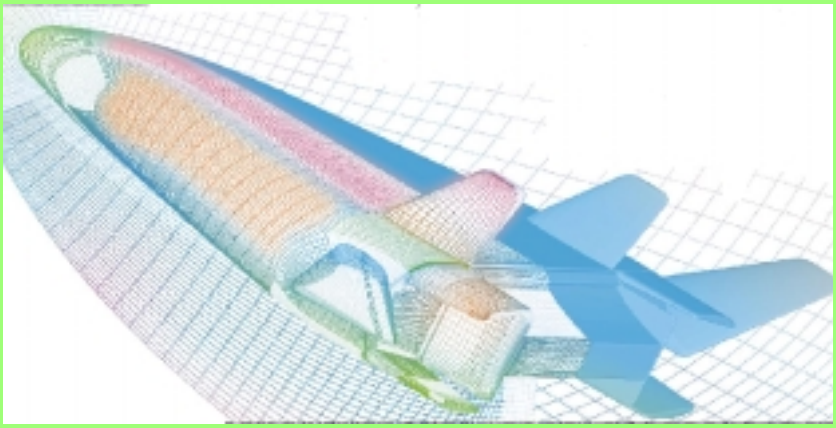
GRIDPRO

THE CFD LINK TO DESIGN

The AZ Graphic Manager Manual

April 26, 2000



Program Development Corporation
300 Hamilton Ave., Suite 409
White Plains, NY 10601

Telephone: (914) 761-1732
Fax: (914) 761-1735
E-Mail: gridpro@gridpro.com

Table of Contents

PART 1 THE RUDIMENTS NECESSITIES

PART 2 BEYOND THE BASICS

PART 1

THE RUDIMENTS

Chapter 1 - Where to Begin

1.1 Introduction

GridPro is unique; it minimizes your work and maximizes your efficiency. Relying heavily on automation and lightly on user input, GridPro generates multi-block, structured grids with relative ease and quickness. You input surfaces and a wireframe topology; GridPro does the rest.

GridPro provides a huge pardigm shift in grid generation technology. With GridPro's top-down approach, users no longer need to concern themselves with massive amounts of detailed, manually inserted data. Rather, they let the program work out the spatial relationships. The mathematical underpinnings that reduce the user effort also account for GridPro's substantially high grid quality. This quality comes in the form of near orthogonality, smoothness, low warpage, and curvature clustering for both concave and convex boundaries.

GridPro is an "automatic" grid generator, which accesses an automatic process earlier than any other "automatic" grid generator. Included in GridPro's automation are such tasks as surface grid generation, zone construction, and the intersection between surfaces.

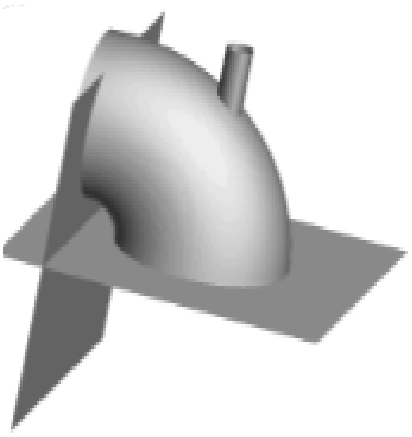
GridPro essentially reduces grid generation to topology generation. The primary user input is a pattern of points referred to as the grid topology. The language that organizes this pattern of points is called "Topology Input Language" (TIL). You can input these points directly in the AZ graphic manager. As you input points in AZ graphic manager, TIL code is automatically generated. Users have choices of using graphic manager and/or TIL code.

GridPro can be used for multiple applications in multiple ways. The program will grid simple cases or complex cases, in two or three dimensions. By producing grids on objects as diverse as a nuclear test missile or a human heart, GridPro showcases its flexibility as well as its depth. Obtaining realistic configurations is an essential industrial need for engineering analysis. These configurations are often complex, and normally require a large amount of human time to produce a grid. GridPro's automated processes handles massive complexity that requires a large number of blocks very efficiently (with a large number of blocks). It can take hundreds or thousands of blocks. There is basically no limit in number of blocks.

GridPro takes the grunt work out of grid generation. The program has features which enable you to perform a variety of basic and complex procedures without having to manually write TIL (Topology Input Language) code with its associated minutiae and rules. Benefit of TIL code, on the other hand, is the versatility of being able to handle all possible scenarios.

Being the only program boasting a fully compiled language, GridPro provides you with the option of working in the TIL code or the AZ (automatic zoning) Graphic Manager. The AZ Graphic Manager provides an interactive environment for the gridding process, an environment that allows the user to see and manipulate the object being gridded. Also, the AZ Graphic Manager is the underlying system that enables GridPro to transform topologies into grids. The resulting grids are structured multi-block grids. In this manual, the effective usage of the Az Graphic Manager is thoroughly presented.

Figure 1.1

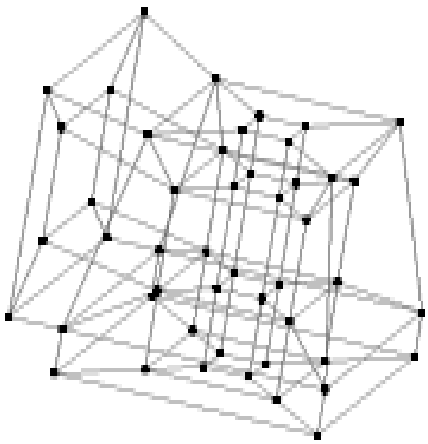


A collection of surfaces bounding a region to be gridded

Your ultimate goal is to perform engineering or scientific analysis using the grids that you have generated. Depending on what analysis you wish to perform, you can vary the quality of your grid at specified points in the gridded region. This manual covers the steps leading up to preparing a grid for analysis: inputting surfaces, placing a topological wireframe about these surfaces, assigning parts of the wireframe to these surfaces, gridding the resulting topology, and viewing the resulting grid. After generation of your grid, you can easily observe cross sections in order to determine if any areas need further refinement. Figure 1.1 illustrates the first step in the gridding process, which is to define a region to be gridded. The figure shows the boundary surfaces.

GridPro contains built-in surfaces, which are ellipsoids and planes, that although simple, are helpful in many practical circumstances. However, because real-life objects rarely mimic symmetrical regions, you cannot always use GridPro's built-in surfaces. When you become sufficiently adept at GridPro, you will want and be able to either create or modify arbitrary surfaces using the program's facilities. You may also want to import surfaces from other design programs such as CAD (Computer Aided Design).

Figure 1.2

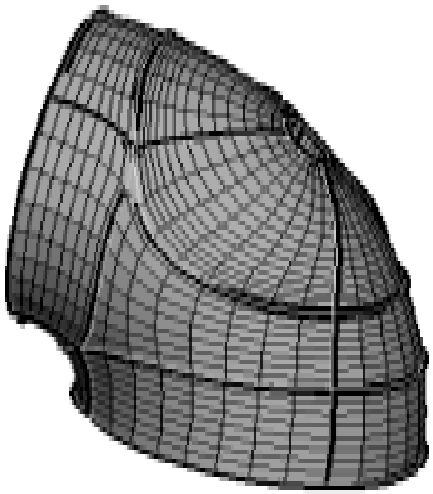


A wireframe for the region enclosed by the surfaces in figure 1.1

After inputting surfaces, whether or not they are built-in from GridPro or from CAD, you create a topology for the region bounded by those surfaces. Topology generation contains two parts: (*a*) wireframe construction, and surface assignments. A topological framework (or wireframe topology - see figure 1.2) is properly designed according to the multi-block scheme Gridpro utilizes. Upon inspection, you will note that the wireframe defines a pattern of grid points. To design a wireframe correctly, you first define your wireframe through the creation of a coarse unstructured quadrilateral (2D) or hexahe-

dral (3D) grid about the region. Then, establish an association between your wireframe and your surfaces by assigning appropriate wireframe corners to these surfaces. Much of this manual is geared toward teaching you how to establish properly topologies for various regions. As with anything else, we will first work on constructing topologies for simple cases, then we will tackle more complicated ones later.

Figure 1.3



The grid (shell view) for the region in figure 1.1

The manual discusses the various graphical actions which are performed to: (1) generate topology interactively: (2) generate grids automatically using that topology: (3) view the generated grids. Automatic grid generation results from interactively or manually generated topology code (TIL code). The topology codes are written to TIL code files. These files can be created by either interactive means or manual text edit, and can be reused, edited, or combined. Grid generation takes place when the user applies Ggrid, a transformation process, to the topology code file (with extension .fra). The interactive launch of Ggrid is performed with a default .TIL file unless the user wishes to name it differently. The grid in figure 1.3 is obtained from the surfaces in figure 1.1 and the topology in figure 1.2. The grid viewing is done in a very flexible manner, allowing for users to view images comprising solids and/or sheets. Trimming and clipping help for the display of solids and sheets. In addition to learning how to utilize various interactive process, being able to choose good topologies is an important part of using GridPro, and will be addressed later in this manual.

The manual is organized to teach you methodically how to use GridPro, and is split up into three parts. The first part describes the essential operations and features of the program, including the drawing area, the cut plane, topology construction, grid generation, and grid viewing. The second part expands on the previously presented concepts, and introduces features such as topological groups, grid slices, surface geometries, surface repair, and grid property assignment.

The third part includes useful appendices. By using the “Menu Navigation” appendix and the “Summary of Operations” appendix, you can find the location and function of each operation. The “Guidelines for Building GridPro Topology” appendix broadly but systematically describes the steps involved in gridding a region from start to finish. This appendix also delineates mistakes frequently made by the user. The “Frequently Asked Questions” appendix provides answers to common problems. The last appendix, “Incorporating User-Defined Solver Formats,” deals with post-processing and the appendix applies to contents in Chapter 9. As the last appendix, “Summary of Operations is attached.

The manual uses the following conventions:

i) sideways carrots, < >, for keyboard keys.
ii) square brackets, [ ], for buttons.
iii) single quotes, ‘ ’, for file names:
iv) double quotes, “ ”, for anything else.

Due to the introductory nature of this manual, only basic skills and basic techniques will be presented. If you are interested in particular case examples, please refer to GridPro’s Case Book. Some objects, such as rotationally symmetric turbines, and some operations, such as assigning boundary conditions to a surface, will not be covered in the manual. GridPro *can* perform such operations and manipulate such objects.

In order to develop the intuition necessary to run GridPro effectively, we will focus on examples. Therefore, in addition to containing tutorial documentation, this manual will guide you through several, simple example problems step by step. Since the visual result of a procedure often enhances understanding of the process in hand, we will provide ample figures as we go through examples.

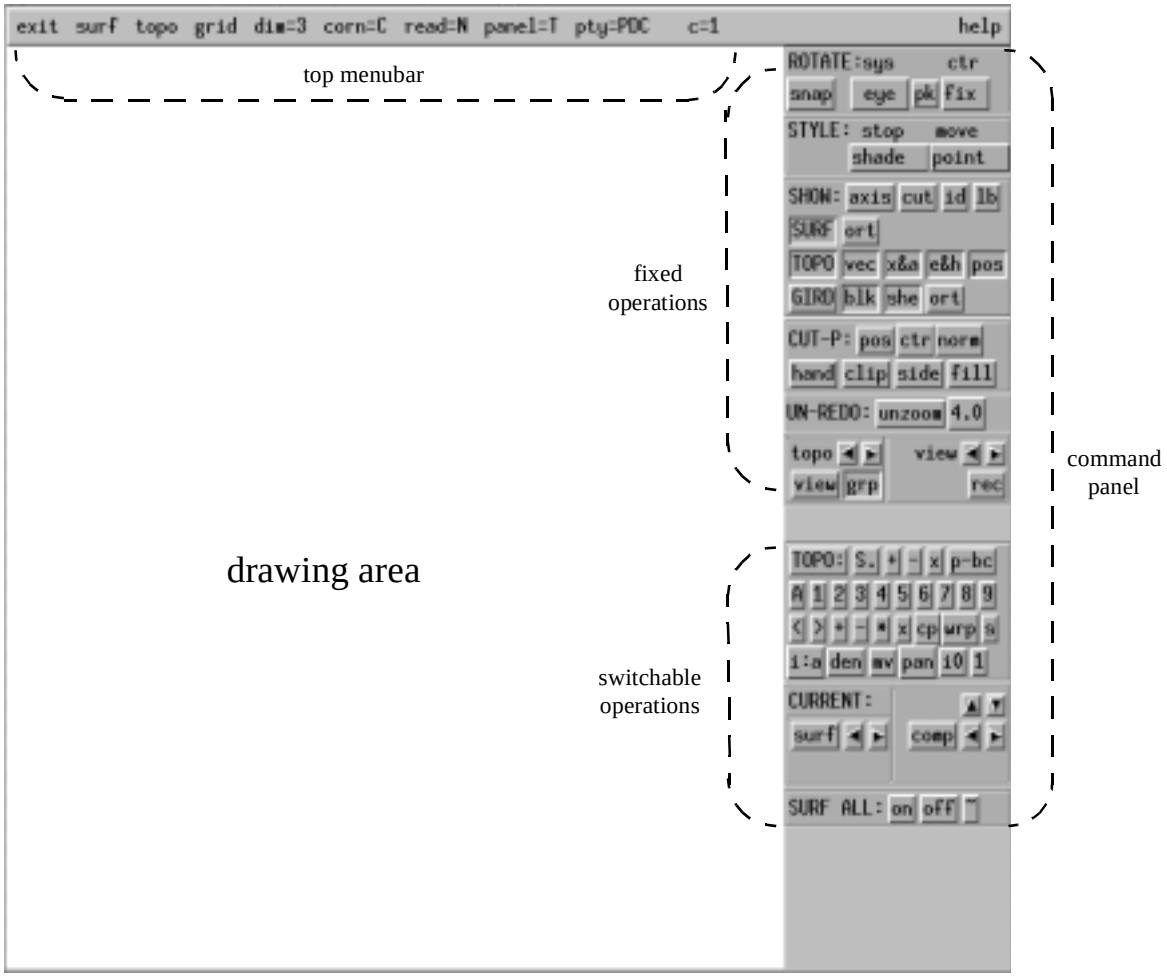
In addition to explanatory prose, this manual contains both example problems and exercises to enhance your understanding. The manual supposes that you have a computer beside you and are able to experiment with an operation or a set of operations as you read about them. It is highly advisable to work on these exercises in order to reinforce your knowledge.

1.2 Menu Format

The menu format for GridPro is user-friendly. The main window consists of a drawing area, a top menubar, and a command panel. The drawing area displays all surfaces, grids, and topology. You perform actions using the top menubar or the operations on the right.

The menus activated from the top menubar put you in different modes. You will be working mostly with the menus dealing with surfaces, topologies, and grids. The “panel=T” menu allows you to toggle among the “Topology Builder,” “Grid Viewer,” “mini CAD,” and “Property Setter” panels (which are modes). The expression, “panel=T”, is an abbreviation for “Topology Builder.” In the “Topology Builder” panel, you can input surfaces and place topologies about them; in the “Grid Viewer” (“panel=G”) panel, you can view and manipulate a gridded object. The “mini CAD” (“panel=S”) panel contains operations which allow you to prepare surfaces for use on GridPro. The “Property Setter” (“panel=P”) panel allows you to assign properties to grids. All modes share the operations found in the upper half of the command panel (the “fixed” operations) and each mode contains unique operations in the lower half (the “switchable” operations). Each half is separated by a large blue bar. GridPro’s window format is illustrated in figure 1.5.

Figure 1.5



Each operation in the command panel has at most one layer of subcommands. Because of the highly automated nature of Gridpro, each operation on GridPro performs functions which would require more button-pushing on most other grid generation programs. Additionally, all the operations for a specific mode are displayed at the same time to maximize efficient application by the knowledgeable user.

Depending on the mode you select from the “panel=T” menu, you will choose to use other menus from the top menubar. The “surf” menu allows you to perform various operations relating just to surfaces. The “grid” menu allows you to perform various operations relating just to generated grids. The “topo” menu contains operations allowing you to manipulate topology and operations relating to the gridding process. Specific operations located within the “surf,” “grid,” and “topo” menus will be explained later in the manual.

Other important basic menus include the “dim” and “help” menus. The “dim=2” or

“dim=3” menu simply places you in the 2-dimensional or 3-dimensional mode. The “help” menu accesses a pulldown menu with three options. The “long help” option opens a dialogue box. This box explains, in detail, the function of whatever operation to which the mouse is pointing. The “short help” option allows you to see a quick description of an operation, and the “no help” option turns off the help feature. The “help” operation comes handy if you prefer to understand the function of the operation without referring to this manual.

We will be back for The “corn=c,” “read=n,” “pty=PDC” and “c=0” menus later.

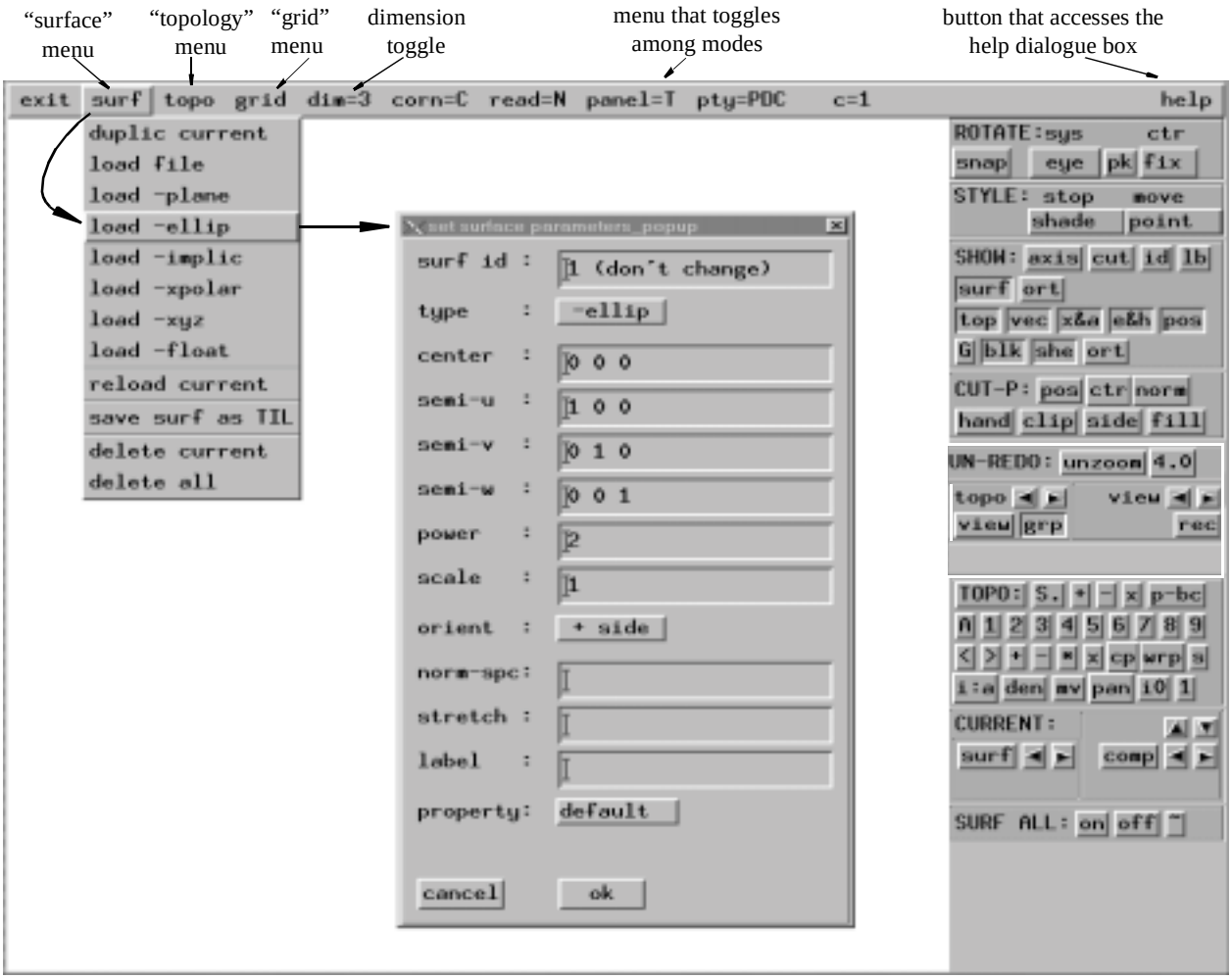
1.3 Getting Started

Like any other computer program, GridPro has a set of defaults. For instance, when the “Topology Builder” panel is accessed, operations allowing you to manipulate topology are available. In general, because of the program design, there is no need to consider the defaults. However, be aware that in the course of your use, you might inadvertently toggle a default. In order to start with the appropriate features, make sure the [cut] and [axis] buttons are not depressed. In the brown “STYLE” box, select “shade” under “stop” and “point” under “move”. In the green “ROTATE” box, select “eye” in the “sys” menu and “origin” in the “ctr” menu.

If you are using GridPro on a PC, you might need to “jiggle” the drawing area after performing an operation to see the results on the screen. In order to “jiggle” the screen, perform a slight translation or rotation (use a mouse button to slightly drag the cursor).

In order to start utilizing, or just even playing around with, the program, you should have a workable surface to manipulate. The superellipsoid is a representative surface with which to work. To see such a surface after you create it, access the “Topology Builder” panel or else you will have to turn on explicitly the display with the [surf] button under the green “SHOW” box. Therefore, click with the left mouse button on the “panel=T” menu, then move the cursor down to the “Topology Builder” selection and click again. Because you desire to create a superellipsoid instead of a superellipse, set the dimension mode to “dim=3”. To create a superellipsoid, access the “surf” menu, then select “load –ellip.” A parameters box will appear. In this box, you can specify the parameters of a superellipsoid. Note that each surface contains a unique parameters box. The sequence of subwindows that appear, as well as the locations of menus, are found in figure 1.6.

Figure 1.6



The first item in the "load –ellip" box is the surface id (identification) number of the object. Every surface created has a numeric identity. Surface ids are useful when you must manipulate many objects, and need to distinguish one object from another. You cannot change the id number of a surface. For now, ignore the "type" option.

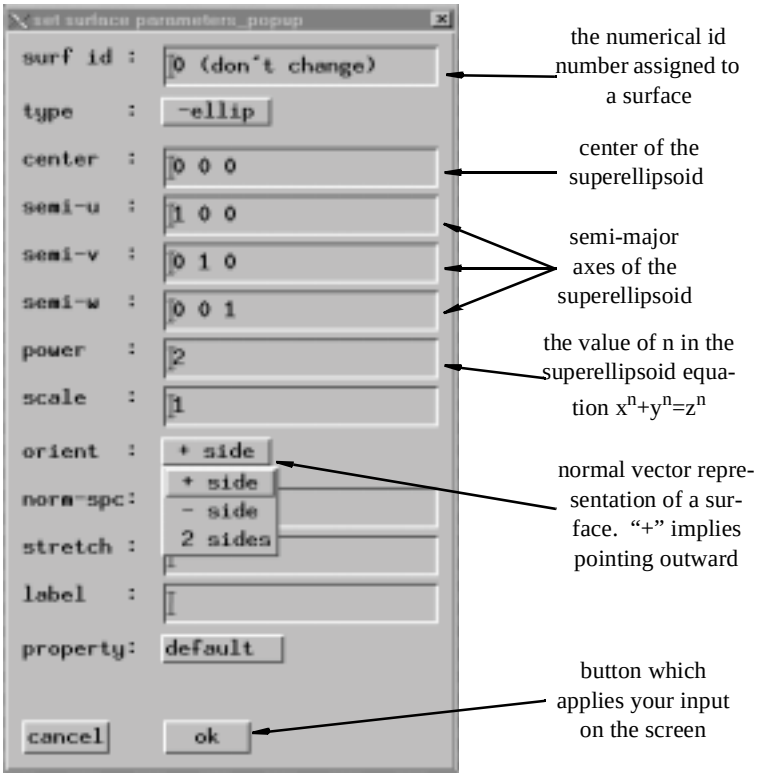
The next four items, "center," "semi-u," "semi-v," and "semi-w," define the shape and position of the superellipsoid using (x, y, z) coordinates. You can input whatever coordinates you wish. If you do not want the superellipsoid to undergo a transformation, let the semi-u axis correspond to the x-axis, with coordinates (x, 0, 0) for some fixed value of x. Also, let the semi-v axis correspond to the y-axis with coordinates (0, y, 0) for a fixed y, and let the semi-w axis correspond to the z-axis with coordinates (0, 0, z) for a fixed z. If you are just learning about GridPro, you might want to keep (simple) the properties of the surfaces you create as simple as possible. For example, to create an ellipsoid with semi-axial lengths of 1, 2 and 3 respectively along the x, y, and z axes, the u, v, and w semi-axes are given by (1, 0, 0), (0, 2, 0), and (0, 0, 3) respectively.

The “power” option defines the relation that provide the shape of the built-in surface and the “orientation” option defines direction of the normal vector to the surface. An ellipsoid is defined by a second order function which render a shape of an oval. A superellipsoid is defined by a higher order function. The simplest analogies in cartesian coordinate system are $f(x, y, z) = 0$,where $f = x^2 + y^2 + z^2 - 1$ for an ellipsoid and $f = x^n + y^n + z^n - 1$ with $n > 2$ for a superellipsoid.

You can alter the n value by changing the number next to “power.” By increasing the surface’s power from its default setting of 2, you are making the resulting superellipsoid look more like a rectangular prism or, in other words, a brick. When you press the [+side] button next to the “orient” heading, a submenu with three items appears. The items are “+ side,” “- side,” and “2 sides.” Each item dictates the direction of all vectors normal to the surface. The + direction is outward and the – direction is inward. The orientation of a surface determines the region to be gridded; that is, either in the region outside or inside the surface. This will be discussed in details later.

Also, ignore for now the “scale,” “norm-spc,” “label,” “stretch” and “property” options; in order to implement your specifications, press [ok]. You must press the [ok] button in order to have your surface created. On the screen, you should see a superellipsoid. If you do not, then perhaps you have positioned your surface too far away from the viewing window; press the [unzoom] button, which is located in the UN-REDO region. To adjust the factor to which you unzoom, click on the [4.0] button found to the right of the [unzoom] button. Then, you can choose a different factor from the resulting pop-up menu. Unzooming repeatedly by a large factor enables you to thrust outward quickly from a microscopic boundary region. The locations of the operations in the “load -ellip” parameters box are illustrated in figure 1.7.

Figure 1.7



Now that you have a surface on the screen, you can translate, rotate, and dilate it using only the mouse. If you click anywhere on the screen using the left mouse button and drag either an object or the space around it, the object will move on the screen just as the mouse moves on the mousepad. Just as the left mouse button is used for translation, the right mouse button is used for zooming in to a portion of the screen in order to dilate that portion for a more detailed view. By clicking and dragging the right mouse button, you form a white zooming rectangle. The point where you click is one vertex of this zooming rectangle. As you drag the mouse, the rectangle expands so that the line between the mouse's original position and current position is the diagonal of the rectangle. When you release the button, the screen zooms in on the specified rectangle automatically. The right mouse button enables you only to zoom in; in order to zoom out, press the [unzoom] button. If you have started to drag the zooming rectangle and then decide to eliminate it, make the rectangle as small as possible. Most likely, the rectangle will then disappear.

The middle mouse button is used for rotation. Rotation can be about the origin, an axis, the center of mass of an object, the center of mass of many objects, or any arbitrary point in space.

EXERCISE #1:

Select the 3-d mode and create any superellipsoid. Rotate and zoom in on different parts of the surface. In other words, play around with the surface and learn to manipulate the mouse effectively.

In order to relate relative positions of surfaces, you can access orthogonal x, y, and z-axes by depressing the [axis] button found in the green "SHOW" box. The axes can be of help to and necessary for some applications.

If you desire to work in 2 dimensions, you should start from scratch. Because you work in the default z = 0 plane in 2 dimensions, do not start a 2-dimensional project if you have been

working in 3 dimensions. In other words, by working in 3 dimensions, you might have changed the defaults necessary for manipulating surfaces in 2 dimensions.

Although knowing how to manipulate 3-dimensional objects is important, learning how to utilize 2-dimensional objects will improve your ability to use GridPro. The best representative 2-dimensional surface to work with is the superellipse. Accessed in the same manner as the super-ellipsoid, a superellipse is inputted by setting the length of the semi-w axis to a very large number (remember, you are working in the plane, z=0). For example, a representative superellipse, the unit circle, would have semi-u axis (1, 0, 0), semi-v axis (0, 1, 0), and semi-w axis (0, 0, 10000). After establishing a 2-dimensional surface, you best select "line" and "point" respectively under "stop" and "move" in the brown "STYLE" box.

EXERCISE #2:

Select the 2-d mode and create any superellipsoid. Rotate and zoom in on different parts of the surface.

Chapter 2 - Introduction to Visual Features

2.1 Rotation, Translation, and Zooming

Because you can learn how to use GridPro best through example, we shall start off the chapter with an example problem. As we do the problem, we will learn about current surfaces, rotation systems, relative sets of axes, and orthogonal relations.

EXAMPLE PROBLEM #1:

Create two identical superellipsoids that do not intersect. Manipulate the surfaces such that the top of the screen cuts into the sea blue surface while the bottom of the screen cuts into the dark blue surface. Then, manipulate the superellipsoids such that one volume completely covers the other. Switch the positions of the volumes.

First, because we are dealing with superellipsoids, we make sure we are in the 3-dimensional mode (dim=3). Then, because we want to start with a blank screen, we clear the screen of all surfaces previously created. To perform this operation, we open the "surf" menu and click "delete all." We then click [ok] on the subsequent confirm message. Also, in order to begin with the necessary defaults, we should make sure "shade" is under stop and "point" is under move in the brown "STYLE" box. The [cut] and [axis] buttons should not be depressed, and we should be in "Topology Builder" mode (accessed through the "panel=T" menu). In the green "ROTATE" box, "rotate by eye system" should be selected under "sys" and "fixed at origin" should be selected under "ctr." Now, we must create two superellipsoids that do not intersect. Therefore, despite the size of the surfaces' axes, the surfaces should be sufficiently far apart. Also, for the surfaces to be superellipsoids, the "power" must be greater than 2.

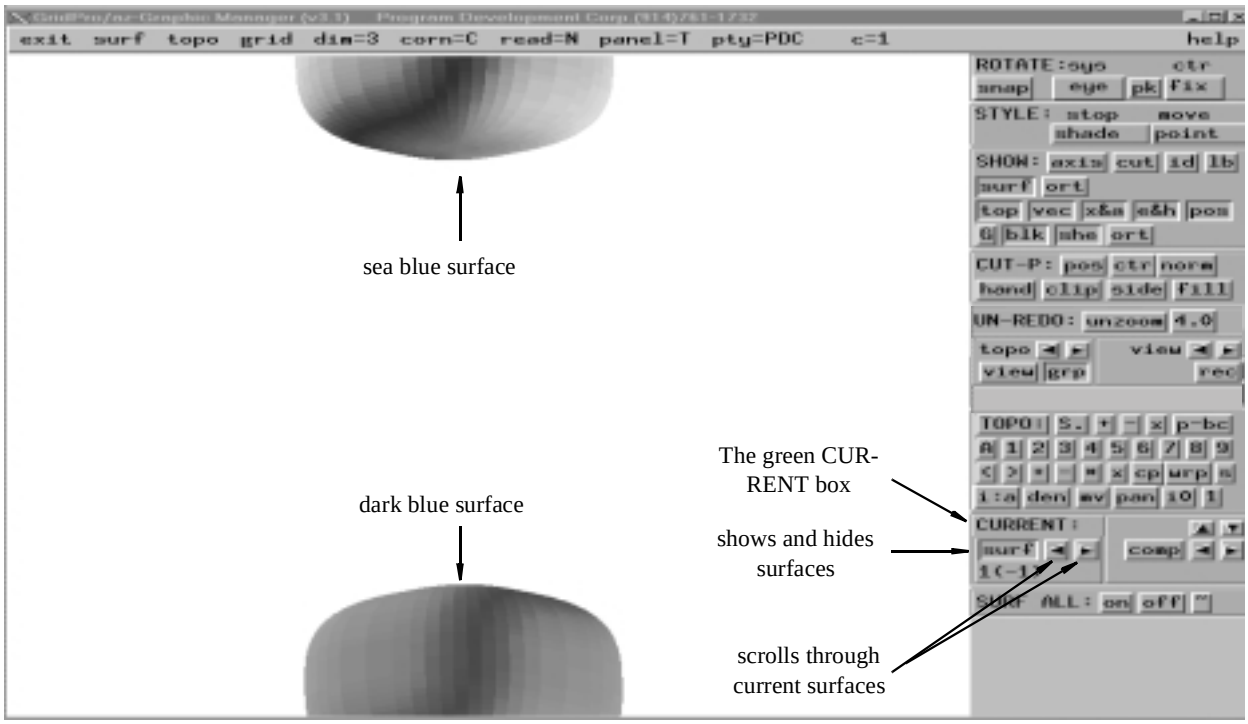
We click "load –ellip" under the "surf" menu, and proceed to set the ellipsoid parameters. Because rotation features allow us to look along one axis, the centers of both superellipsoids should be on the same axis (because we want to look at them on the screen such that one completely covers the other). With this rationale in mind, let us type in our current center's surface as (5, 0, 0), and leave the "semi-u", "semi-v", and "semi-w" axes at their default settings. Also, let us change the number in the "power" box from 2 to 3. Ignoring the "orient" box, we click [ok]. If the superellipsoid appears small, we use the right mouse key and zoom in on it. If the superellipsoid does not appear or is too big, we press, [unzoom]. Because the surface you created is offset from the origin, you will most likely have to unzoom in order to see the superellipsoid.

Now, we will create the other surface. After accessing "load –ellip" again, we notice that the surface id is different. The parameters of the problem imply that an axis and the centers of the ellipsoids should be colinear; let us type in the coordinates (–2, 0, 0) in the "center" box. Leaving the semi-axes at their default settings, we replace the 2 in the "power" box with a 3, then click, [ok].

Both superellipsoids are now on the screen. We notice that one is sea blue and the other is a darker blue. The current surface is defined as the one that is sea blue. The current surface's id number is shown in the top left corner of the viewing window. Following the id number is a -1 in parenthesis. This number reveals to the user that the current surface was created from a built-in database, and not from the TIL code. In order to make the other surface current, we go to the green "CURRENT" box. The (<, >) arrows to the right of the [surf] button scroll through the current surfaces. Using these operations causes each surface to switch colors. When the [surf] button (in the "CURRENT" box) is not depressed, the current surface disappears. Therefore, a default of the "Topology Builder" panel is to have the [surf] button depressed.

By using the mouse buttons, we can translate, rotate, and zoom in on the surfaces so that the screen cuts each surface. The final result is shown in figure 2.1.

Figure 2.1



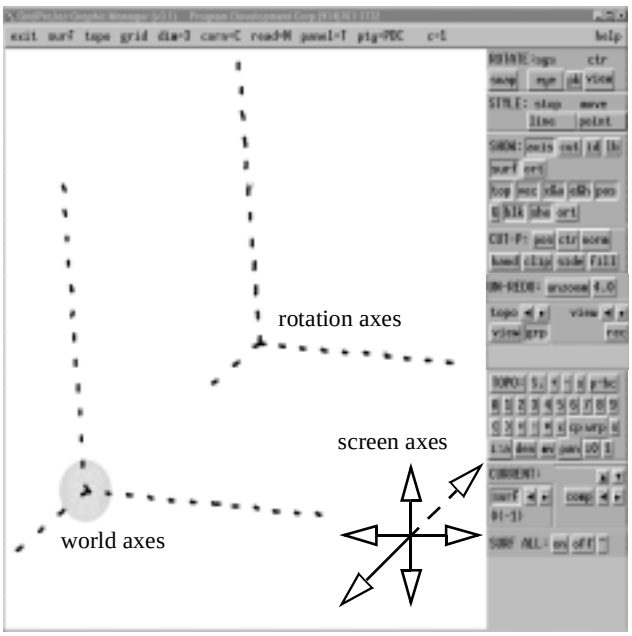
In order to do the second part of example problem #1, we must first learn about the different GridPro rotation operations. All the rotation operations are located in the green "ROTATE" box. The box contains the "snap," "sys" (system), and "ctr" (center) menus as well as the [pk] (pick) button.

There are four sets of axes: the screen axes, the world axes, the cut plane axes, and the rotation axis. Ignore the rotation and cut plane axes for now. The screen axes are fixed to the screen; the screen x-axis travels left to right, the screen y-axis travels bottom to top, and the

screen z-axis travels out of the screen. The world axes are fixed to the model. They can be translated and rotated relative to the screen axes.

The [snap] menu enables us to perform a transformation on the world axes so that they correspond to the screen axes. In the [snap] menu, the "world: xyz" operation rotates the model such that the world axes go to the closest possible screen axes. For example, "world: xyz" might align the world y-axis to the screen z-axis, the world x-axis to the screen y-axis, and the world z-axis to the screen x-axis because that transformation requires the least rotation. The "x," "y," and "z" operations (following the "world: xyz" operation) snap the world x, y, or z-axis to the nearest screen axis, but leave the other two axes unsnapped. The last three operations in the "snap" menu perform specific transformations from the world axes to the screen axes. For example, the "w:xyz->s:zxy" operation aligns the world x-axis to the screen z-axis, the world y-axis to the screen x-axis, and the world z-axis to the screen y-axis. The "w:xyz->s:xyz" and "w:xyz->s:yzx:" operations perform similar transformations. Ignore, for now, the "cut-p" rotation options. Of course, in order to have an understanding of the differences between these operations you should experiment with them on the computer.

