# Chapter 18

# Three Dimensions

## 18.1 Three-Dimensional Drawing

Three-dimensional illustration programs are rather like their two-dimensional counterparts except that the range of tools and options is necessarily broader.

A low-end program like Dimensions employs a limited suite of strategies:

1. Fixed suite of "primitives": cubes, spheres, cones, ellipsoids
2. Extrusion of 2D shape perpendicular to the drawing plane
3. Rotation of 2D shape about an axis
4. Bevel library

Ray Dream Studio offers a wider range of options:

1. "Primitives": (i) cubes, spheres, cones, ellipsoids
   (ii) exotics such as Fog, Fountain and Fire
   (iii) formula-objects [user-entered mathematical formula in terms of parameters $p_1, ..., p_4$, which are then controlled by sliders to allow one to rapidly adjust the shape.
2. Free form modeller: extrusion and rotation of 2D shapes
3. Editable extrusion curves
4. Scaled extrusion: a cylinder that is wide at one end and narrow at the other , for example.
5. Morphing between cross-sections so that an extruded object may have a circular cross-section at one end and a square or elliptical cross-section at another
6. Mesh modeller: network of vertices connected by segments

The "primitives" are the shapes that can be drawn by clicking on a tool. Matlab is a powerful alternative to drawing programs because Matlab can draw "primitives" that are defined by very complicated mathematical expressions.

Raydream Studio's "formula" primitives are an acknowledgment of the need for user-defined tools. The bad news is that the expression-parser in Studio is rather simple, and cannot for example deal with Bessel functions or anything more exotic. However, the parser does recognize $p_1, \ldots, p_4$ as parameters to the curves, which must otherwise be expressed as polynomials, trigonemetric functions, exponentials or algebraic

functions in the Cartesian coordinates $(x, y, z)$. One can then vary these parameters by moving sliders to see how the shape changes when these parameters are varied.

For complicated objects that arise in a scientific project, Studio can be combined with Matlab or Mpale. After a complicated shape has been computed in Matlab or Maple or Mathematica, one can do a polynomial curve-fit or the like to obtain a formula that can be used to recreate the shape in Studio as a "formula primitive".

Studio also supports some rather exotic primitives such as "Fog", "Fire" and "Fountain". These are not terribly useful in science, but are reminders that the simple primitives of most illustration programs are inadequate to create any but the simplest shapes. Illustrator allows the creation of exotic objects through clip art libraries.

A skilled artist can draw three-dimensional shapes in a two-dimensional program like Illustrator. One can, for example, create a template of isometric lines to make it easier to draw shapes in perspective, then import the grid pattern into an illustration as a separate layer, and finally discard the layer from the final drawing.

However, three-dimensional drawing programs rather strongly discourage this sort of direct three-dimensional drawing. The reason is that to be able to rotate, scale and manipulate a shape in three dimensions as a vector graphics object, the program must know the three-dimensional location of each vertex of a polygon (or whatever). An Illustrator drawing of a cube-in-perspective may look fine to the eye, but the program knows only how the figure projects onto the two-dimensional screen surface, not its true three-dimensional shape.

Instead, tactic is draw a two-dimensional shape which is then "lifted" into three dimensions by EXTRUSION and/or ROTATION. The word "extrusion" is borrowed from manufacturing where objects are often fashioned by forcing a viscous liquid through a die that forces the object to have the cross-sectional shape of the die when it hardens. One can also "extrude" by cutting many identical copies of a two-dimensional shape out of paper or cardboard; when the copies are stacked and glued together, the result is an object of finite thickness with the cross-section of the two-dimensional sheets from whence it came (Fig. 18.1.

