1. Overview

GM - A Geometric Modeling Program

1. Overview

This appendix describes the geometric modeling program GM that is on the companion CD for this book. The GM program should be considered as a starting point for a modeling program or a test bed for trying out geometric algorithms and in no way as an end product. It was created to provide the reader with an easy to use and easy to extend environment. The program comes with some basic functionality for creating and modifying geometric shapes. On the other hand, it is also organized in such a way that the reader can extend the code in new directions. Only minimal knowledge of Windows programming and C++ is required (unless of course the user wants to get fancy). Hopefully, the reader will have become interested enough in one or more of the topics that we covered to want to implement their associated algorithms. Only a brief overview of GM is presented here. For more details one will have to consult the documentation on the CD that comes with the program.

2. Overview of User Interface

Figure 1 shows the basic layout of the screen when the GM program is started. We are in 3-dimensional perspective mode with the camera at (10,10,10) looking towards the origin of the world coordinate system that is displayed as a small frame with unique markings for the x-, y-, and z-axes. The main menus appear along the right hand side of the screen. It is convenient to think of the rest of the screen as being divided into several areas.

Figure 1 The main screen
GM - A Geometric Modeling Program

The TITLE line: This shows the icon and name of the GM program along with the standard three Windows buttons on the extreme right.

The STATUS line: This is the line immediately below the TITLE line. It shows such things as the name of the current object and the current frame (as reference coordinate system). See Figure 2.

The GRAPHICS area: This is the black area immediately below the STATUS line where the display takes place. It has a title line at the top which specifies the view mode.

The GRAPHICS menu: There are two parts to this menu, the VERTICAL GRAPHICS menu along the right side of the GRAPHICS area shown in Figure 3 and the HORIZONTAL GRAPHICS menu on the line immediately below the GRAPHICS area shown in Figure 4.

The IO line: This is the area between the HORIZONTAL GRAPHICS menu and the HINT line at the very bottom of the screen. Here is where the typical program/user interaction would take place. For example, if choosing a menu item for creating a box, this is where the query for the dimensions of the box would appear. Pop-up menus are placed in the menu area. Alternatively, one could have the menus pop up below items listed along a line at the top of the screen as is done in programs following the standard Windows interface. There are advantages and disadvantages with either scheme. The advantage with the former is that one can see the entire graphics area and the current menu at all times. The advantage with the latter is that one has more room for the graphics area but it will take two keystrokes or mouse clicks by the user to activate commands and part of the graphics area would get obscured all the time.

Figure 2 The items on the STATUS line above the graphics area

Figure 3 The choices of the GRAPHICS menu to the right of the graphics area

Figure 4 The GRAPHICS menu and scroll bar below graphics area
2. Overview of User Interface

![The MAIN menu](image)

**Figure 5** The MAIN menu

**The HINT line:** This is the line on the bottom of the screen where hints are shown. System messages may also appear on this line.

A close-up of the main menu is shown in Figure 5. Clicking on a menu item will either pop up another menu with its heading below the heading of the previous menu or carry out an action that may involve interacting with the user on the IO line. For example, clicking on the Create item would pop up the Create menu shown in Figure 6(a). Clicking on the Block item to create a block of the form $[0,a] \times [0,b] \times [0,c]$ centered at the origin will prompt the user on the IO line with the message

Enter dimensions of block: $x, y, z$;

as shown in Figure 6(b). In the figure the user has specified $a = 5$, $b = 4$, and $c = 3$. There would be another prompt

Enter block name:

![Create menu](image)

(a)

![Create block](image)

(b)

**Figure 6** Creating a block
because every object that is created has a name so that it can be referenced. After the name has been entered the block would appear as shown in Figure 7. Switching to orthographic mode by clicking on the N button in the VERTICAL GRAPHICS menu would lead to the view shown in Figure 8.
2. Overview of User Interface

To exit GM one needs to click on the standard Windows exit button at the top right-hand corner of the screen. If the world is nonempty, then the user will be asked to reconfirm his/her desire to quit before the program actually exits.