Figure 2.2



The "sys" and "ctr" menus relate to the rotation action performed using the middle mouse button. The "sys" menu defines the system we wish to rotate under, and the "ctr" menu defines the object or objects we wish to rotate about. In the "sys" menu, the "rotate by eye system" operation allows us to rotate objects such as surfaces in the same general "direction" we move the mouse. Technically, the "rotate by eye system" operation allows us to rotate about an on-screen rotation axis perpendicular to the cursor's movement. The "rotate by world axis" operation (in the "sys" menu) allows us to rotate about a world axis of our choice. With this operation accessed, we can click on whatever world axis we wish to be our axis of rotation. When we select many operations in the "ctr" menu, we will see a set of rotation axes appear apart from the world axes. A pair of parallel axes is always present. However, when the rotation center is at the world origin, both sets of axes are coincident and we can see only one. In the "ctr" menu, the "fixed at origin" operation fixes the rotation center at the origin of the world axes, and the "fixed at surf" operation fixes the rotation center at the center of mass of the current surface (remember, we can change current surfaces in the green "CURRENT" box). The "fixed at model" operation fixes the rotation center at the mass center of a group of surfaces, and the "follow view" operation fixes the rotation center at the center of the viewing window. If we translate an object under "follow

view," the origin of the rotation axes will remain in the center of the viewing window while the world axes move away. The "fixed here" operation enables the origin of the rotation axes to remain a fixed distance away from the origin of the world axes when surfaces are translated. In addition, you can pick a rotation center at any point in space, on a topological wireframe, or on a grid with the cursor when the [pk] button is depressed. The [pk] button is quite convenient because it offers the user great flexibility in viewing options; one can pick points on surfaces, wireframe corners and edges, grid block edges, grid sheets, the cut plane axes, etc. Figure 2.2 illustrates some sets of axes.

Notice how effeciently you can manipulate the drawing area. You can translate, rotate, and zoom in on the contents of the drawing area *without taking your eyes off* the drawing area. Unlike other grid generation programs, GridPro contains no annoying dials, which take your eyes off your work as well as take up unnecessary space.

Returning to the second part of example problem #1, we notice that we must rotate the two surfaces such that one surface blocks the other from view. Therefore, we need to look along the world x-axis; the world x-axis should be sticking out of the screen. Thus, we must align the world x-axis with the screen z-axis. To do this, we open the [snap] menu and select "w:xyz->s:zxy." Automatically, our view of the surfaces is changed and one superellipsoid completely covers the other. In order to switch the positions of the objects, we can use another operation in the [snap] menu. The "w:xyz->s:zxy" operation cannot switch the positive-negative orientation of the axes. In other words, it cannot transform the positive z direction into the negative z direction. Therefore, that operation will always put the same superellipsoid "in front" of the other. In order to accomplish our objective, we first position one superellipsoid almost in front of the other (we do not have to be precise) by rotating the volumes about 180 degrees around each other. Then, we click the "world: xyz" operation under the "snap" menu, and obtain the desired result. ***EXAMPLE PROBLEM #1 COMPLETED***

Often, you will want to have surfaces peering out of the sides of the screen to look evenly between them or to attain a better perspective on the topology.

GridPro helps you keep track of multiple surfaces with some operations under the "surf" menu (located on the top toolbar). In this menu, the "reload current" operation reloads the surface dialogue box of the *current* surface you originally accessed using "load -ellip." You can make any desired changes to the parameters or even switch surface types. If you switch surface type, then you must of course insert the appropriate parameters for the new choice. Once you have completed your specifications, press [apply] to implement them. If you want to delete a surface, make this surface current and click "delete current" under the "surf" menu. If you want to delete all the surfaces, click "delete all" under the "surf" menu. If you prefer to hide all surfaces rather than delete them, depress the [surf] button found in the green "SHOW" box. Press the button again to make the surfaces reappear. Recall that the [surf] button in the green "CURRENT" box hides only the current surface; the [surf] button in the green "SHOW" box hides all the surfaces.

You should experiment with these operations using the superellipsoids created in example problem #1. You must remember which surface is current and which surfaces are hidden, especially with similarly positioned surfaces.

EXERCISE #3:

Create three superellipsoids with colinear centers. Make one of the surfaces cover the other two. Repeat this last procedure for a different surface.

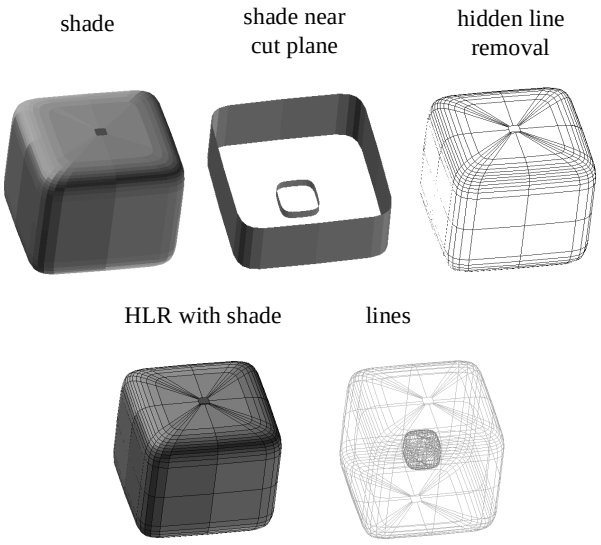
2.2 Drawing Styles of Objects

EXAMPLE PROBLEM #2:

Create a superellipsoid inside another, larger superellipsoid. Manipulate the screen so that the user can observe outlines of both ellipsoids.

In order to simplify the geometry, we should make both superellipsoids concentric. We click, "load- ellip," change the number in the power box from 2 to 4 (or any number greater than 2), and leave the axes and center boxes alone. Pressing [ok], we create a fourth-order superellipsoid centered at the origin with axes length = 1. Then, we access "load- ellip" again, change the number in the power box from 2 to 8, and give each axis a length of 4. Therefore, the semi-u box should read (4, 0, 0), the semi-v box should read (0, 4, 0), and the semi-w box should read (0, 0, 4). Finally, we press [ok].

Figure 2.3

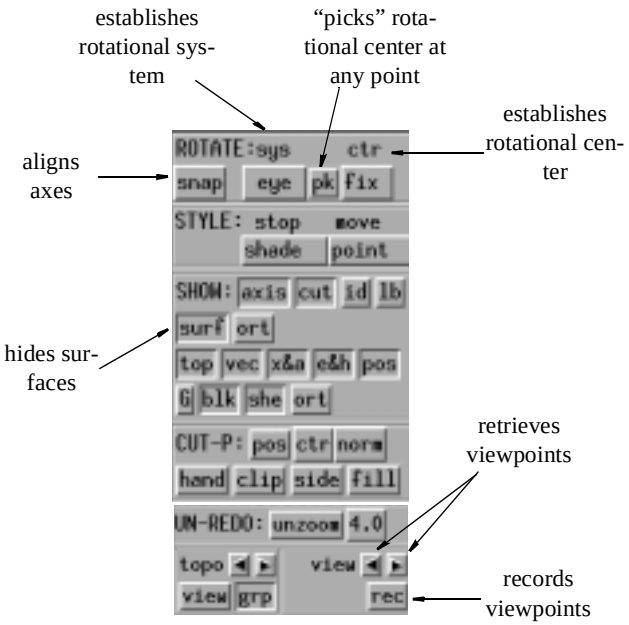


The operations in the brown, "STYLE" box dictate how we see geometry. Whatever operations are accessed in this box determine what we see on the screen. The "stop" menu determines how surfaces are outlined when they are still. The "move" menu determines how surfaces are outlined when they are moving. Five of the seven types of outlines are displayed on the same surfaces in figure 2.3. The two outlines not displayed, "points" and "1/10 points," cover surfaces with points. The "1/10 point" outline covers surfaces with one tenth of the points displayed from using the "point" outline. In general, the "move" menu should always have the "point" or "1/10 point" option accessed because these options allow surfaces to be moved quickly and smoothly. However, GridPro is a flexible program, one that can be used on both graphically fast and graphically slow computers. If you are using a fast computer, the

computer will not slow down when certain operations under the “move” menu, such as “shade,” are chosen. For graphically slower computers, you best choose “point” or “1/10 point.”

As can be seen from figure 2.3, the “shade near cut plane,” and “lines,” operations provide us with transparent outlines (the “point,” and “1/10 point” operations do as well). Therefore, we can see both superellipsoids when one of these four operations is accessed. ***EXAMPLE PROBLEM #2 COMPLETED***

Figure 2.4



Often, you will discover a useful viewpoint on your geometry. To save such viewpoints for future use, press [rec] in the “UNDO-REDO” box. For the same geometry, you may press [rec] as many times as desired. The computer will save each viewpoint *chronologically.* When you desire to retrieve a saved viewpoint, you can scroll back and forth through the saved list using the [<] and [>] buttons found above the [rec] button.

Some aforementioned operations are illustrated in figure 2.4.

EXERCISE #4

Create two superellipsoids that lie within another, larger superellipsoid. Outline all three such that you are able to see them on the screen at the same time.

Chapter 3 - The Cut Plane

3.1 Overview

The cut plane is, simply, a plane in space that is used for interactive topology generation and for graphical viewing. It represents a tool that allows you to place topology around, behind, and below objects. Essentially, the cut plane is at the heart of creating and visualizing three-dimensional topology. For example, if you want to create a wireframe of a cube enclosing a sphere, you need a device that can move you around the six faces of the cube. The cut plane allows you to place topology at any (x, y, z) coordinates in space. The plane can be rotated and translated to fit any position. Like all planes, the cut plane is infinitely long. However, to facilitate easy operational use, GridPro represents the cut plane as a finite rectangle. This rectangle is called the cut plane rectangle. This rectangle can be shrunk or expanded in order to work on a global or local scale.

The cut plane should be thought of as a tool that guides your cursor. Cursor movement is *always* along the cut plane. In other words, although you see the cursor move on a flat, two-dimensional screen, it is actually moving along the surface of the cut plane. Essentially, the cursor is "attached" to the cut plane regardless of the position or size of the cut plane rectangle. Stated in a slightly more formal way, the mouse space (your mouse pad) is mapped onto the cut plane. Before you input or remove topology, always think of where the cut plane is positioned.

In order to access the cut plane, depress the [cut] button found in the green, "SHOW" box. GridPro outlines the cut plane rectangle in white. To keep the cut plane hidden, leave the [cut] button undepressed. If the rectangle is too large, you will see nothing because you are looking inside the rectangle. In this case, push the [unzoom] button. By contrast, sometimes the rectangle will appear as a white speck. In this case, zoom in on it using the right mouse button.

The cut plane is used in both two and three dimensions. However, when working in two dimensions, make sure the cut plane is aligned with the plane of the screen (the plane on which you are working, or the $z = 0$ plane).

3.2 Movement

Moving the cut plane is similar to, but more precise than, moving surfaces. Unlike less advanced programs, GridPro allows you to translate, rotate, and dilate the cut plane without taking your eyes off the drawing area. If you press the "hand" button in the brown, "CUT-P" box, features appear on the cut plane. A set of beige, sea blue, and red axes appear (the cut plane axes)

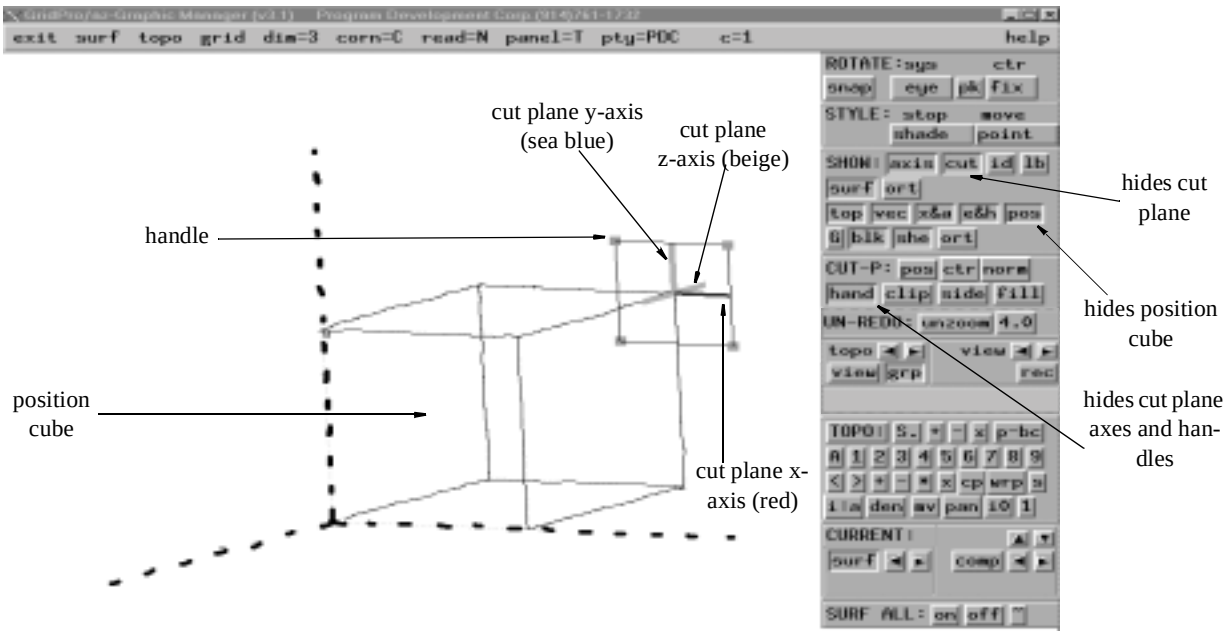
along with four dark blue squares (the handles), each of which is pasted on a vertex of the cut plane rectangle. The beige axis is the cut plane z-axis, the sea blue axis is the cut plane y-axis, and the red axis is the cut plane x-axis. When you move the mouse to an axis, the cursor will transform into two curved arrows:  Make sure the cursor becomes these two arrows before you try to click on an axis.

By dragging an axis with the translating mouse button (the left mouse button), you can translate the cut plane along that axis. You will use this imprecise but simple method of translation most often to position loosely the cut plane. If you wish to rotate the cut plane about one of its axes, click on the axis with the left mouse button. Notice that the specified axis becomes thin. Then, drag one of the two fat axes with the rotational mouse button (the middle mouse button) in order to rotate the cut plane about the thin axis. You can change the size of the cut plane rectangle by dragging inward or outward one of the handles (four dark blue vertex squares). When you position the cursor on a handle, the cursor becomes a square with a center dot:  Because you will need to master these manual translation, rotation, and dilation operations, start practicing them now.

When the center of the cut plane rectangle is displaced from the center of the world axes, a cube is formed between the centers. Each center is at an endpoint of a diagonal of the cube. This cube gives you a visual indication of how far away the cut plane is from the center of the world system. If the [pos] button in the green, “SHOW” box is depressed, this cube will be seen. In order to deactivate the cube, un-depress the [pos] button.

Figure 3.1 illustrates operations relating to the cut plane.

Figure 3.1



EXERCISE #5:

Create two ellipsoids (which do not intersect): one with center (3, -8, -1) and axes length = 15, and one with center (-5, 12, -6) and axes length = 6. Position the cut plane between the surfaces.

EXERCISE #6:

Create one superellipsoid inside another, and position the cut plane so that the cut plane rectangle cuts the inner surface but does not touch the boundary of the outer surface.

All operations in the brown, "CUT-P" box relate to the cut plane in some way. The [pos] menu positions the cut plane, the [ctr] menu centers the cut plane (with respect to different reference frames), and the [norm] menu aligns the cut plane's axes with the world axes or at angle of 45 degrees from them. You already know the [hand] operation reveals the cut plane axes and vertex squares. The [clip] and [side] operations enable you to cut off part of surfaces and grids. Finally, the [fill] operation shades the cut plane rectangle in translucent white.

When you access the [pos] menu, a window will appear. Ignore the "NOTE:" box. The next two boxes, the "normal:" and "center:" boxes, define the position of the cut plane. Like any plane, the cut plane can be positioned by inputting the coordinates of a normal vector to the plane and the coordinates of one point on the plane. The point on the cut plane for which you provide coordinates becomes the center of the cut plane rectangle. You can input (x, y, z) coordinates of the normal vector in the "normal:" box, and input (x, y, z) coordinates of the point in the "center:" box. When you finish inputting coordinates, press [apply] in order to implement your input. The cut plane should immediately conform to its new properties.

EXERCISE #7:

Repeat exercise 5. However, position the cut plane using the [pos] menu.

EXERCISE #8:

Repeat exercise 6. However, position the cut plane using the [pos] menu.

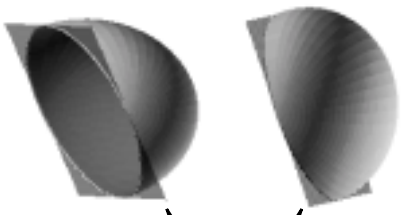
The [ctr] menu contains five options. For now, just know that the "view" option aligns the center of the cut plane rectangle with the center of the drawing area and rescales the cut plane rectangle so that its area is half that of the drawing area. The "view" option is often useful when you do not know where the cut plane is, and do not care about altering the cut plane rectangle's position or size.

The [norm] menu contains nine options. Each option allows you align cut plane axes with world axes by transforming the normal to the cut plane. If you select the "x" operation, for example, the normal of the cut plane will align with the x-axis. As a result, the cut plane will snap to the yz plane. The "y" and "z" operations work similarly. If you select the "xy" operation, the normal of the cut plane will conform to *both* the positive x and y-axes, forming a 45 degree angle

with each one. The "-xy" operation rotates the cut plane positioned by the "xy" operation 90 degrees about the z-axis. In order to understand fully these "norm" operations, you must experiment with each one and pay attention to the relationship between axes.

The [clip] and [side] operations can be used to look inside different parts of surfaces. By positioning the cut plane so that it cuts through a surface, you can actually clip away part of that surface by pressing, [clip]. Only the portion of the surface on one side of the cut plane will remain. To reveal the portion of the surface on the other side of cut plane, press [side]. The previously showing portion of the surface will disappear while the other side of the surface will appear. Since the clipped graphic display pierces objects, you can clearly observe 2 dimensional slices of a surface.

Figure 3.2



"filled" cut plane clipping a surface

The [fill] operation, like the [clip] and [side] operations, serves as a visual aid. Depressing the [fill] button will leave the cut plane rectangle shaded in translucent white. Usually, the [fill] operation is most useful when a side of the surface is clipped and you want to better orient your eyes to the viewscreen. Essentially, the [fill] operation locates the cut plane for you, and allows you to view better 2 dimensional slices of a surface. Figure 3.2 displays the result of a filled cut plane clipping a surface.

EXERCISE #9:

Recreate the surfaces from exercise 5. Clip each one and experiment with the [side] and [fill] operations.

EXERCISE #10:

Recreate the surfaces from exercise 6. Clip both at the same time and experiment with the [side] and [fill] operations.

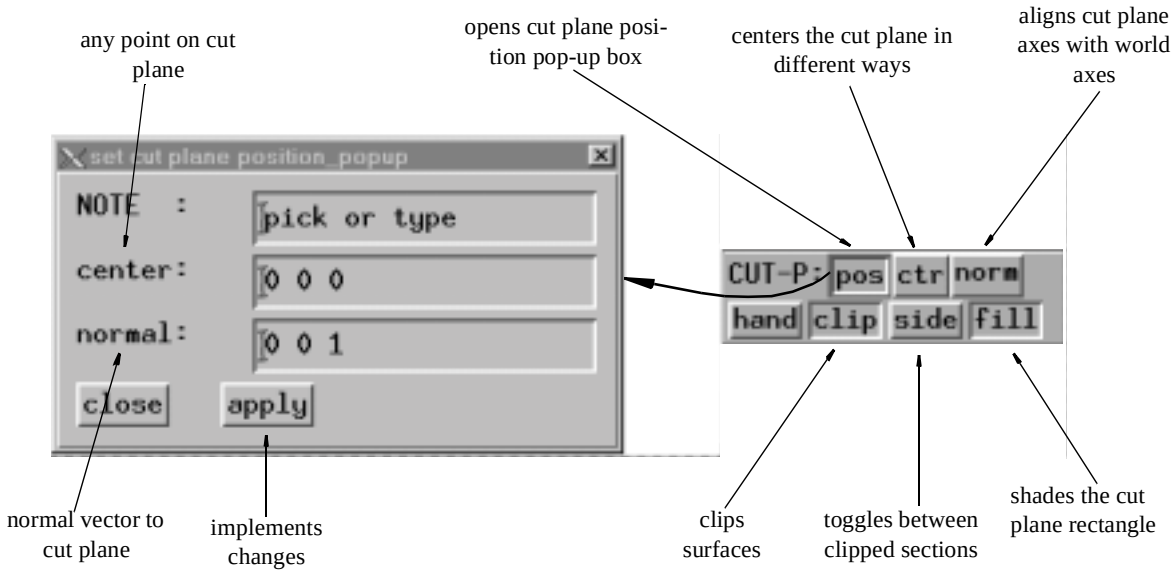
Many rotation options deal with the cut plane. The [snap] menu in the green, "ROTATE" box contains four cut plane operations, the "cut-p: xyz" operation, and the following "z," "y," and "x" operations. Just as the "world: xyz" operation aligns the world axes with the screen axes using the least rotation needed, the "cut-p: xyz" operation aligns the cut plane axes with the screen axes using the least rotation needed. Similarly, just as the subsequent (world) "x," "y," and "z" operations snap the world x, y, or z axis to the nearest screen axis (and leave the other two axes unsnapped), the "x," "y," and "z" operations following the "cut-p: xyz" operation snap the cut-plane x, y, or z axis to the nearest screen axis, leaving the other two axes unsnapped.

In the green, "ROTATE" box, cut plane operations exist under the "ctr" and "sys" menus as well. Under the "ctr" menu, the "follow cut-plane" operation assigns the rotational center to be the center or origin of the cut plane rectangle. Therefore, after you select this option, the middle mouse button rotates the drawing window around the center of the cut plane rectangle. Under the

"sys" menu, the "rotate by cut-p axis" operation allows you to rotate the drawing window around one cut plane axis. After selecting this "rotate by cut-p axis" operation, click on one cut plane axis with the left mouse button. Now, you can rotate the drawing window around that axis.

The cut plane can also influence how surfaces are represented. The "shade near cut-p" operation under the "stop" and "move" menus in the brown, "STYLE" box actually shade in parts of the surfaces that are close to the cut plane. This operation is useful when you need to analyze the region around an intersection of the cut plane and a surface. When you resize the cut plane rectangle by dragging one of its vertex squares, you also resize the distance that the surface is drawn from the cut plane in its normal direction. For example, an enlargement of the cut plane rectangle results in a proportionally larger band of surface display on each side of the cut plane. Figure 3.3 illustrates other properties of the cut plane.

Figure 3.3



Chapter 4 - Introduction to Topology and Grids

4.1 Overview

In order to grid regions, you must cover them with wireframe topologies that crudely follow their boundaries. The majority of GridPro operations deal with creating and modifying topology. Remember, a topology consists of a wireframe *and* surface assignments.

GridPro is a unique multiblock grid generation program because it revolutionizes how volumes are gridded. Currently, multiblock grid generation programs require users to assign topological attributes (separate parts of a topological framework) laboriously to individual blocks. GridPro *automatically* assigns separate parts of a topological framework to individual blocks.

Remarkably, GridPro can also grid loosely positioned topology. In other words, GridPro will grid topology whose position only roughly conforms to the corresponding geometry. A wireframe topology does not have to exactly fit the boundaries for the region to be gridded. However, the topology itself should be precise. If a topology is positioned too far from a surface, the gridding process might take longer than it normally should. Thus, as long as your topology is positioned with some care, the surfaces will be gridded with relative ease. When creating topology, you would do well to remember this golden rule: the topology for a region has to be precise but does not have to be positioned precisely, just positioned to roughly follow the geometry of its bounding surfaces. The allowed imprecision of the wireframe corners is referred to as putting them in "general position". This is a well-known concept from topology, a subject that deals with classes of surfaces defined in a loose, imprecise fashion, like stretchable rubber surfaces.

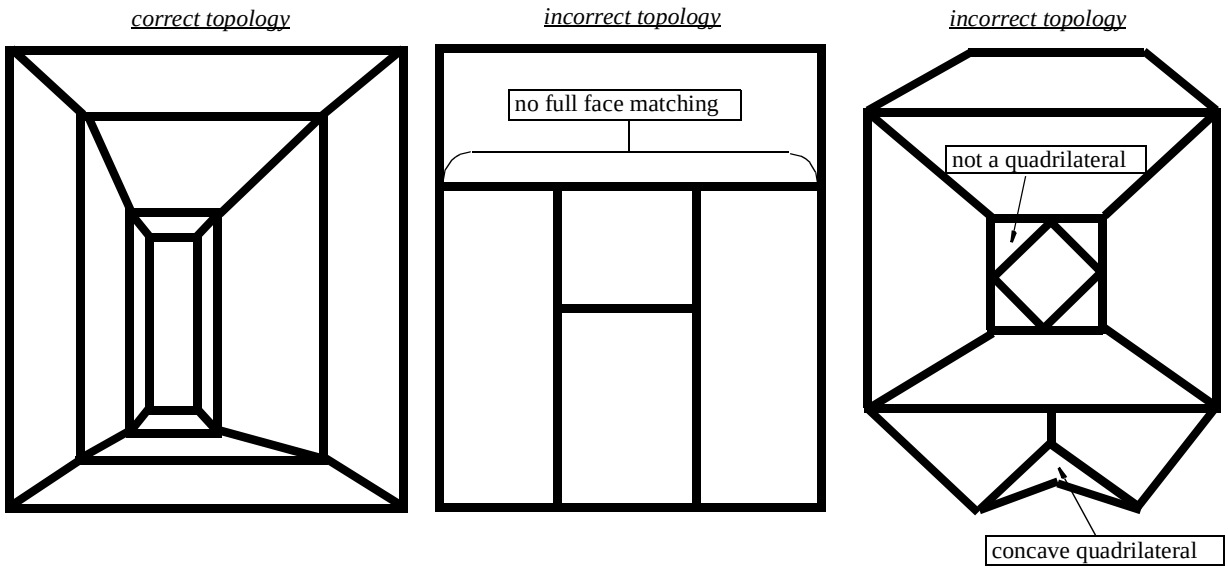
Like all other multiblock grid generation programs, GridPro requires you first to *sketch out the topology for a surface before inputting that topology.* You *must* know how a volume will be decomposed into blocks; GridPro can only work with your results, not create those results. In other words, you should decide upon the pattern of grid points before you can instruct GridPro to generate the corresponding grid.

GridPro grids volumes in both 2 and 3 dimensions. Although all real life objects are three-dimensional, sometimes two-dimensional grids can serve as useful models. Also, first creating topology in two dimensions better prepares you for creating topology in three dimensions.

In two dimensions, gridded blocks are represented by convex quadrilaterals, or quads. In three dimensions, gridded blocks are represented by convex hexahedrons, or hexes. When you create a quad or hex, GridPro will automatically recognize it as such. Unlike other grid generation programs, GridPro does not require the you to identify a geometrical object after you create

it. GridPro will read only topologies containing “full face matching.” These topologies include intersecting quads with intersecting faces having the same dimensions, or intersecting hexes with intersecting faces having the same dimensions. In other words, one side of one quad cannot intersect one side of two other quads, and one face of one hex cannot intersect one side of two other hexes. Figure 4.1 illustrates some examples.

Figure 4.1

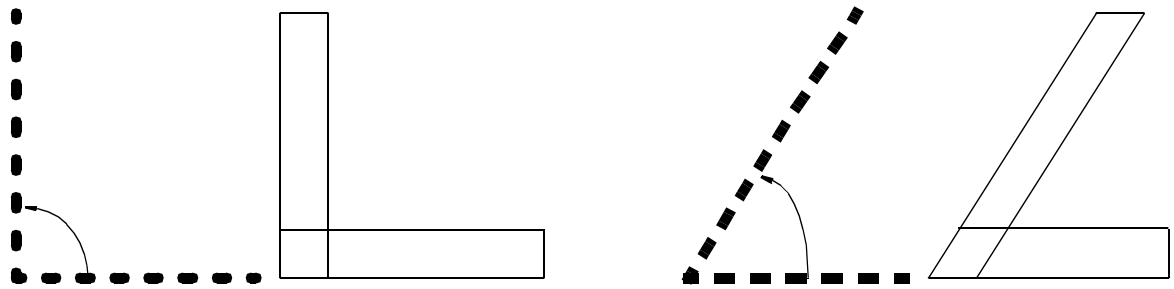


Now that you know the topological form expected from GridPro, you should make sure that you understand how to decompose a region and create a topology.

Because surfaces can be complex, they may require complex topologies. Unfortunately, no set of rules can define how all topologies could be constructed. There are some rules, however, which define how most topologies should be constructed when there is no other special feature or concern. When you create topologies about angles in surfaces, you should consider the concavity or convexity of the angle. This is seen from the viewpoint of the region to be gridded rather than from outside that region looking inward. In the following examples, the dashed structures represent surfaces, and the solid structures represent a possible, correct topology for those surfaces which is on the side that is to be gridded. Differences between correct topologies and a discussion of what topology to use will be covered later. In general, preferred topologies about angles between 0 and 90 degrees have the structures shown in figure 4.2

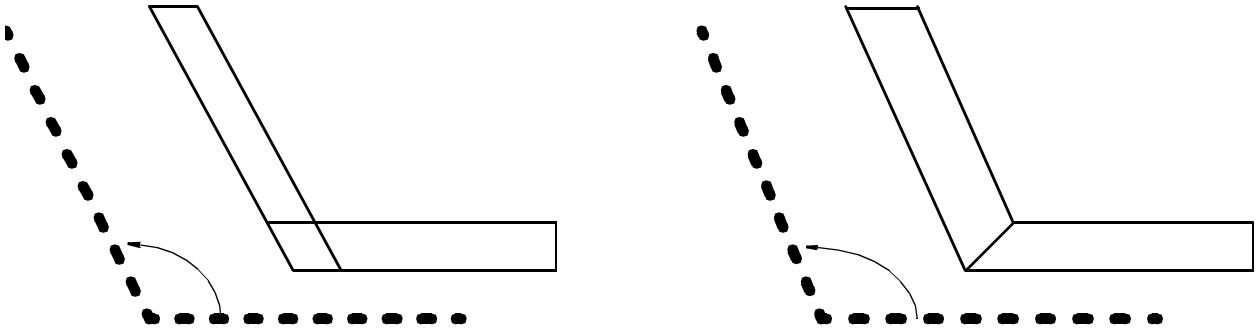
:

Figure 4.2



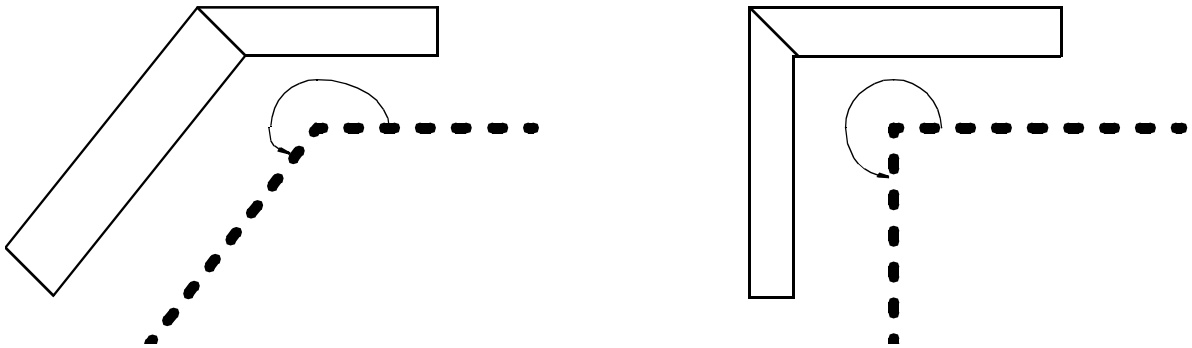
Angles between 90 and 180 degrees typically have two different structures. These structures are illustrated in figure 4.3; the topology on the right side of the figure is used more often for these angles:

Figure 4.3



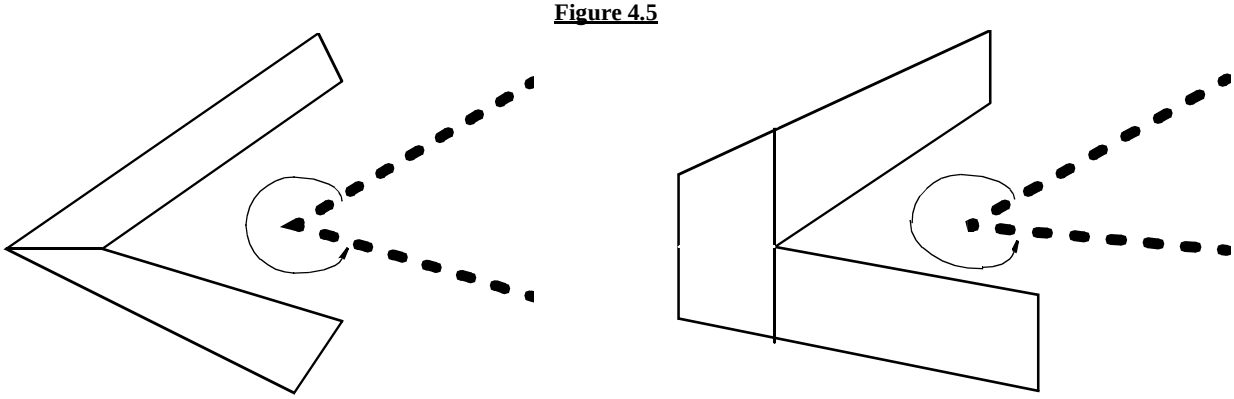
Preferred structures for angles between 180 and 270 are illustrated in figure 4.4:

Figure 4.4



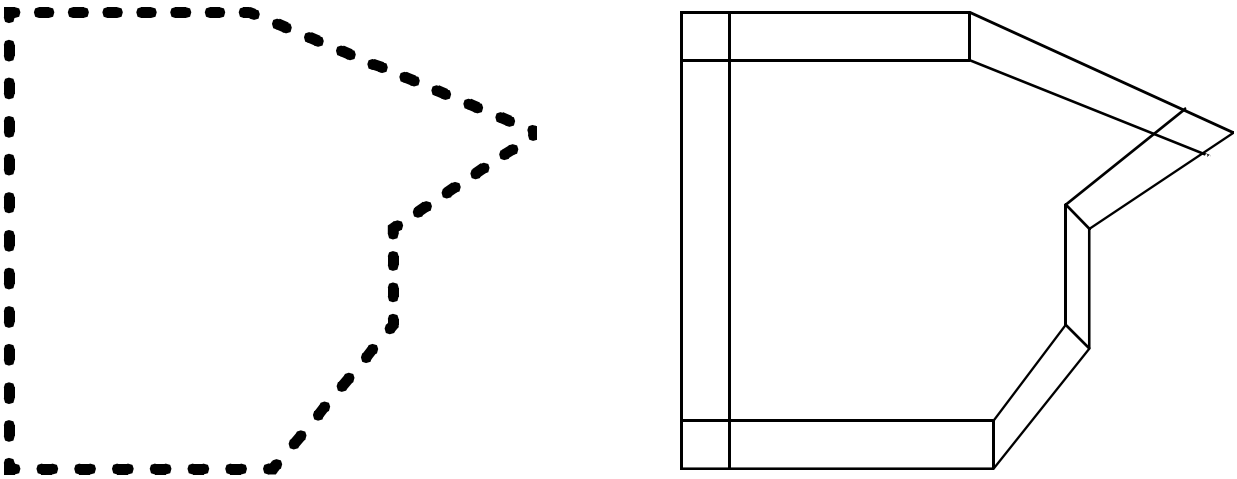
Angles between 270 and 360 degrees can have several structures, the most common of which are illustrated in figure 4.5:

Figure 4.5



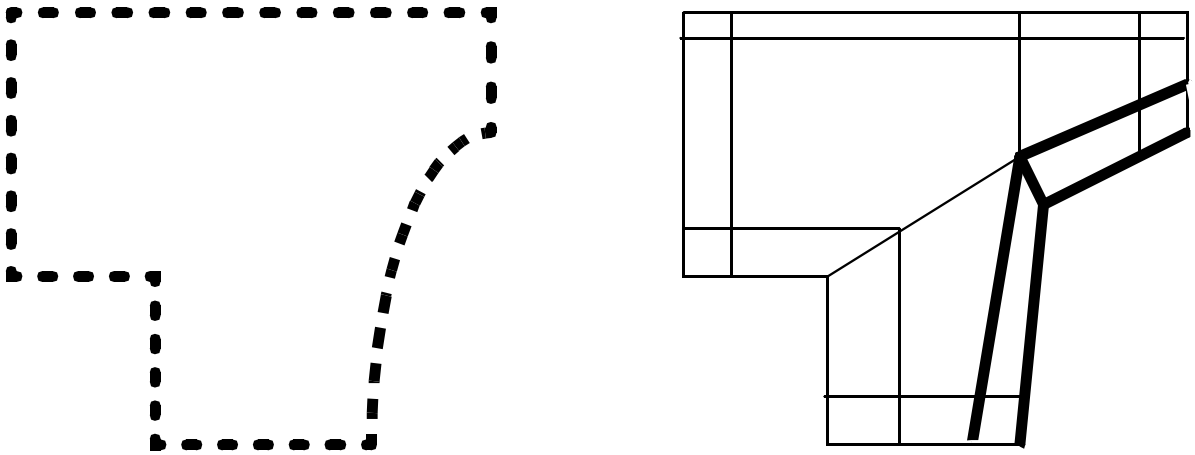
Notice that all of the topologies are created using *convex* quads, and all quads have full face matching. Figure 4.6 incorporates ideas you have seen:

Figure 4.6



If a surface is smooth, decompose the surface as you normally would, then create an "inner layer" between the smooth surface and decomposed block. The "inner layer" is illustrated in dark shade. Figure 4.7 illustrates a correct topology for a smooth surface:

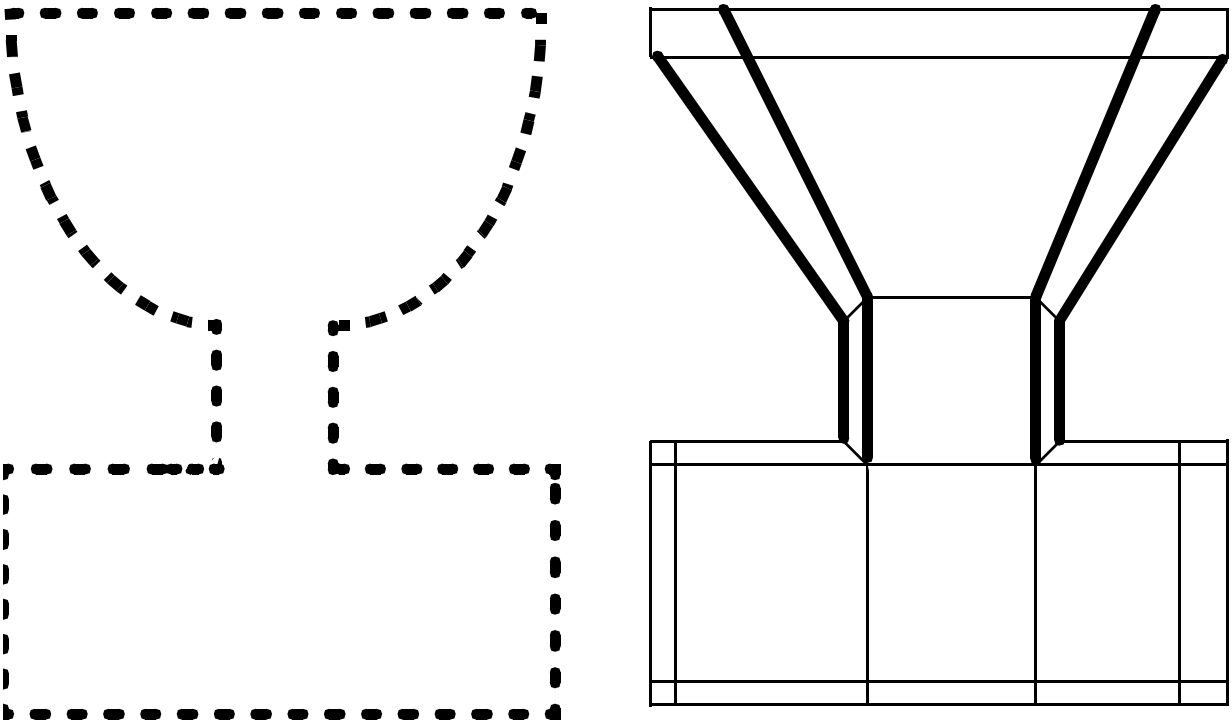
Figure 4.7



Notice how the darkened lines follow the path of the circular region. In figure 4.7, the topology conforms well to the corresponding geometry. Also, this example shows how topology is wrapped around the continuously curved portion of the surface. The figure formed by the dark lines is considered a partial "wrap." Looking at the angle variations above, you will notice that the angles close to 180 degrees are wraps. In fact, at every point of a smoothly varying surface, the angle is computed from the corresponding surface tangent line.