"Rotation" is called "lathing" in Studio. Whatever the name, a solid-of-revolution can be made by rotating a two-dimensional cross-section about an axis. Such solids are

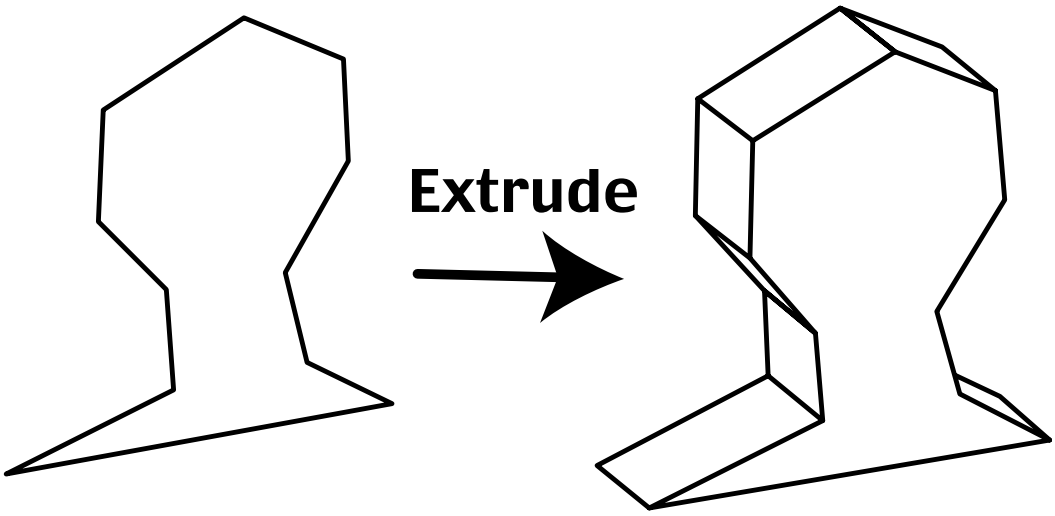


Figure 18.1: Left: two-dimensional shape. Right: the three-dimensional shape created by extruding the two-dimensional shape. The extruded figure can then be appropriately shaded or covered with a texture map.

manufactured by putting a block of wood on a lathe where the block is turned rapidly at high speed, and then pressing a metal chisel into the wood as it rotates. Intricate patterns can be cut by pressing the chisel different depths at different points along the shaft of the lathe, but the result is always axisymmetric in cylindrical coordinates (Fig. 18.2).

Extrusion and lathing are rather limited ways to make three-dimensional figures, so additional strategies are needed. A "bevel" in carpentry is a rounding off of a sharp corner. Dimensions employs a whole library of bevel shapes so that the corners of extruded objects can be rounded in different patterns.

Studio allows very complicated extrusion or "sweep" paths. One can, for example, specify that the cross-section will shrink as the extrusion proceeds so as to obtain a cylinder which is wide at one end and narrow at the other. One can specify smooth variations between different beginning and ending cross-sections so that a shape can "morph" between a circle and a square. The axis of the area swept by the cross-section can be drawn on the sides of the three-dimensional grid box to inform the program that the sweep path should twist, zigzag or spiral. Such complicated extrusion paths would be almost impossible with a physical die-and-plastic, but are easy in the computer.

The other option is draw vertices directly in three dimensions — in such a way that the computer knows the true three-dimensional location of each point — and then connect the vertices by lines to make polygons. The magic is that one must specify a "drawing plane" when moving the mouse to specify a new vertex. By moving the mouse a certain distance along one drawing plane, changing the plane, and moving again, one can reach any point in $(x, y, z)$ space. Because the move is split into two steps, rather than a single move as in drawing isometrically in two dimensions, the program knows the three-dimensional location of the vertex.

The Mesh Modeller in Studio is a vertex-based method for creating three-dimensional objects. Of course, drawing each vertex for a complicated object is very tiring and boring, so Mesh Modeller allows one to begin with primitives (such as cubes, spheres,

