were also included as part of the course.

As commercial CAD systems have grown in power to handle both 2D and 3D representations, many of the basic functions of creating drawings and models are fairly well taken care of. The aspect of the these systems I wanted to develop in this course was how could A CAD system be used to investigate a 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 that 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 a program modification. The programs were meant to hold all the rules needed for the form being generated. Manual modifications were not desired at this stage. 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*".

The following sections describe two such 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 investigated basic 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 and that variations could be easily investigated.

c. The forms generated could be somewhat anticipated. Students should be able to easily visualize each form.

d. The forms would move into the third dimension as soon as possible.

e. The exercises would also demonstrate a number of 2D and 3D representations and the variety of ways each can be modeled.

f. The exercises would investigate how to develop forms based on rules and randomness.

To generate the forms the programming language used for the exercises was AutoLISP within AutoCAD R13. 3D Studio 4 was used for the renderings and animations.

EXERCISE 1 - GEOMETRIC FORM GENERATION

This first exercise uses the basic geometric function of sine and cosine to control the edges of a simple form. Each step in the exercise includes the a description of what is to be done and the instructional intention.

1. The students were given an initial program which asked for two points which defined a rectangular boundary and the number of lines to be drawn across the X axis (Figure 1).

Intent: Input functions, looping construct, and the computation of line start and end points were highlighted. This program duplicates the array command found in most CAD systems.

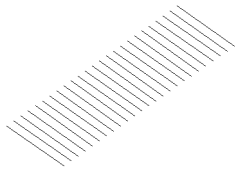

Figure 1 - Initial boundary

2. Add a thickness to the lines to extrude them into 3D (Figure 2).

Intent: The entry of the height of the extrusion is added. The 2D lines now are represented by 3D planes.

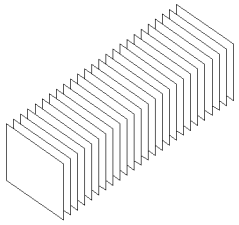

Figure 2 - 3D extrusion

3. Vary the length of each line by multiplying the original line length by the sine of an angle. Assume that the angle covers 360 degrees over the repetition of lines (Figure 3).

Intent: The computation of an angle and line length is added. Variation is added to the edge of the planes.

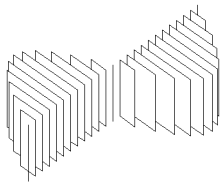

Figure 3 - Vary length by sine

4. Modify the line length computation by changing the sine function to the cosine function (Figure 4).

Intent: A related variation is investigated.

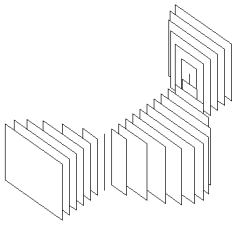

Figure 4 - Vary length by cosine

5. Modify the line length computation to use the absolute value of the cosine function (Figure 5).

Intent: After reviewing the last two variations and possibly wanting to use symmetry, it seemed appropriate to only concentrate on half of the form.

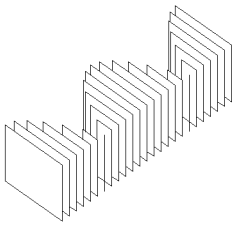

Figure 5 - Absolute value of length

6. Modify the line length computation so that only 1/4 of the length is determined by the cosine function and the remaining 3/4 of the length is based on the original length specified (Figure 6). Further modify this version by adding a prompt which determines what this ratio should be.

Intent: Since planes tended to decrease to zero and a desire to have more control over the depth of the function used, a length factor is added.

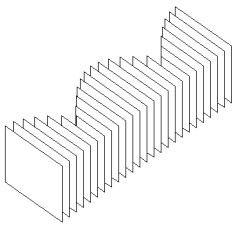

Figure 6 - Add length factor

7. The current version is based on the cosine being computed through 360 degrees. Add a prompt to get the number of cycles. Compute the total angle by multiplying 360 times this factor. Try values less than 1.0 and values greater than 1.0 (Figure 7).

Intent: Add more control over the edge of the planes.

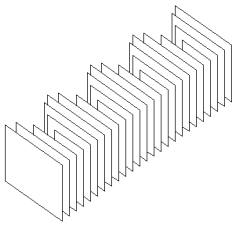

Figure 7 - Vary angle of length

8. Use the same type of computations that are used on the line length to modify the height. Modify the Z coordinate by this new value for both the start and end of the line (Figure 8).

Intent: Only the edge of planes were considered up to this point, now we also consider the height of each plane.

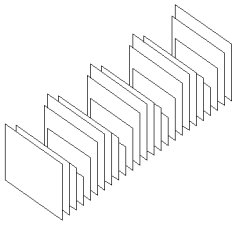

Figure 8 - Vary height

9. Modify the program to mirror the entire design across the straight edge (Figure 9).

Intent: Add symmetry to form.

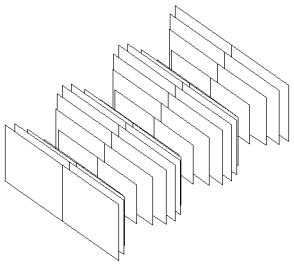

Figure 9 - Mirror planes

10. Modify the program by changing the cosine function in the computation of the height to the sine function. Leave the length using the cosine function (Figure 10).

Intent: Include a new rule that differs the height from the length.

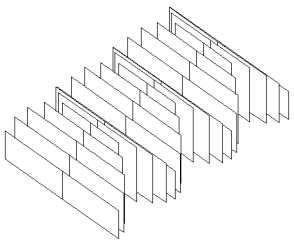

Figure 10 - Change sine to cosine

11. Modify the program by switching the cosine and sine functions in the computation of the line length and height (Figure 11).

Intent: This is a variation of the previous height and length computations.