Look back at how topologies are created for angles between 90 and 360 degrees. Notice that some of those topological configurations are "wraps" as well. In figure 4.8, the partial wraps formed for the continuously curved surfaces are illustrated in dark shade.

Figure 4.8



Notice that in the above example, wraps are formed around the 270-degree angles as well. Figure 4.9 displays the full wraps in dark shade:

Figure 4.9

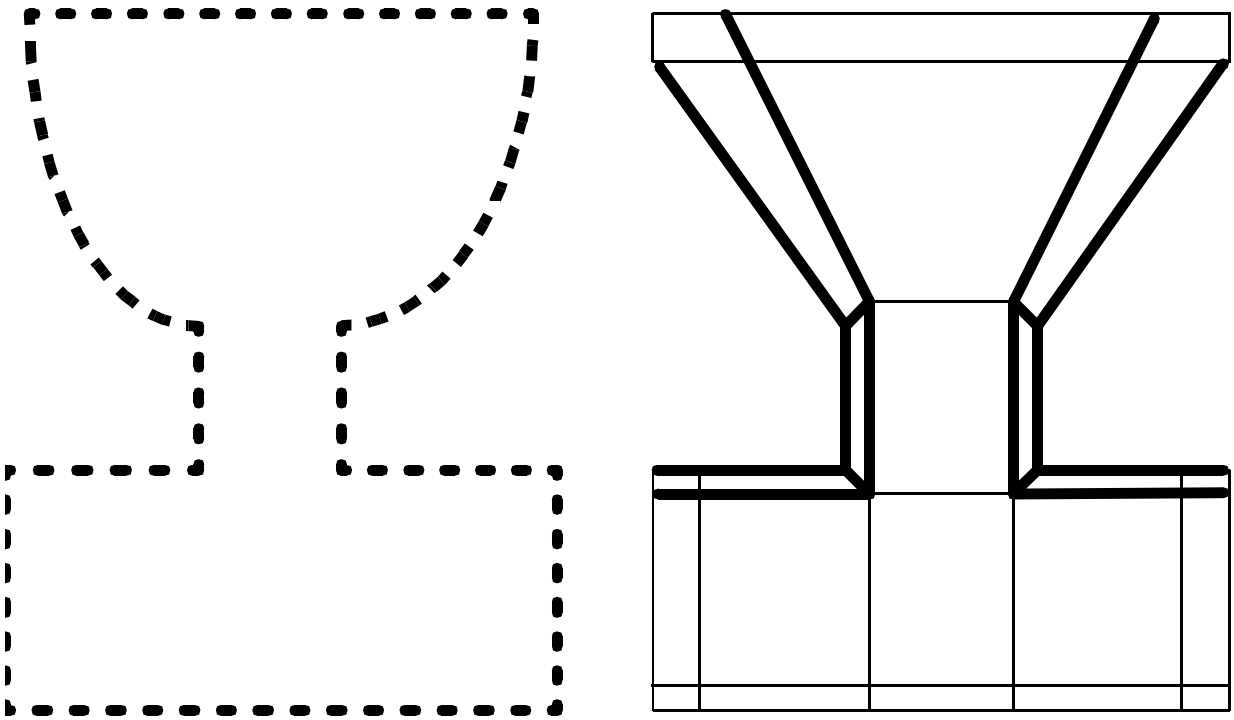
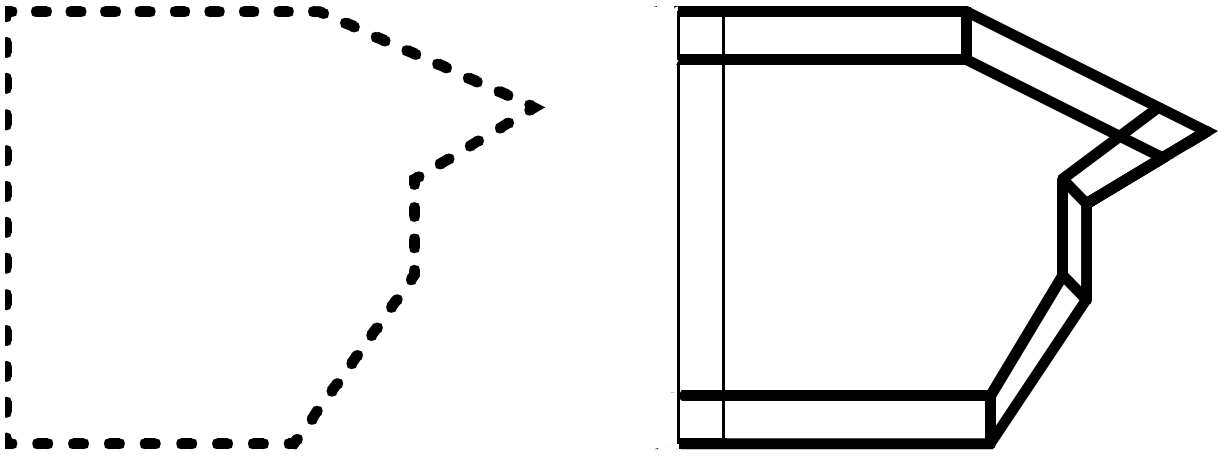


Figure 4.10 displays the full wraps in dark shade from the first example in this chapter:

Figure 4.10



In order to develop an intuitive understanding of what topology to create, you should learn through example. You will successfully prepare regions for gridding by first learning the rudimentary rules described above. Of course, in order to obtain the phenomenal grid accuracy GridPro can provide, you have to learn gradually the nuances of different surfaces and be able to

produce topological frameworks using intuition rather than rules.

4.2 Corners and Edges

Wireframe topologies are created entirely by corners, or vertices, and edges, or links. Any two- or three-dimensional structure can be constructed with just corners and edges. Corners are displayed as small orange squares, much like the cut plane rectangle vertex squares. Edges are displayed simply as yellow lines with two corners as endpoints.

Corners are inputted either on surfaces or on the cut plane. For cut plane input, you must know where the cut plane is in relation to your surfaces. You will also need to be in the mode for cut plane input, "corn=C," which can be accessed on the top tool bar. Because this mode is the default mode, first-time users will not need to change the mode settings. Selecting the "corn=S" mode allows you to input corners on surfaces. When in the "corn=C" mode, you will most likely need to change the cut plane's position before inputting corners. Edges can then be created after corners are inputted; the cut plane has no effect on edges.

Most of the operations dealing with corners are done on the keyboard and not in the command panel. In order to input a corner, depress the <C> key and click the left mouse button. An orange square should appear at the location of the cursor. In order to remove a corner, move the cursor to a corner, depress the <R> key, and click the left mouse button. Because of screen tolerances, you might have to try repeatedly to remove a corner. When you depress the <R> key and move the cursor to the corner you wish to remove, the cursor becomes a ring:  If the ring turns white and you click the left mouse button, the corner will definitely disappear. If the ring turns red and you click the left mouse button, the corner will probably disappear. If the corner does not disappear, keep on trying to remove it.

When you want to create an edge between two corners, use the <E> key. Move the cursor to one corner, hold down the <E> key, and click the left mouse button (in order to select a corner, the cursor must become a ring). Still holding down the <E> key, move the cursor to the other corner and click the left mouse button. An edge should appear between the two corners. Remember, if the cursor does not become a white ring when you click on the corners, an edge might not form. In this case, click one or both corners again to create the edge. If you want to remove an edge, depress the <R> key and move the cursor to any point on that edge. The cursor will become a two-sided vertical arrow:  This arrow can become red or white similar to the ring; clicking a red double arrow will probably eliminate an edge, and clicking a white double arrow will definitely eliminate an edge.

In order to create a string of edges, all you must do is depress the <E> key for the duration

of the process and click the mouse strategically. By clicking the endpoint corner of the string, double-clicking every successive corner you wish to link, and finally clicking the other endpoint corner of the string, you can form many edges quickly and efficiently.

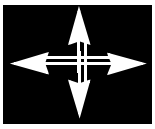
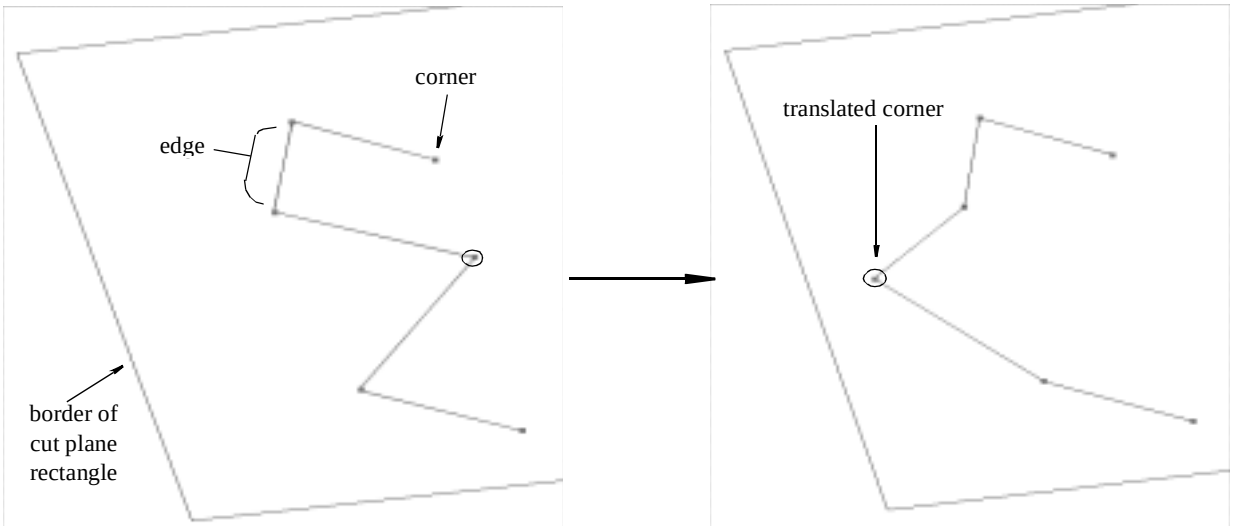
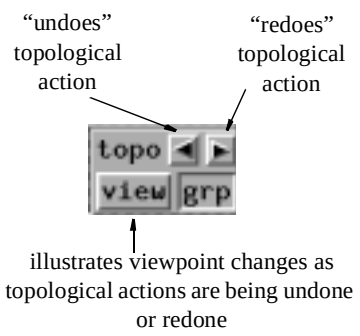
After you create a corner, you can translate it by moving the cursor to the corner and dragging the corner by using the left or right mouse button. If you use the right mouse button, you will translate the corner perpendicular to the cut plane. As you translate a corner, the cursor becomes cross-hatched, double arrows:  Using the left mouse button enables you to drag a corner parallel to the cut plane, and using the right mouse button enables you to drag a corner perpendicular to the cut plane. After you create at least one edge, you can deform it simply by clicking on one of its endpoints (a corner) with the translation mouse button and dragging the endpoint. The edge or edges will drag with the endpoint. As you translate corners and edges, the cursor becomes a white cross. Figure 4.11 illustrates the effect on edges of translating a corner with the left mouse button.

Figure 4.11



When you input corners and edges, you will often make mistakes. GridPro remembers every topological operation you perform, and enables you to go back and scroll through every operation. Therefore, you can erase all topology you have created from any point in the process. Also, you can scroll forward; if you realize that you should not change anything, you can return your screen to its previously current state. In the green, “UNDO-REDO” box, the arrow button (<, >) operations to the right of the “topo” heading allow you to scroll through all your previous topology actions.

Figure 4.12



Most likely, you will create topology using different viewpoints. In addition to topology operations, GridPro can remember the viewpoints you used when you performed the operations. You can include viewpoints you would like GridPro to remember as you scroll through previous topology actions by depressing the [view] button found below the "topo" heading in the green, "UNDO-REDO" box. Figure 4.12 displays the location of these topology operations,

Positioning corners with the mouse might seem too imprecise to be practical. However, remember that GridPro will grid loosely positioned topologies, or topologies that do not conform exactly to a geometry. Even so, you can examine precise corner locations using the query facility, accessed with the <Q> key. When the <Q> key is depressed, the coordinates of the cursor on the cut plane are displayed in the upper-left hand corner of the screen. When the cursor is on a corner, it becomes a circle and the corner coordinates are displayed, regardless of weather or not the corner lies on the cut plane.

Remember, when you create topologies in 2 dimensions, you should align the cut plane with the plane of the drawing window (the plane z = 0). When "corn=C" is selected in the top menubar, corners are inputted only on the surface of the cut plane.

EXERCISE #11:

In 2 dimensions, create an octagon using corners as vertices. Split the octagon into three quadrilaterals, then clear the segments used to create the quadrilaterals using two different methods.

4.3 Surface Assignments

Unlike other grid generation programs, GridPro automatically generates a multiblock grid within a region bounded by surfaces after the wireframe topology is given. However, to complete the topology, you must tell GridPro which collection of corners are to be assigned to each surface. Even if you are gridding a region bounded by only one surface, you have to assign the boundary corners of your inputted wireframe to that surface.

For the purpose of simplifying our present discussion, we will facilitate our understanding by following some restrictive rules to illustrate better topology creation. If a boundary is composed of connected straight lines, each line should represent a surface. If a boundary is composed of circular arcs, each arc should represent a surface. Therefore, a circle or ellipse is one surface and a square or a rectangle is four surfaces. If an endpoint of a straight line intersects with the endpoint of an arc, you must determine if the tangent lines to each endpoint are colinear. If the

tangents are colinear, the arc and line are part of the same surface. If the tangents are not colinear, the arc and the line are each one surface. Remember: these rules represent a basic guideline only for the present discussion and will be altered as you learn more about GridPro.

In order to understand fully the process of surface assignment, you should intuitively understand what occurs. Think of the surface to which you are giving an assignment as a huge vacuum, one that attracts all corners assigned to it along with all logically corresponding edges and faces. The corners and their corresponding elements become attached to the surface, stop moving, and reside where they stick. All of this occurs in the grid generation process. When the multiblock grid is viewed with corners and edges only, we see the resultant block decomposition. This image can also be thought of as the final resting spot of the original wireframe.

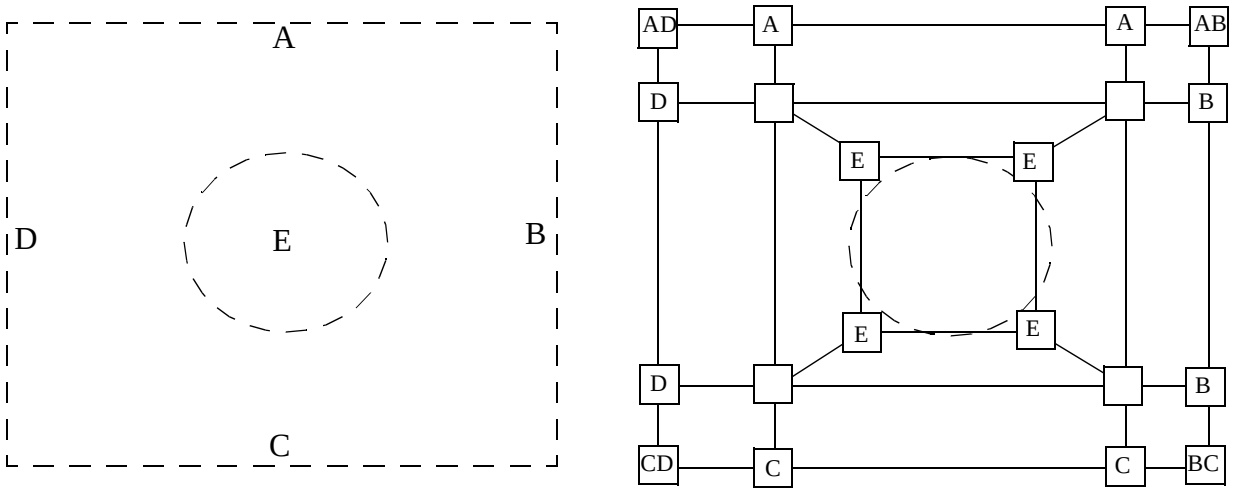
Corners can be assigned to zero, one, two, or more surfaces. For our discussion, all surfaces are bounding surfaces; only corners that are wireframe boundaries are assigned to surfaces. If the two corners on either end of an edge are assigned to a surface, then the edge is automatically assigned to the surface (in 2D or 3D). Likewise, if a closed loop of four corners is assigned to a surface, then the corresponding face and edges are automatically assigned to the surface (in 3D). This means that the grid on each automatically determined part also goes to the surface.

If a corner is assigned to each of two intersecting surfaces, then the corner will go to the intersection between the two surfaces. For example, when gridding a rectangle, the double assignment to both a horizontal and a vertical side will attract the corner to the vertex of the rectangle formed by the intersection. If it were the case of a box in three dimensions, then the attraction would be to a box edge. A triple assignment, however, would send the corner to a box vertex.

Figure 4.13 displays surfaces, corners, edges, and surface assignments in two dimensions. The surfaces are outlined in dashes and their corresponding topologies are in bold print. Each surface is lettered. The corners composing the topological wireframe are gray squares, and the lines connecting the corners are edges. If a corner is assigned to a surface, the letter of that surface appears inside the corner. Notice that the surfaces are transposed over the wireframe topologies. From those illustrations, you can see the corners and edges are loosely placed; their positions relative to one another and to surfaces are not precise or exactly proportional.

When you are gridding more than one object, you must be absolutely sure what region you wish to grid (when you grid one object, you have to grid the region inside that object). The surfaces in figure 4.13 can be gridded in many ways. For our purposes right now, we make the region to be gridded outside of the hole and inside of the large rectangle. In other words, we grid a rectangular plate with a center hole. *If we grid the region outside of a surface, the boundaries of that surface are defined by the internal corners of that surface's topology. If we grid the region inside of a surface, the boundaries of that surface are defined by the external corners of that surface's topology.* In shorter, simpler terms: when we grid outside, we assign inside; when we grid inside, we assign outside. Therefore, because we grid the region outside of the circle, we assign

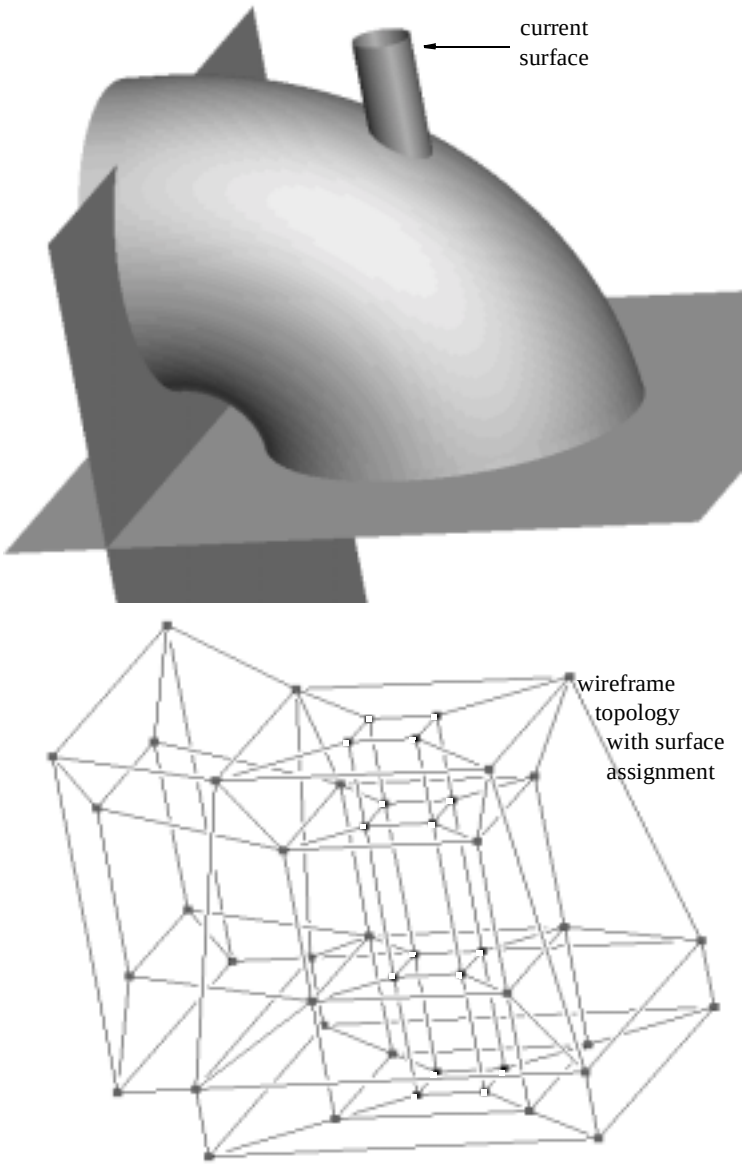
Figure 4.13



the internal corners of that topology to the circle.

Now that you know where to assign corners to surfaces, you can learn how to assign corners to surfaces. GridPro enables you to utilize the keyboard or the menus for assignment purposes. Before assigning any corners to a surface, you must make sure that surface is current (using the toggle operations in the green, “CURRENT” box). Remember, when a surface becomes current, its color changes to sea blue.

Figure 4.14



After a surface is made current, you may assign corners to that surface by depressing the <S> key and clicking on those corners. Notice, as you move the cursor toward a corner, the cursor becomes a white or red ring. Also, notice when a corner is assigned, a small white box appears within the corner. This white box provides you with a visual indication of confirmed surface assignment, which proves to be useful often when many surfaces and many corners are involved in a topology. If you wish to deactivate a surface assignment, depress the <S> key and click on the assigned corner. The small, white internal box should disappear. Figure 4.14 illustrates how these internal boxes are displayed.

Although you can use the <S> key for surface assignment, operations in the brown, "TOPO" menu will expedite your task. The [S.] operation performs the same task as does the <S> key. However, the [S.] button only needs to be depressed once in order to be activated (as opposed to holding the <S> key every time you want to assign a corner). The [S.] operation can assign corners to surfaces *or* deactivate corner assignments.

The three operations found to the right of the [S.] operation, the [+], [-], and [x] operations, assigns or deactivates many corners at a time. After depressing the [+] button, you can use the right mouse button to create a *purple* rectangular box (recall that the right mouse button normally creates a white zooming box). The point where you click is one vertex of this purple box. From that point, you can expand the box by dragging it. When you finish dragging and release the mouse button, all corners inside the box will be assigned to whatever surface is current. The [-] operation performs the opposite function; when depressed, this button deactivates all surface assignments in the purple box you create. Only one purple box can be created when the [+] or the [-] button is depressed. In other words, you must remember to re-depress the [+] or [-] buttons

every time you wish to use them. The [x] operation, when accessed, simply deactivates all corner assignments to the current surface.

In order to understand the usefulness of the surface assignment operations, look again at figure 4.13. You best assign all corners to surface D by depressing the [+] button, then dragging a long, thin purple box so that it encompasses the left-most column of corners. By using this method, you assign four corners to one surface with one operation and one mouse click. The alternatives are to use the <S> operation four times with four mouse clicks, or the [S.] operation once with four mouse clicks. If, by mistake, you assign all corners in the second column to surface D, you can deactivate the assignments using the [-] operation in the same manner as you used the [+] operation.

Often, you will be in an understandable rush to complete a grid. As a result, you might forget one step in the process of assigning corners to a surface. A common mistake people make is to forget to depress the [+] or [-] button each time they wish to use a purple box. As a result of this oversight, a white zooming box appears. Remember: in order to deactivate the zooming box, make the box as small as possible before releasing the right mouse button.

4.4 Miscellaneous and Example Problems

As you create a topology for a region in space, sometimes you might wish to hide the topology in order to obtain a better view of its bounding surfaces. If you depress the [TOPO] button in the green “SHOW” menu, the topology in the drawing window will disappear, or hide. The topology will reappear when you un-depress the [TOPO] button.

If you want to eliminate the topology displayed on the drawing window, select the “delete current” operation under the “topo” menu found in the top toolbar.

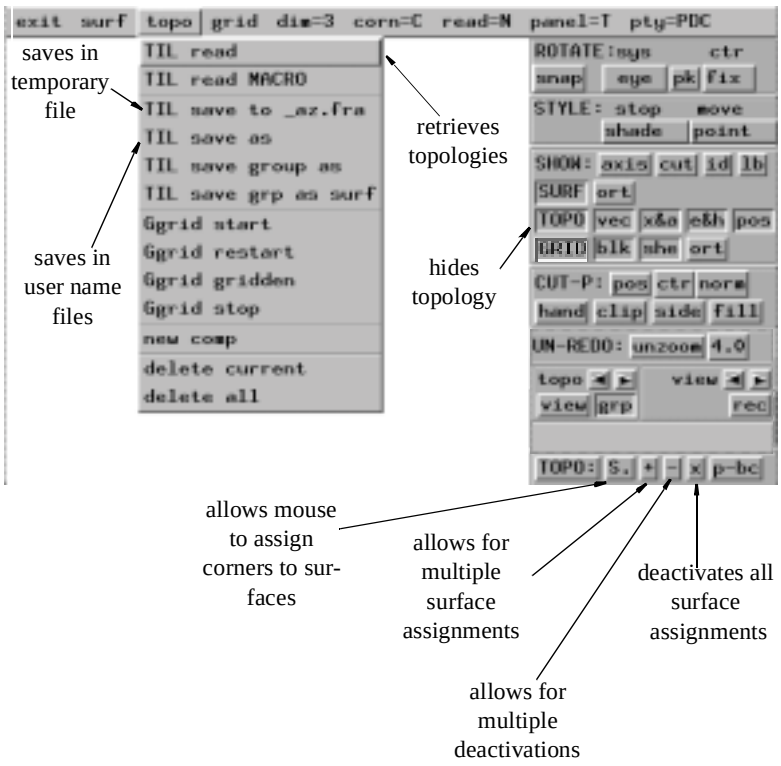
When you save a topology, you save surface data as well. Every time GridPro reads a surface, whether or not GridPro created that surface, the program defines that surface data using TIL code. Topologies are saved along with the TIL code that defines the surfaces. That is, corresponding surfaces are saved with the topology.

Remember that references to surfaces in TIL code are really numerical labels for surfaces. For example, a large data set may reside in a separate file and define the intricate details of some surface geometry. Conversely, one surface definition in TIL code is just one line that associates a numerical label to the surface data file. Thus, if you want to use a new geometry for a surface, you can perform one of two sets of actions: i) put the new geometry data in the existing file without altering the TIL code ii) put the new geometry data in the new file, then alter the TIL code such that the numerical label is pointing to the new file. In some cases, you will need to swap sur-

face types (and possibly other parameters) in the surface definition line in the TIL code. As a user of the graphic manager, you will not explicitly see the TIL code unless you wish to inspect the '.fra' file which contains it.

You will most likely wish to save the majority of topologies you create. However, some topologies you might wish to save temporarily in a file that will be eventually overwritten. Such topologies you might work on only a few times for testing purposes. Or, perhaps you might be progressing through a step by step construction that you are not ready to save until it is complete or at some acceptable stage. Under the "topo" menu, found in the top toolbar, there are options to save topologies both as file names you specify, or as given file names that will be eventually overwritten. One option, the "TIL save as" operation, allows you to save a topology under a file name you specify. Another option, the "TIL save to _az.fra" operation, allows you to save a topology under the file name '_az.fra' until you overwrite the file. The '_az.fra' file has four backups, '_az.fra. 1', '_az.fra. 2', '_az.fra. 3', and '_az.fra. 4'. When you save a new file to '_az.fra', the last file saved under '_az.fra' becomes a backup file, and the oldest backup file is erased. In essence, GridPro saves five files under the '_az.fra' name, then erases them as new '_az.fra' files are saved.

Figure 4.15



In order to retrieve topologies, access the "TIL read" operation under the "topo" menu and then access the file. You can access '_az.fra' files or user named files using the "TIL read" operation. *All user-named topology files must have the extension '.fra'*. Figure 4.15 displays the position of some operations you have learned so far.

GridPro can load topology files visually in two ways. Still, in either case, retrieving a topology file requires you to select the "TIL read" menu item. The GridPro default allows you to read in a file directly, so that all the data appears in the drawing area at once. However, selecting the "Demo read topo" mode from the top menubar "Read=N" menu will allow you to read in a topology file in a movie format; topology data will stream in one by one, providing a perspective on how the topology was built.

When you prepare surfaces that bound the region for gridding, you must orient the surfaces as well as create topologies about them. Any GridPro surface has two sides, and GridPro will assign them "+side" and "-side" designations. You need to indicate which side of the surface you will grid. This indication represents a surface orientation. If a surface is oriented in the "+" direction, the normal vectors at each surface point points to the "+side" of the surface. If a surface is oriented in the "-" direction, the normal vectors at each surface point points to the "-side" of the surface. In the case of super-ellipsoids, the "+side" normal vectors point radially outward. Thus, if you wish to grid the inside of a super-ellipsoid (with the –ellip type), you must flip normal vectors with the "-" so that they point into the region to be gridded. Thus, we use the "+" and "-" selections to choose the side of the surface that will have the grid on it. Implicit in this selection is the assumption that all surfaces are orientable. That is, there is an inside and an outside to each surface. While there are more than enough orientable surfaces to deal with the boundaries of any physical region, there are surfaces that do not have this property. For example, Mobius strips (i.e. twisted belts) are not orientable since you cannot distinguish their sides. They, of course, are not used here.

Sometimes you may want to define a surface within a grid. For example, you may wish to analyze the heat transfer between a fluid and a solid. In this case, you would have a surface separating the fluid from the solid. That surface would then have grid on both sides. Thus, at each point, there would be two normal vectors: one pointing to each side. So, surfaces that have grid on both sides are internal to the grid and as such are quite naturally called "internal surfaces." In order to impart double orientation to an object, select "2 sides" from the "orient" submenu in the surface's property menu. In the generation process, GridPro will optimize the grid quality and produce a grid sheet that is so smoothly embedded within the overall grid that you will be hard-pressed to find any discrepancy.

In order to discover or to change the orientation of a surface, you first make that surface current, then access the properties of that surface by selecting, "reload current" under the "surf" menu. In the subsequent window, you can change the orientation of the surface using the "orient" submenu.

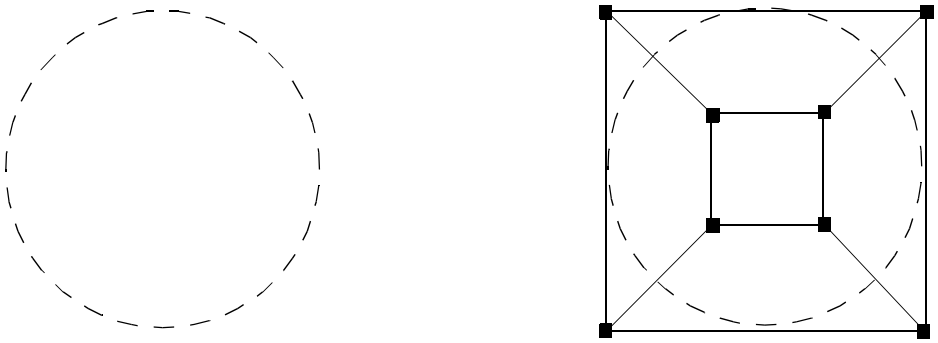
EXAMPLE PROBLEM #3:
Prepare a two-dimensional circle for gridding, and save the resulting topology.

First, because we are working in two dimensions, we should run GridPro from scratch in order to have workable default settings. After doing so and selecting "dim=2" from the top toolbar, we ask ourselves, what region do we want to grid? Because we are gridding only one object, we grid the region inside that object. Therefore, we must give the object a proper orientation. Accessing the "load –ellip" operation under the "surf" menu, we first should determine the surface's orientation. We select "- side" because the natural orientation of the built-in "-ellip" type comes with normals pointing in the outward convex direction away from the region to be gridded.

Since we are working in two dimensions, we are required to set the surface's semi-w axis to a large number. Thus, in the semi-w box, let us input (0, 0, 100000). Keeping the semi-u and semi-v boxes at their default settings of (1, 0, 0) and (0, 1, 0), and leaving the number 2 in the "power" box, we click [ok] to produce the unit circle.

Now that we have created a surface, we can create a topology about this surface. Because the whole boundary of a circle is curved, it is best to wrap the whole boundary. Therefore, this wrap forces us to decompose the circle into at least five blocks. The configuration is shown in figure 4.16.

Figure 4.16

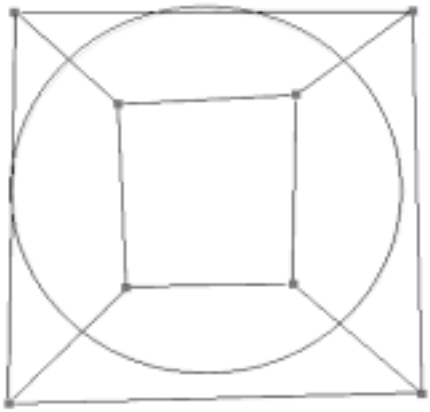


Notice that the circular hole in figure 4.13 is wrapped in the same manner as the one above (most circles have the same topology). We construct a topology similar to that illustrated above by positioning 8 corners, and creating 12 edges among the corners using the <C> and <E> buttons. Remember: we can translate corners and edges by dragging them. Also, if we wish to remove a corner, we can use the <R> key. The corners of the topology do not have to be relative to one another by exact measures, and edges do not have to be precisely perpendicular or parallel to one another.

After creating the topology, we now should assign corners to surfaces. Since there is only one surface, it is current by default. Because we are gridding the region inside of the circle, we assign the outer four corners to the surface. In order to assign most efficiently these corners, let us use the [+] operation. After depressing the [+] button, we drag a purple box over the first column of two corners, assigning them to the surface. Then, we depress the [+] button again and drag a purple box over the last column of two corners, assigning them to the surface.

Now, the surface is ready to be gridded. The actual result (seen on the drawing window) is shown in figure 4.17. Notice how loosely the topology can be positioned about the surface.

Figure 4.17



Because we desire to access this topology only once or twice more, we place it in an '_az.fra' file by accessing the "TIL save to _az.fra" operation under the "topo" menu.