3. The Shape Tree

GM is a program which allows users to represent geometric objects, specifically, a world of objects. The fundamental data structure type in GM which represents these primitives, the geometric objects, is called a “shape.” The world of objects becomes a tree of nodes, each one of which corresponds to a named shape. From the programming point of view, a shape may or may not be “in” the “world”, which is the tree of shapes that the GM program has been told to keep track of and basically corresponds to those shapes that are displayed on the screen. The point is that a shape does not automatically get added to the world. The programmer has to do this explicitly. We shall use the word “object” to denote a shape which is IN the WORLD. Users of the GM program are typically only concerned with objects.

The different types of shapes that are possible in GM are structured in the form of a tree. The top level part of this tree is shown in Figure 9. Shapes are first differentiated by dimension. 0-dimensional shapes are obviously discrete and are either lists of points or lists of labeled points. 1- and 2-dimensional shapes are separated into discrete and or parametric ones (with one special class of robotic-related shapes). In the 1-dimensional case we also distinguish between single curves and multiline/polyline shapes (lists of line segments or polygonal curves). An example of a multiline shape is a frame like the coordinate system shown in Figure 1 (it is a list of line segments).

Discrete shapes are shapes that are completely defined by a finite set of points and their relationships. For example, a discrete surface has points and a data structure that specifies the facets (lists of edges) and edges (list of two points). Parametric shapes are defined by parametrizations that are given either explicitly, as in the case of splines, or implicitly, as in the case of a circle. The actual curve and surface types that are implemented in GM are discussed in Sections 4 and 5.

In order for GM to support a tree of objects, object names are path names similar to the path names for files. For example, an object X named “box/top/center” would refer to an object which is a child of an object named “box/top”, which in turn is a child of an object named “box.” Since all objects are children of a global dummy object named “world”, one could also reference X with the name “world/box/top/center”; however, the prefix “world” is optional.

Note: All items created from the menus in GM are made children of the current object which initially is the “world.” This means that users of GM do not have to worry about path names.

![Figure 9 The top levels of the shape tree](image-url)
4. The Implemented Curves

Figure 10 shows the currently implemented curve tree.

The discrete curves are the polygonal ones. It was convenient to have two special subcases.

The discrete circle: This is a polygonal curve that represents a circle. The main reason for this type is that one can create versions of the GM program that omit some of its functionality. Specifically, one can create a version that does not contain any parametric curves or surface types. In such a case the discrete circle type comes in very handy. The motivation for leaving out some code in the compilation process is that it might not be needed for the task at hand and would therefore produce a smaller and perhaps faster program.

The parametrized polygonal curve: Here, we associate to a polygonal curve the natural parametrization induced from the standard parametrizations of its segments. This curve type was useful for animation and the robot module.

There are three parametric curve types. First, there is the ellipse with its special subtype of circle. Next, there are the parametric spline a rational B-spline curves, both of which are modeled on the corresponding curves defined in IGES.

5. The Implemented Surfaces

Figure 11 shows the discrete surfaces that have been implemented. The only that is not obvious is the one referred to as “general.” This is a general discrete surface type in the sense that it is any surface defined by a facet/edge/vertex structure. It is intended for a person programming for the GM program and not a user of the program.

Figure 12 shows the currently implemented parametric surface tree. Most of the types in the figure
5. The Implemented Surfaces

Figure 12 The parametric surface types

should be fairly self-explanatory. Generalized cylinders are surfaces obtained by sweeping a segment along a curve. The parametric spline a rational B-spline surfaces are, like the curves, modeled after the corresponding objects defined in IGES.

In order to make the GM program a little more interesting, we have added a robotics module. A user can create a robot and then have it move along curves. The forward kinematics needed to achieve this has been implemented. Although a user can only create complete robots, a programmer would use the robot shape types shown in Figure 13. Currently, robots only consist of discrete surface parts, but it would be easy to extend this and allow parametric surfaces to get round shapes for the robot’s limbs.

6. The Graphics Menu

The GRAPHICS menu along the bottom and right side of the GRAPHICS area are available to the user at all times.

First, consider the items in Figures 3.

The S button: Pressing it will print a query on the IO line that will let the user change between wireframe, constant shading, or smooth shading mode. The default mode is wireframe mode as shown in Figure 7.