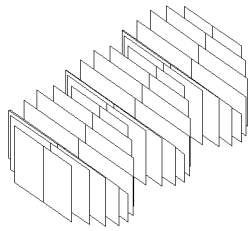

Figure 11 - Switch sine with cosine

12. Instead of using a thickness to extrude the lines into planes, replace the planes with frames: two horizontal lines for the top members and two vertical lines at either side for the columns (Figure 12).

Intent: The form is now represented as a frame.

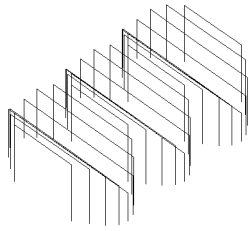

Figure 12 - Change planes to frames

13. The top member height is computed by the cosine function. The height is the same value for the edge of the structure as it is the midpoint. Modify the program to compute a new height for the midpoint based of the sine function (Figure 13).

At this point the length of the top is computed by the sine function, edge height by the cosine and midpoint height by the sine. Another variation would be to try cosine, sine, and cosine respectively.

Intent: The midpoint of the frame is independently computed from the edge points.

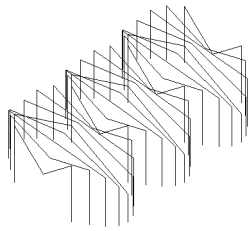

Figure 13 - Vary midpoint height

14. The height of the edge and the midpoint are both computed from one entered height. Modify the program to accept two different heights; one for the edge and one for the midpoint of the frame (Figure 14).

Intent: The midpoint height is independent from the edge height.

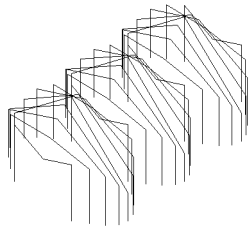

Figure 14 - Vary edge height

15. The columns currently start at the edge of the top members and go straight down at that location. Modify the bottom point of the columns so that it is located along the edge of the boundary first specified (Figure 15).

Intent: Variation on how the boundary is used to define the frame.

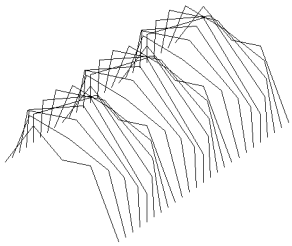

Figure 15 - Change column base locations

16. We started with lines, then developed them into vertical planes, and then into frames. Now replace the frames with a surface based on the computed points (Figure 16). The students are asked to generate a series of forms based on this last variation.

Intent: The frame representation is now modified to a surface.

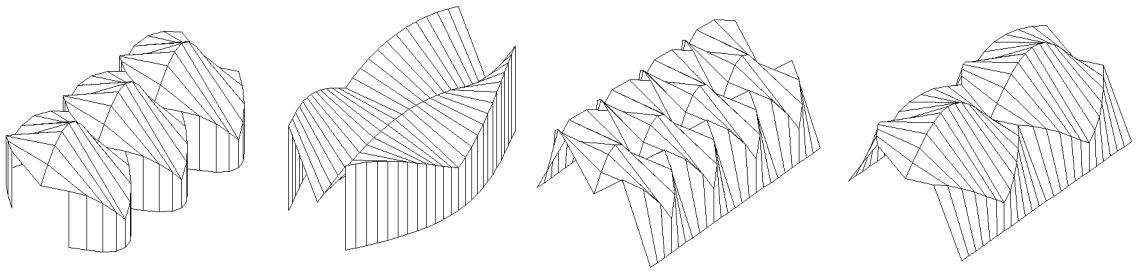

Figure 16 - Change frames to a surface

Once these steps were completed, discussions were held on the different representations that the form evolved, how each could be used; from 2D CAD, 3D CAD and rendering to animation.

The discussion then moved to other geometric functions that could be used to develop forms. Since this initial series used a simple linear form, we started to discuss other ones based on a circle, ellipse, spiral and an entire family of curves based on complex sine and cosine functions as described by (Lawrence 1972) (Figure 17).

The students were asked to consider how the basic form should be generated, how the edge should be determined, and how the height will be determined and how to will related to the edges. Simple increasing and decreasing relationships were discussed, continued use of the sine and cosine functions, and using two or more of the complex curves to control the edges and midpoint of the form.

Student work in Figure 18 demonstrates a series based on a cardioid and Figure 19 a series based on a nephroid.