EXAMPLE PROBLEM #3 COMPLETED

EXAMPLE PROBLEM #4:

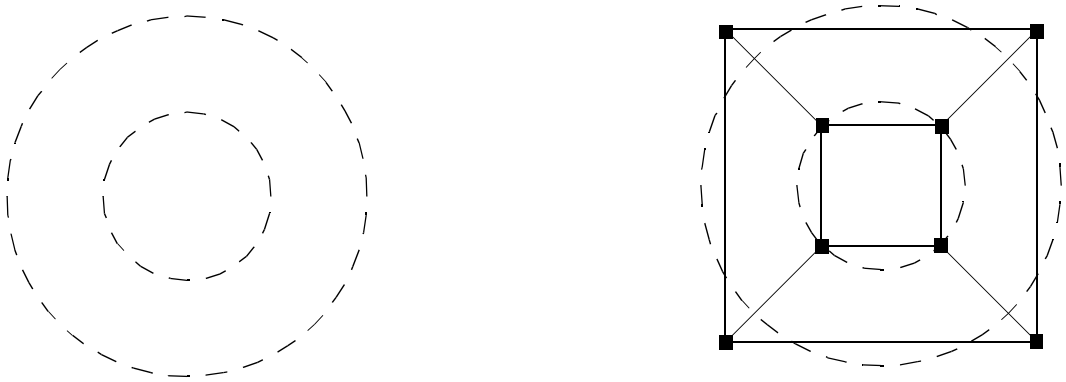
Prepare a 2-dimensional ring with a circular hole for gridding, and save the resulting topology.

Similar to the last example problem, we should run GridPro from scratch in order to have workable default settings. Then, after selecting "dim=2" from the top toolbar, we ask ourselves, what region do we want to grid? Because we suppose that we are working with finite surfaces, we want to grid inside of the ring. Because the inner object is a hole, we must grid the region outside of the inner object. Therefore, we are gridding the region between the two surfaces. The orientation of the ring should be negative (inside) and the orientation of the inner circle should be positive (outside).

In order to create the surfaces in correct proportion to each other and with the correct orientations, we must keep track of the different surface properties. Accessing the "load –ellip" operation under the "surf" menu, we choose to make surface 0 (id number =0) the hole, or the inner surface. We set the semi-w axis to (0, 0, 10000) and make sure the "orient" box reads "+ side." Otherwise, we change nothing else and press [ok], creating a circle of radius 1. Accessing the "load –ellip" operation again, we choose to make surface 1 the ring, or the outer surface. We set the semi-w axis to (0, 0, 10000), the semi-u axis to (3, 0, 0), and the semi-v axis to (0, 3, 0). Then, we change the "orient" box to read "-side," and click [ok].

We now decompose the "donut" region we are gridding. We can split the region into *four* blocks, and then wrap the curved surfaces. Therefore, we can use a wireframe similar to the one in example problem #3. This wireframe is illustrated in figure 4.18.

Figure 4.18



Although this wireframe matches that of example problem #3 exactly, the topology does not match that of example problem #3 exactly; each topology has different surface assignments. At this stage, the region is ready to be gridded. Because we want to save this topology in a temporary file, we now access the "TIL save to _az.fra" operation under the "topo" menu. ***EXAMPLE PROBLEM #4 COMPLETED.***

4.5 The Gridding Process

GridPro is well known for its revolutionary ability to generate grids from a topological basis, one that is not encumbered by detailed geometric demands, decisions or judgements. GridPro requires no manual construction of geometric pieces such as block edges or faces, or their associated pointwise distributions. All gridding is done in one automated process; there is no separate process for surface grid generation or surface to surface intersections. Even within the context of a wireframe, there is no projection algorithm required to send wireframe edges or faces onto the geometric boundary of a physical region. Such projection algorithms require tight constraints upon the placement of wireframe corners - the antithesis of true topology, which relies upon a much looser and more flexible structure.

The GridPro topology engine "Ggrid" provides a map of true topology into multiblock grids. Under its action, the topology code (which is stored in a '.fra' file) is compiled and run under the command "`Ggrid` *`TIL_file_name`* `<return>`". The resulting grid is optimized to be smooth and nearly orthogonal throughout an entire region, and to be properly clustered to a region's boundary curvature. The optimization is, of course, relative to the constraint of topology choice (usually a rather mild constraint). Generally, you will get smooth, nearly orthogonal grids that are appropriately clustered to both concave and convex boundary curvature. The resulting grid is so natural, you will become accustomed to high quality grids every time you use GridPro.

With the interactive Graphic Manager, the entire gridding process is purely visual. The

“Ggrid” launch is started, stopped, and re-launched graphically. The Ggrid process is tightly coupled with interactive topology generation; topology errors are automatically detected and highlighted in red when you try to launch an incorrect TIL code. The highlight appears as an overlay on top of the topology wireframe, and a white box appears over the specific problem corner or corners. The error and highlight display points precisely to the problem area that prevents Ggrid from running. Concurrently, a more technical error message is also displayed at the upper left-hand portion of the screen. Once you know the location of an error, the error and highlight display that helped you find the error will sometimes be in the way of your corrective actions. To remove the highlight, go the green SHOW box and click off the error and highlight button, [e&h]. Once the topology is fixed, you will be given a pop up box that will tell you that your topology is correct and ask you if you wish to run it. When you click [ok], the Ggrid process will commence. However, you will not explicitly see the gridding process itself. Instead, GridPro will display a status window that will dynamically give information about your run as it progresses.

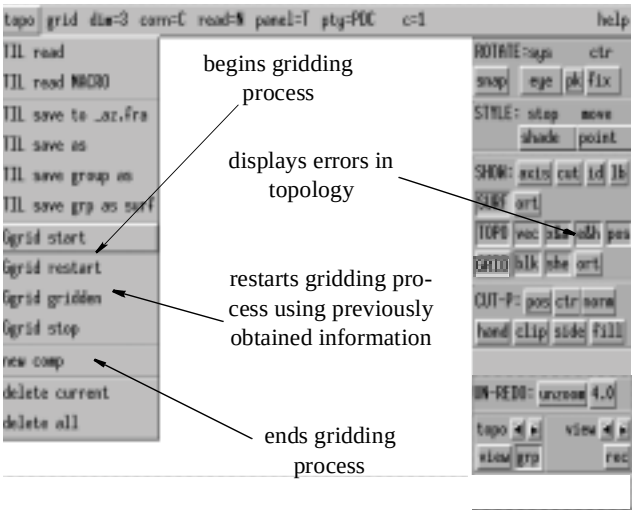
The status window contains seemingly garbled code. However, the numbers displayed convey information about the quality and convergence of the grid being processed. The first three letters shown, “swp,” form an abbreviation for “sweep.” The number proceeding “sweep” illustrates the number of times GridPro has visited your topology in its attempt to optimize the grid in the corresponding region. Every subsequent sweep relies on information processed in previous sweeps in order to improve the quality of the grid. In other words, the more sweeps you allow GridPro to perform, the better your resulting grid.

In order to numerically measure convergence for the grid optimization process, look at the residual number found in the status window. This number follows the “(r” expression found in every sweep, and indicates the disparity between the actual grid and an ideal grid. An ideal one would require a limitation on generality and an infinite number of points. Because this ideal is unobtainable, GridPro compromises by creating the best possible grid for the chosen topology, a grid that is consistent with the number of points and specified parameters of the topology. As the grid generation process converges, the grid’s residual number becomes constant. As the number of sweeps increase, the residue number will often oscillate and then converge to a constant. Often, however, the residual does not completely converge. Still, the resulting grid quality is far better than what you would expect from traditional methods.

Sometimes, if the gridding process is stopped too quickly, it will produce a grid whose coordinate sheets fold in on themselves at some location or locations. Therefore, you must allow GridPro to sort out all surfaces. In the status window, a stream of consecutive sweep messages are periodically interrupted by the reading, “fold count.” When the number following “fold count” reaches zero, all surfaces have been sorted out. The fold count gives the user a coarse means to judge the convergence of the process. Generally, in the convergence, non-zero fold counts appear only in the early stages, well before it is worth while to inspect the residual; the residual becomes the main focus of your monitoring once you are at zero folds. Your attention is then redirected to seeing when the residual settles down to a constant value. While the grid is

converging, you have the option to take the grid for use at that intermediate stage or just wait until a more full convergence of the optimizer has taken place.

Figure 4.19

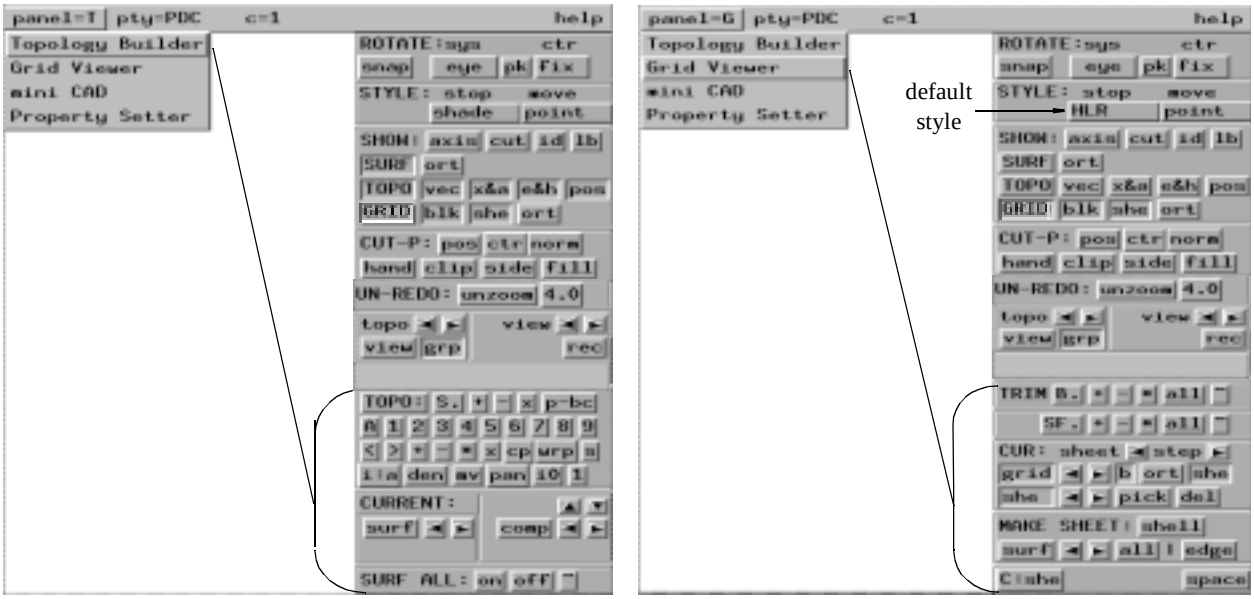


The gridding process is controlled by four “Ggrid” operations found in the “topo” menu from the top toolbar. The “Ggrid start” operation begins the gridding process, and the “Ggrid stop” operation ends the gridding process after the current sweep is completed. If you end the gridding process, then decide that you need a more converged grid, you may resume gridding *using grid information already processed* by accessing the “Ggrid restart” operation. The “Ggrid gridden” operation will be discussed later. Figure 4.19 displays the location of these operations.

As mentioned above, GridPro will refuse to begin the gridding process if a topology is incorrect. Common mistakes people make include not assigning corners to surfaces, not assigning corners to the right surface, and failing to create wireframe quadrilaterals or hexahedrons. Also, the topology itself could be implausible for a given region. If you make a mistake, GridPro will guide you to that location by automatically turning on the error and highlight (the [e&h] button which is found in the green, “SHOW” box) which serves to show your mistake. Note: errors are displayed only when you attempt to start the gridding process of an incorrect topology.

You can load a grid manually or let GridPro load a grid automatically. Before you do load a grid, it is convenient to first go to the “panel=T” menu and select the “Grid Viewer” mode. Now you get “panel=G” in the top tool bar along with new operations in the switchable operations area of the control panel. These operations all relate to the grid viewing process. Figure 4.20 illustrates the effect on operations of switching panels. Also, notice in the figure that switching panels will change the defaults in the “STYLE” box; hidden line removal is the default grid viewing style.

Figure 4.20



During the gridding process, after every 100 sweeps, GridPro saves the grid to the file, ‘blk.tmp’. Therefore, after you stop the gridding process, you can load the grid by accessing the ‘blk.tmp’ file. Under the “grid” menu, found in the top toolbar, the “load new” operation opens a window that allows you to find and access the file, ‘blk.tmp’. Note: you need not stop the gridding process in order to access the grid. In fact, since the file, ‘blk.tmp’, is being updated after every 100 sweeps, you can reload an updated version of ‘blk.tmp’ by accessing the “reload current” operation found in the “grid” menu. GridPro will automatically perform the “reload current” operation after every 100 sweeps if you select the “auto read grid” mode in the “read=N” menu. In this same menu, the “normal read” mode (the default mode) allows you to perform the manual loading operations stated above. Note that when you select “auto read grid,” the “read=N” designation becomes “read=A”.

The Ggrid process really represents the running of a schedule file, with extension ‘.sch’. Thus, when transfering topology data from one terminal to another, be sure to include the schedule file with the ‘.fra’ (topology) file.

Like topologies, grids can be saved under file names. Because GridPro saves grids to the default file, ‘blk.tmp’, you must alter the name ‘blk.tmp’ when you retrieve your grid if you wish to save it under a user name. To assign a user-name to a grid, access the “save grid as” option in the “grid” menu (located on the top menubar). You can arbitrarily choose the extension name for any grid file. By altering the file name, you will be able to retrieve your file whenever you wish because all other grids produced will be filed under ‘blk.tmp’.

Each grid file is accompanied by a connectivity file. This connectivity file specifies how the grid blocks are connected, and has the extension ‘.conn’. GridPro automatically runs the con-

nectivity file when you access ‘blk.tmp’.

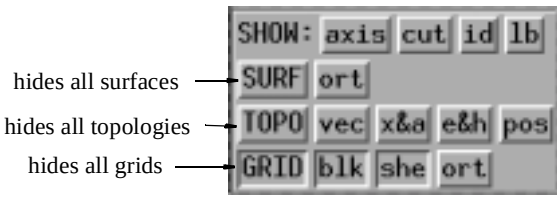
4.6 Grid Viewing

Aside from boasting a faster gridding process and higher quality grids than all of its competitors, GridPro attains a high watermark in the area of viewing and manipulating grids. Cross-sectional cuts, block coloring, and partial shells are some of the many features that can be displayed in the “Grid Viewer” panel. In this mode, the grid stays invariant as you manipulate its various features.

When you load a grid, the outline of the grid appears first. At first, this outline looks similar to its corresponding topology; always keep in mind the mode you are in (“Topology Builder” or “Grid Viewer”). GridPro allows you to manipulate this grid visually and to provide the grid with property labels before releasing it for engineering analysis. The “Grid Viewer” can hold many grids in the same space; if you do not remove a previous grid from the “Grid Viewer,” a current grid you load will overlap the previous grid. Just as you make surfaces current, you can make grids current. The arrow buttons to the right of the [grid] button (found in the green, “CUR” box) toggle between the grids. The grid which contains some sea blue color is the current grid. The [grid] button itself, when depressed, hides the current grid. The [GRID] button (found in the green, “SHOW” box), when depressed, hides all grids. In the “grid” menu, the “delete current” operation eliminates the current grid and the “delete all” operation eliminates all grids in the “Grid Viewer.”

Notice in figure 4.21 how the green, “SHOW” box acts as a global control center; the box contains operations that can hide or show surfaces, topologies, or grids. You should also think of the “SHOW” box as a screen clutter manager; every button hides something from the drawing area.

Figure 4.21



In order to see quickly the overall grid, or mesh, press the [shell] button found in the brown, “MAKE SHEET” box. Pressing the [del] button in the green, “CUR” box will eliminate this mesh display.

EXAMPLE PROBLEM #5:

Retrieve and grid the topology and surface from example problem #3, then save the resulting grid.

Because the region in example problem #3 is two dimensional, we should exit and reenter the az graphic manager in order to begin with the proper default settings. After reentering the az manager, we are automatically put into the “Topology Builder” mode. You can confirm this

default mode choice by observing the “panel=T” on the top menubar and the associated switchable operations relating to topology on the lower part of the command panel. Then, in order to access a topology, we select the “TIL read” operation in the “topo” menu. Because we saved the topology from example problem #3 as an ‘_az.fra’ file, and have saved the topology from example problem #4 similarly, we look for the most current backup ‘_az.fra’ file. We access each of the four backup files until we find the desired topology and display it on the screen. As we look for the desired topology, we successively eliminate unwanted topologies, should they be brought in by mistake.

After checking to see if we have correctly assigned corners to surfaces, we access the “Ggrid start” operation under the “topo” menu. A window should appear and inform us that our current topology is complete. After pressing [ok] in this window, the status window should start displaying the results of sweeps. Most likely, we will have to translate and shrink the main window in order to see the status window. Although we could wait only until the fold count equals zero and the residual number varies to within a couple thousandths of a point, here let us wait until 500+ sweeps have been completed before we stop the process. After ending the gridding process by accessing the “Ggrid stop” operation in the “topo” menu, we look in the status window and observe closely the residual numbers produced from successive sweeps and the fold counts produced between sets of sweeps.

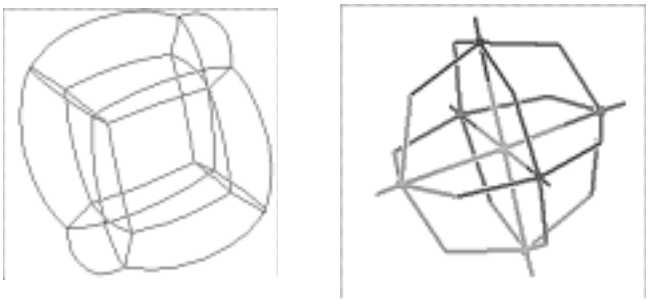
In order to see our grid, we first access the “Grid Viewer” mode by going to the “panel=T” menu and switching it to “panel=G.” Then, we access the “load new” operation in the “grid” menu. In the resulting window, we find the file ‘blk.tmp’, access it with a click, and accept it with a click on [ok] which then displays the outline of the grid on the screen. In order to see a mesh representation of the outline, we press the [shell] button. ***EXAMPLE PROBLEM #5 COMPLETED***

All grids, two-dimensional and three-dimensional, contain blocks. You determine the number and general configuration of blocks through the topology you create. For example, the wireframe topology of a circle includes five quads. Therefore, the grid for a circle will be composed of five blocks positioned relative to one another in the same fashion as the topology quads. However, each “side” of the blocks will be curved so that the blocks conform to the circle’s geometry and represent the current geometry of the block regions. With these regions, each of the internal edges will likewise be curved, but without a geometric constraint. Altogether, these curved edges represent the outlines of the block structure. Quite naturally, you can see extra information in the block edges (curved lines) as compared to the topology (straight lines). With the uniqueness of straight lines, the topology wireframe is determined by only a sequence of corners; the block edges in the resultant multiblock grid are too complex to be distilled to such a simple level.

All grid points, which are displayed when the [shell] button is depressed, define “miniblocks,” or cells. Corners are to blocks as grid points are to cells.

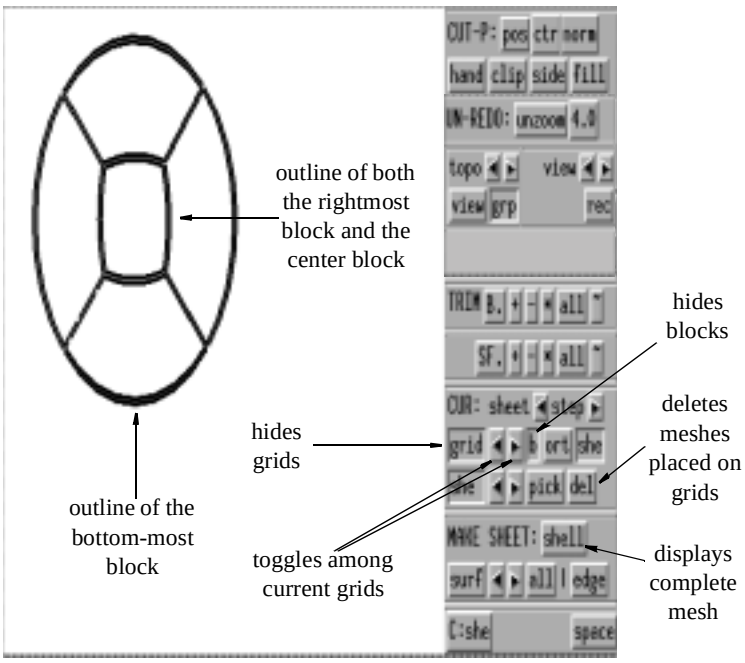
Each block has a center, outline, and a skeleton. The center of each block is the center of mass of each block, and is represented by a small square. The outline of a block is simply the boundary of the block formed by its edges. The skeleton of a block is the lines connecting the indexed center of the block with the centers of the block's bounding edges (2D) or faces (3D). Each block skeleton is drawn in its own color. When collected, each block skeleton forms a skeleton for the entire multiblock grid. The skeleton represents the minimal amount of information required to show a broad representation of a multiblock grid. Also, the skeleton illustrates the "flow" of the block structure better than the outline. In three dimensions, the number of lines produced by a skeleton is 6, while the number of edges of a block is 12. Therefore, you can greatly remove clutter in large grids by activating block skeletons. Figure 4.22 displays the broader physical representation provided by the skeleton. The flow is the block connectivity that is witnessed as simple curves in the skeleton. In the case of the circle with a hyperquad topology, the skeleton shows one circular loop and four radial like lines. You can view grids as a collection of outlines and/or a collection of skeletons. However, you best use only one of these display forms at a time in order to avoid screen clutter.

Figure 4.22



block outlines block skeletons

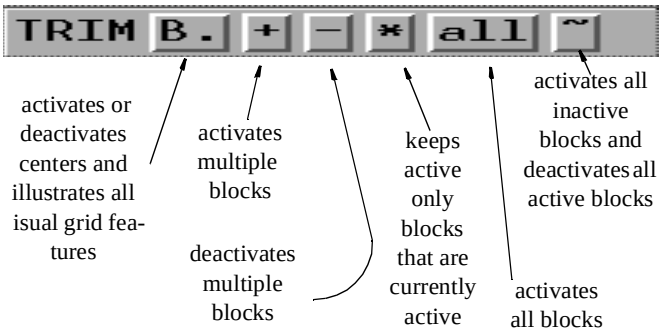
Figure 4.23



The outlines of blocks are the first visual features of a grid you will see. The outline of each block is displayed once you retrieve the grid from the 'blk.tmp' file. By depressing the [B.] button in the brown, "TRIM" box, you make the skeleton appear in addition to the outline. Therefore, if you want to see just the outline of a grid, leave the [B.] button un-depressed. If you want to hide the outline of a grid, depress the small [b] button in the green, "CUR" box. Thus, if you want to see just the skeleton, first depress the [B.] button (to show both outlines and skeletons), then depress the [b] button (to hide the outlines). Figure 4.23 displays the location of previously mentioned operations.

All the operations in the [B.] row deal with activating and deactivating blocks. In order to deactivate a block, deactivate the skeletal center of the block. If a block's center is deactivated, the outline of the block will not be shown (when the [b] button is on). However, the skeleton of a deactivated block will still be displayed if the [B.] button is depressed. Visually, the cut out blocks will allow you see the inside of (3-dimensional) grids as solid objects or to see slices of the grid which do not propagate beyond the active blocks. If a block skeletal center is a filled, colored square, the center is activated. If a center is a hollow, colored square, the center is deactivated. The [shell] operation places a mesh on the exterior boundaries of the activate blocks. The exterior block boundaries are those that have only one active block attached to them: not two. The result of pressing [shell] is the display of the active blocks as if they were a solid object. The propagation is limited to the active blocks.

Figure 4.24



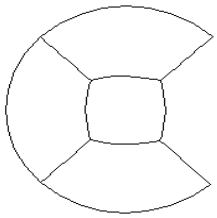
Because most grids will contain numerous blocks, you will often need to activate or deactivate many blocks at a time. If you require point by point, or center by center, activation or deactivation, depress the [B.] button and click on whatever centers you wish to activate or deactivate. The [+] button to the right of the [B.] button functions similar to the other [+] you have encountered: when depressed, the button allows you to create a purple box with the right mouse button and enclose the box around all centers you wish to activate. The [-] complements the [+] button. When depressed, the [-] button allows you to create a purple box with the right mouse button and enclose the box around all centers you wish to deactivate. Remember that you must drag the purple boxes to enlarge them. The [all] button, when depressed, simply activates all centers. The [~] button, when pressed, activates all deactivated centers and deactivates all active centers. The [*] button is an intersection operation. Similar to the [+] and [-] operations, the [*] operation allows you to create a purple box and enclose it around centers. Any centers not enclosed by this box will be deactivated. Activated centers enclosed by this box remain activated and deactivated centers enclosed by this box remain deactivated. Figure 4.24 illustrates the operations in the "TRIM" box.

Due to the design of the grid viewer, the possibilities for observing grids are virtually endless. Figure 4.25 displays different ways to view grids, and describes what is shown and how to show it.

Figure 4.25

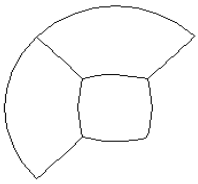
- outlines of five blocks are displayed.
- "b" button is depressed

Partial Grid



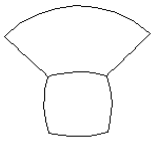
- outlines of four blocks are displayed.
- rightmost block is deactivated.
- "b" button is depressed

Partial Grid



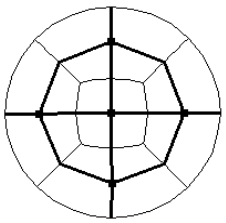
- outlines of three blocks are displayed.
- rightmost and bottom-most blocks are deactivated.
- "b" button is depressed

Partial Grid



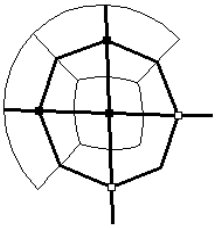
- outlines of two blocks are displayed.
- rightmost, leftmost, and bottom-most blocks are deactivated.
- "b" button is depressed.

Partial Grid



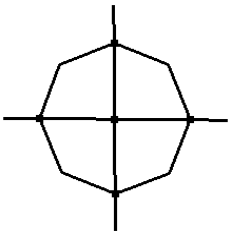
- outlines, centers, and skeletons of five blocks are displayed.
- "b" button is depressed
- "B" button is depressed

Partial Grid



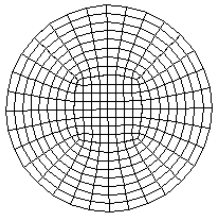
- outlines of three blocks, and centers and skeletons of five blocks are displayed.
- rightmost and bottom-most blocks are deactivated.
- "b" button is depressed
- "B" button is depressed

Partial Grid



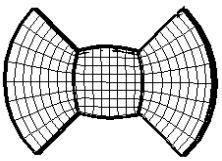
- centers and skeletons of five blocks are displayed.
- all blocks are deactivated.
- "b" button is depressed
- "B" button is depressed

Partial Grid



- complete mesh is displayed.
- "shell" button has been pushed
- "b" button is depressed.

Complete Grid



- outline of three blocks, and a partial mesh, is displayed.
- upper-most and bottom-most blocks are deactivated.
- "shell" button has been pushed

Partial Grid

EXERCISE #12:

Retrieve the grid from example problem #5, deactivate any four blocks two different ways, and encase the resulting figure in a shell.

EXERCISE #13:

Grid the topology saved in example problem #4, deactivate the four innermost blocks two different ways, and encase the resulting figure in a shell.

PART 2
BEYOND
THE BASICS

Chapter 5 - Constructing Topology

5.1 Working in Three Dimensions

All real-life configurations have three dimensions. Therefore, in order to use effectively GridPro, you must understand the structure of three-dimensional topology and become proficient in manipulating three-dimensional topology.

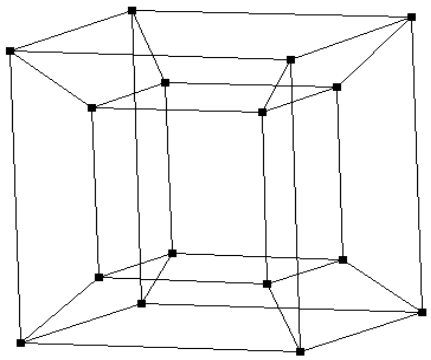
At a fundamental level, topology in three dimensions is much like topology in two dimensions. Also, the process of creating a wireframe, assigning corners to surfaces, gridding the topology, and manipulating the grid in three dimensions mirrors the process in two dimensions. The first step in creating any topology is always to define the region which must be gridded. Then, boundary geometry is decomposed into surfaces, and the surfaces describe a block pattern; the resulting wireframe topology must represent this pattern. In three dimensions, although the rules governing surface assignment do not change, you will need to assign many more corners. Also, the gridding process does not change. However, you will receive error messages much more often. Therefore, you must be more meticulous when planning topologies and assigning corners to surfaces.

GridPro boasts powerful three-dimensional graphics. By utilizing these graphics to your advantage, you can create topologies for virtually any geometry. Rotation increases depth perception; because viewing a three-dimensional scene on a two-dimensional screen can disorient you, constantly rotate images so that they maintain their three-dimensional appearance. Also, remember that the <Q> key allows you to view the coordinates of your cursor on the cut plane. The <Q> key is especially useful when the cursor is in between many clustered together corners and edges.

Because topologies in three dimensions are sufficiently complex, GridPro provides tools to help you create these topologies. The first, and most important tool, is the cut plane. The cut plane allows you to place corners anywhere in space. Also, the cut plane helps you orient yourself. Remember to use the [fill] operation in order to distinguish the cut plane from the surrounding topology and regional geometry.

EXAMPLE PROBLEM #6:
Prepare an ellipsoid for gridding, and save the resulting topology.

Figure 5.1

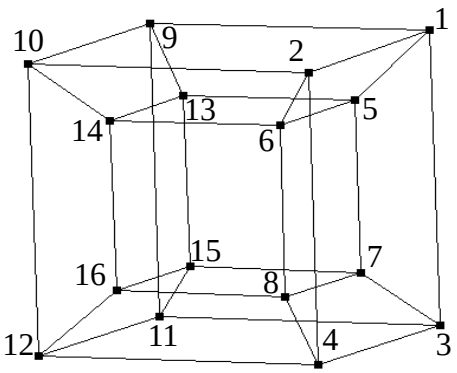


After accessing the “Topology Builder” panel (in the “topo” menu) and the “dim=3” mode (in the “dim=” menu), we must think about what topology we should create. Keeping in mind that all three-dimensional topologies are composed of hexahedrons, we visualize the topology composing seven blocks: one cube enclosing the ellipsoid, and six hexahedrons serving as a wrap for the curved border of the ellipsoid. Figure 5.1 displays the resulting shape, commonly known as a hypercube. We notice that the hypercube has 16 corners and 32 edges, a considerable amount of topological componentsforonesymmetricallysimplesurface(theellipsoid).

There are few properties of the ellipsoid we should change from the default settings. In fact, we only should change the surface orientation; because we are only gridding one surface, we grid inside that surface. We access “load –ellip” under the “surf” menu, then choose “- side” in the orient box, and press [ok].

Now, we must create the topology illustrated above about the ellipsoid. As shown in figure 5.2, if corners are grouped into the sets {1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11 12}, and {13, 14, 15, 16}, the points in each set are coplanar.

Figure 5.2



A total of four planes contain the 16 corners of the topology. Therefore, we must align the cut plane with each of the four planes above. At each alignment, we input four corners at the positions illustrated above. We should remember that exact positioning of corners and exact alignment of cut planes is unnecessary because GridPro grids loosely positioned topologies. The topology does not have to be exactly symmetric and our point clicks do not have to be precise.

In order to access and be able to maneuver the cut plane, we depress the [cut] and [hand] buttons, found respectively in the green, “SHOW” box and the brown, “CUT-P” boxes. We remember that the cut plane can be translated or rotated about one of its axis, and can be dilated by dragging its blue vertex squares.

Figure 5.3

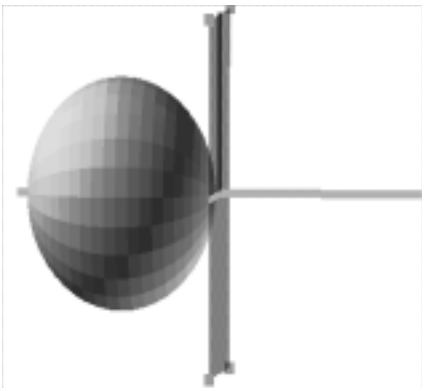


Figure 5.3-5.10 displays the steps in creating the topology. We maneuver the cut plane so that it lies almost tangent to the ellipsoid (figure 5.3). Then, we rotate the surface and use the [snap] menu (found in the green, "ROTATE" box) in order to align the cut plane with the plane of the drawing window. Finally, we create four corners, and connect them with edges (figure 5.4). Now, we repeat this process, but first must move the cut plane to a different position (figure 5.5). In this position, we create four corners and connect them with edges (figure 5.6). We must realize that the four corners we now input are on a different plane than the four corners we previously inputted even though all corners look like they are on the same plane. By depressing the [fill] button, we can better visualize the plane in which we are working (figure 5.7). Now that we have created eight corners in two different planes, we have finished creating corners for one "side" of the ellipsoid (figure 5.8). We repeat the procedures above in order to obtain the same general positioning of corners on each side of the surface (fig-

Figure 5.4

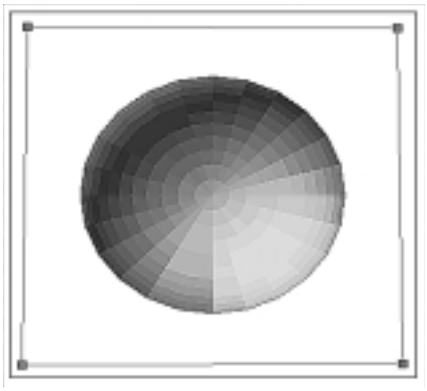


Figure 5.5

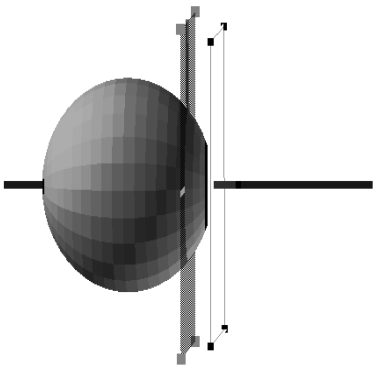


Figure 5.6

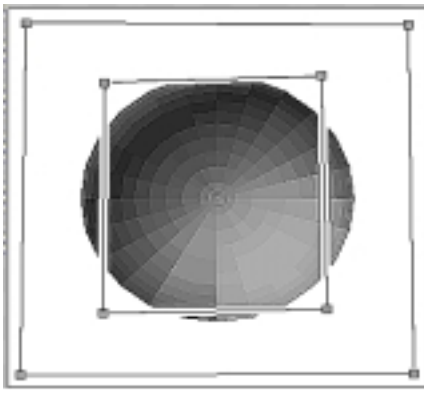


Figure 5.7

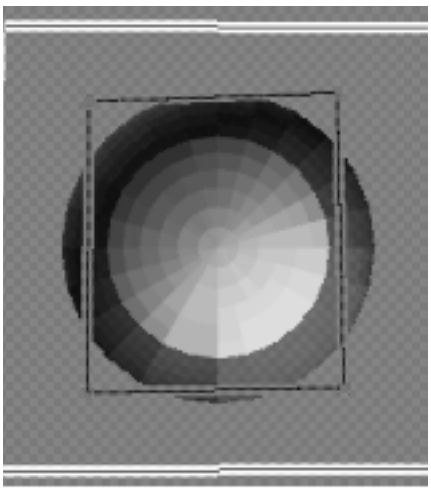
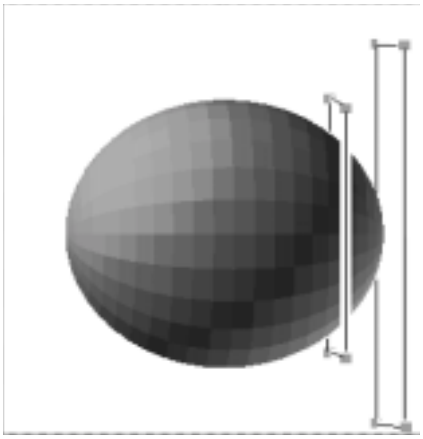


Figure 5.8



ure 5.9). By correctly linking all corners with edges, as we first outlined, we complete the wireframe (figure 5.10).

Figure 5.9

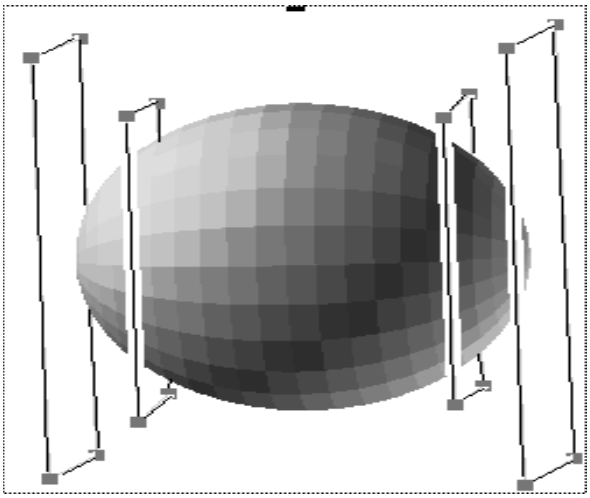
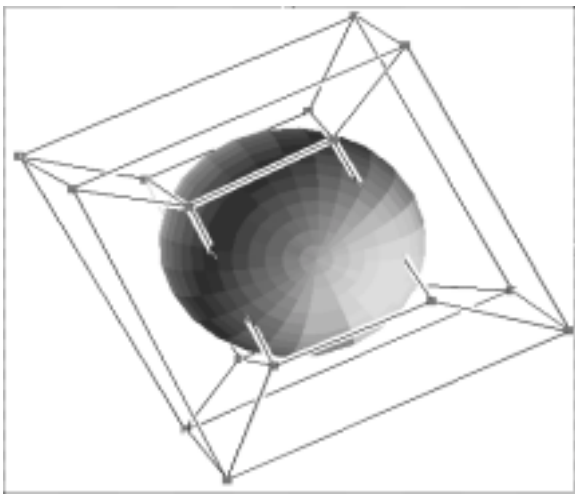


Figure 5.10



However, a completed wireframe does *not* represent a completed topology. Corners must still be assigned to the surface. Because we are gridding the region inside of the ellipsoid, we must assign the outermost corners to this ellipsoid. Using the numerical identification labels for all corners from figure 5.2, we conclude the outermost corners are those in the set, {1, 2, 3, 4, 9, 10, 11, 12}. Therefore, we assign these corners to the surface. Because the [S.] operation involves assignment of individual corners, we use the [+] operation in order to quicken the assignment process.