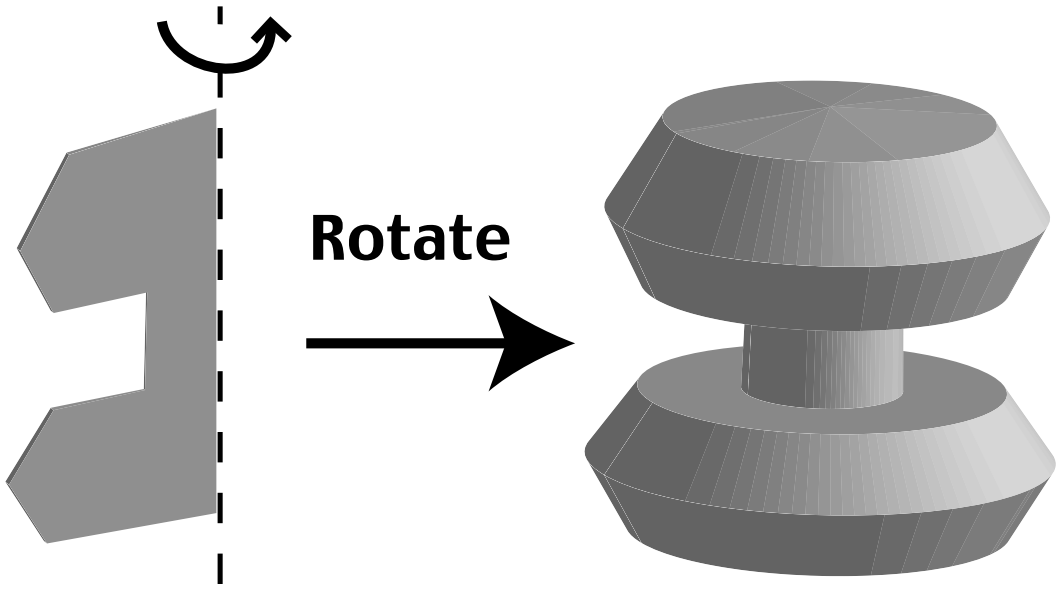


Figure 18.2: Left: two-dimensional shape. Right: the three-dimensional shape created by rotating the two-dimensional shape about the vertical axis indicated by the dashed line.

etc.), fuse different shapes together, and apply the extrusion and lathing concepts. The "Loft" command stretches shapes between two existing polygons to make a new, united complex surface.

Drawing programs are much better than Matlab at aligning, rotating, scaling, and shading 3D objects. They are much inferior to Matlab at creating objects which are the outcome of a complicated scientific algorithm or numerical experiment. As in two dimensions, drawing and scientific graphics software are complementary with partially overlapping but also partially distinct regions of usefulness.

## 18.2 Three-Dimensional Plots

### 18.2.1 Isosurfaces

The three-dimensional generalization of a contour plot is an isosurface graph, that is, a visualization of a two-dimensional surface in a three-dimensional space where $q(x,y,z) = q_0$ where $q_0$ is a constant. Matlab's **isosurface** command will generate such plots, and most high-end scientific graphics packages can do the same.

An ordinary contour plot is viewed from above so that many isolines can be seen simultaneously in the $x-y$ plane. Unfortunately, the only way to see the entire $x-y-z$ volume, and thus see a lot of nested isosurfaces, would be to view the plot from the four dimension.

Consequently, it is usually possible to view only a SINGLE isosurface at a time, which is very annoying. Possible remedies include:

1. Multi-panel graph, each showing the isosurface for a different level $q_0$
2. Slide show animation, such as a Quicktime image sequence, allowing the viewer to switch rapidly between different isosurfaces so as to "fly in" to an object.
3. Cutaway of one isosurface to reveal part of another.
4. Transparency.

Another difficulty is that a given view reveals only one face of the isosurface while the other is hidden like the Dark Side of the Moon. Again, a two-panel graph, showing front and back, or a Quicktime or VRML movie that allows one to rotate around the isosurface at the click of a slider, can help.

### 18.2.2 Isocaps

If an isosurface is, for example, an infinitely long cylinder, then the surface will be clipped by the edges of the viewing box. The isosurface then appears hollow with holes at the two caps where the surface touches the edge of the viewing box.

Table 18.1: Three-Dimensional Equivalents of Two-Dimensional Plots

| 2D plot | 3D equivalent |
|---|---|
| contour (isoline) | isosurface, isosurface with caps<br>contour slices |
| pseudocolor | mini-cubes<br>[pseudocolor] slices |
| quiver | coneplot |

These holes are rather distracting, and furthermore represent areas of non-information. It is therefore common to fill the holes or "isosurface caps" with contour plots or pseudocolor plots of the same quantity where these flat, two-dimensional graphs are parallel to the edges of the viewing box, and thus plug the holes completely.

These "isocap" plots do give an indication of interior isosurfaces, and therefore add information to the isosurface itself. Furthermore, the eye seems a closed volume, bounded at the ends by the isocaps, rather than a hollow surface, which tends to draw the eye out of the viewing box. Matlab has an **isocaps** command which adds caps to isosurfaces in a single line.