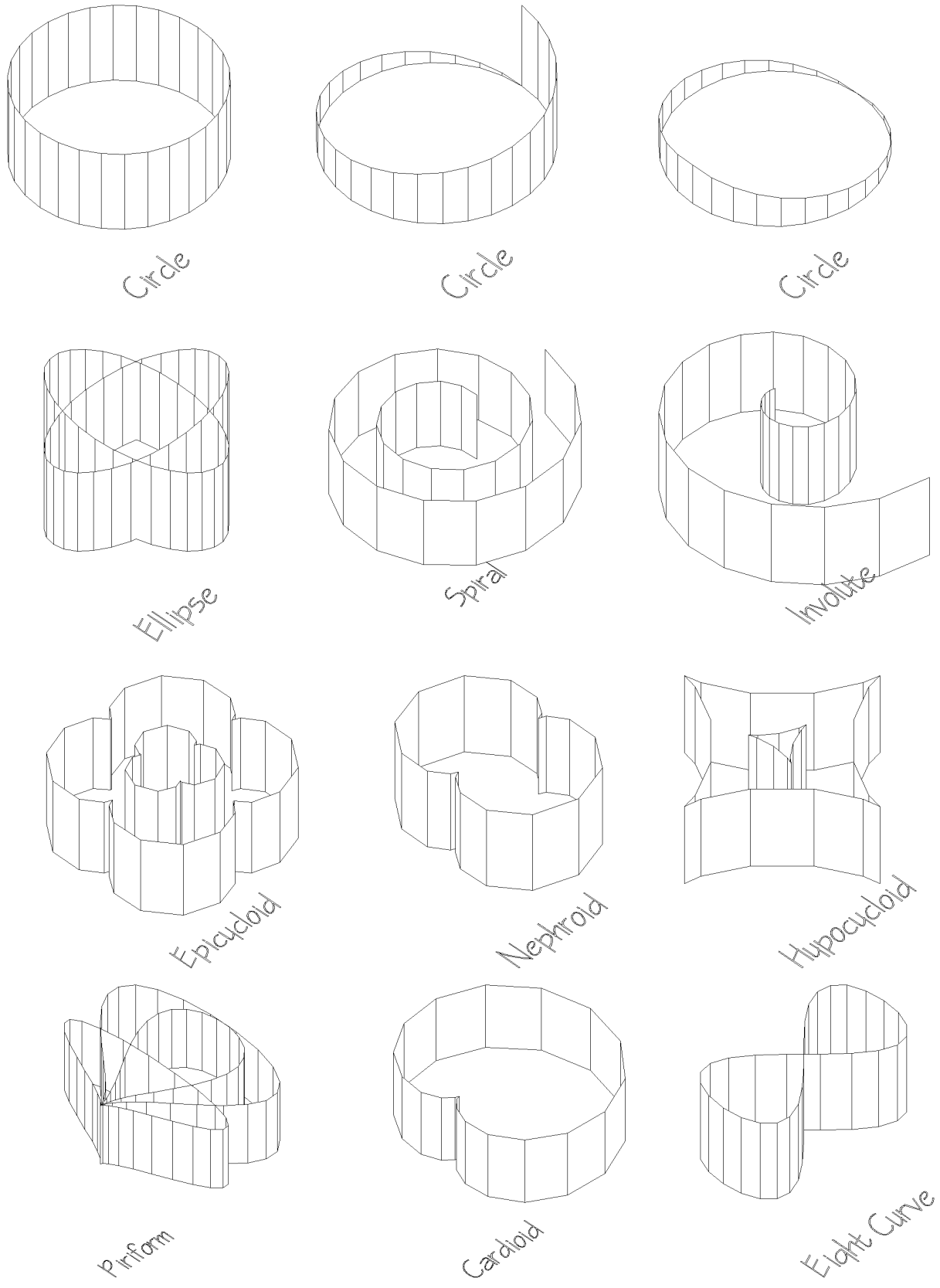

Figure 17 - Examples of complex curves

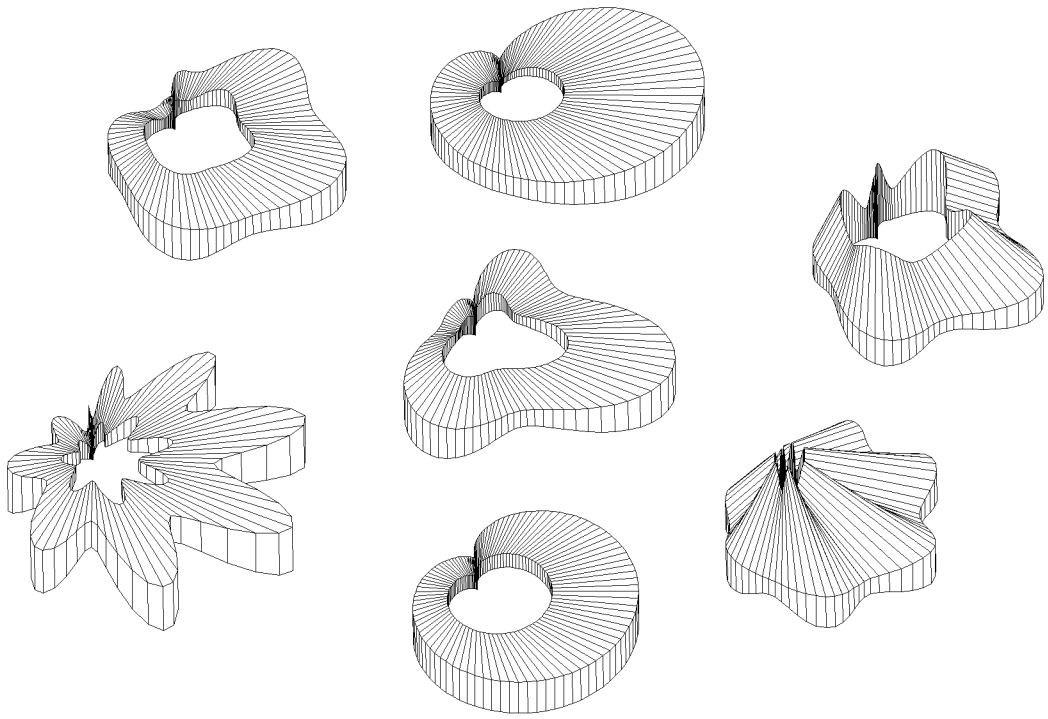

Figure 18 - Forms based on a cardioid

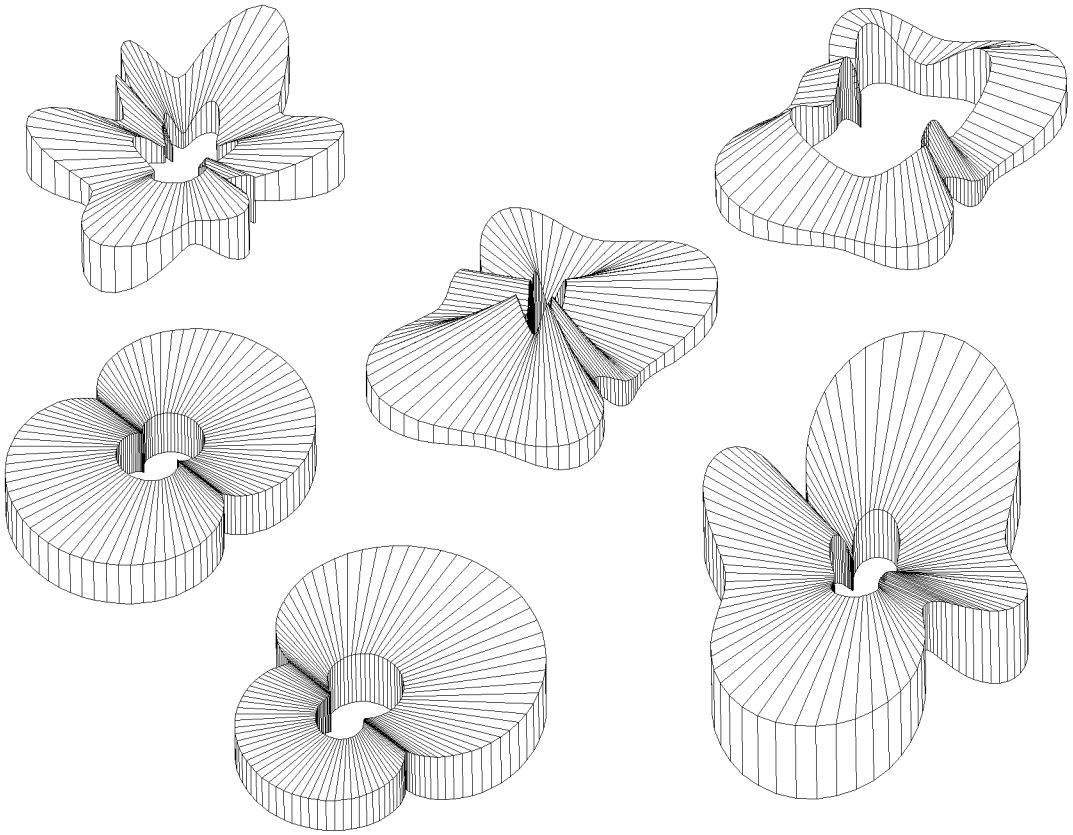

Figure 19 - Forms based on a nephroid

EXERCISE 2 - RANDOM CONSTRUCTION

The second exercise uses the simple blocks and randomness to develop forms. Again each step in the exercise includes the a description of what is to be done and the instructional intention.

1. The students were given an initial program which asked for two points which defined a rectangular boundary and the number of randomly generated rectangles to draw (Figure 20).

The location of each rectangle and its length and width were to be randomly generated.

Intent: The program introduced the concept of random numbers and how to use them in computations.

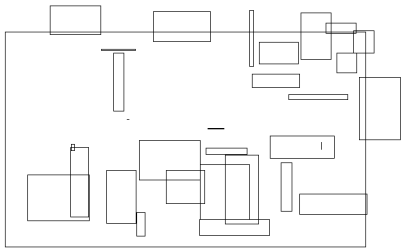

Figure 20 - Initial rectangles

2. The problem with the previous version is that the rectangles extend beyond the borders of the boundary. The program is modified to only draw rectangles that are fully inside the defined boundary (Figure 21).

Intent: Adding the concept of exclusion and begin to consider other rules that can control the arrangement.

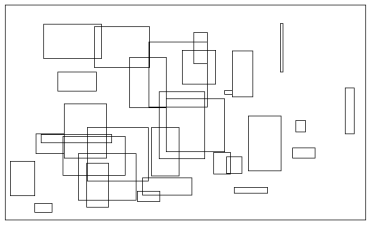

Figure 21 - Add boundary check

3. Because of the great range of values that can be randomly selected, the rectangles have a great variety of sizes. In this version, the program is modified so that a module size is asked for and that the rectangle location and size is computed based on this module (Figure 22).