Now that the topology is complete, we save it under a user-defined file name. In the "topo" menu, we access the "TIL save as" operation, create a file name, and press [ok]. ***EXAMPLE PROBLEM #6 COMPLETED***

EXERCISE #14:
Prepare a superellipsoid of power 10 for gridding, and save the resulting topology.

5.2 Introduction to Groups

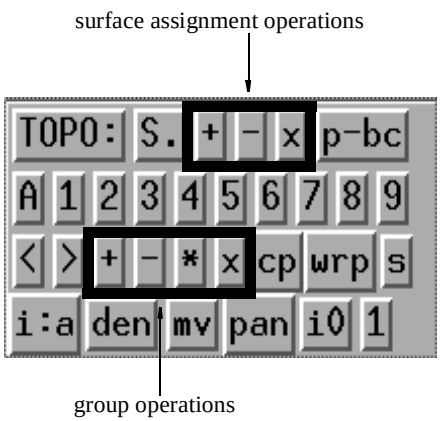
One tool that helps you create three dimensional topologies is the cut plane; another tool is a group. *Groups are, simply, collections of corners and edges*. Groups allow you to create topologies many times faster than does sequential corner by corner, edge by edge constructions on the cut plane. Often, you will repeat procedures for creating topologies (in the last example problem, corners were inputted four separate times at four different positions of the cut plane) because grid topology often requires the user to repeat operations. GridPro has developed the group feature in

order to help you circumvent the procedural repetition so prevalent in the field of grid generation.

More than any other feature, groups help you manipulate three-dimensional topology and use graphics effectively. Creating a typical topology composed of hundreds of corners and edges without utilizing groups is ill-conceived, time consuming, and unnecessary.

Before learning what a group can do, you should first learn how to access a group. A maximum of nine groups can exist for any topological components in a single topology file. Each group is numbered, one through nine. The corresponding buttons, one through nine, appear in the brown, "TOPO" box. When you depress one of these buttons, the number on that button activates that number group. Therefore, when the [6] button is depressed, group #6 is active. Only one group can be active at a time.

Figure 5.11



Corners can be added to, or removed from, only *active* groups. Initially, no corners exist in any group. Think of groups as empty buckets. Therefore, at first, you can only add corners to groups. After depressing a (one through nine) button, you may add corners to that corresponding group by depressing the [+] button found below the [2] button. Like all other [+] features, this [+] operation allows you to drag a purple box with the right mouse button about corners. The corners within this box will be assigned to the group number depressed. Be careful to use the correct [+] when assigning corners to groups. Often, people depress the [+] button to the right of the [S.] button instead, producing an undesired result. Also, notice that unlike assignment of corners to surfaces, adding corners to groups cannot be accomplished by clicking on individual corners; you must use purple boxes. Just as the [+] button allows you to add corners to groups, the [-] button lets you remove corners from groups. Similar to other [-] buttons you have encountered, this [-] button requires you to drag a purple box around corners. Of course, you can only take corners out of groups to which you have already added corners.

Similar to the block-related [*] and [x] operations you have seen, the group-related [*] and [x] operations help you more efficiently add corners to, or remove corners from, groups. Both buttons are located in the brown, "TOPO" box. The [*] button is found directly below the [4] button, and the [x] button is found directly below the [5] button. When a group already containing corners is active, depressing the [*] button allows you to drag a purple box with the right mouse button. Any corners inside this box that are part of the active group will remain active. All other corners, no matter their position or state, will be removed from the group (if they are not removed already). When a group already containing corners is active, pressing the [x] button will automatically remove all corners in that group.

The button located above the [<] button helps you visually distinguish active groups from

the surrounding topology. This button will display [A], [G], or [R]. Each letter represents a topology mode, one that controls how active groups are distinguished from inactive groups. By repeatedly pressing the button, you scroll through the three modes. When a group is active, and the button displays [A], the corners in the active group and the edges connected between those corners are highlighted while all of the surrounding topology is *dimmed*. When a group is active, and the button displays [G], the corners in the active group and the edges connected between those corners are highlighted while all of the surrounding topology is *hidden*. When a group is active, and the button displays "R," the corners in the active group and the edges between those corners are highlighted while all of the topology in *group #1* is *dimmed*. Figure 5.12 illustrates some examples.

In particularly complex topologies, you may have difficulty focusing on one set of corners. By setting the button mode to [R], you localize your work, eliminate screen clutter, and effectively manipulate specific corners. If you want to dim topology contained in many different groups, add the topology in these groups to group #1 (using the [+] button), and set the button mode to [R]. Because group #1 is linked with the [R] mode, this group is called the reference group. For example, suppose you want to work on the topology surrounding the wing of a plane. In the drawing area, the topology of the whole plane is displayed. First, isolate the topology surrounding the wing by adding this topology to group #1. Then, toggle the button mode to [R]. Now, when you work with active groups *within* the wing topology, the inactive topology groups of the wing will be present but dimmed, while the topology for the rest of the plane will be hidden.

The [s] button, located below the [9] button in the brown, "TOPO" menu, is a special group. You might consider it a tenth group. However, this group *always contains the corners assigned to the current surface*, and is thus named the surface assignment group. Therefore, you can change the contents of the group just by changing the current surface. Often, when surfaces are close together, the corners assigned to each surface appear to be right on top of one another. Using the [s] button will allow you to pick out the corners assigned to one of those surfaces, and allow you to manipulate *just* the wireframe about that surface. The [s] group contains all other properties common to groups.

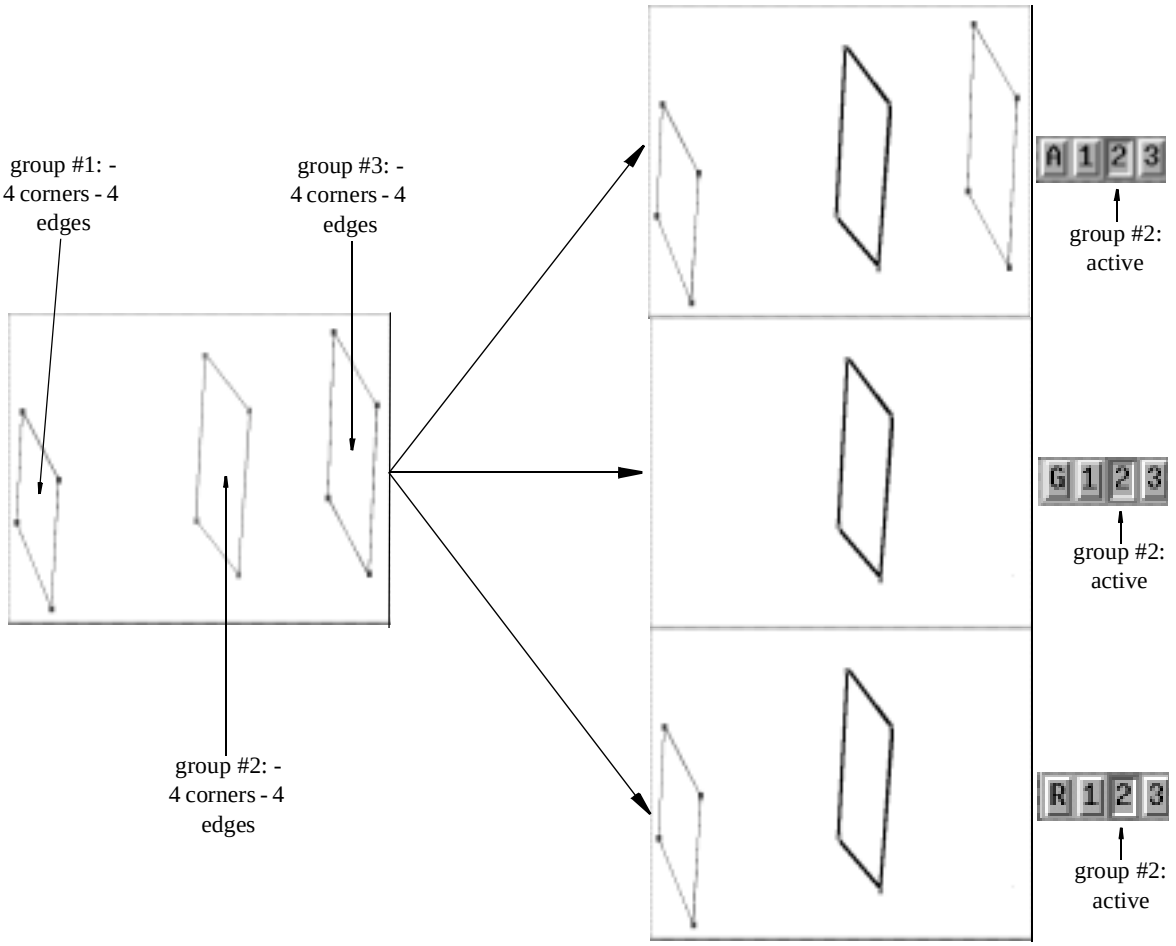
To save the topology in a group as a topology file, select "TIL save group as" from the topo menu in the top menubar.

Notice that "TOPO" is a button as well as the title of the brown box in the "panel=T" mode. The "TOPO" button enables you to visualize better a wireframe. When you depress the "TOPO" button, the secondary wireframe appears. The secondary wireframe has brown (not yellow) edges, and yellow (not orange) corners. This wireframe represents nothing more than a visual aid; you can reposition corners and edges in the secondary wireframe without affecting the original wireframe. If you are afraid that repositioning some components of your original wireframe might irrevocably damage it, perform this repositioning with the secondary wireframe to

see the effect. If you approve of the effect, then you can change your original wireframe according to the secondary wireframe.

Often, users take advantage of the “TOPO” button when manipulating complex wireframes. The best way to untangle a complex wireframe is to create orthogonality among the edges. In other words, repositioning corners in a rectilinear fashion can help you identify an error with the topology or see where another layer of edges must be inserted. To reposition corners in such a fashion without permanently altering your original wireframe, use the secondary wireframe.

Figure 5.12



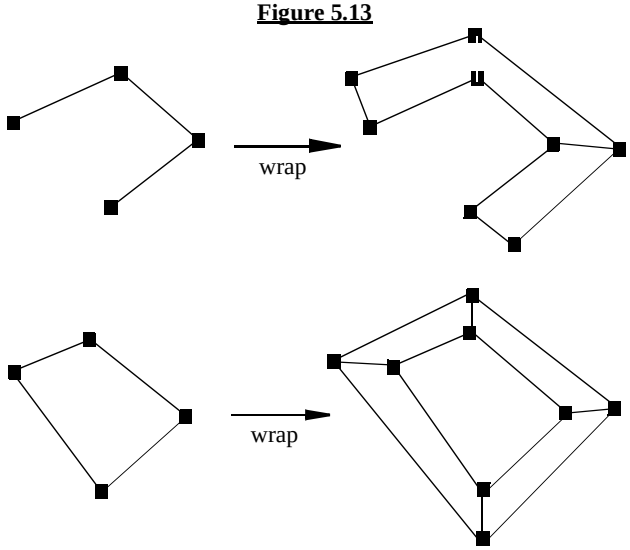
5.3 Wrapping with Groups

Groups often confuse beginning users because groups, at first, may appear to be abstract. In effect, the activation of a group is like the creation of an empty bucket. This empty bucket discards its abstract label when you fill the bucket with corners and edges. As already described, you can add corners to groups with the [+] button and the purple box created by dragging the cursor

over the corners that you wish to add.

Correctly thought out topologies for most surfaces include many wraps. You first saw wraps in figures 4.6 and 4.7. Two more examples are displayed in figure 5.13. A wrap is a dilated or shrunk duplication of a string of segments or quadrilaterals translated from and linked to the original string such that no links intersect one another. Because wrapping strings or faces of topology is common and can be a laborious process, GridPro has developed a feature that allows you to create a wrap automatically using groups. Since you must place a wrap around every curved boundary, as well as most boundary angles, you will more likely be wrapping wireframe than not.

Figure 5.13

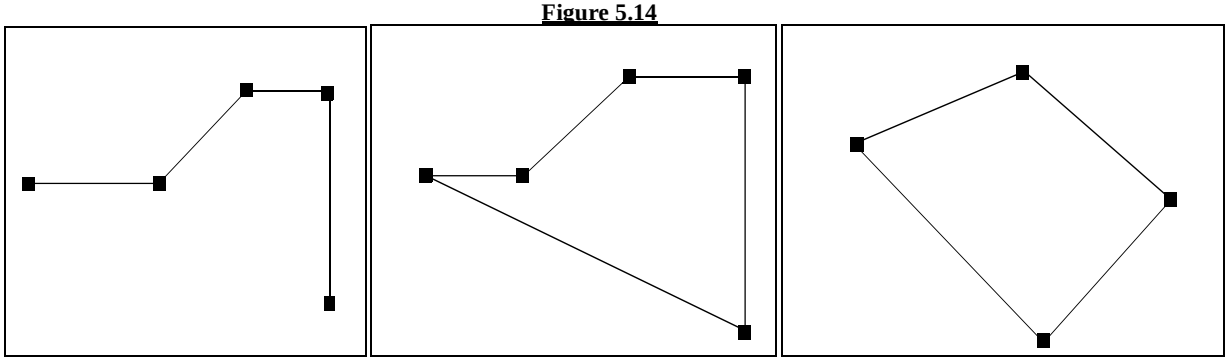


Before you learn how to wrap, you must first learn what to wrap. On a plane, a closed or open string of segments is wrappable. Each corner in the string must be the endpoint of one or two edges, not three or more edges. All edges must be part of the same string, and no more than two corners can define an edge. There is no restriction on the direction of the string, or of the angles formed from connected edges. In space, a closed or open string of quadrilaterals is wrappable. Each connected edge in the string must be common to one or two quadrilaterals, not common to three or more quadrilaterals. All quads must be part of the same string. There is no restriction on the direction of the string, or of the angles formed from connected edges.

Sometimes, you will want to exclude a face of a wireframe in order to achieve a certain wrap. Excluded faces contain a diagonal represented by a dashed red line; these faces are ignored by GridPro during the wrapping process. Thus, a string of one-dimensional segments, which form a rectangle, could be considered a two-dimensional quad if the face is not excluded. If you depress the “F” key and click two opposite corners of a face, a dashed red line will appear, signifying that the face is excluded. Section 5.7 will cover face exclusion extensively.

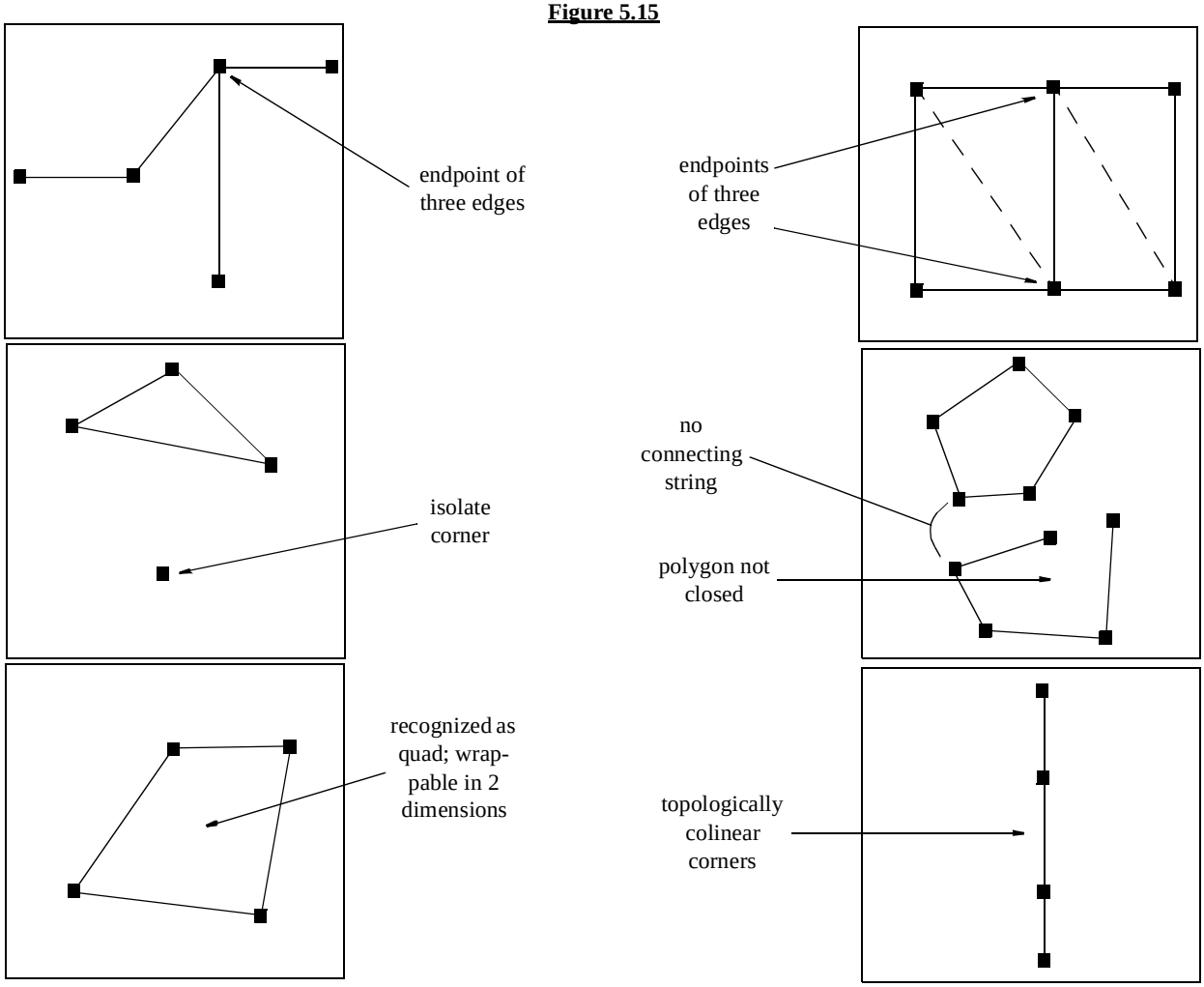
A wrap is better understood by sight than by explanation. However, the definition provides a good backdrop to the pictures. Figure 5.14 illustrates wrappable one-dimensional objects, and figure 5.15 illustrates unwrappable one-dimensional objects.

Figure 5.14



all objects wrappable in one dimension

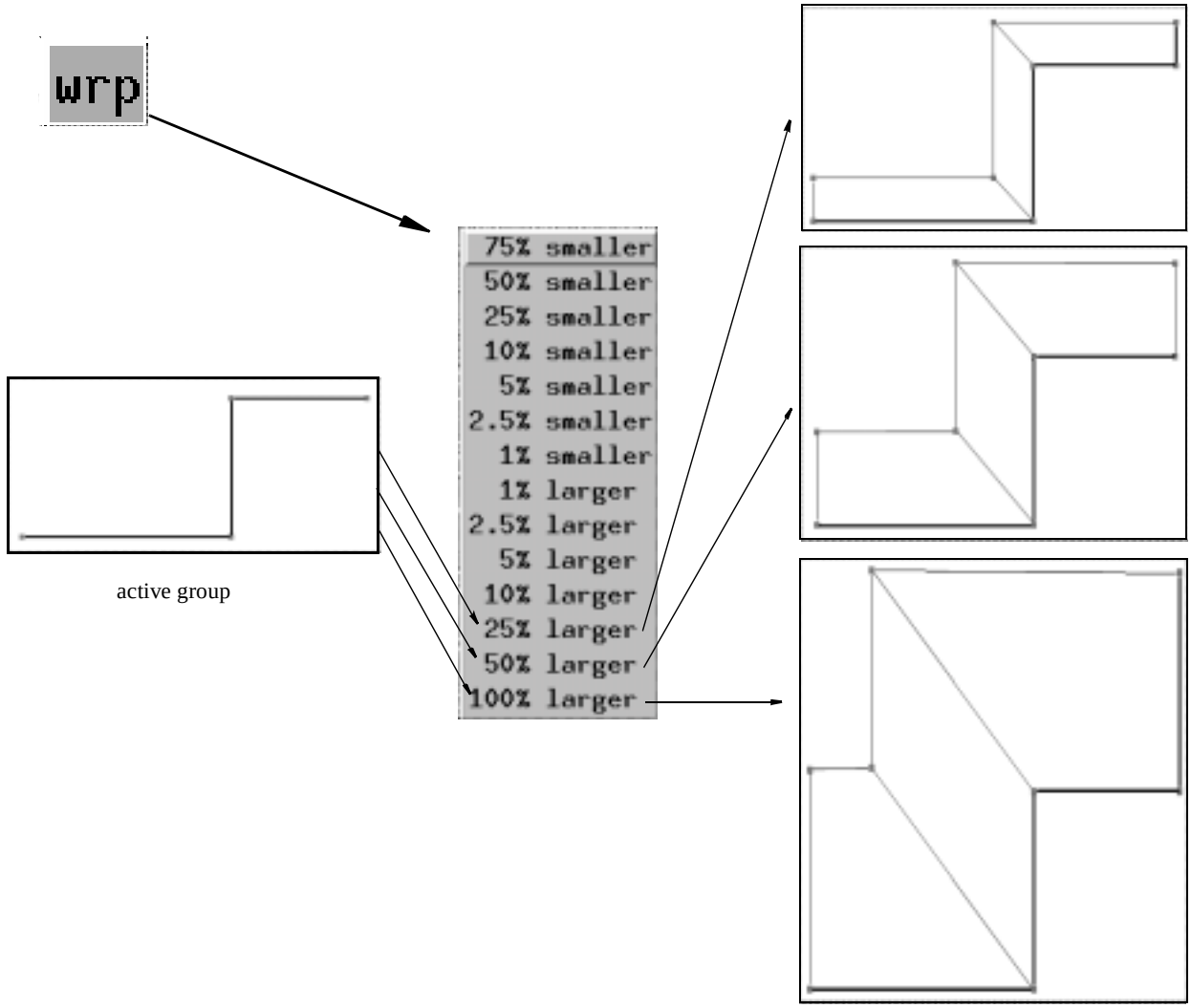
Figure 5.15



GridPro can automatically wrap pieces of a wireframe in one swift topological action; you do not need to input the corners and edges of a wrap one by one. However, these “pieces” of wireframe must be “wrappable,” and must be inside a group. If these conditions are met, you

have a wrappable group. In order to wrap a group, activate the group, then press the [wrp] button found in the brown, “TOPO” menu. The button activates the submenu shown in figure 5.16. By selecting one of the values in the submenu, you can dilate or shrink your wrap by the percentage shown. A message, found on the top left of the viewing window, will inform you if a group is “wrappable” or “unwrappable.” . .

Figure 5.16



5.4 Editing and Projecting Groups

Because most topological operations are done with groups, you might wish to retrace previous operations as group actions. Rather than scrolling backwards corner by corner or edge by edge, you can scroll backwards and see just group actions by depressing the [grp]/[one] button, found in the green, “UNDO-REDO” box. When this button is depressed, the expression, [grp], appears. Conversely, when the button is not depressed, the expression, [one] appears; the un-

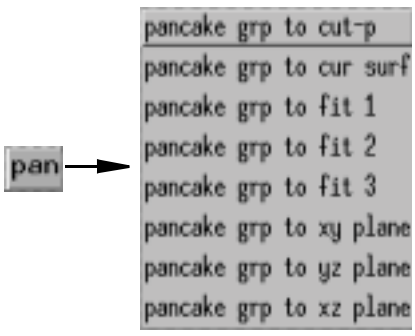
depressed button allows you to retrace only individual, non-group actions.

In addition to group actions, actions within a group can be retraced. If you have added corners to a group using more than one purple box (adding corners to a group at different times), you can retrace your previous steps by pressing the [<] and [>] buttons, found above the [grp]/[one] button. Practically speaking, these buttons are very useful. By enabling you to visit previous configurations in a group, these buttons allow direct access on the parts of a group that were the products of your actions (e.g. wraps, copies, adds, subtracts, etc).

Using the [grp]/[one] button is essential when you are constructing a wireframe. Say you are working on the wireframe surrounding one blade of a turbine, and you want to determine how to best place corners around the blade's edge by wrapping a group. If your wrap is too large, you will want to undo the whole wrap at once (not piece by piece). In order to do so, depress the [grp]/[one] button and click the [<] button. Say you have completed your wireframe and are assigning corners to surfaces with the [+] button. By mistake, you press the wrong [+] button (see figure 5.11), and add 50 corners to the active group. Because these corners were between others that were already in the group, you can't visually identify which corners were originally in your group. Rather than un-doing actions (subtracting corners from the group) one by one, the [grp]/[one] button (with the [<] button) allows you to subtract, at once, all 50 corners you mistakenly added to the group.

The [<] and [>] toggle buttons in the green, "UNDO-REDO" box affect only the secondary wireframe when the "TOPO" button is depressed.

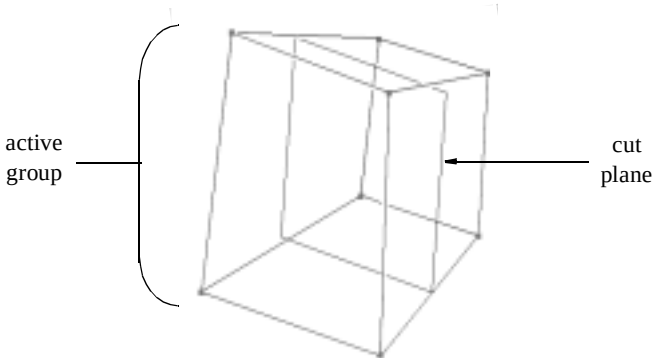
Figure 5.17



As you encounter increasingly complex topologies, you will need to project groups onto the cut plane. When you project a group, you are projecting all corners and edges of that group onto the cut plane. The [pan] button, found in the brown, "TOPO" box, opens a submenu containing operations allowing you to project *active* groups. Figure 5.17 illustrates these operations. The expression, [pan] is the abbreviation for pancake, which also means projection in most cases.

Figures 5.18-5.23. illustrate some operations in the [pan] submenu. The original active group and cut plane position for all figures is shown in figure 5.18. Six operations in the [pan] submenu, "pancake grp to fit 1," "pancake grp to fit 2," "pancake grp to fit 3," "pancake grp to xy plane," "pancake grp to yz plane," and "pancake grp to xz plane" project groups onto planes. The "grp fit" operations project an active group on either the first, second, or third least square fitted planes. Recall from statistics that least square fitted planes are to points in space as best fit lines are to points on a plane. Figure 5.19 illustrates a projection with the "pancake to grp fit 1" operation, figure 5.20 with the "pancake to grp fit 2" operation, and figure 5.21 with the "pancake to grp fit 3" operation. The other three operations project active groups to the xy, yz, or xz plane. Often, when a group's axes are aligned with the world axes, a projection on the xy, yz, or xz plane will reduce a rectangle in the group to a line, or a cube in the group to a rectangle. The "pancake grp to cut-p" operation, shown in figure 5.22, projects the active group to the cut plane such that the center of mass of the active group and the cut plane are relative. The "pancake grp to current surf" operation projects the active topology group to the current surface. In figure 5.23, the active group is projected on an ellipsoid.

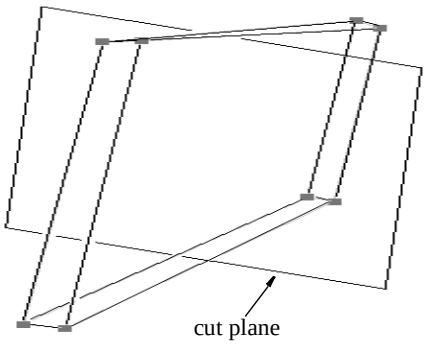
Figure 5.18



EXAMPLE PROBLEM #7:

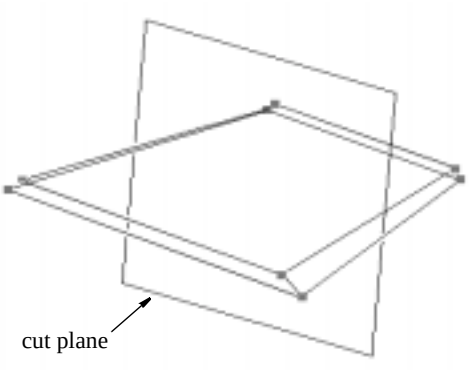
Redo example problem #3 [Prepare a two-dimensional circle for gridding, and save the resulting topology]; however, this time do an inner or outer wrap using the [wrp] operation.

Figure 5.19



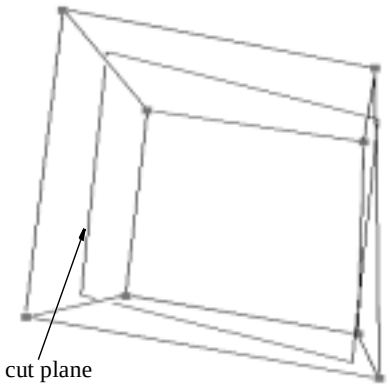
pancake to grp fit 1

Figure 5.20



pancake to grp fit 2

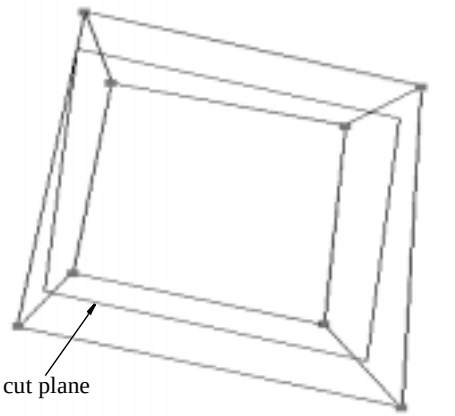
Figure 5.21



pancake to grp fit 3

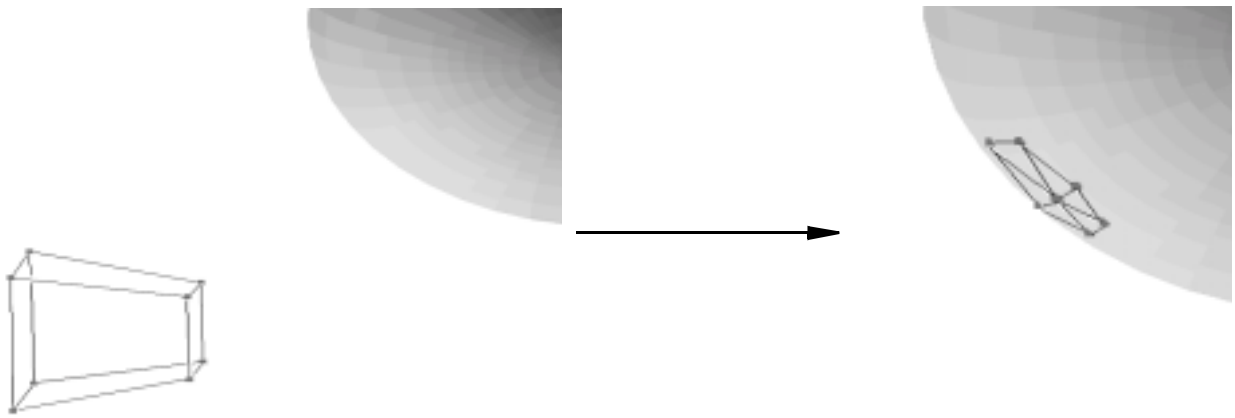
Because we are working in two dimensions, we first exit then enter the az manager in order to start with the proper default settings. We select “dim=2” from the top toolbar, and decide that the region we are gridding is inside of the circle. We access the “load –ellip” operation, make the “orient” box read: “- side,” and enter in the coordinates (0, 0, 100000) for the semi-w box. After clicking [ok], we are ready to construct the topology.

Figure 5.22



pancake to cut plane

Figure 5.23



pancake to cur surface

Because we are gridding a circle, and the circle’s whole circumference is curved, we must wrap about the circle’s whole circumference. We recall from example problem #3 that the correct wireframe looks similar to figure 5.24.

Figure 5.24

Essentially, all we must do is wrap a square. After creating the square with four corners and four edges, we can use the [wrap] operation to complete the wireframe. By using the <C> and <E> buttons to create four corners and link them, we create a quadrilateral that looks like a square (remember, the topology can be loosely positioned).

Now, we must add the corners in the quad to a group and wrap the group. We depress any one of the buttons numbered one through nine in order to activate a group, then depress the [+] button below the number [2] button. By dragging a purple box over the four corners with the right mouse button, we add these corners to the active group. Now, we are ready to wrap the contents of the active group. Note that a square is wrappable because it is a string of segments such that every corner is the endpoint of only one or two edges. After we press [wrap] (found in the brown, "TOPO" menu), a subwindow displaying dilation values appears. The value we pick depends upon the position of our quadrilateral. If we want to make an outside wrap, we pick a "larger" wrap. If we want to make an inside wrap, we pick a "smaller" wrap. If we pick the wrong dilation value, we can undo the wrap action by depressing the [<] button found to the right of the "topo" heading. For this problem, let us make an inner wrap; we choose the "25% smaller" operation.

Now, we must assign corners to the surface. Because the region we are gridding is inside the surface, we assign the four outermost corners to the surface (preferably by using the [+] button, found next to the [S.] button). Now, the topology is prepared for gridding. In the "topo" menu, we access the "TIL save as" operation and save the topology under a user name we input in the subsequent window. ***EXAMPLE PROBLEM #7 COMPLETED***

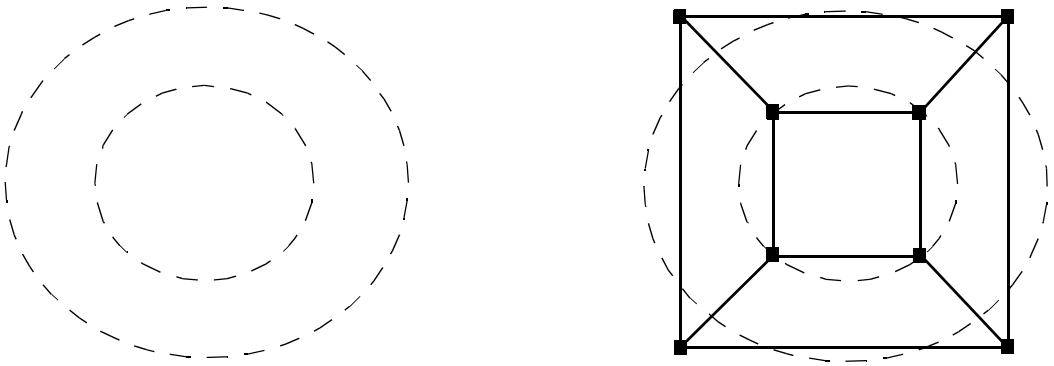
EXAMPLE PROBLEM #8:

Redo example problem #4 [***prepare a 2-dimensional ring with a circular hole for gridding, and save the resulting topology]; however, wrap both objects using the [wrp] operation***

We exit and enter the az manager in order to start with the proper default settings in two dimensions, then select "dim=2" in the top toolbar, and decide that we are gridding the region outside the hole and inside the ring. Planning to first create the hole, we access "load –ellip" opera-

tion and see that no default settings need to be changed (even the “orient” box) except the semi-w box, which should read (0, 0, 100000). Therefore, we press [ok] and access “load –ellip” again in order to create the ring. We change the coordinates in the semi-u and semi-v boxes to (3, 0, 0) and (0, 3, 0) respectively, and the orient box to “- side.” After clicking, [ok], we recall in figure 5.25 the correct topology for the surfaces:

Figure 5.25



The wireframe has one wrap. We can make the wrap the inner or outer quadrilateral. For this problem, we choose to make the outer quadrilateral the wrap. Thus, we first create the inner quad by using the <C> and <E> operations. Then, we want to add the corners of this quad to a group. After Depressing any one of the buttons numbered one through nine, we depress the [+] button, and drag a purple box over the corners in the quad. Now, the corners of the quad are added to the active group. Pressing [wrap], we pick the value “25% larger”, and create an outer wrap. Notice: the wrap becomes part of the active group. Also, notice that our wireframe has been completed.

After completing any wireframe, we must always remember to assign corners to surfaces. Here, as in example problem #4, we assign the four inner corners to the hole and the four outer corners to the ring. The topology is now complete. We access the “TIL save as” operation in the “topo” menu, and create a file name in order to save the topology. ***EXAMPLE PROBLEM #8 COMPLETED***

5.5 Grouping with the Cut Plane

In three dimensions, you will almost surely use both groups and the cut plane. Therefore, GridPro provides you with operations that integrate group and cut plane features. Mastering these operations will help you maneuver quickly and efficiently in the az graphic manager. You have seen the pancake operations, which alter group features according to the cut plane's position. In this section, you will learn more of these features, as well as features enabling you to alter the cut plane's position according to group structure.

Sometimes, you will want to move a group along with the cut plane. After depressing the [mv] (move) button found in the brown, "TOPO" box, you can transform (translate, rotate, or dilate) an active group the same way you transform the cut plane. First, when you depress the [mv] button, the center of mass of the active group becomes aligned with the center of the cut plane. Then, by shrinking the cut plane to 50% of its original size, for example, the active group will shrink to 50% of its original size as well. Note that the [mv] button is *not* automatically undepressed after being used. Figure 5.26 displays an example of the dual transformation possible with the [mv] button.