The V button: Pressing it will print a query on the IO line that will let the user change between having a one or two view display. The default is to display one view in either perspective or orthographic mode as shown in Figures 7 and 8. Switching to the two view mode in that example would split the graphics area into two windows. The top window would have a perspective view of the world and the bottom, an orthographic view. See Figure 14.

The N button: Pressing it will toggle between perspective and orthographic projection mode. The default mode is perspective mode.

Figure 13 The robot-related types
The ∧ and ∨ buttons: These are really a shrunk scroll bar and control zoom.

The scroll bar below the zoom buttons controls spinning the camera about its focal point in the vertical direction.

Next, consider the items in Figure 4.

The VwAll button: This will position the camera so that all objects in the world are visible and centered on the screen. The orientation of the original camera is preserved and the camera is only translated appropriately.

The VwObj button: This allows the user to pick an object and the view will then be changed via a translation of the camera so that the picked object appears at the center of the screen.

The Point button: This lets the user pick a point on the graphics area with the mouse that will then become the new center of the screen.

The StdVw button: This resets the view to the standard view where one is in perspective mode and the camera is at (10,10,10) looking towards the origin with the y-axis as the “up” direction.

The Color button: This lets the user change the color of an object.

The H/Shw button: This lets the user pick an object whose visibility will be toggled.

The Remove button: This lets the user pick an object that will then be removed from the world.

Finally, the scroll bar on the right controls spinning the camera about its focal point in the horizontal direction.
7. The Main Menu

We give a brief description of the main menu items.

**The Create menu:** Bringing up this menu presents a large list of shapes that can be created. These include polygons, circles, spline curves, blocks, spheres, cylinders, cones, facets, surfaces of revolution, extrusions, and spline surfaces. One can also make copies of shapes and read shapes from a text file whose format is described in the documentation on the CD.

**The Move menu:** In this menu one can translate or rotate the current object interactively with a mouse with respect to the current frame. One can also specify transformations more precisely by entering translation vectors or rotations on the IO line. For example, Figure 15 shows the user specifying a rotation about the z-axis. One sees a circle in the xy-plane and the user dragging a dotted line to specify the angle which will be the angle that this dotted line makes with the x-axis.

**The Edit menu:** The items in this menu let the user attach textures to objects, affix objects to one another (so that moving any one in the group will move them all), or change the resolution or domain of parametric objects.

**The Save menu:** This gives the user the choice of saving a single object or the whole world to a binary file.

**The Restore item:** Clicking this menu item will display a list of all the binary files that were saved previously in the Save menu. If the chosen file was an object, then it will be restored into the world. If it was a saved world, then the saved world will replace the current one.

![Figure 15](image.png)  
*Figure 15* Rotating an object about the current frame
GM - A Geometric Modeling Program

The **Utils menu**: The items in this menu include letting the user write an object to a text file or toggle the coordinate axes visibility.

The **Animate menu**: In this menu the user can define some simple animations. For example, the user can have objects move along paths. The motions will take place in the background while the user continues to perform other tasks. One can start and stop the motions.

The **Robots menu**: The actions in this menu are an application of the animation and forward kinematics capabilities of the GM program. The user can define one or more robots that will move along paths. One can specify paths, speeds, and start and stop the motions.

The **Examples menu**: The only item in this menu is one that will display the well-known teapot from the University of Utah.

The **Render menu**: The purpose of this menu is to let the user change various aspects of the rendering process. For example, one can toggle whether lighting computations are performed, toggle whether lights are local or located at infinity, toggle whether or not lights are used, define new lights and their properties, and change the material properties of objects.

The **Tests menu**: The purpose of this menu is to be the place for test code. Some currently included items are tests to see if picking shapes or surfaces works.

### 8. Miscellaneous

The general philosophy of the GM program is that entering any sort of input to it can be aborted by pressing either the ESC key or the right button of the mouse. The effect of pressing certain other keys depends on the mode of the active window.

**Perspective mode**: Pressing the left or right arrow keys will pan the picture in the corresponding direction along the x-axis. Pressing the up or down arrows will move the camera along its z-axis or view direction. Pressing the END or PGDN keys will rotate the camera in the xz-plane. Pressing the HOME or PGUP keys will rotate the camera in the zy-plane.

**Orthographic mode**: Pressing the left, right, up, or down arrow keys on the keyboard will pan the picture in the corresponding direction.