Intent: Adding rules which control the size and location of the rectangles.

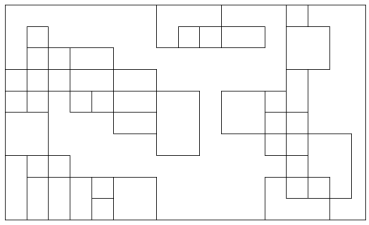

Figure 22 - Add module size

4. In this variation the number of rectangles parameter is changed to a percent fill. The area of the rectangles is being drawn is computed and compared to the percent fill entered. Rectangles are drawn until one exceeds the percent fill (Figure 23).

Intent: Present another method of determining how many rectangles are to be placed.

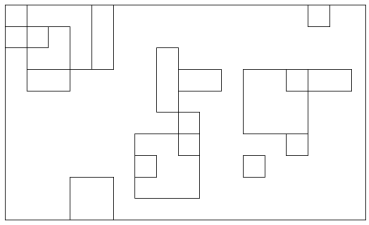

Figure 23 - Placements by percent fill

5. In this variation rectangles are created that are either vertically or horizontally oriented, all being a constant thickness. Only a single dimension is required, the other will be set to a single module (Figure 24). To determine if the rectangle should be vertical or horizontal, a simple even/odd rule on rectangle count is used.

Intent: Add rules for orientation and placement.

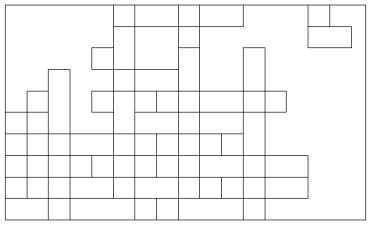

Figure 24 - Vertical and horizontal placement

6. The thickness of each rectangle is entered as input (Figure 25).

Intent: This is the last in the 2D series. The discussion moved to the type of rules that could be added, the addition of color, development of 2D images and its implications as a basis of architectural forms.