Figure 5.26

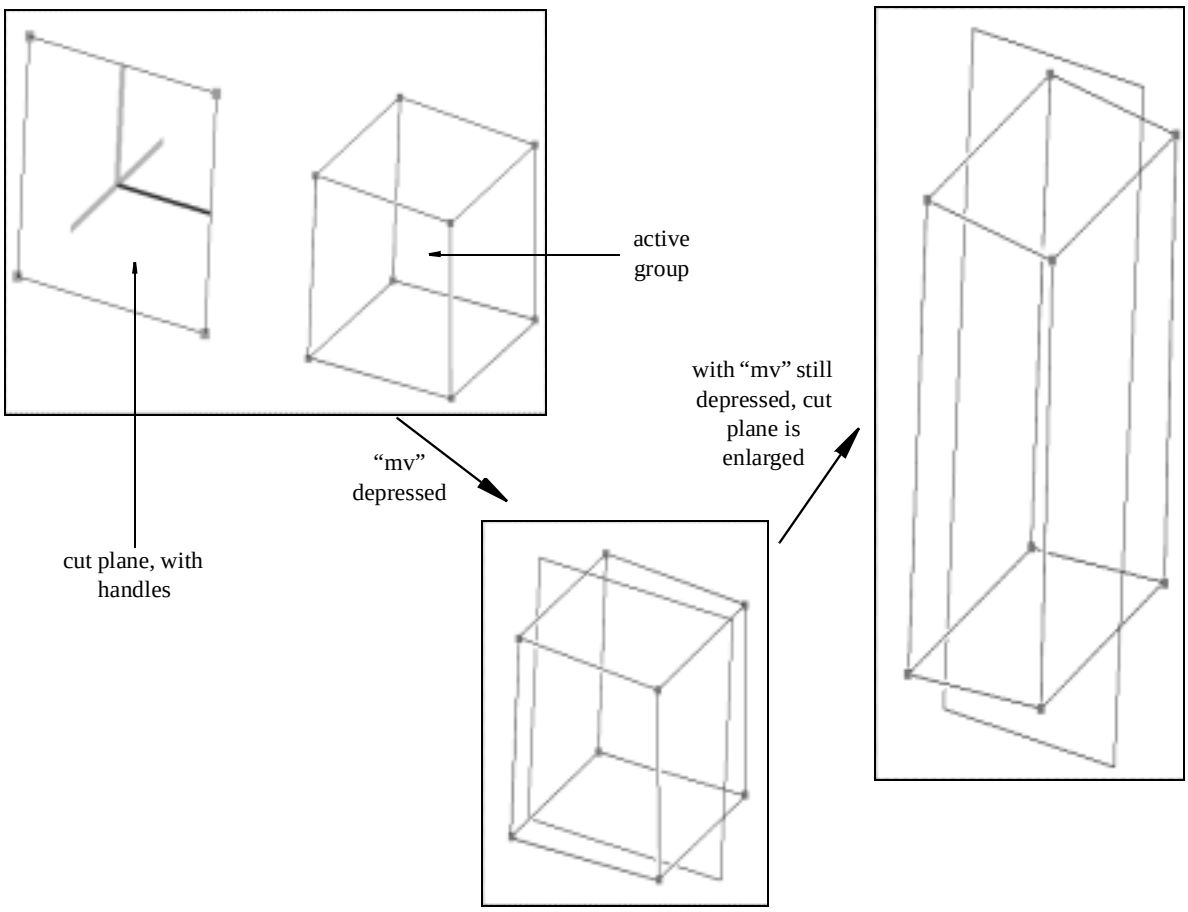
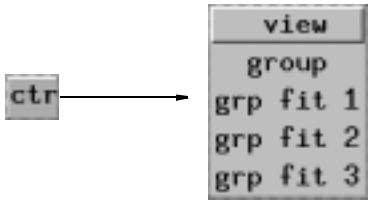
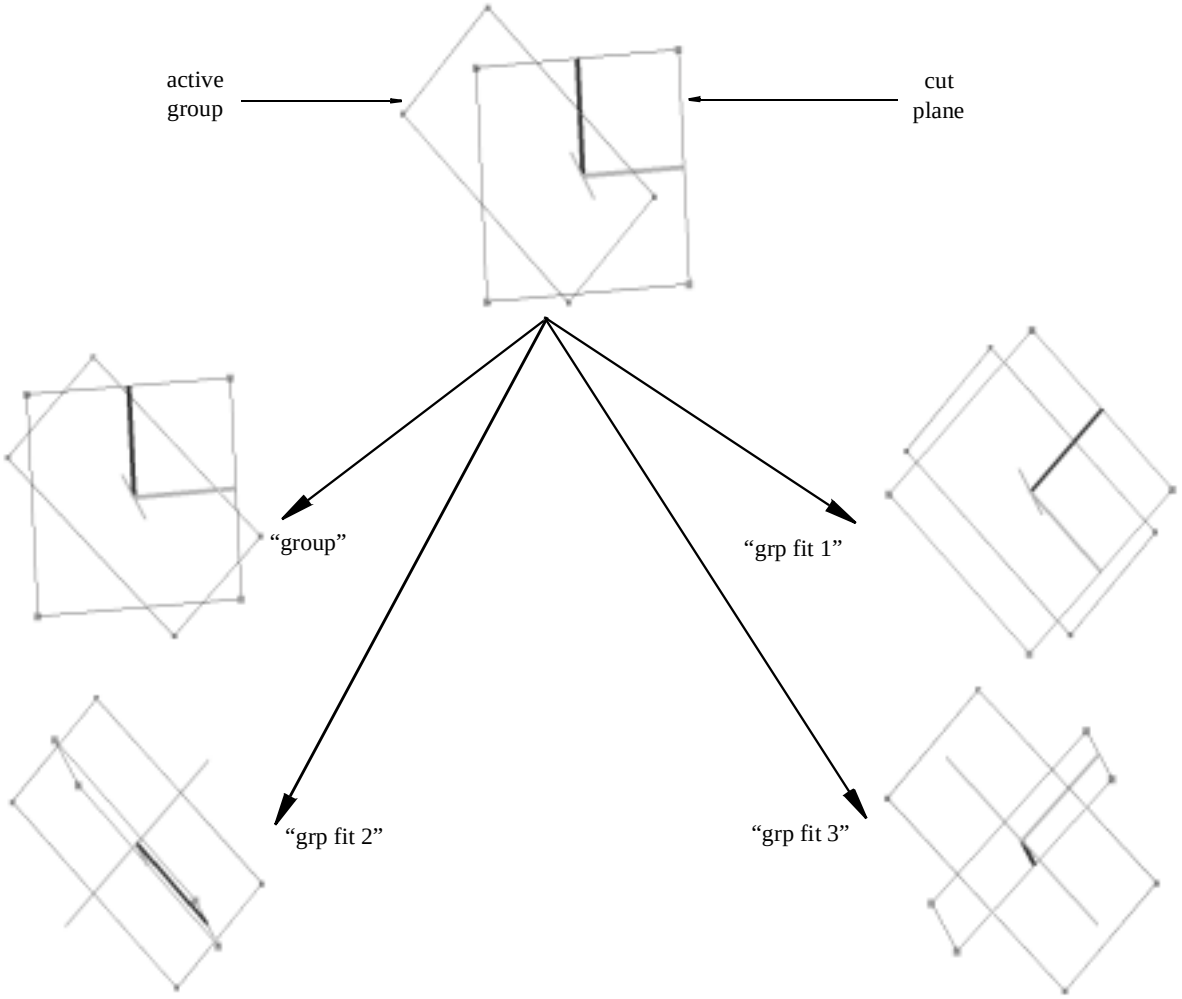


Figure 5.27



Often, you might want to align the cut plane with a group. When you press [ctr] in the brown, “CUT-P” box, a submenu containing the “view,” “group,” “grp fit 1,” “grp fit 2,” and “grp fit 3” operations appear. Remember from chapter 3 that the “view” operation aligns the center of the cut plane rectangle to the center of the drawing area and rescales the cut plane rectangle such that its area is half that of the drawing area. In order to the other operations, a group must be active. The “group” operation aligns the center of the cut plane rectangle with the center of mass of the active group. The “grp fit 1” operation performs the same function as the “group” operation, but also aligns the cut plane normal axis to the shortest major group axis and the other two cut plane axes to the other two group axes. The “grp fit 2” operation performs the same function as the “group” operation, but also aligns the cut plane normal axis to the second shortest major group axis and the other two cut plane axes to the other two group axes. The “grp fit 3” operation performs the same function as the “group” operation, but also aligns the cut plane normal axis to the longest major group axis and the other two cut plane axes to the other two group axes. Figure 5.28 displays how the cut plane is affected by the operations in the [ctr] submenu.

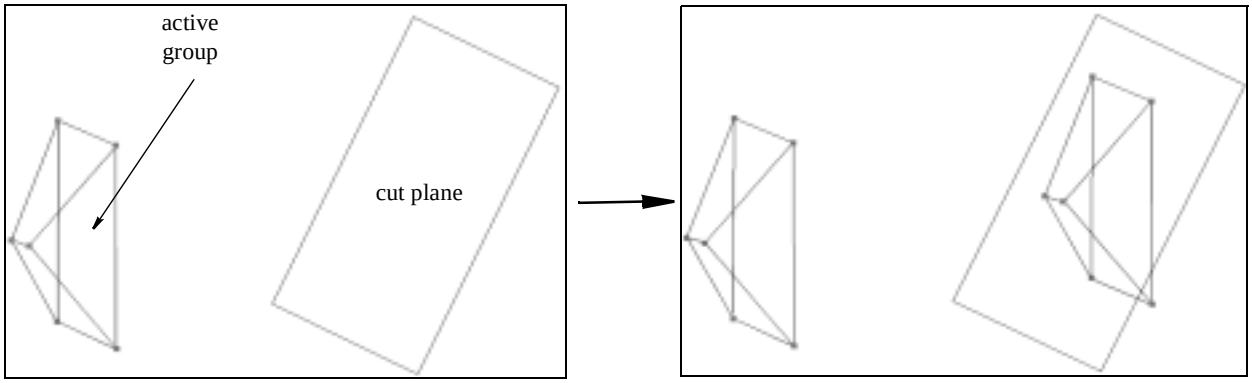
Figure 5.28



Note the difference between the "grp fit" operations in submenu contained under the [ctr] button and the submenu contained under the [panc] button (in the brown, TOPO box). The [ctr] button deals with operations that align the cut plane to the active group; the [panc] button deals with operations that align the active group to the cut plane.

In addition to aligning the cut plane with a group, you can add corners to a group by using the cut plane. In the brown, "TOPO" box, the [cp] (copy) button opens a submenu containing operations that allow you to produce transformed copies of a group by using the cut plane. Of course, in order to use these operations, a group must be activated. The first operation, "translate to cut-p," simply duplicates the group structure on the cut plane such that the center of mass of the copy of the group and the center of the cut plane rectangle coincide. Figure 5.29 displays the result. After this action, a new group is formed, one comprised of the original group and its duplicate. This new group replaces the old group. In order to retrieve the old group, you must retrace an action in the new group by pressing the [<] button in the brown, "TOPO" box.

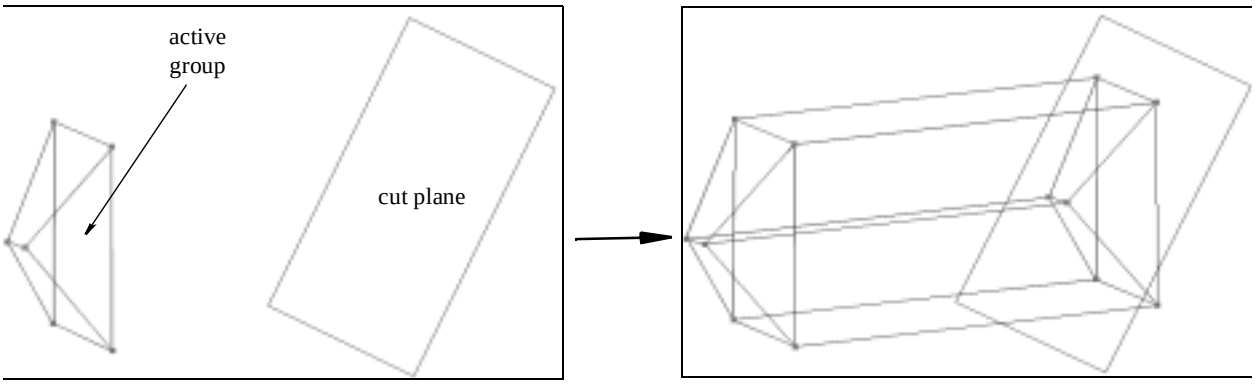
Figure 5.29



"translate to cut plane"

The second operation, "+drop back edges," represents perhaps the most useful single tool for the graphic manager. This operation first performs the "translate to cut-p" operation, then links each original corner with its corresponding duplicated corner. Figure 5.30 displays the topological action.

Figure 5.30



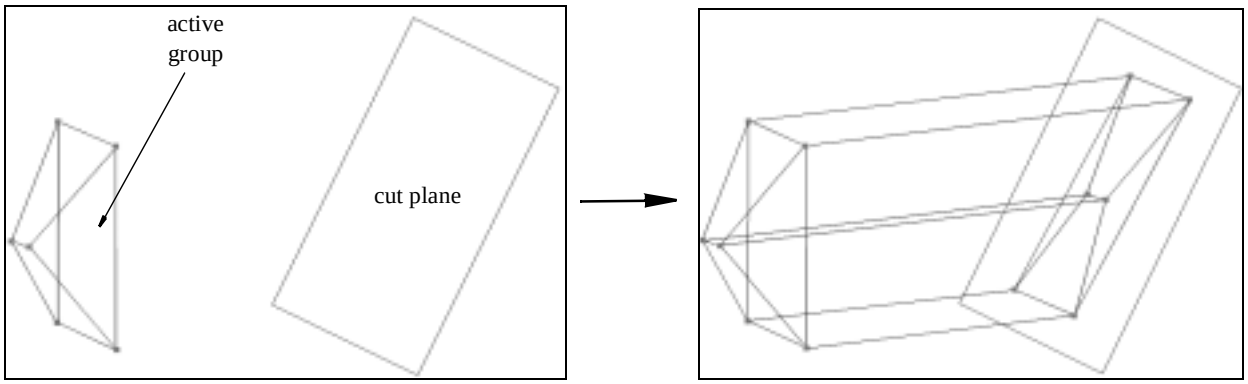
"+drop back edges"

Essentially, the "+drop back edges" operation saves you time because the operation creates a duplicate of a wireframe in a different plane, then creates edges between the wireframe and its duplicate automatically. As with the "translate to cut-p" operation, every corner and edge created becomes part of the active group. Also, remember to use the [<] button in the brown, "TOPO" box when you wish to retrieve the old group.

The other operation, "+pancake to cut-p," defines another often used feature. This operation duplicates the active group by first moving its center of gravity to the cut plane center and then projecting each corner onto the cut plane in the direction of the original movement. Then, the operation drops back edges, or links the original corners to the corners which compose the flattened copy of the group. Despite what you might think, the "+pancake to cut-p" operation has

tremendous value; inputting corners along a curved trajectory seldom can be accomplished with the same efficiency using other operations. Like the other two operations in the [cp] (copy) menu, the "+pancake to cut-p" operation adds duplicated corners to the original group. Figure 5.31 illustrates how "+pancake to cut-p" works.

Figure 5.31



"+pancake to cut-p"

EXAMPLE PROBLEM #9:

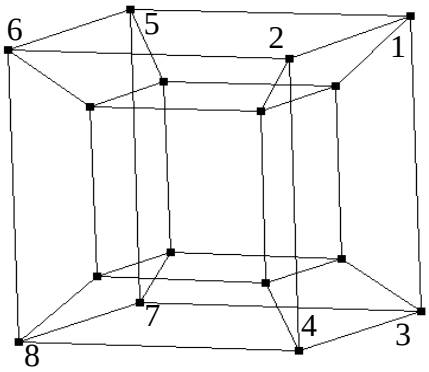
Redo example problem #6 [prepare an ellipsoid for gridding, and save the resulting topology]; however, use groups, operations from the [cp] menu, and the [wrap] button in order to create your wrap.

By using groups and their associated operations, we will be able to complete the topology for an ellipsoid with great ease. However, before we begin creating the topology, we must access the "dim=3" mode (if not accessed already) and "Topology Builder" mode, then ask ourselves: what region are we gridding? Because there is only one surface, we will grid inside that surface. Accessing the "load –ellip" operation and changing the "orient" box so that it reads, "- side," we click [ok].

The topology for this surface mirrors the topology in example problem #6; the wireframe itself represents a hypercube and the surface assignments do not change. However, instead of having to translate the cut plane three times and to input corners manually, we can now use group operations to translate and create corners automatically.

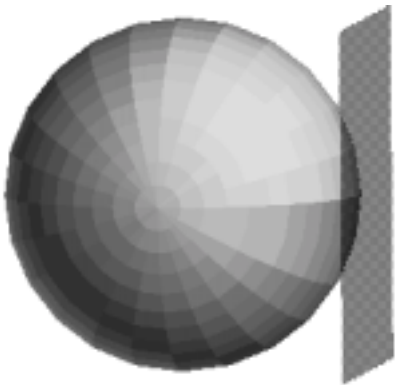
First, we will plan how we are going to make the topology. Looking at figure 5.32, we see that by creating corners 1, 2, 3, and 4 by traditional means and add those corners to a group, we can duplicate the group in order to produce corners 5, 6, 7, and 8. By dropping back edges (connecting the original corners to the duplicated corners), we create a cube:

Figure 5.32



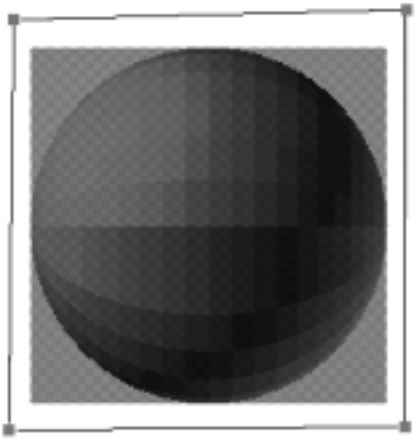
Then, after having created the outermost portion of the figure, we automatically have all eight corners in our group and can then perform an inside wrap on the group. Figures 5.33-5.38 illustrate visually the steps involved in creating the wireframe.

Figure 5.33



In order to create the first four corners, we position the cut plane so that it lies near the ellipsoid. We depress [cut] and [hand] buttons in order to see both the cut plane rectangle and its handles. Then, we position the cut plane by manipulating its axes, or by using the [pos] button found in the brown, "CUT-P" box. The result is shown in figure 5.33. Notice that we have pressed the [fill] button and removed the cut plane axes by pressing [hand] so that we can see the cut plane more clearly. We can best position corners by aligning the cut plane axes with the screen axes. Therefore, after the cut plane has been positioned, we should rotate the system and use the [snap] menu in order to align the axes.

Figure 5.34

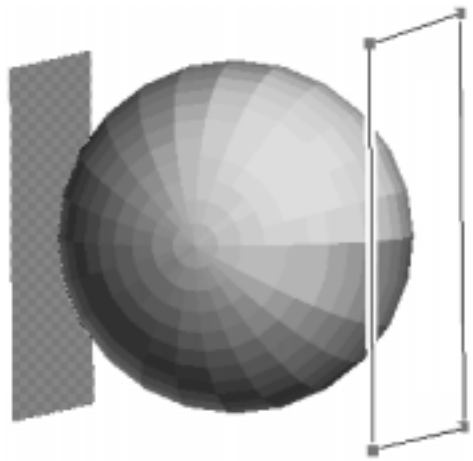


After positioning the cut plane, we can now input and link corners. Because we are creating outermost corners (corners 1, 2, 3, 4), for visual convenience we place the corners such that any link between them does not cover any part of the ellipsoid. Using the <C> and <E> keys, we create the corners and edges. Figure 5.34 displays a possible correct configuration. The cut plane rectangle in the figure appears smaller than the quadrilateral formed by the four corners; remember, the actual size of the cut plane rectangle is irrelevant because the rectangle just represents the whole plane. Dilating the cut plane rectangle is nothing more than a visual aid.

Before we translate the cut plane and apply the [cp] (copy) operation, we should align the cut plane with the group center of gravity. Without this alignment, our drop back edges will be skewed (not perpendicular to the cut plane). Thus, we press the [ctr] button in the brown, "CUT-P" box, and choose the "group" operation in the resulting submenu. Although unnecessary, this procedure produces a visually more attractive and thus

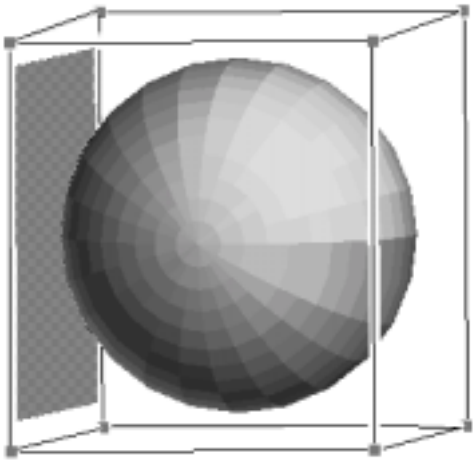
distinguishable wireframe, an essential component to effectively manipulating corners and edges in more complex situations.

Figure 5.35



From the four corners we created, we can construct the whole topology. First, we must add the existing topology to a group. By depressing any button numbered one through nine, we activate a group. Then, by depressing the [+] button (found below the [2] button) and dragging a purple box over the four existing corners, we add those corners and their corresponding edges to the active group. Next, we depress "hand" to access the cut plane axes. Then, we translate the cut plane in order to position it on the other "side" of the ellipsoid. Figure 5.35 displays the correct translation after we hide the cut plane axes. Now, by duplicating the existing topology on the cut plane and dropping back edges, we can create an outer cube.

Figure 5.36

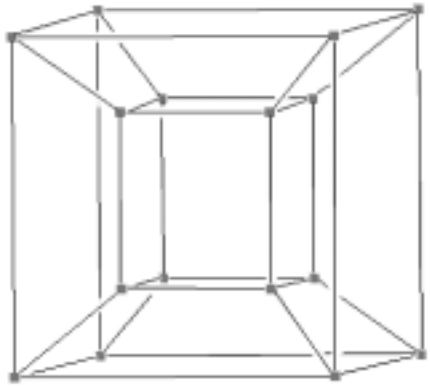


We can now use the cut plane rectangle in order to drop back edges. After pressing the [cp] button found in the brown, "TOPO" menu, we see a window allowing us to choose among the "translate to cut-p," "+drop back edges," or "+pancake to cut-p" operations. Because we want to duplicate the topology on the cut plane, and then link the duplicated corners with the original corners, we choose the "+drop back edges" operation. Actually, the "+pancake to cut-p" operation would have the exact same effect as "+drop back edges" operation because the axes of the cut plane are parallel to the corresponding axes of the group. The "translate to cut-p" operation would just duplicate the topology on the cut plane. Figure 5.36 illustrates the desired result, eight outer corners linked so that a cube is formed. Notice that the perpendicularity of the drop back links results from our previous use of the [ctr] button.

By wrapping this topological cube, we can create a hypercube. We now remember that any operation from the [cp] menu performed on a group *automatically* adds duplicated corners to that group. Therefore, the four duplicated corners we created were automatically added to the group containing the original four corners when we used the "+drop back edges" operation. Realizing that all corners of the cube are now grouped, we can wrap the group. Accessing the wrap operation, we choose whatever dilation value is appropriate. Remember, the [<] button to the right of the "topo" heading in the "UNDO-REDO" box will undo a topology mistake, or specifically here undo an incorrectly dilated wrap. Figure 5.37 illustrates the topology created after a

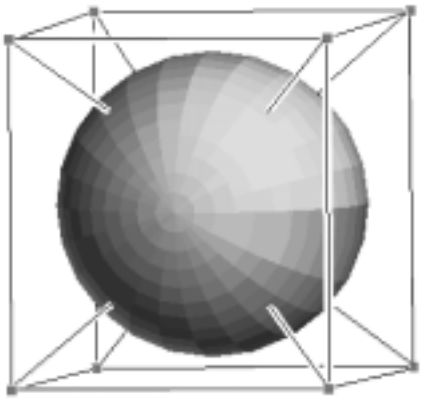
"25% larger" wrap. Notice that the surface and cut plane are hidden (because the [surf] button in the green, "SHOW" box and the [cut] button in the same box are undepressed). Figure 5.38 displays wireframe's position relative to the surface.

Figure 5.37



Now that our wireframe is complete, we must assign the appropriate corners to the surface. Because we are gridding inside the ellipsoid, we must assign the outermost eight corners to the surface. These corners are numbered, 1, 2, 3, 4, 5, 6, 7, and 8 in figure 5.32. Because there is only one surface, it must be the current surface. Using the [S.] button or the [+] button (found to the right of the [S.] button), we can assign these corners to the surface. Now, the topology is complete.

Figure 5.38



We access the "TIL save as" operation under the "topo" menu and save the topology under a user-defined name. ***EXAMPLE PROBLEM #9 COMPLETED***

EXERCISE #15:

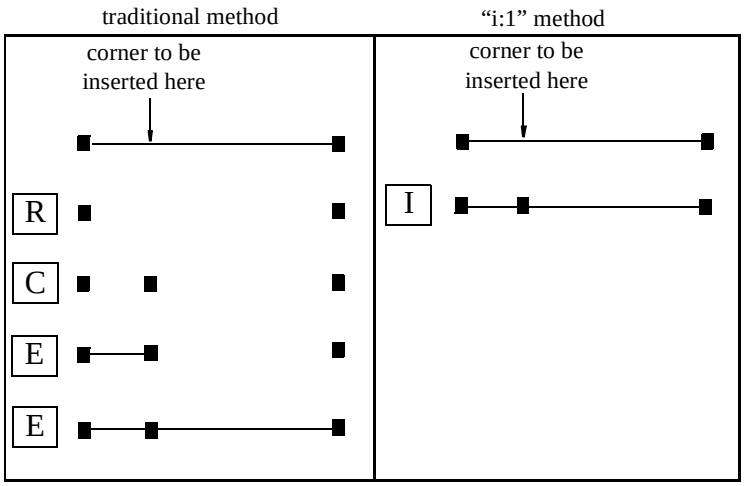
Using groups and operations from the [cp] menu, prepare any superellipsoid of power 15 for gridding.

5.6 Inserting Edge Groups

When you create topologies for multiple surfaces, you will often construct grid-like wireframes. Because adding corner layers individually for these wireframes consumes time and sacrifices precision, GridPro has a feature which enables you to create quickly these (often multiple) corners within the boundary of an already existing wireframe.

Corner inserts are created using a feature of the insert mode. While the default insert mode will give you an entire layer that propagates throughout the entire extent of *all* possible wireframe edges, the basic structure can be seen from the insert of a single corner along an edge. Located in the bottom left corner of the brown, "TOPO" box is the button [i:1]. The expression [i:1] means "1 edge insert." By pressing this [i:1] button, a submenu appears with two other modes, the "group insert" ([i:g]) and the "all insert" ([i:a]) modes. The [i:1] mode allows you to input a corner *on* an edge in order to split that edge by depressing the <I> key. The other two modes allow you to input multiple edges by depressing the <I> key and clicking on an edge.

Figure 5.39



By enabling you to place a corner on an edge, the [i:1] mode lets you compartmentalize numerous operations and much maneuvering into one click of the mouse. For example, in order to place a corner in between two others, we traditionally had to perform many operations. We would eliminate the existing edge between the corners with the <R> key, then input a new corner between the two old ones using the <C> key. By clicking several times with the <E> key depressed, we could create links between the corners. Finally, we would have to translate the inputted corner such that all three corners were colinear. An alternative to this somewhat laborious procedure is accessing the [i:1] mode, depressing the <I> button, and clicking at the point on an edge where you wish to place a corner. Figure 5.39 illustrates the differences between the two methods.

Exactly what multiple edges which are inserted are better understood through visual example rather than explanation. Therefore, look at figures 5.40-5.43 *after* reading the following explanation:

The "i:g" and the "i:a" modes can be accessed in both two and three dimensions. If the "i:a" mode is accessed, when you depress the <I> key and click on an edge, a corner forms on that edge at the point you click and proportionally on every edge topologically parallel to the edge you

clicked. If the “i:g” mode is accessed, and a group is active, when you depress the <I> key and click on an edge in the active group, a corner forms on that edge at the point you click and proportionally on every edge both in the active group and topologically parallel to the edge you clicked. Notice, every point you click goes on an edge. If the “i:a” mode or “i:g” mode is accessed, when you depress the <I> key and click on an edge, not only are corners formed proportionally along every edge parallel to the one you clicked; edges are created between the newly formed corners. If the “i:g” mode is used and a group is not activated, the mode functions similar to the “i:a” mode.

Figures 5.40 and 5.41 are two-dimensional examples, and figures 5.42 and 5.43 are three-dimensional examples. Figure 5.41 illustrates a common occurrence: when the whole wireframe represents an active group, the actions performed under the [i:g] and [i:a] modes are equivalent.

Figure 5.40

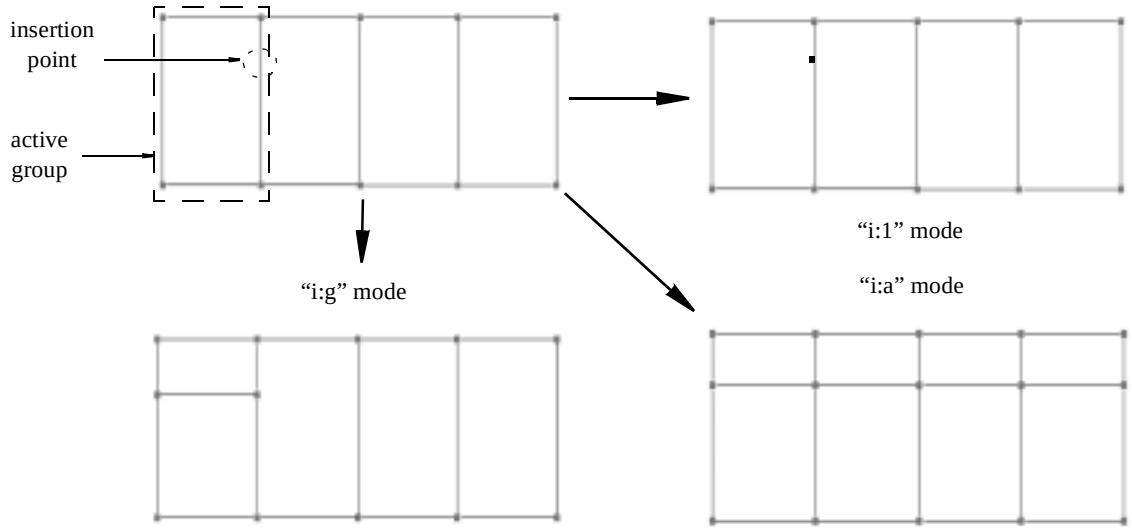


Figure 5.41

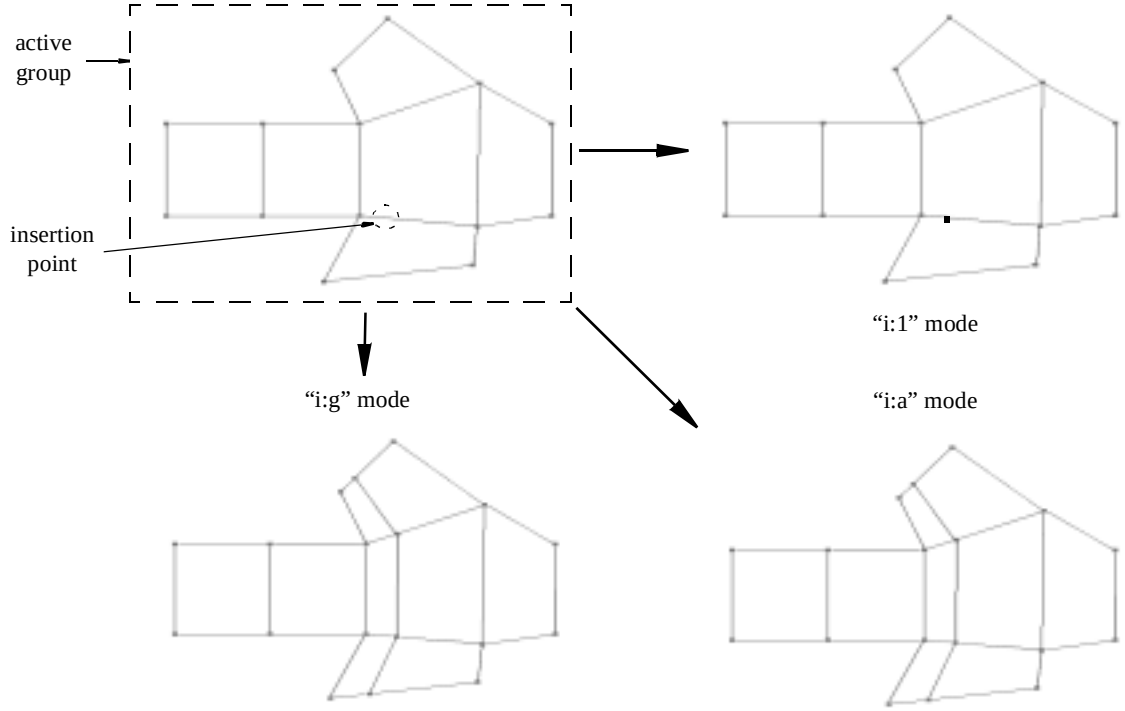


Figure 5.42

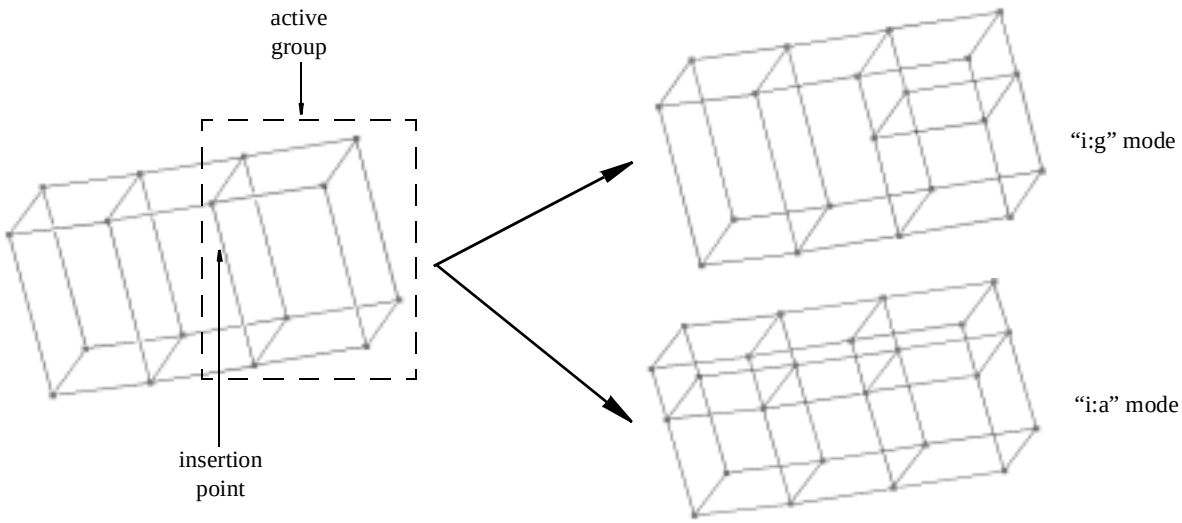
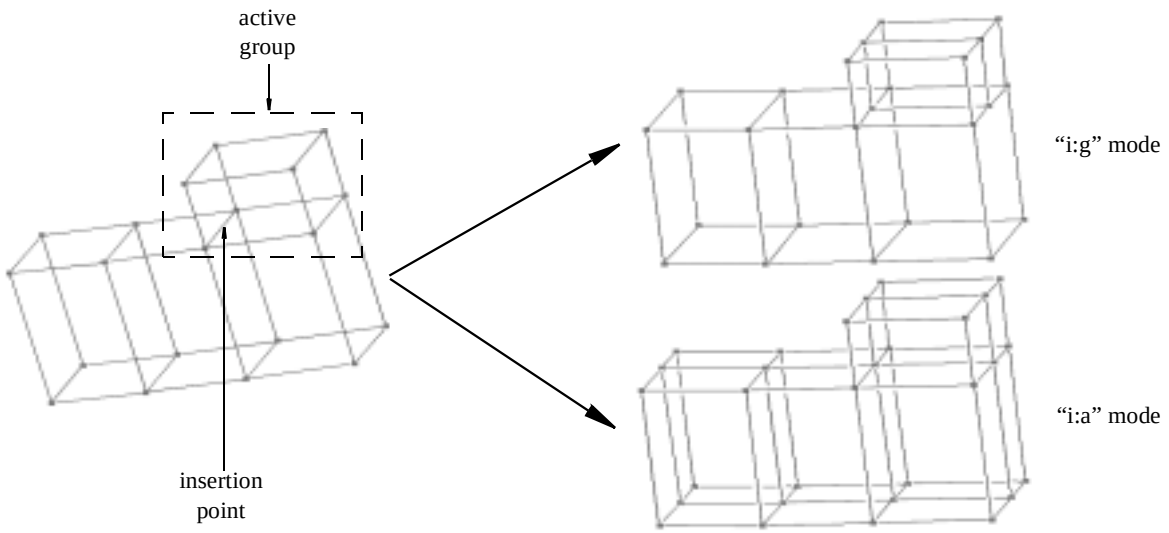
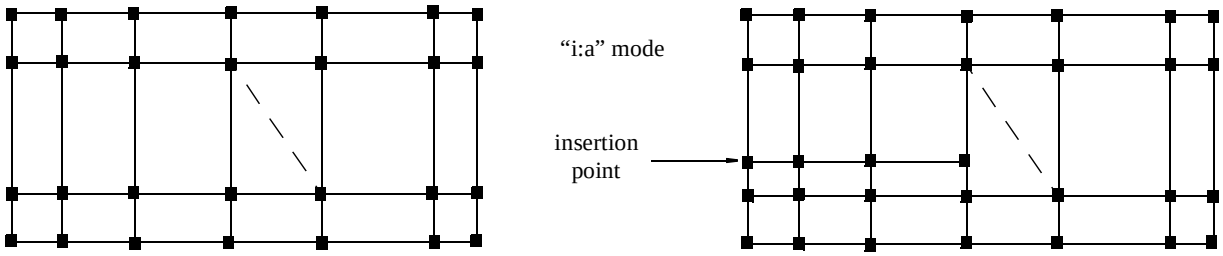


Figure 5.43



Additionally, the edges formed as a result of the [i:g] and [i:a] modes do not penetrate excluded faces. See figure 5.44.

