Lofting 3D Shapes

Madeup is a new and exciting 3D modeling software that provides a programmatic approach towards creating geometrically intense objects. It aims to be robust, flexible, and accessible for young minds. For instance, the extent of Madeup's accessibility can be demonstrated by its use in elementary classroom environments as a tool to provide a visual representation for basic programming concepts. Beyond the educational use, Madeup's programmatic approach allows the user to generate geometrically complex shapes that would be arduous using other methods. Such methods traditionally consist of dragging around nodes in 3D space. To fully grasp the differences between these models, it is best to explore how a computer interprets and stores these 3D objects.

On the most fundamental level, the computer understands 3D objects as cluster of connected triangles. In fact, the first half of the popular OBJ file format simply consists of a list of vertices (as x, y, z - coordinates), and the second half of the file follows with a list of triangular faces (as indices into the vertex list). In theory, it is possible to construct OBJ file for a torus within a text editor, but this would be a tremendous waste of time. In Madeup, on the other hand, the task of creating a torus, along with an extensive list of other shapes, can be completed using a few lines of code and some basic geometric intuition.

For this research project, we are designing a built in function into Madeup titled *loft* that provides another method to solidify objects. The goal of *loft* is best described in an inquisitive manner outlined in the paragraph that follows.

Imagine a triangle and a pentagon of similar size are plane parallel and not coplanar. How would we connect the vertices of the triangle and the pentagon to create a closed 3D shape? The issue here is that the choice isn't obvious, especially for a computer, for these two shapes differ in vertice cardinality. Furthermore, as the problem is generalized allowing various shapes, sizes, orientations, and nonplanar paths, choices for these vertice connections become increasingly less obvious. This is the exact problem the *loft* function aims to solve. In order to complete this task of generating a 3D object from an arbitrary contour, we must remove all ambiguity by defining the aesthetically desired properties of a general 3D shape. Creating these definitions lends itself to some interesting questions that blur the line between intrinsic beauty and mathematics.