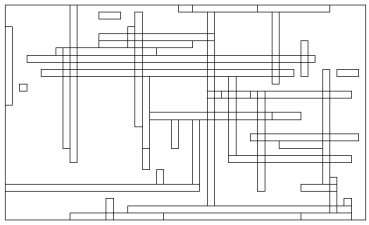

Figure 25 - Set thickness

7. The rectangles are created as 3D volumes. All the volumes start at a elevation of 0. No need to check if the height is within the boundary. Also include a "base" and "ground" for the model (Figure 26). The volumes are created as two entities, first one as an extruded side, the second as the top surface.

Intent: Modify the representation from 2D to 3D; discuss methods to create the volumes, and introduce layering to separate parts of the construction.

Figure 26 - Rectangles as volumes

8. Change the program for a circular boundary (Figure 27).

Intent: Discuss alternate methods for boundary exclusion.

Figure 27 - Circular boundary

9. Modify the volumes to be a single entity using AutoCAD's 3DMESH option. Compute the bottom elevation and thickness for each volume elevation randomly selected based on the boundary height (Figure 28).

By changing the axis to which the length applies, we could create planes along the XY, XZ, and YZ planes. In this program only ask for the maximum length and width of the plane and its thickness.

Intent: Begin to discuss the volumes as planes and the methods CAD systems to create single entity volumes.

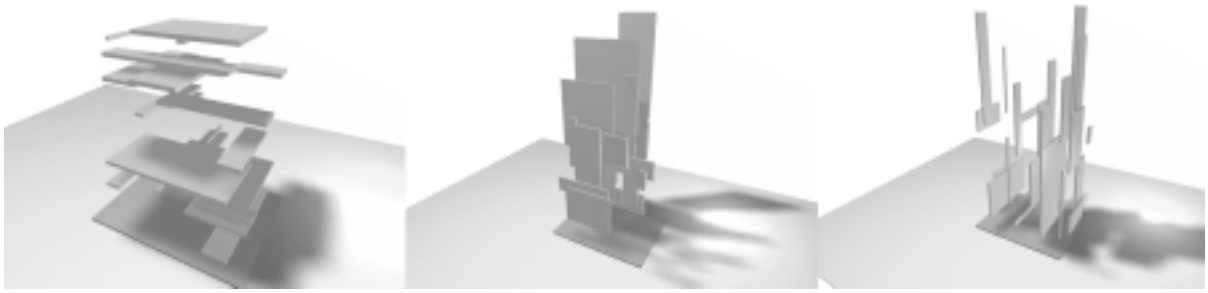

Figure 28 - XY, XZ, and YZ planes

10. Select along which axis each plane will be created by randomly by selectly a number from 1 to 3; representing XZ, YZ, XY planes (Figure 29). Instead of ignoring planes that do not fit in the boundary, modify this program to reset the plane dimension so that it stays within the boundary. Place the base and each type of plane on individual layers. Also count how many are created in each plane and display the counts at the end of the program.

Intent: Introduce randomness in the selection of orientation, location, and size.

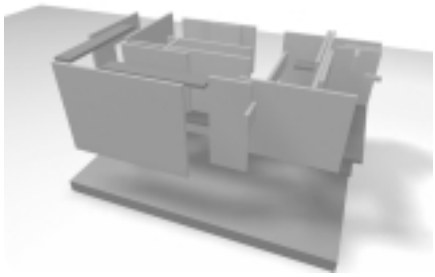

Figure 29 - Select plane axis

11. Treat the vertical planes, XZ and YZ, as walls; set their bottom elevation to 0 in all cases (Figure 30).

You will also notice that the method to select an axis does not give us the ability to skew the selection to any one plane. Modify the selection to randomly select from a list, such as, (1 2 1 2 3 3 3 3 3 3).

Intent: Introduce architectural elements and control over random selection.

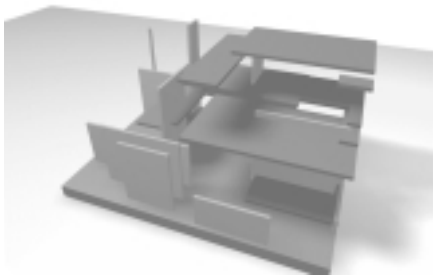

Figure 30 - Planes become walls

12. Since each plane is defined by a simple length and width, the random selection will give us a great range of plane sizes. To better control these dimensions, input both a minimum and maximum length and width. The plane dimension will now be based on the minimum plus the difference of the maximum and minimum randomly computed (Figure 31).

Intent: In this version only include planes which are at least the minimum dimension to more closely model architectural elements.

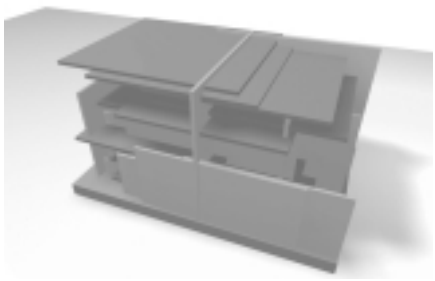

Figure 31 - Minimum plane length

13. In this version add the ability to rotate any of the three type of planes around the Z axis. Figure 32 includes XY plane only rotation and a combination of XZ and YZ plane rotation.

Intent: Add rules for plane orientation.