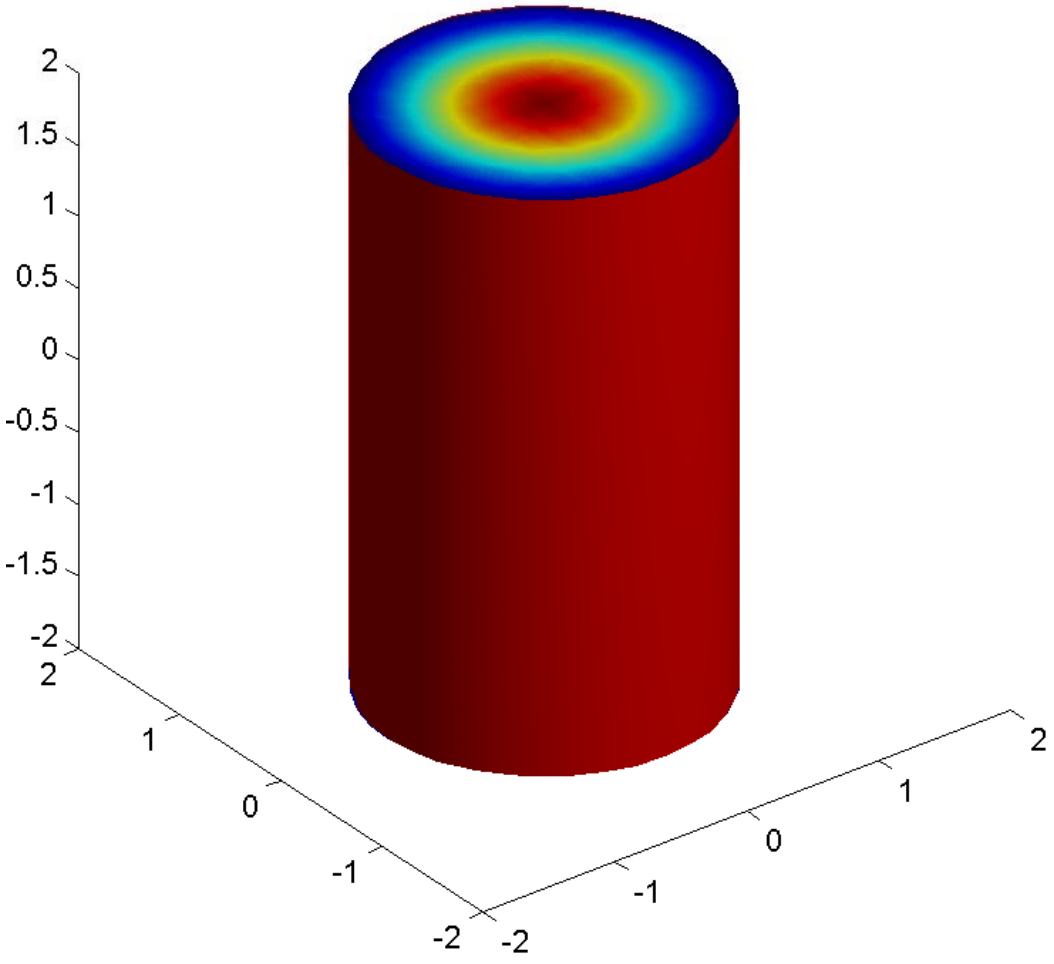


Figure 18.3: A cylindrical isosurface (red) with pseudocolor caps. The isosurface is $q = 1/4$ where $q = exp(-x^2 - y^2)$.

### 18.2.3 Glyphs, Arrows and Cones

To go beyond isosurfaces, one needs to make use of glyphs, which are little icons whose shapes, size and/or color conveys additional information about the fields at the location of the glyph. The most familiar glyph is the arrow which is used in an ordinary two-dimensional vector or quiver plot. The length of the arrow is magnitude of the vector while the orientation of the arrow indicates the direction of the vector field.

Unfortunately, three-dimensional arrow plots, though an option **quiver3** in Matlab, are not very effective. The problem is that an arrow is a two-dimensional shape, and it is difficult to perceive the three-dimensionality from a sea of arrows, even with a surrouding three-dimensional grid box to provide perceptual clues. It is therefore common to replace the arrow glyphs by cones. Matlab 6 has a coneplot option. It seems to be easier to visually decode the vector field from the pattern of cones, which are three-dimensional shapes, than from arrows.