Figure 5.44

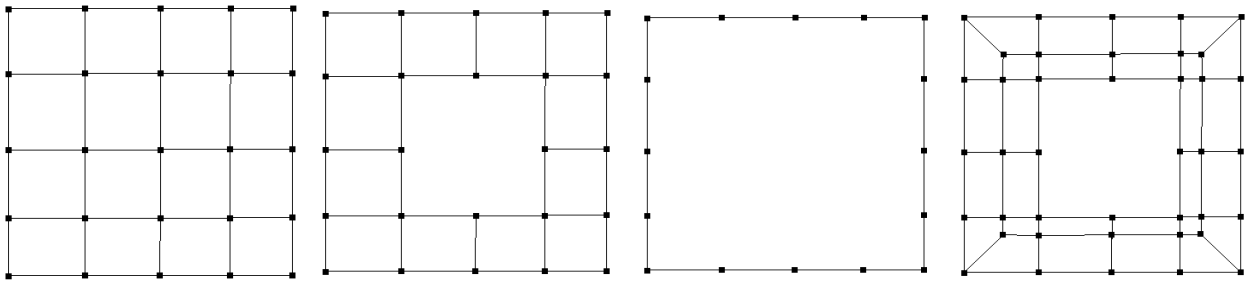


EXERCISE #16:
Grid, in two dimensions, a superellipse with two circular holes.

Also, you must distinguish between corners inserted on an edge and corners in free space that appear on that edge. Be sure to take these considerations into account when you wrap. If the edges produced as a result of a wrap intersect other edges, you will want corners to be *inserted* at the intersections. In order to make GridPro recognize these intersections, you must use the reference group (group #1). The contents of the reference group should contain the corners and edges you wish to wrap *and* the edges that will contain the intersections. *Each corner of the wireframe you wish to wrap can be connected to either 0 or 1 edges that will contain an intersection point.* You should put only the corners and edges you wish to wrap in any other group. Then, you must repeatedly click the button located above the [<] button in the "TOPO" box until the button reads [R]. Now, you can activate the group containing the wireframe you want to wrap, and wrap it. Figure 5.45 provides an example. In the figure, notice that the four verticies of the outermost rect-

angle of the final wrap are connected to edges containing no intersection points. All other corners on the edges of the outermost rectangle are connected to edges containing exactly one intersection point. Remember, to perform the wrap successfully, the button located above the [<] button must read [R].

Figure 5.45



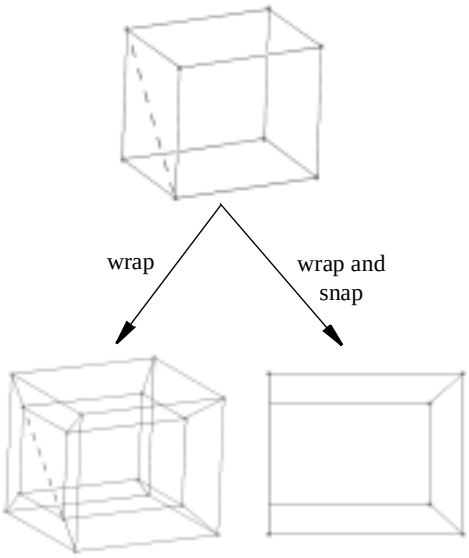
original wireframe | group #1 | other group | wrapping "other group"

5.7 Face Exclusion

You have learned to distinguish basic wrappable surfaces from basic unwrappable surfaces; this subsection describes exceptions arising from the rules stated previously regarding the wrappability of a surface. In order to account for these exceptions, you must "exclude" faces of a wireframe. A face here is defined as the area enclosed by four corners and the edges among them. Face exclusion is necessary in two cases:

First, you must exclude faces so that GridPro can read correctly the *dimension* of a surface being prepared for wrapping. For example, a wireframe rectangle can be thought of as a string of four segments (a one-dimensional wrapping) or as a quadrilateral (a two-dimensional wrapping). Therefore, you must specify the dimension of the wrap. In order to convey to GridPro what you are wrapping, you must exclude faces of a wireframe when the edges of those faces are to be wrapped as one-dimensional segments.

Figure 5.46



Second, you must exclude faces in order to exclude a portion of the wrap. Figures 5.46 and 5.47 illustrates how wraps are affected by excluding faces.

Figure 5.47

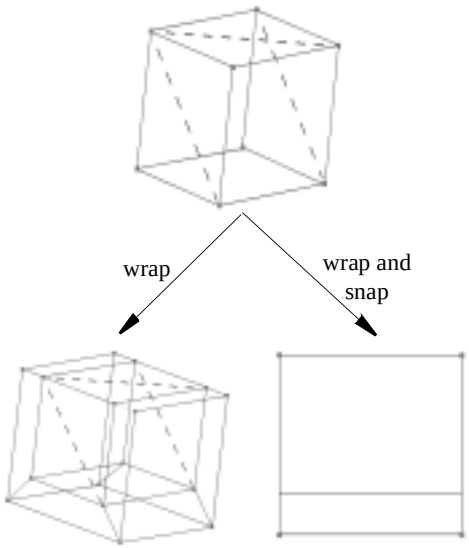
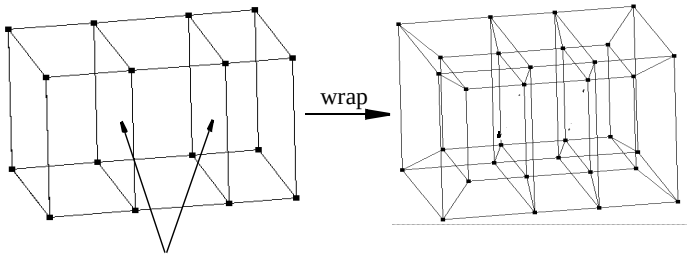


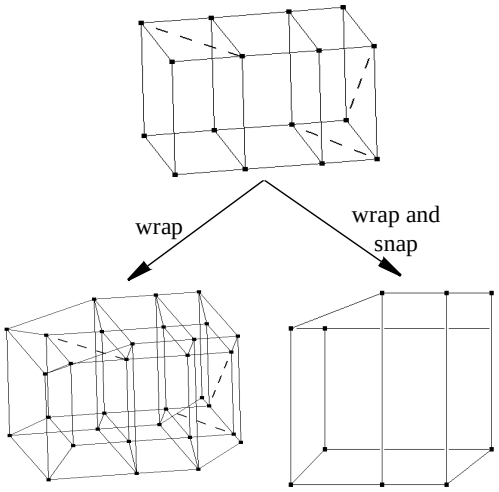
Figure 5.48



You might believe that in order to wrap the outside faces of a three-dimensional object, you have to exclude the object's internal faces. However, GridPro automatically excludes these faces for you. See figure 5.48.

Face exclusion allows some corners and edges to be wrapped one-dimensionally and other corners and edges to be wrapped two-dimensionally. The combination of dimensionality and positioning determine the shape of the final wrap. When one face is excluded (wrappable in one dimension) and an adjacent face is not (wrappable in two dimensions), the resulting wrap will form as follows: the excluded face will attempt to pancake the wrap, and the other face will attempt to extrude the wrap. The result is a compromise: a somewhat extruded but somewhat pancaked configuration, like that in figure 5.49.

Figure 5.49



In order to exclude a face, you must hold down the <F> key click two corners of a diagonal of the desired face. A dashed red line appears between these corners, signifying that the quad, or face, is excluded. As the cursor approaches a corner, the cursor becomes a red or white circle. Remember: a red circle implies that you probably will access a corner, and a white circle implies that you definitely will access a corner. In

order to remove the dashed, red line, hold down the [R] key, move the cursor to the line, and click the left mouse button. As the cursor approaches an excluded face diagonal, the cursor will turn into a red or white X symbol:  Often, exclusion markers will clutter your screen. To remedy this problem, you can hide the exclusion markers by undepressing the [x&a] button, found in the green SHOW box.

Chapter 6 - Viewing Grids

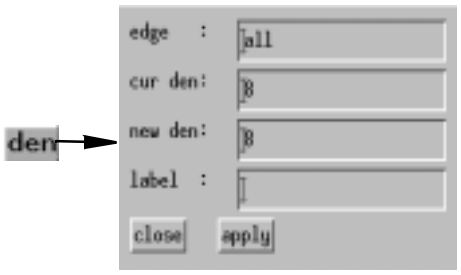
6.1 Grid Resolution

The density of a grid is of great importance; a grid can be used for analysis only if the gridded region is sufficiently resolved. Before you begin the gridding process, you can specify the density of the grid you wish to create. Also, you can specify what portions of the grid should contain more or less detail. This "detail" is represented by a number, a number which describes the number of grid cells along an edge (and thus all compatible edges).

The grid cell count is directly proportional to the grid's overall density. Including both endpoints of an edge, the number of mesh lines present is one more than the number of cells inputted. You must set the density of the grid in the "Topology Builder" panel; you will not see the results of any density alterations until you actually see the grid (in the "Grid Viewer" panel). By accessing the [den] operation in the brown, "TOPO" box, you open a window containing density options. You can alter the grid density of sets of topologically parallel edges, or all edges. You cannot alter the density of only one edge or a pair of topologically perpendicular edges.

If you wish to alter the density of a set of topologically parallel edges, first open the density window, then click on one of the parallel edges. All parallel edges will then become red. If you wish to alter the density of all edges, do not click on any edge.

Figure 6.1



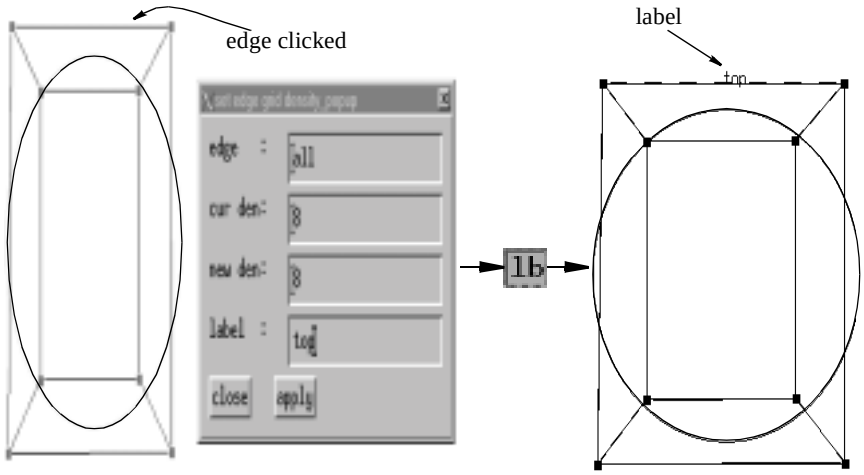
the density pop-up menu

The density window contains four boxes: the "edge," "cur den," "new den," and "label" boxes, as shown in figure 6.1. The edge box reads "all," or delineates specific edges. If you are changing the density of all edges, the edge box will read "all"; if you are changing the density of a set of parallel edges, the edge box will display every edge being altered. The "cur den" box displays the current density of either the parallel edges being altered or all edges. The default setting is 8, or 9 mesh lines per set of parallel edges. The "edge" and "cur den" boxes contain fixed information; you cannot type new information in these boxes. The "new den" box illustrates the new density of the edges you wish to alter. Therefore, you must type the desired density value of the edges in this box. When you press [apply], the density of the edges you specified will become the number you typed in the "new den" box.

The greater the density of the topology, the longer the Ggrid process will take to complete. For two-dimensional grids, every time you double the density of the topology edges, you increase the Ggrid processing time by a factor of 4. For three-dimensional grids, every time double the density of the topology edges, you increase the Grid processing time by a factor of 8.

If you have initiated the Ggrid process, but notice that you wish to change the density of your grid, you can do so without interrupting Ggrid. As Ggrid is running, change the density to the desired value using the [den] button, then select the "Ggrid gridden" operation in the "topo" menu. Ggrid will automatically update the grid being created.

Figure 6.2

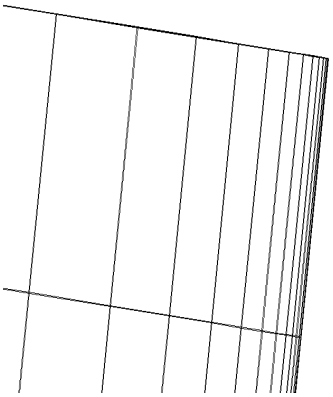


By typing a designation in the "label" box, the edge you clicked will adopt your designation as identification. In order to see this identification (on the wireframe edge), depress the [lb] button in the green, "SHOW" box. See figure 6.2.

Labeling edges on your wireframe can help you weave your way through a complicated topology. Usually, you label edges which represent two very different parts of the region to be gridded, yet look too similar in its surroundings to be distinguished from each other. Additionally, if the "bottom" of one surface is facing the "top" of another, labeling the corresponding topological edges can help you navigate through the topology. Also, labeling could be used as a tool to identify which edges have unique densities. For example, say you are gridding an airplane, and want a fine mesh around just the wings. In order to make sure every topological edge around the wing will be gridded with a greater density, label each edge when you change its density. Then, when you *think* you have changed the density of all the appropriate edges, you can observe whether you have done so on the drawing area. Whatever "wing" edges that are not labeled have not had their densities changed.

Labels specify edges; identification numbers specify corners and grid blocks. GridPro automatically assigns an identification number to a corner or grid block as soon as one is created. To see these identification numbers, depress the [id] button, found in the green, "SHOW" box.

Figure 6.3



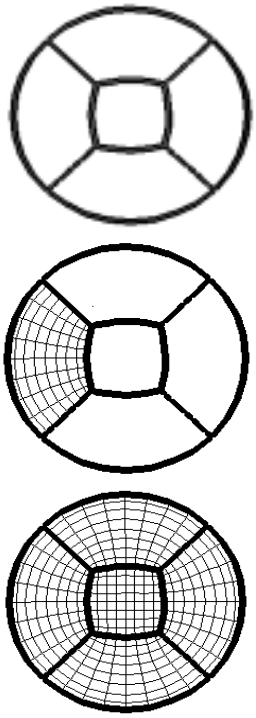
clustering at a boundary layer

Undoubtedly, before performing CFD (Computational Fluid Dynamics), you will want to resolve certain parts of your grid to a greater extent than other parts. In general, for viscous flow, boundary surfaces need to have a high resolution; body surfaces can have a low resolution for non-viscous flow. Most physical changes occur at the boundaries of a region. Thus, substantial fluid dynamic variations occur at these boundaries. Therefore, in order to perform CFD calculations, you must place enough nodes (or cells) at these boundaries to resolve the flow physics. Think of each node as an observer. You need many observers at the boundaries of a region because a rapid change is occurring there. So much happens in such a small space - you need to sort out individual occurrences. The only way to resolve the action is to position numerous nodes appropriately. An example of this type of boundary node clustering appears in figure 6.3. In this figure, the clustering resolves the boundary layer along the right side of the figure. Clustering is included as a surface definition parameter, and as a utility in GridPro (the “clu” utility). The “clu” utility converts Euler grids into Navier-Stokes grids.

6.2 Introduction to Blocks, Sheets, Faces, Shells and Slices

As you have already learned, a grid can be split up, sliced and represented in many different ways. The block traces composing a grid contain edge outlines, centers of mass (remember the filled in or hollowed out little square boxes), and skeletons (links from the center of mass to the center of each face). These various features can be hidden or activated in various ways in order to display different parts of a grid. Using the [shell] button, you can see an outline of the grid.

Figure 6.4



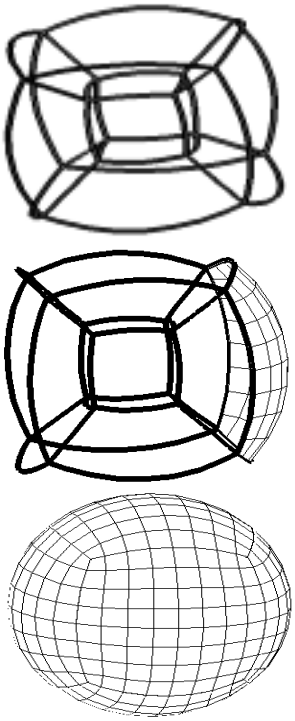
five blocks

five blocks and one face

five blocks, five faces, one face sheet, and the shell

Because meshes are used extensively for analysis testing, GridPro allows you to visualize them in different ways. First and foremost, in order to understand meshes, you must understand the difference between blocks, faces, slices, sheets, and shells; so far, you have learned about blocks, their skeletons, their outlines, and their centers. When the "Ggrid" process creates a grid, GridPro carves up the region inside each block into IJK coordinates (index coordinates) in order to create a smooth grid. The IJK coordinates of each point are integers; each node thus lies on a lattice point. The maximum and minimum values of these IJK coordinates appear on the boundary of the block. Note that IJK coordinates differ for *each* block (within one grid). A face is the grid about a boundary of a block. More technically, a face is a network of links along indexed IJK lattice points such that I, J, or K is a maximum or minimum. A slice is a grid that cuts through a block. More technically, a slice is a network of links along indexed IJK lattice points such that I, J, or K is any constant integral value between a certain maximum and minimum. A sheet is simply a collection of linked faces or a collection of linked slices. A sheet composed of linked faces is called a face sheet, and a sheet composed of linked slices is called a slice sheet. A face sheet composed of the faces of the outer boundaries of *all* outer, activated blocks is called a shell (which is a sheet). A face sheet composed of the faces of the outer boundaries of *some* outer, activated blocks is called a partial shell (which is also a sheet). Figures 6.4, 6.5 and 6.6 illustrate some different ways in which grids can be viewed. Figures 6.4 and 6.5 display faces and face sheets. Figure 6.6 displays slices and slice sheets. Notice that sheets can take on many shapes.

Figure 6.5

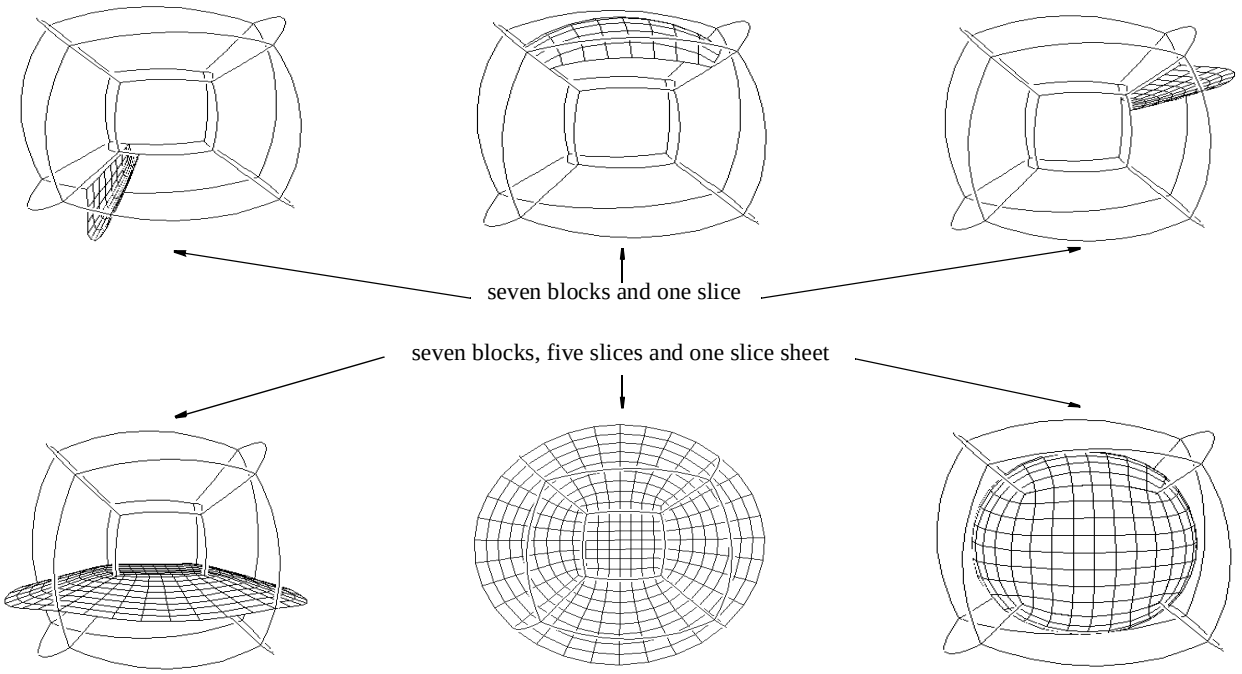


seven blocks

seven blocks and one face

seven blocks, seven faces, one face sheet, and the shell

Figure 6.6



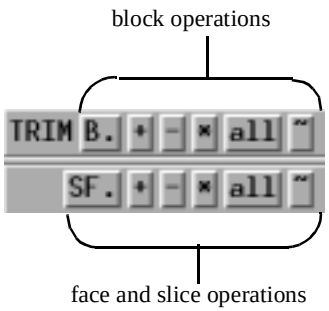
Let us summarize: distinct blocks have distinct faces and distinct slices. Many linked faces or many linked slices comprise a sheet (either a face sheet or a slice sheet). A face sheet can be either a shell or a partial shell.

Manipulating blocks, faces, slices, and sheets will help you to look inside and analyze parts of grids. For complex grids, you will want to check numerous cross sections to observe specific features such as the density of the mesh lines or the number of singularities in a subregion.

6.3 Using Blocks, Sheets, Faces, Shells and Slices

Now that you understand the difference between a face, slice, and sheet, you will learn how to manipulate each grid feature. You will use operations from the brown, “TRIM” box, the green, “CUR” box, and the brown, “MAKE SHEET” box. Before doing so, realize that blocks are at the core of grid viewing; faces, slices, sheets, and shells are derived from the block structure. Because blocks are so important, each is given an identification number (similar to surfaces). In order to see every block’s identification number, depress [id] in the green “SHOW” box.

Figure 6.7

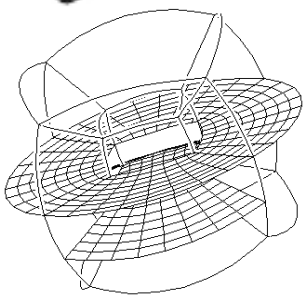


You can manipulate faces or slices just as you manipulated blocks by using the operations in the brown, "TRIM" box. Like blocks, faces and slices are activated and deactivated. Also like blocks, faces and slices can be activated and deactivated using [+], [-], [*], [all], and [~] buttons. These buttons are found to the right of the [SF.] button. The [SF.] button functions for faces and slices just like the [B]. button functions for blocks, as shown in figure 6.7. If you do not remember how to manipulate blocks, look back in chapter 4; manipulating blocks is analogous to manipulating faces and slices. Before you can use the buttons in the [SF.] row, however, you must already have created a sheet; the buttons in the [SF.] row can manipulate only the faces on the current sheet. Note also that a shell is composed of linked, *activated* faces – if one or more faces are not activated, the sheet is not a shell (but rather a partial shell).

Figure 6.8

The center, rightmost, and bottommost slices of the slice sheet are deactivated. The "HLR with shade" operation is accessed.

The center slice of the uppermost slice sheet, and the rightmost and leftmost slices of the lower slice sheet, are deactivated.



The [SF.] button is depressed - the center and skeleton of every slice on the slice sheet is displayed. The "point" operation is accessed

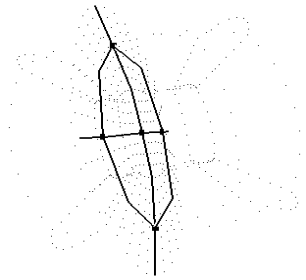
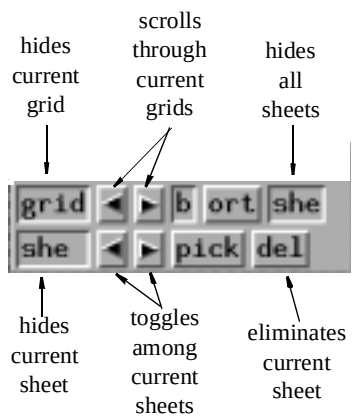


Figure 6.9



The green, "CUR" box houses some useful operations for manipulating grids. First, remember that we have used the [del] button found in the box in order to eliminate shells. However, this [del] button actually eliminates the current sheet, not just the shell. Like surfaces, sheets and grids can be current. When many sheets are on the screen, you can press the [<] and [>] buttons (found to the right of the [she] button) in order to toggle between current sheets. In order to hide the current sheet, un-depress the [she] button. In order to hide all sheets for the current grid, undepress the [she] button found to the right of the [ort] button. Just as the arrow buttons to the right of the [she] button toggle among current sheets, the arrow buttons to the right of the [grid] button toggle between current grids. When undepressed, the [grid] button itself hides the current grid. Because you will most likely have one grid in the "Grid Viewer" panel, you will use the last three mentioned buttons rarely. The location of these operations is shown in figure 6.9.

The brown, "MAKE SHEET" box allows you to manipulate grids about multiple surfaces and to create slices, shells, and sheets. Understand that by default, surfaces are hidden in the "Grid Viewer." You can toggle among the hidden surfaces in order to make one current by using the arrow buttons found to the right of the [surf] button. The upper-left corner of the drawing area displays the current surface id number. The [surf] button, when pushed, will place a sheet about the current hidden surface. The [all] button, when pushed, will place a sheet about all hidden surfaces. These operations are conveniently located next to one another, as in figure 6.10.

Figure 6.10

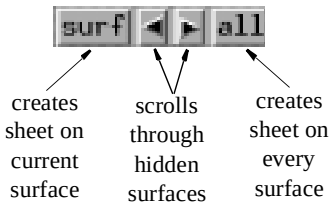
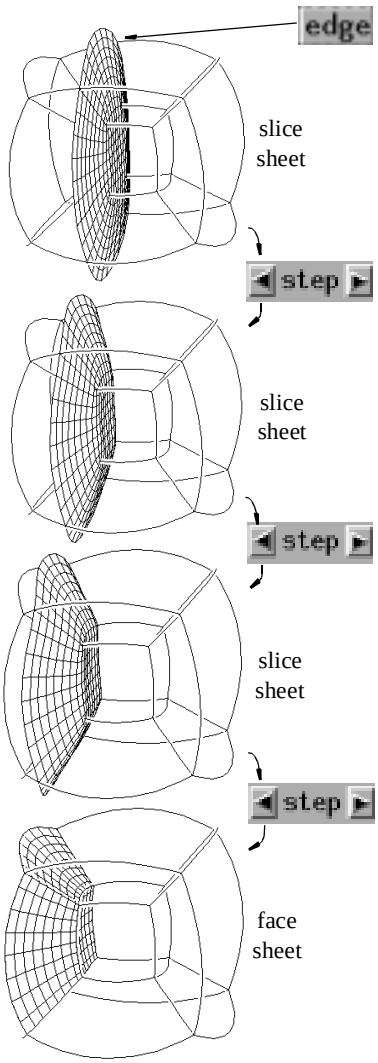
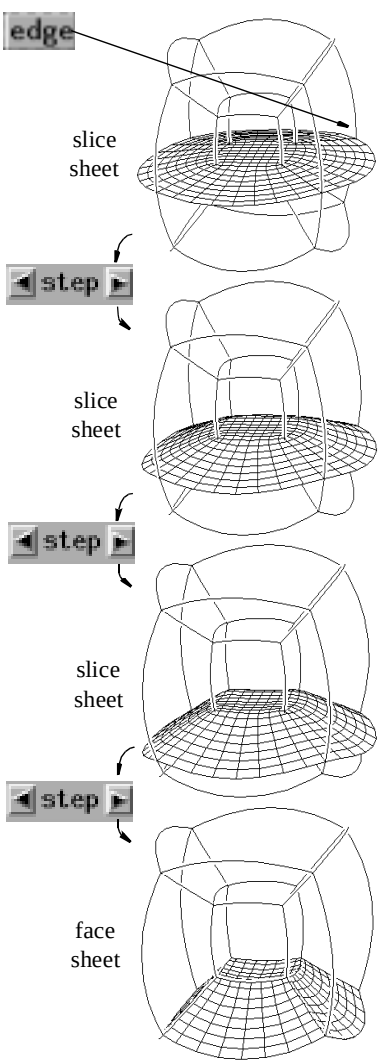


Figure 6.11



Aside from creating sheets about surfaces, you can also create sheets at any desired IJK setting. By pressing [edge], then clicking any point on an edge, you create a sheet composed of slices. When you move the cursor to an edge, the cursor should become a vertical, double arrow:  The sheet will appear probably close to where you click on the edge, at the closest IJK constant index. After creating a sheet, you can use the arrow buttons located on both sides of the [step] operation in the green, "CUR" box to "step through" different sheets. All but two of the sheets you "step through" will be slice sheets; the sheets present when I, J, or K is a maximum will be face sheets. Remember, there are a finite number of slices because each node of a slice is on an IJK lattice point. Figures 6.11 and 6.12 illustrate the "stepping" process.

Figure 6.12

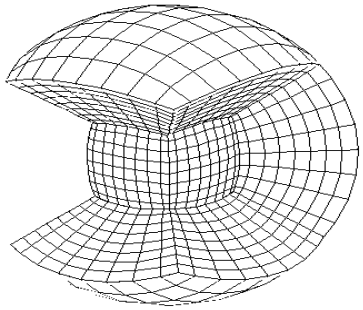


When you have one or more sheets on display, you can select a color scheme that will fit your needs. You can color each sheet by slices or faces, or just color every sheet differently. In the green, unnamed box, the [c:she] button opens a menu containing the "color by blk," "color by

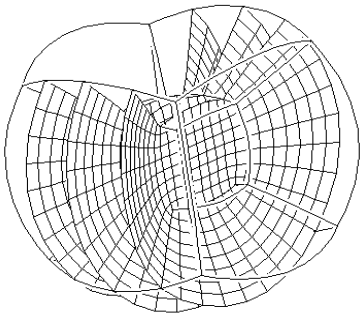
sheet," and "color by IJK" operations. The "color by sheet" operation, when accessed, colors each displayed sheet differently (there is one color on each sheet). The "color by block" operation, when accessed, colors each *slice or face* of each sheet differently.

By creating slices, shells, and sheets after deactivating some blocks, you can obtain useful cross-sections of grids. When you create a sheet using the [edge] button, the sheet only cuts through activated blocks. Also, when you press [shell], a sheet is formed from the outer faces of only activated, outer blocks. Figures 6.13 and 6.14 display a small sampling of cross-sections of grids.

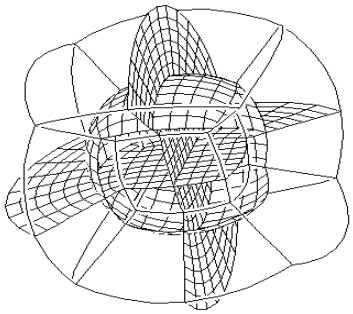
Figure 6.13



One face sheet is displayed. Three blocks are deactivated and the [shell] button has been pushed. Notice, the sheet *does* represent a shell because the outer faces of all outer activated blocks are activated. Also, notice that the center block is now an outer block



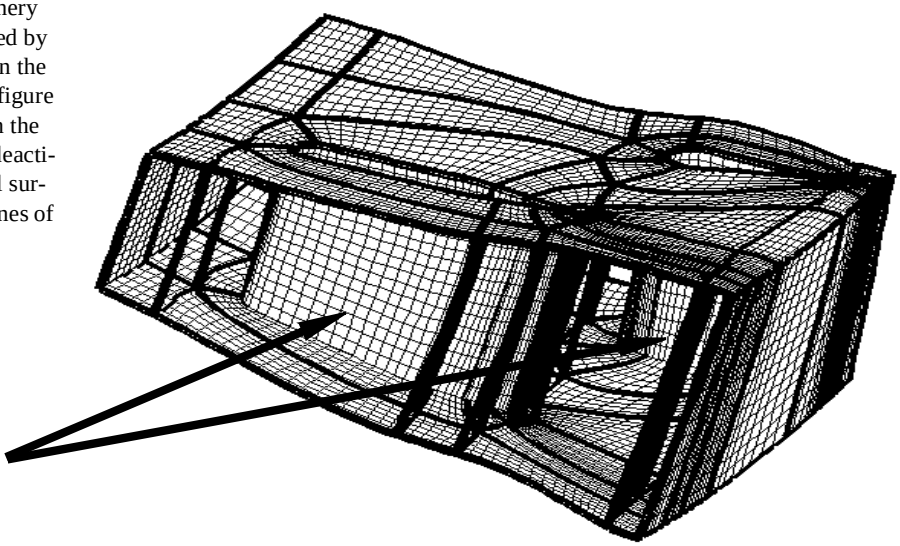
Three slice sheets are displayed and the uppermost block is deactivated



Three slice sheets are displayed. The front-most and rightmost blocks are deactivated.

Figure 6.14

This grid is one of turbomachinery blades. Each blade is represented by an interior surface. The faces on the top, bottom, right, and left of the figure are activated while the faces on the front and back of the figure are deactivated. Therefore, a partial shell surrounds the figure. Also, the outlines of all blocks are displayed.



Locations of hidden surfaces about which a grid has been created.

GridPro enables you to save blocks and sheets just as it allows you to save grids. To save a set of grid blocks, make the grid current, activate the set of blocks, then select the “save trimmed blk” option in the top menubar “grid” menu. To save grid sheets, make the grid current, display all and only the sheets you wish to save, and select “save sheets” in the top menubar “grid” menu.

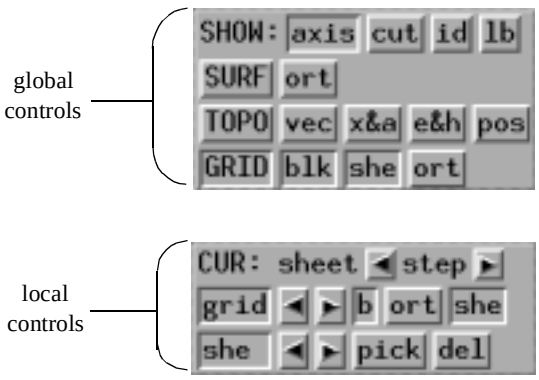
You can print out grids in color or in black and white. When you choose “print color” or “print BW” from the top menubar “grid” menu, GridPro writes the displayed blocks and sheets to the file ‘sheet.tmp’. A hidden line removal (HLR) is automatically performed on this file, and the results are written to a postscript file, ‘eps.tmp’. You can send this postscript file to a printer.

6.4 IJK Orientation, Global Control and Spacing

As you already know, each block has unique IJK positioning, one allowing slices and faces to be created. In order to see the IJK axes for each activated block, press the [ort] button found in the green, “CUR” box. The IJK axes will appear near the center of each block. For each set of axes, the I axis is pink, the J axis is green, and the K axis is yellow.

You know that you can color a sheet either uniformly or by faces or slices. Sheets can also be colored by the ILK orientation of their faces or slices. Because each face or slice is created when I, J, or K is a constant, each face or slice can be colored depending on which coordinate is a constant. The [c:she] menu, found in the unmarked green box, contains the “color by IJK” operation. This operation allows you to color faces and slices as described above.

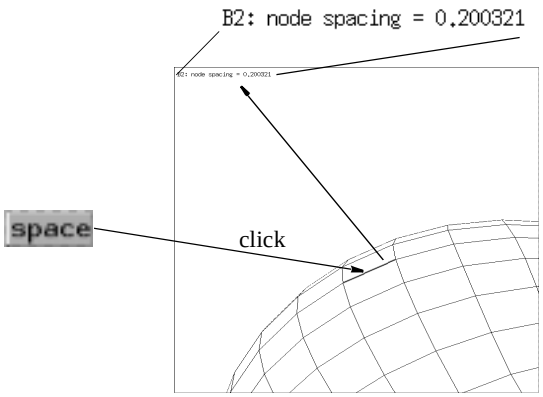
Figure 6.15



In complex grids, you will need to hide some grid regions in order to observe others. Hiding objects, such as sheets and blocks, are done on both a global and a local level. You cannot hide an object on a local level unless you have depressed the appropriate button on the global level. The green, “SHOW” box contains global operations and the green, “CUR” box contains local operations. See figure 6.15. You can toggle between *being able* to hide or to display blocks, sheets, and IJK axes only if the [blk], [she], and [ort] buttons in the green, “SHOW” box are depressed. You toggle between actually hiding or displaying blocks, sheets, and IJK axes with the [b], [ort] and two [she] buttons in the green, “CUR” box. Remember, one [she] button hides the current sheet, while the other [she] button hides all sheets on the current grid.