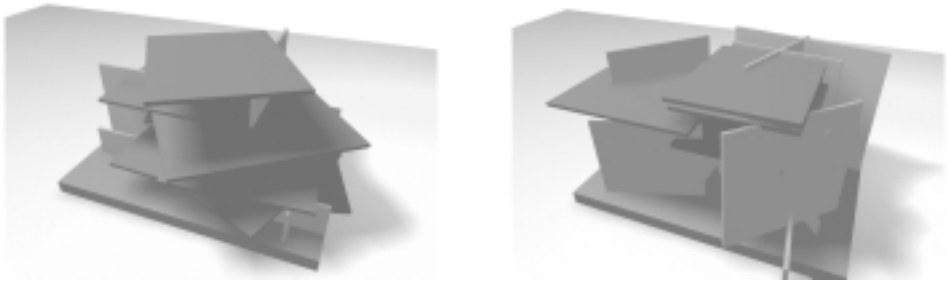

Figure 32 - Rotation of planes

The discussion continued to other variations and type of geometry that could be used. The following were some ideas:

a. Instead of randomly picking a height for the place location, use a level scheme, set level heights then generate objects for each level randomly.

b. Consider rectangular volumes, random X and Y, fixed Z to level heights with or without rotation around the Z axis at the midpoint.

c. Consider triangular volumes, random 3 XY points, create as extruded polyline, Z set to level height, cover polyline with 3– faces with or without rotation around the Z axis at the midpoint.

d. Consider random 4 points, created as extruded polyline, set Z to level height.

e. Consider random vertical planes, Z set to level height, with or without Z rotation, set minimum and maximum for plane length.

The discussion then moved to creating possible volumes which were not part of any 3D CAD primitive set. Polyhedra were investigated, (Pugh 1976) and (Mortenson 1990). Figure 33 describes a series of polyhedra that were produced. Each is a surface model controlled by a set of dimensions which determine
the sizes of each exterior and interior face.

Using the previous concepts of random sizing, location, and rotation a few preliminary models were created. Figure 34 displays one with random locations and size, Figure 35, 36, and 37 display models with rotation around the Z axis, X and Y axis, and X, Y, and Z axis.