Hydrodynamicists have been especially clever in drawing more elaborate glyphs to simultaneously convey vector and scalar information. A simple instance is to use color on a code to visualize the pressure. However, much more elaborate glyph-schemes have been developed.

Matlab allows a lot of creativity because one has a subroutine to draw a certain glyph in a unit box, one can then call this function from a glyph-plotting function that will lay down many copies of the glyph, each with the orientation, shape, color and size appropriate to that point in 3D space. The difficulty is that novel glyphs force the viewer to learn a new visual code or paradigm. This means that elaborate glyphs will only succeed in conveying useful information to viewers who are strongly motivated, and will to invest a lot of time learning the new visual metaphor.

### 18.2.4 Mini-cubes

The direct analogue of a pseudocolor plot would be to color each volume (or "voxel", in computer graphics jargon) with a color appropriate to $q(x, y, z)$. Unfortunately, in the absence of transparency, only the colors of the outer wall would be visible.

One remedy is to render a lot of non-continguous small cubes in color instead. If the gaps between the cubes are sufficiently large and the cube is shown in the proper perspective, one can get a feeling for how $q(x, y, z)$ varies over the whole volume. Since the cubes must be rather small to allow sufficient viewing channels into the interior, this type of graph is called a "mini-cubes" plot.

### 18.2.5 Slices

Another variation is to visualize three-dimensions through a stack of oriented, flat two-dimensional slices, spaced so that one can view each member of the stack. Matlab uses the terminology of "slice" plots, but some others prefer "cut-planes".

Matlab allows the slices to have an arbitrary orientation; the cuts need not parallel a pair of coordinate axes. Matlab offers two routines, **contourslice** and **slice**. The former does what the name implies: it draws a fairly standard Matlab contour plot on the slice plane. The unusual feature is that the output of **contourslice** is a fully three-dimensional object, so it can be rotated and viewed at an angle. (In this respect, it is really a **contour3** plot computed along the slice plane.)

The **slice** routine is really a pseudocolor plot of the function on the slice plane.

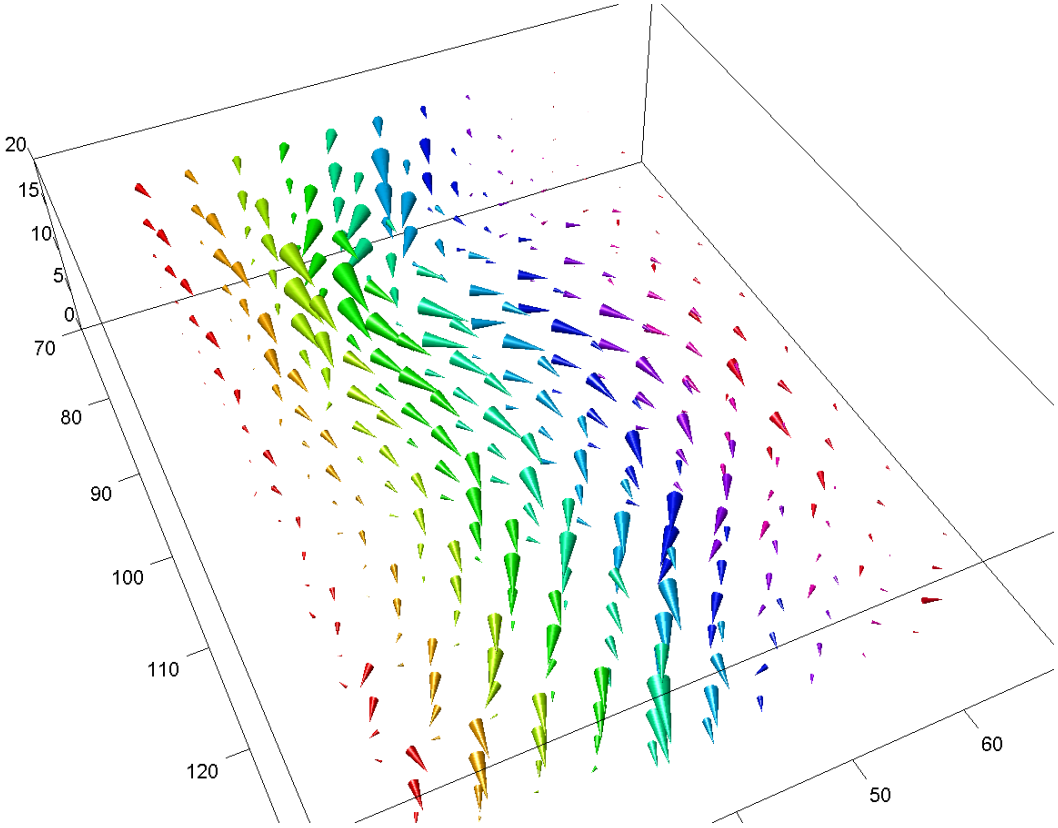


Figure 18.4: A cone plot of a three-dimensional wind field.

## 18.3 Topology

In a fluid flow, certain "critical points" have place-of-honor. These include stagnation points, where the flow is zero, and the centers of vortices. Other linear features, such as the streamline between fields of two vortices and shocks or fronts, are important, too. One strategy for wrestling with three dimensions is to plot only these topologically-crucial points, curves, and surfaces rather than attempting to visualize the entire surface.

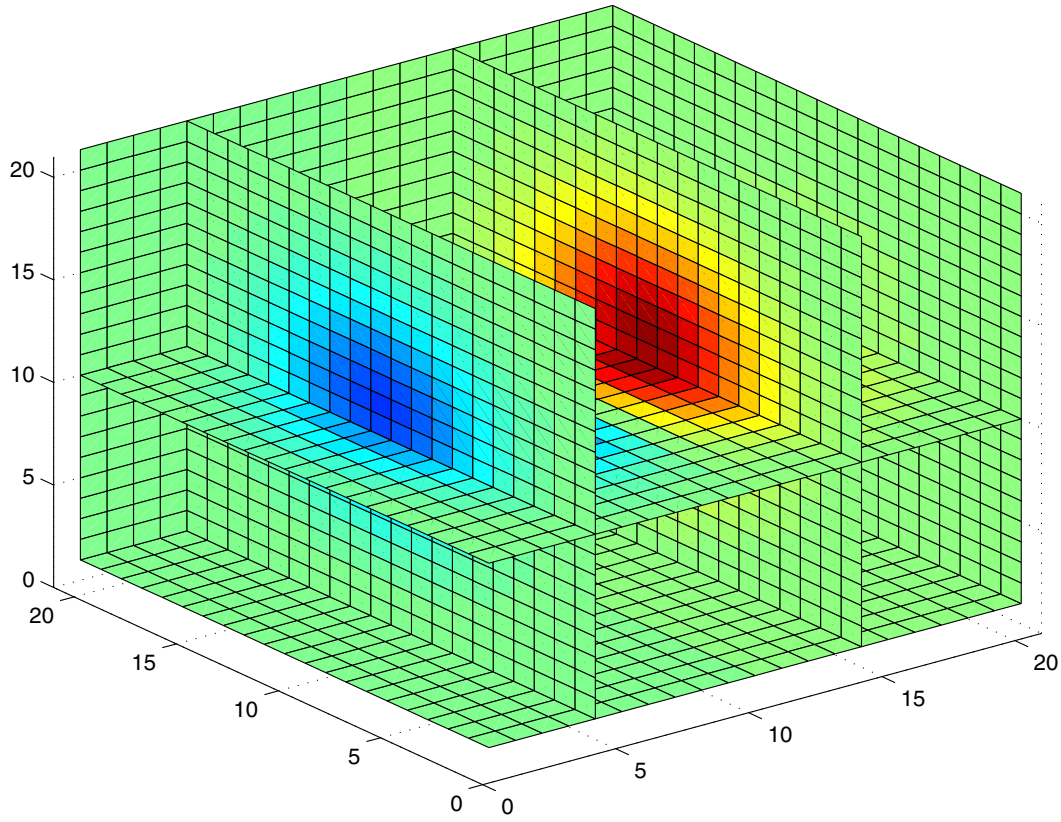


Figure 18.5: A slice plot of the function $q(x, y, z) = exp(-x^2 - y^2 - z^2)$.