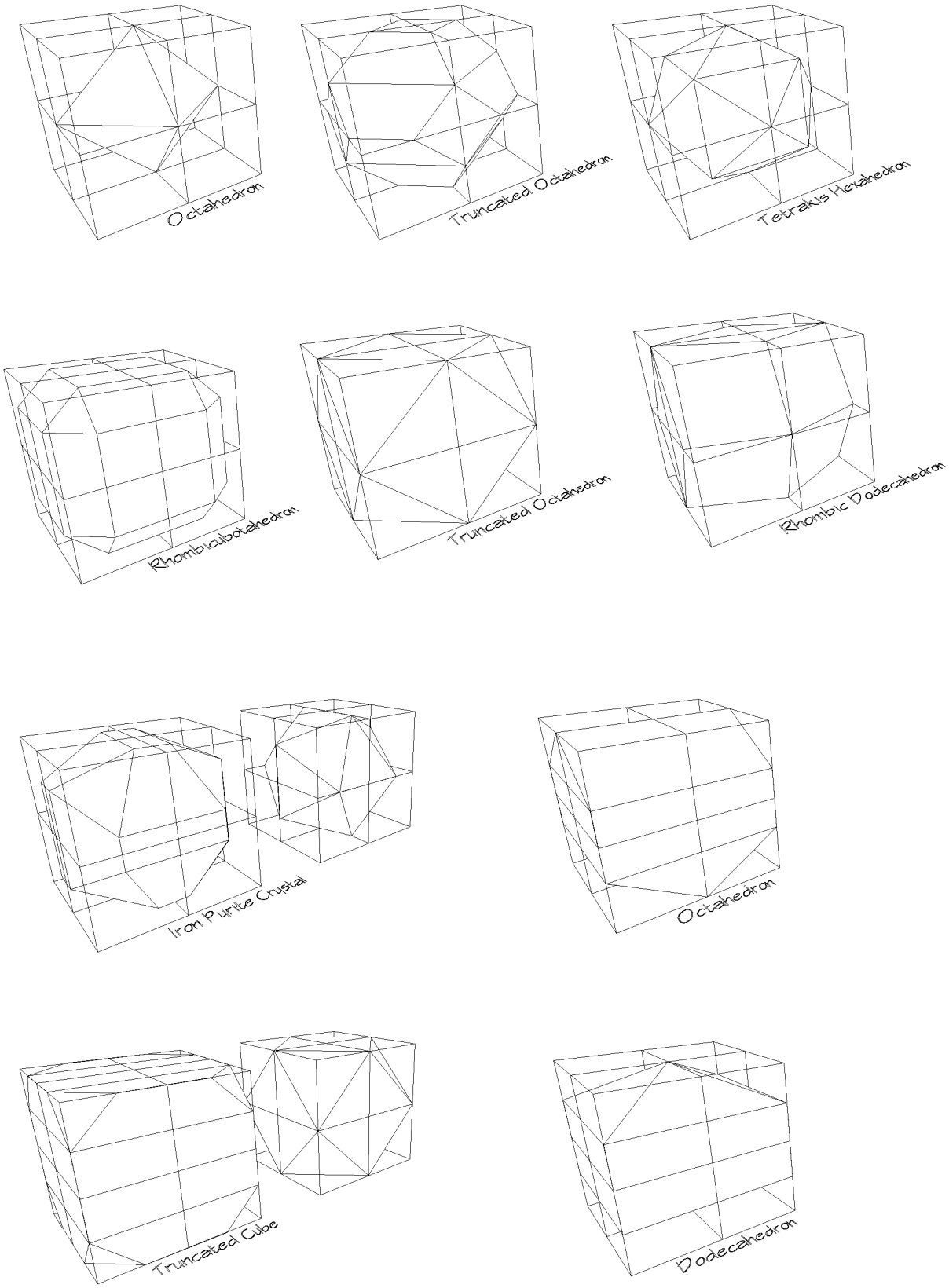

Figure 33 - Examples of polyhedra

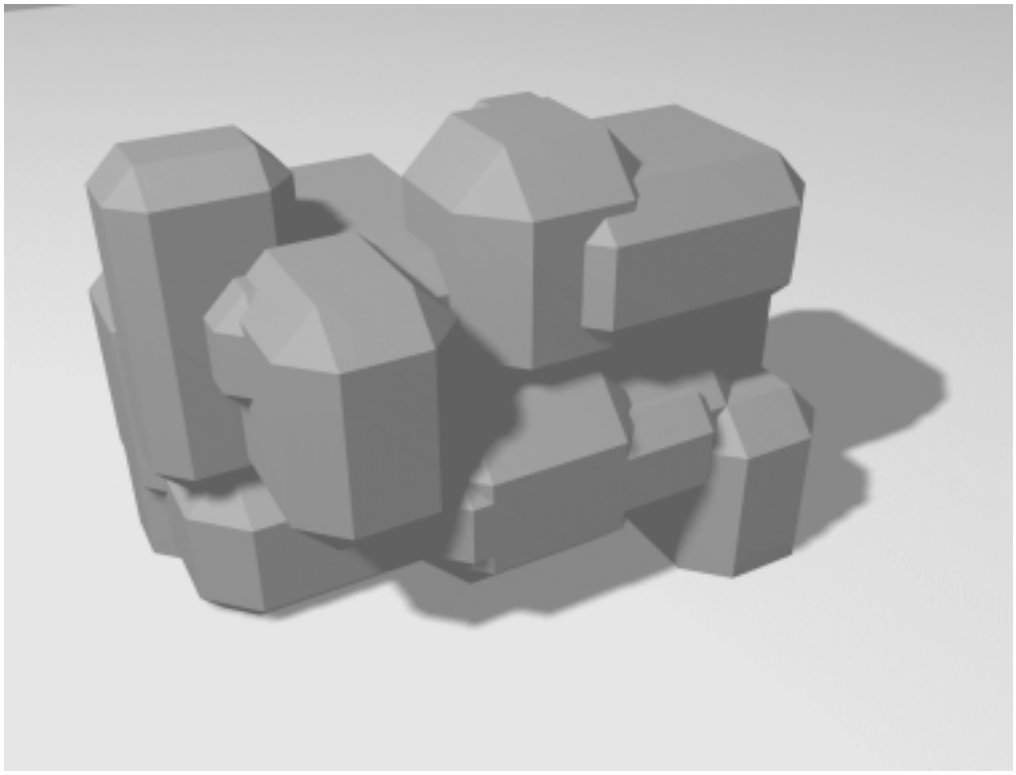

Figure 34 - Random location and size

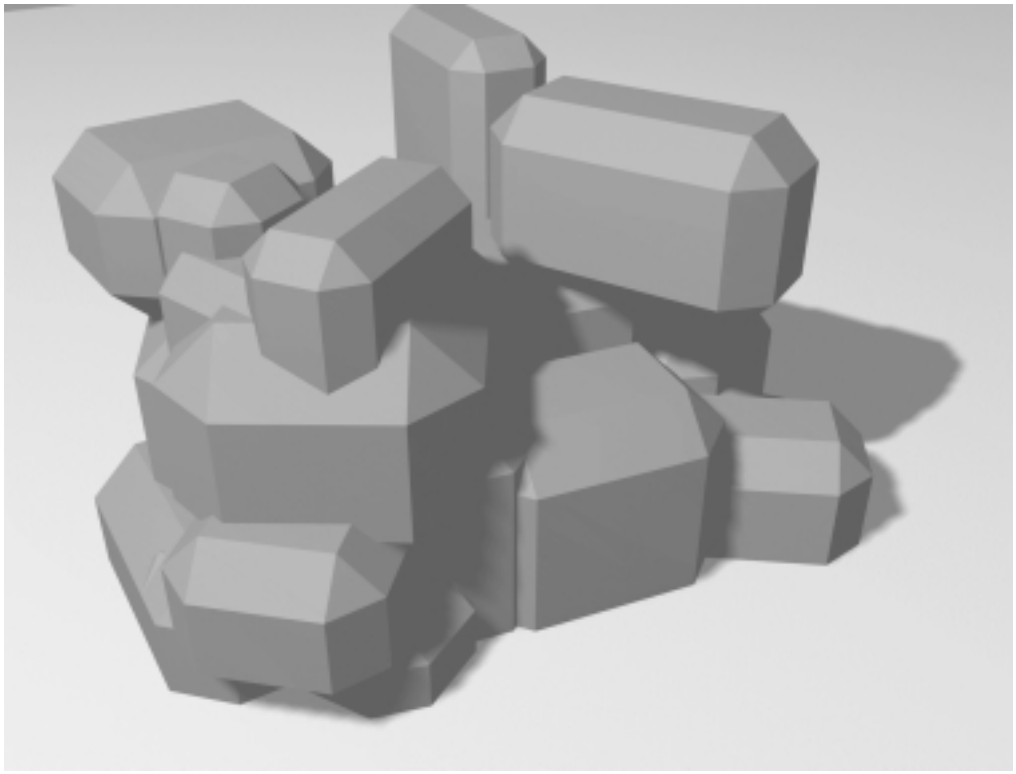

Figure 35 - Rotation Z axis

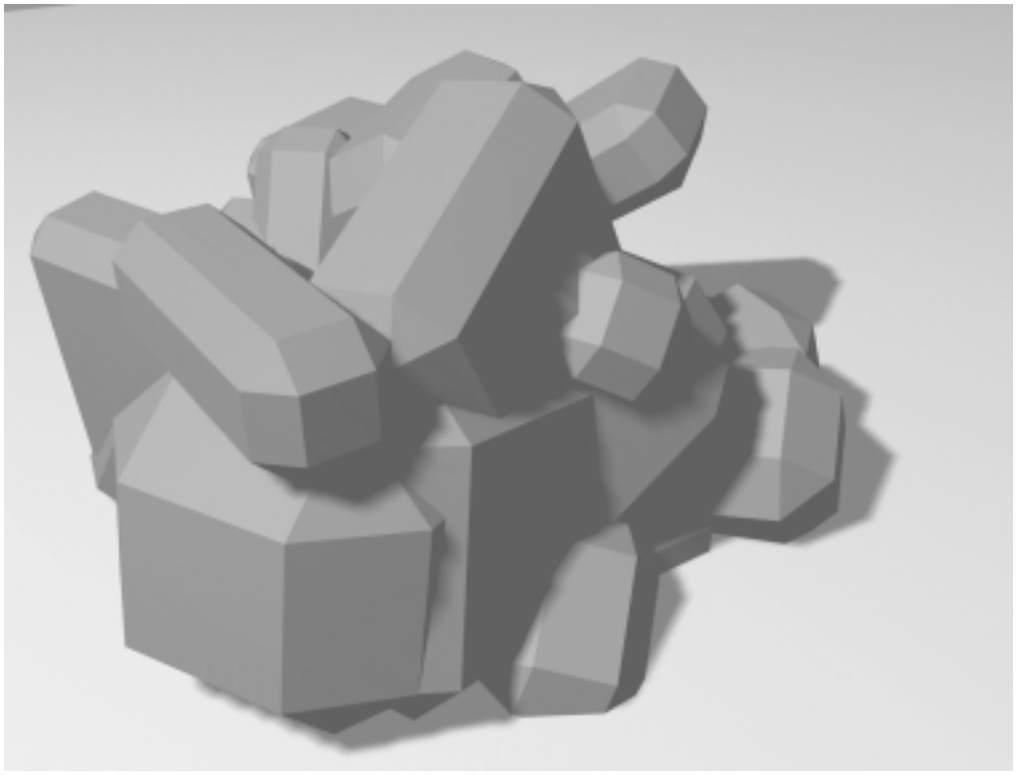

Figure 36 - Rotation X and Y axis

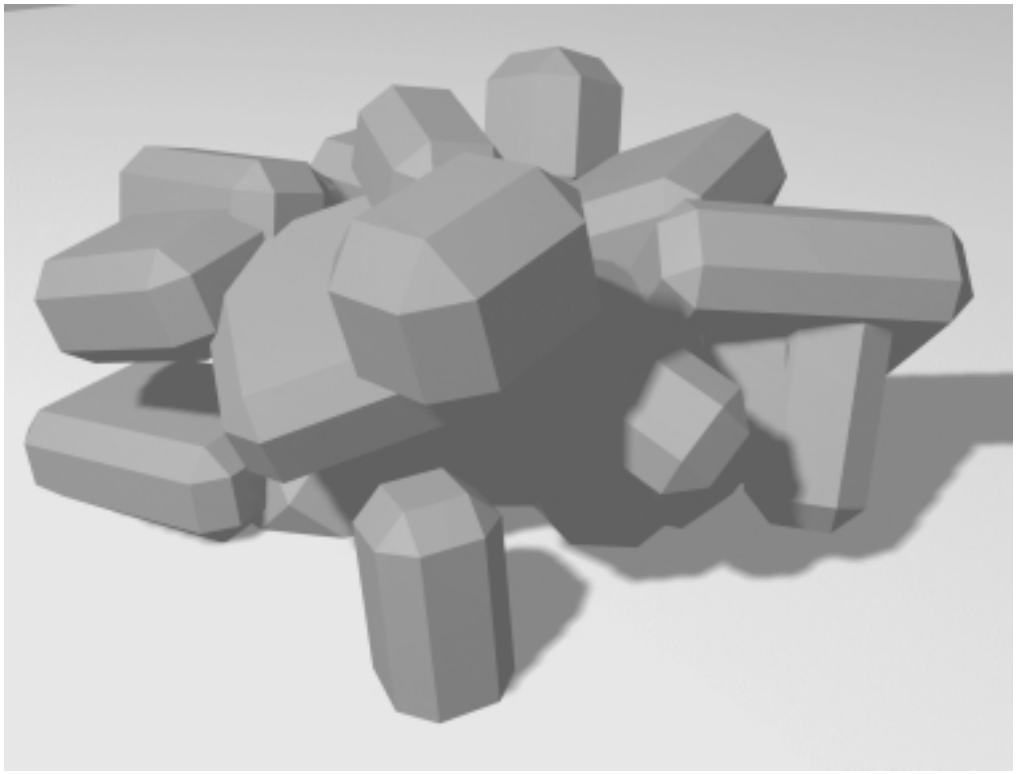

Figure 37 - Rotation X, Y, and Z axis

CONCLUSION

The process of developing these forms was the most important aspect of these exercises. The step-by-step procedure, the evaluation on each as we encountered them, 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 really made the student think about how they approached each variation.

The discussions at the end of the course clearly indicated to me that the students now consider the development of programs as their own personal expression of an idea, 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.

ACKNOWLEDGMENTS

Special thanks to the following students who developed examples and encouraged me to develop these ideas: Samir Abdelmawla, Amel Abdulla, Christian Badea, Paul Docka, Alphonso Peluso, and Catherine Zuercher. Additional thanks to Samir Abdelmawla for preparing the final format of the paper and the reviewers for their comments.

REFERENCES

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

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

Mortenson, Michael. 1990. *Computer Graphics Handbook*. Industrial Press, Inc.

Pugh, Anthony. 1976. *Polyhedra, A Visual Approach*. University of California Press