Sometimes, you will need to test the smoothness of a grid or to measure precisely dimensions of a grid. You can measure precisely the distance between two nodes by first pressing the [space] button found in the unmarked, green box, then moving your cursor to the line connecting two nodes and clicking the left mouse button. The cursor will turn into a red or white double arrow,  , as it approaches this line; you might have to zoom in so that the double arrow becomes white. *You must be in the [c:she] mode in order to use [space].* Also, the sheet on which you are measuring distance must be current. When you have successfully clicked a grid line after pressing [space], the line becomes red. The length of the line (or curve), or the node spacing, is displayed in the top left corner of the drawing area, as in figure 6.16.

Figure 6.16



The [space] operation proves useful when you use clustering in your grid. The button allows you to check precisely off-the-wall spacing near a boundary.

EXERCISE #17:

Grid a superellipsoid of power 5, then represent the grid as a shell. Cut open the grid shell by deactivating faces. Then, deactivate the whole shell and create a sheet composed of slices. Use the [step] operation until you have observed at least 5 other, corresponding slices.

Finally, find the distance between any three pairs of nodes.

Chapter 7 - Surface Geometries

7.1 Overview

Until now, you have seen only a small sampling of GridPro's surface definitions in the form of ellipsoids. More generally, GridPro contains two types of surfaces: implicit (analytic) surfaces and explicit surfaces.

Implicit surfaces are defined as equal potential surfaces of a scalar valued analytic function of the (x, y, z) position vector. These surfaces are defined by the equation, $u(x, y, z) = 0$. Hardwired into GridPro are four special cases: the -plane, -ellip, -xpolar, and -xyz surfaces. You have already learned about ellipsoids; planes are discussed in the next section. The other two surfaces deal with periodic boundary conditions and are discussed in section 7.5. GridPro also allows you to create a generalized, non-built in implicit surface, named -implic. This rarely used surface plays a small part in the AZ Graphic Manager, and is discussed in the TIL manual.

Explicit surfaces are defined by parameters. GridPro contains four explicit surface types: the -linear, -quad, -tria, and -tube surface types. Explicit -linear surfaces are single patch bilinear parametric surfaces, or surfaces composed of multiple patches of bilinear parametric surfaces. A -quad surface is composed of unstructured quadrilateral elements (-tria surfaces are defined similarly). A -tube surface is a surface of revolution about a cartesian curve. This surface type is often used for internal surfaces. Chapter 8 describes -tria and -quad surfaces; the TIL manual describes -linear and -tube surfaces.

7.2 Planes

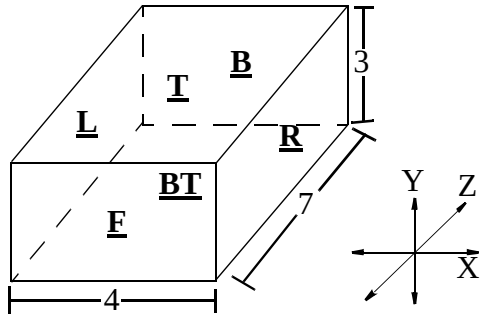
Planes represent one of the most simplest surfaces. You should learn well the operations associated with planes because of their central role they play in defining regions to be gridded. Keeping track of and careful positioning of planes is essential.

Planes can be defined either by an equation, or by a center point and a normal vector. The general equation for a plane is $ax + by + cz - d = 0$. Planes are inputted in the TIL code according to this equation. However, you need input only a plane's center and normal vector in the AZ Graphic Manager.

We will best learn about planes through an example problem.

EXAMPLE PROBLEM #10
Grid the region inside a rectangular box 7 units long, 4 units wide, and 3 units high.

Figure 7.1



We first restart GridPro in order to obtain the default settings. Then, we have to create surfaces and a topology to grid the box. Because a rectangular box has six faces, we can define the region to be bounded by six planes. Therefore, we have to input six surfaces (six planes). To simplify our task, we should center the box around the origin. Figure 7.1 organizes our region. Each letter in the figure represents a face (front, back, left, right, top, and bottom). The wireframe we will construct only need be a single block.

In order to input a plane, select the "load -plane" operation from the "surf" menu. The box shown in figure 7.2 appears on the screen. Notice that this property box is different for planes than for ellipses. In fact, each type of surface has a unique property box.

Figure 7.2

plane property box

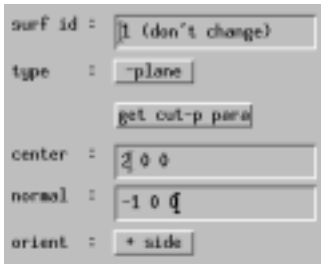
Let us first create the "L" plane, part of which will represent the "L" face on our prism. A plane can be defined by its normal vector and a point, and is of course infinitely long. Like the cut plane, surface planes are represented by rectangles. The center of this rectangle is the point that helps define the plane. With the center of the prism being at the origin, the center of the left face is at the coordinates (-2, 0, 0). Thus, we input (-2, 0, 0) in the "center" box. Of course, we could input any other point on the plane. Then, although the grid will not be changed, the visual representation of the plane will. The normal of a plane helps define the size of the plane's rectangle representation and the plane's orientation. The larger the magnitude of the normal vector, the larger the rectangle will appear. Because we want all planes to be visually represented equally in this problem, we will set the magnitude of the normal equal to one for each plane. Because this plane represents the left face, and we want to grid inside the prism, the normal must point in the positive x direction; we input in the "normal" box the coordinates (1, 0, 0).

Now, we must set the surface orientation. However, because we already defined the direction of the normal vector (and thus where we want to grid), we leave the default setting of "+side" in the "orient" box. *If we changed the orientation to "-side," we would reverse the direction of our normal vector.* Ignoring the other options for now, we press [ok]. A rectangle should appear on the drawing area in the specified position. To see the rectangle better, we hide the cut plane and rotate the drawing area.

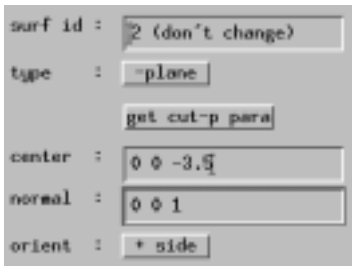
Now, we will repeat the procedure for five other planes, altering only their “center” positions and normal coordinates. Figure 7.3 illustrates what we input for the other five planes.

Figure 7.3

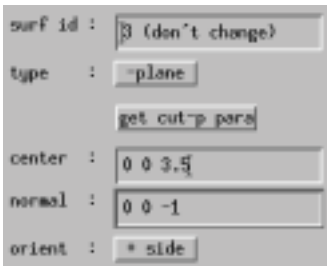
R



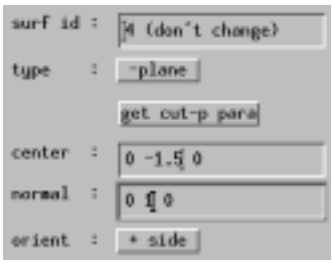
B



F



BT



T

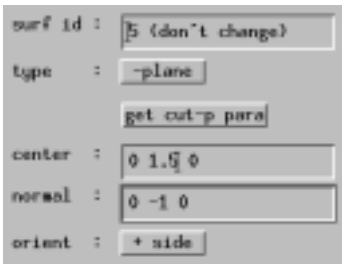
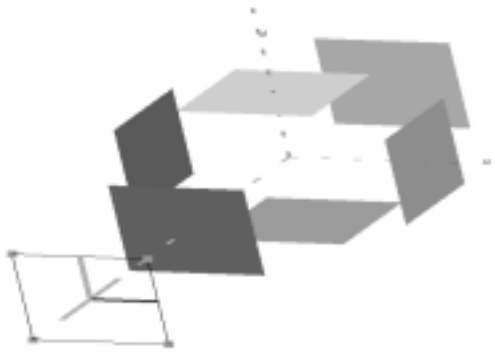


Figure 7.4

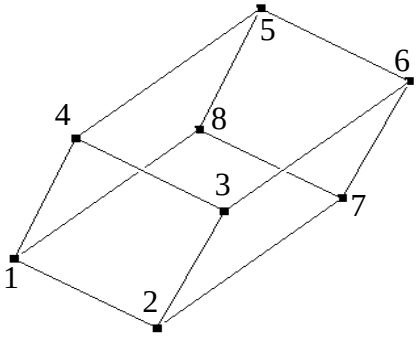


surfaces used to grid a rectangular prism

Now, all of our surfaces are implemented; we can construct the topological wireframe. We first depress the [cut] button to see the cut plane, then press [hand] to make the cut plane handles appear. Because the cut plane’s normal is along the z-axis by default, we will construct the wireframe just through translation of the cut plane on this axis. By dragging the appropriate cut plane axis, we move the cut plane just outside the front prism face. We also press [pos] to remove screen clutter. Our drawing area should now look like that of figure 7.4.

The wireframe we are constructing is composed of just a single block. We create four corners on the cut plane near the handles, then translate the cut plane again to a point right behind the back face. Instead of creating four more corners and linking them to the original corners, we can group the original corners and use the "+drop back edges" operation. To do so, we depress the group [1] button, then the [+] button (located under the [2] operation), and draw a purple box around the original four corners. We then press the [cp] button in the beige "TOPO" box, and select "+drop back edges" from the corresponding menu. Figure 7.5 displays the completed wireframe, with corners numbered for reference.

Figure 7.5



completed wireframe

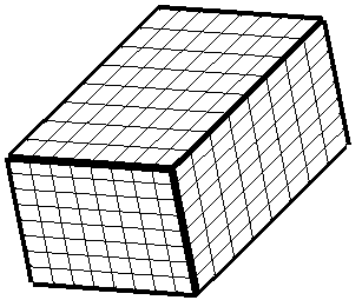
Figure 7.6

surface	corners
F	1 2 3 4
B	5 6 7 8
L	1 4 5 8
R	2 3 6 7
T	3 4 5 6
BT	1 2 7 8

Our next task is to assign corners to surfaces. Because four corners will be assigned to each surface, we use the [+] button (near the [S.] button) each time to be more efficient. Figure 7.6 displays a list of the surfaces and the corners to which we assign them.

Finally, activate Ggrid by selecting the "Ggrid start" operation under the "TOPO" menu. Figure 7.7 displays the completed grid. ***EXAMPLE PROBLEM #10 COMPLETED.***

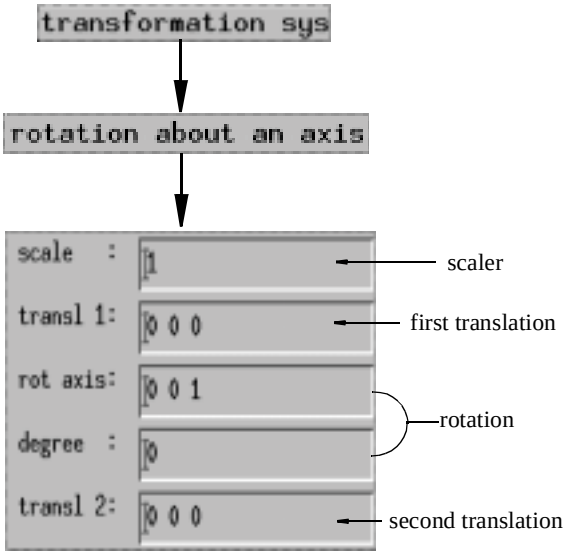
Figure 7.7



completed grid

For easy positioning, GridPro allows you to translate and rotate surface planes using two transformation systems. In the surface properties box, the "transformation sys" operation opens a submenu containing the "rotate+translate" and "rotate about an axis" modes. The default mode is the "rotate about an axis" mode.

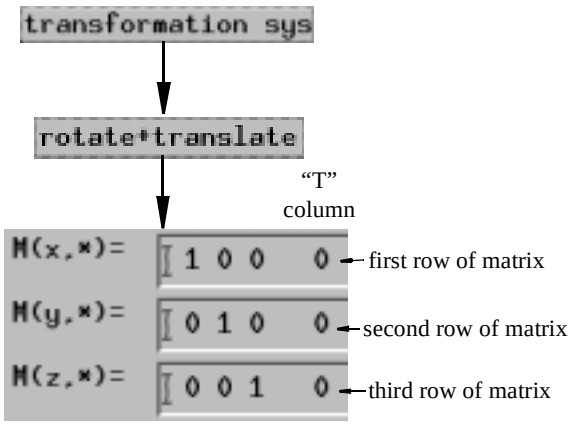
Figure 7.8



The default mode allows you to scale, translate, rotate, and translate again a plane's rectangle representation, as shown in figure 7.8. By entering a number other than 1 in the "scale" box, you can enlarge or shrink the rectangle accordingly. Then, you can translate the rectangle by a certain number of units along each XYZ axis by entering coordinates in the "transl 1" box. After this translation, the rectangle will undergo a rotation. You specify the axis of rotation in the "rot axis" box, and the number of degrees to which the plane is rotated in the "degree" box. Then, after the rotation, the rectangle can undergo another translation. The coordinates in the "transl 2" box specify this second translation.

The "rotate+translate" mode allows you to position a surface plane with a transformation matrix. See figure 7.9. The equation governing the transformation is illustrated in figure 7.10.

Figure 7.9



The three operations in the property box not mentioned are common to all surfaces, not just planes. Located above the "transformation sys" button is the "label" box. Similar to labeling wireframe edges, labeling surfaces involves nothing more than inserting a name in the "label" box. Remember, you can label every type of surface, not just a plane. Remember, in order to see a label, depress the [lb] button in the green, "SHOW" box.

7.3 Scaling

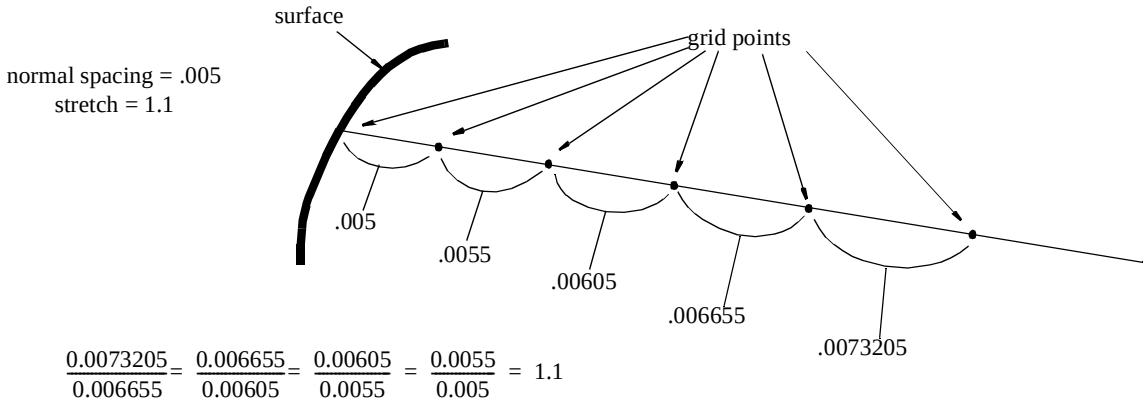
Figure 7.10

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} Mxx & Mxy & Mxz & Tx \\ Myx & Myy & Myz & Ty \\ Mzx & Mzy & Mzz & Tz \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

equation with translation+rotation matrix
XYZ - original position
X'Y'Z' - new position

Above the "label" box are two important boxes central to boundary conditions. These two boxes, the "norm-spc" and "stretch" boxes, allow you to specify the quality of a grid near a surface. Depending on your choice of obtaining an Euler grid or a Navier-Stokes grid, you will want to refine the boundary conditions for one or more surfaces. This refinement involves two components: the number of grid points and the rate at which the spacing between these grid points increases. Recall that setting the density also defines the number of grid points. However, density deals with the number of grid points on a global scale. The "norm-spc" box, short for normal spacing, deals with the number of grid points on a local scale (specifically, near boundary surfaces). The density constraints and the normal spacing constraints work in sync with one another. *GridPro allows for this conjointment because of the cells created by the density constraint, only the one containing the boundary in question will be affected by normal spacing*. Of course, all other cells will remain unchanged. Specifically, the number you insert in the "norm-spc" box is the length of the cell edge measured between a boundary grid point and the next closest "off-wall" grid point. The ratio between this length and the cell edge length measured between the off-wall grid point and the next off-wall grid point is the number you input in the "stretch" box. This number becomes the ratio for a geometric progression of cell edges, which propagates outward from the boundary surface along off-wall grid points. Typically, the value of normal spacing is less than 10^{-2} of the length of the boundary. Also, the "stretching" value usually lies between 1.1 and 1.5. See figure 7.11.

Figure 7.11



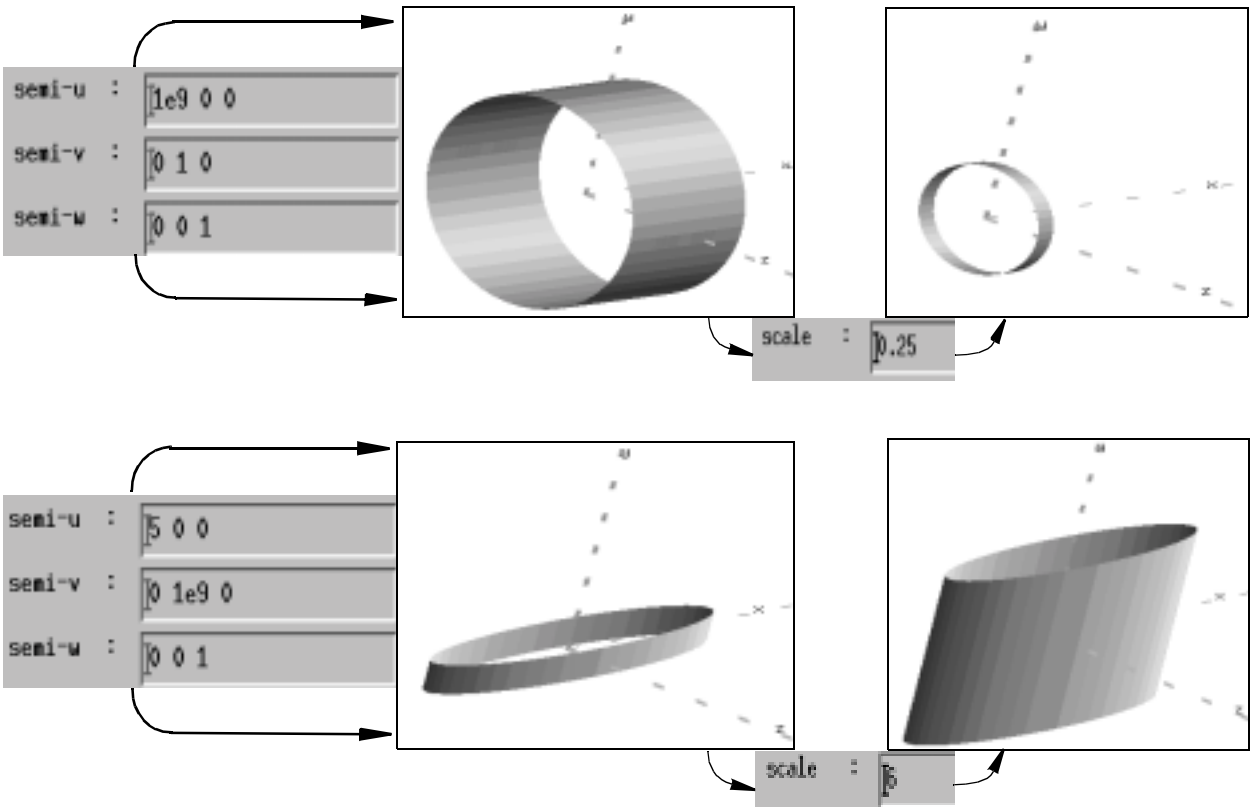
$$\frac{0.0073205}{0.006655} = \frac{0.006655}{0.00605} = \frac{0.00605}{0.0055} = \frac{0.0055}{0.005} = 1.1$$

If you choose, you do not need to input numbers into the "norm-spc" and "stretch" boxes. Gridpro will, by default, set the "norm-spc" value to

7.4 Cylinders

Like a plane and ellipsoid, a cylinder represents a commonly used surface in GridPro. GridPro interprets cylinders as ellipsoids with one infinitely long axis. To create a cylinder, simply access the “load -ellip” operation in the “surf” menu, and give one axis a very “long” length. To input a “large” number, for example 10^9, type “1e9.” In general, GridPro’s shorthand scientific notation for the expression ($A \times 10^B$) is (AeB), where B is an integer. Remember, cylinders are infinite; what you see on the drawing area is a representation of the cylinder, just as the cut plane rectangle is a representation of the cut plane. Also recall that you vary the size, but not proportions, of a representational cylindrical piece by changing the number in the “scale” box from its default value of 1. Figure 7.12 provides some examples.

Figure 7.12

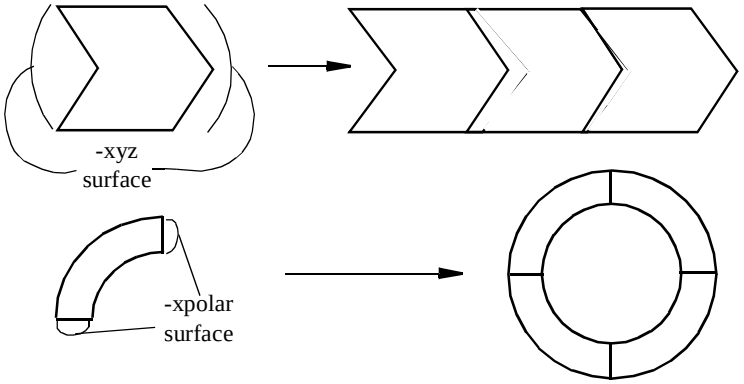


7.5 Symmetric Surfaces

Some geometries are translationally symmetric or rotationally symmetric. Rather than computing the whole grid about such an object at once, GridPro can grid a “section” of the object, then duplicate that section to construct easily the grid about the whole object. This process saves you time and effort, and allows you to construct topology for only a fraction of the region you

want gridded. Gridding a symmetric object can be done in the Graphic Manager. However, in order to duplicate your grid "section," you must use the "trf" utility.

Figure 7.13



The required "section" of symmetric objects are created with pseudo-surfaces, called "-xpolar" and "-xyz." The "-xpolar" surface is used for rotationally symmetric geometries and the "-xyz" surface is used for translationally symmetric geometries. These surfaces comprise nothing more than the *periodic* borders of the object. See figure 7.13. *These surfaces do not have a physical representation until you input the geometry.* However, you can still input these periodic surfaces before the geometry; you specify only the surface's period, normal spacing, stretch, and label. You can specify any period for "-xyz" surfaces. The "xyz" period number corresponds to a length, and will relate to the length of the "section." For "-xpolar" surfaces, you specify a period in degrees. This degree measure must be a factor of 360. In order to produce the figures on the right side of figure 7.13, you input the periods given in figure 7.14.

Figure 7.14

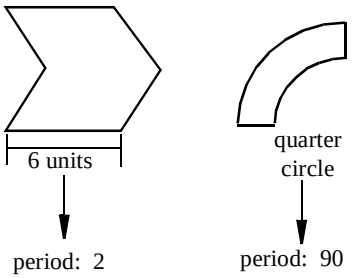
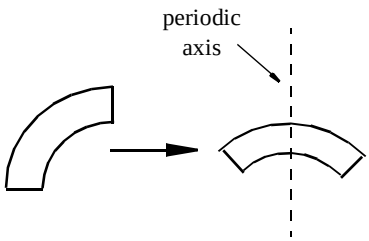
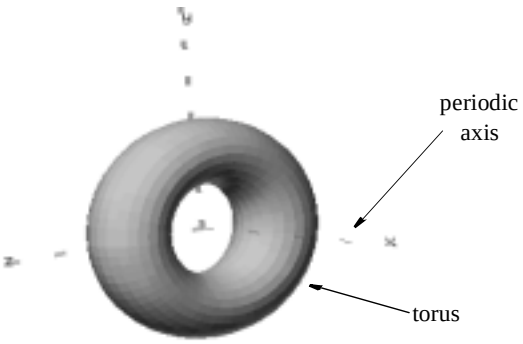


Figure 7.15



In addition to specifying a period, you must specify the axis about which you wish to make a "-xpolar" object periodic. However, no part of the "-xpolar" surface should lie on the periodic axis. The x-axis serves as the default periodic axis. However, you can change this axis by setting the translation and rotation properties in the "-xpolar" surface property menu, similar to what figure 7.8 displays. Figure 7.15 displays an example of how an "-xpolar" surface should be aligned.

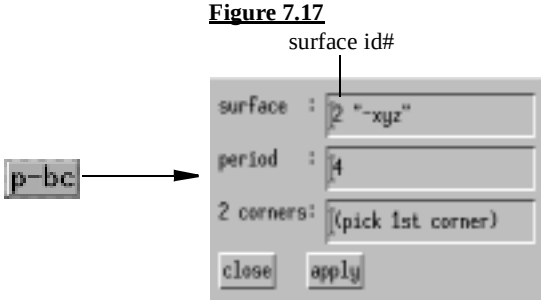
Figure 7.16



Unlike other surfaces you have seen up to now, *you cannot see periodic surfaces as soon as you input them*. Instead, for "-xpolar" surfaces, GridPro automatically displays a representation of the periodic axis. This representation is a torus. The axis extending through the torus' center hole is the periodic axis. Keep in mind that the torus serves solely as a visual aid, and in no way represents the physical properties of a surface. You can scale the torus to any size. No representative surface exists for "-xyz surfaces;" they are completely invisible.

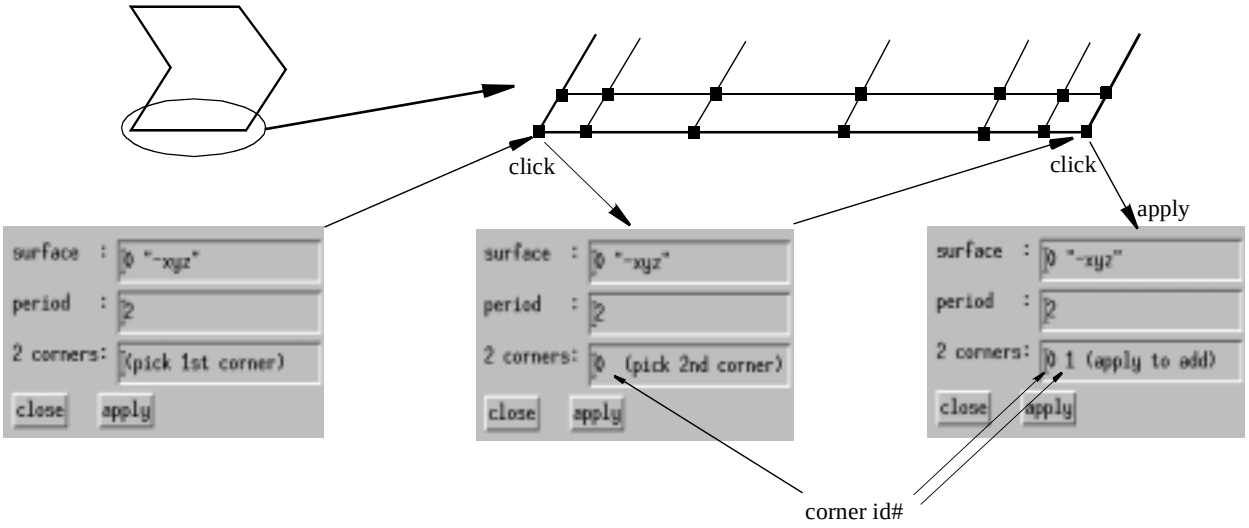
Similar to any other surface, you must assign corners to the "-xpolar" and "-xyz" surfaces. Unlike other surfaces, you have to use the [p-bc] (periodic boundary condition) button in the brown "TOPO" box when assigning corners to periodic surfaces. Usually, inputting your periodic surface is the *first* step in creating a grid; assigning corners to this surface is the *last* step. Just as the periodic surface is on the periodic border of an object, the corners assigned to the periodic surface are on the periodic border of the wireframe.

Figure 7.17



After pressing the [p-bc] button, the pop-up window in figure 7.17 appears. The contents of the box describe the current surface. The current surface in figure 7.17 is an "-xyz" surface. Its period is 4 and surface identification number is 2. When the pop-up window is activated, you assign corners by merely moving the cursor to the corners and clicking the left mouse button. However, you must assign corners in pairs. These pairs of corners should be roughly symmetrical about the periodic axis. That is, one corner should reflect on its twin about this axis. After you click on both corners in the pair, you must press the apply button. If you mistakenly click on the wrong corner in a pair, click on the corner again to deactivate it. Figure 7.18 illustrates the process. Remember that periodic corner assignments are always on the periodic border of the wireframe.

Figure 7.18



7.6 A Rotationally Symmetric Turbine - an Example

You will better understand the procedure of producing grids on periodic surfaces through a practical example problem.

EXAMPLE PROBLEM #11
Grid a rotationally symmetric turbine containing 12 curved blades.

First, we must be absolutely sure of the region we wish to grid. If we are not, then we cannot properly place the periodic axis, and Ggrid will distort our desired grid. Because we are dealing with a rotationally symmetric grid, we access the “load -xpolar” operation under the “surf” menu. In order to produce 12 blades, we will need to create a representative “section” which can be rotated 360/12 = 30 degrees. So, we enter the number 30 in the “period” box. Now, we should define our periodic axis. For the sake of simplicity, we use the default setting (the x-axis) and press [ok].

Remember, we cannot see the -xpolar surface yet because we have not defined our geometry. However, we should see the torus encircling our periodic axis. As we build the topology, this torus will serve as a reminder as to where the periodic axis lies.

Figure 7.19

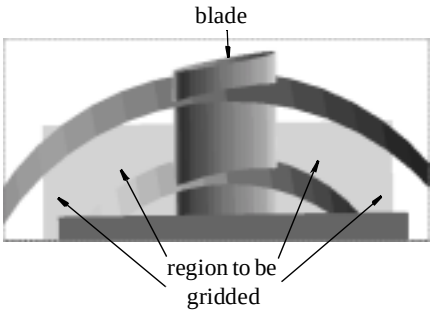


Figure 7.19 illustrates how our cross-sectional "section" should look. Remember, by gridding only the region in figure 7.19, we can obtain the grid of the whole turbine. The turbine has 12 blades, not one. By analyzing the figure, we realize that, aside from our -xpolar surface, we have to input 5 surfaces. Two of these surfaces are planes and three are cylinders. Figure 7.20 illustrates the surface each geometry represents. When we actually input the surfaces, the torus will also be present.

Figure 7.20

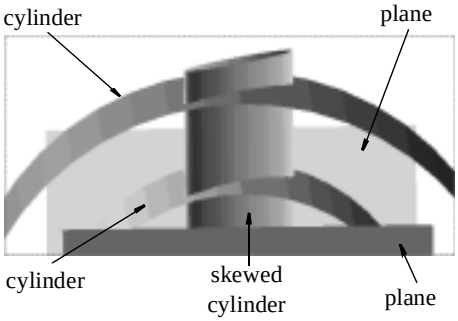
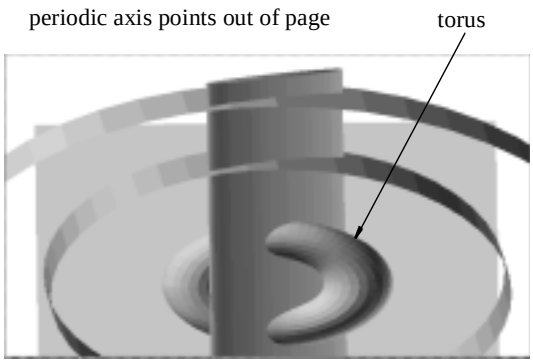


Figure 7.21



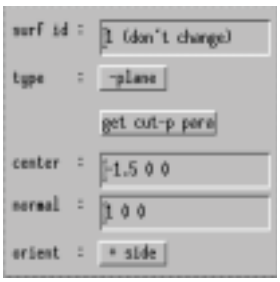
Now we have to input the surfaces correctly so that they are in correct relation to one another and the periodic axis. Remember, the periodic axis should not cut through any part of the periodic surface. In figures 7.19 and 7.20, the invisible planes connecting the two cylinders will compose the -xpolar surface. Therefore, the periodic axis must cut through the center of the planes, perpendicular to the long dimension of the blade. In figure 7.21, the front plane is hidden and the -xpolar representative torus is exposed. In the figure, the periodic axis points out of and into the page.

As we position our figures, we must keep in mind that the periodic axis is the x-axis. Also, we should remember that the torus only represents the more abstract "-xpolar" surface, and that the planes and cylinders are infinite. Because we want the periodic axis to cut through the center of the two planes, we let these centers be offset from the origin by, say, 1.5 units in the x direction. The inner and outer cylinders (or the hub and shroud) for our grid are infinitely long in the x direction. Let us define the outer cylinder to be 3/2 times as far from the origin as the inner cylinder. The curved blade is a skewed cylinder that is infinite in the y direction. To add perspective to our drawing area, we resize some of the surfaces. We shrink the torus, stretch the blade, and slightly elongate the hub. The properties of the surfaces are illustrated in figure 7.22.

Figure 7.22

periodic -xpolar surface

left plane



right plane

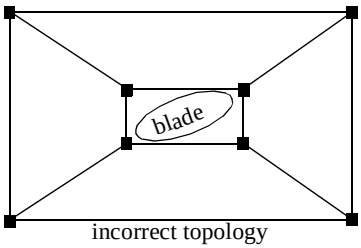
hub

shroud

blade

Notice the orientation of the surfaces and the normals of the planes in figure 7.22. Because we are gridding the region outside of the hub and inside of the shroud, we define the orientation of the hub to be positive and the shroud to be negative. Also, because we are gridding around the blade, the blade's orientation is positive. To account for the grid in the region between the two planes, we could define the normal vector for each plane differently, or we could define the same normal vector for each plane with opposing orientations. We chose the former option for simplicity.

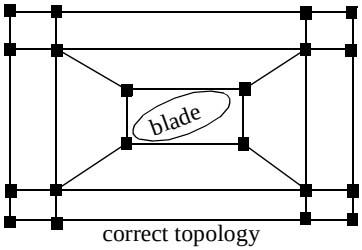
Figure 7.23



incorrect topology

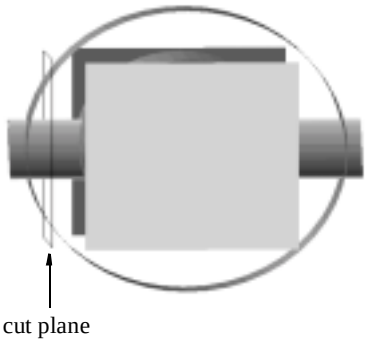
After inputting our surfaces, we are now ready to sketch our wireframe. Wireframes are normally supposed to "flow" with the region to be gridded. However, GridPro automatically curves an -xpolar surface so that the grid is symmetrical with respect to the surface's period. Therefore, instead of having to curve our wireframe to fit the contours of the desired region, we can just construct a rectilinear wireframe border. In fact, a single block wireframe would be sufficient if not for the presence of the blade. The blade forces us to wrap inward our block, so that we can assign the wrapped corners to the blade. In addition, we need another wireframe rectangle in between our inner wrap and the outer boundary. If we attempted to grid the wireframe in figure 7.23, grid lines would ripple out from the blade and form singularities at the periodic boundaries. However, we greatly improve our grid quality by constructing the wireframe in figure 7.24.

Figure 7.24



correct topology

Figure 7.25



Now, we can create the wireframe. First, we will create the outer face of our block. We position the cut plane so that it cuts through a small portion of the shroud, as in figure 7.25. Then we create four corners such that each corner is roughly aligned with a cut plane. We connect the corners with edges, as in figure 7.26.

Figure 7.26

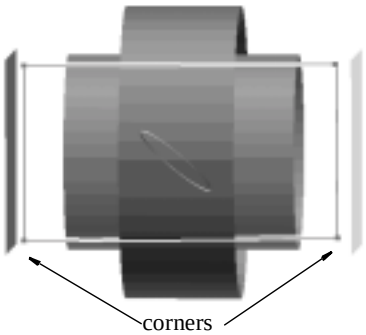
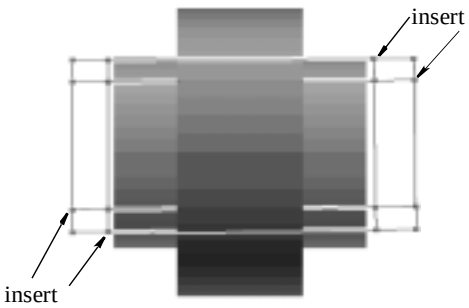


Figure 7.27



By leaving the cut plane in the same position, we can create the middle layer and inner wrap for the outer face of our wireframe. To best construct the middle wrap, we use the [i:a] button. We press this button, and select "all insert" in the menu. Then, we hold down the keyboard <I> key, and click the four positions specified in figure 7.27; our middle layer is complete. Now, we must create an inner wrap. Because we want the wrap to take place on a plane, our wrap is defined as a one-dimensional wrap. Therefore, we must exclude the face of the string of edges we are wrapping. To do so, we hold down the keyboard <F> key and click on opposite vertices of the middle layer rectangle. A dashed red line should appear between these corners. We place the middle layer in a group, press the [wrp] button, and select "50% smaller." If we look at the blade in relation to the wrap, the wrap seems a little too small; we enlarge it slightly by dragging the each of the four innermost corners a little outward. The shroud is hidden in figure 7.28 so that we could see the result of the wrap.

Figure 7.28

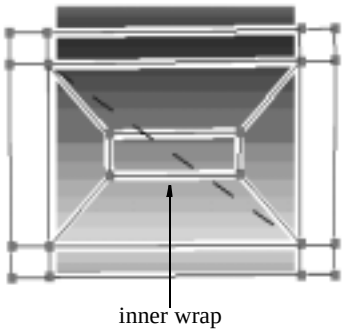
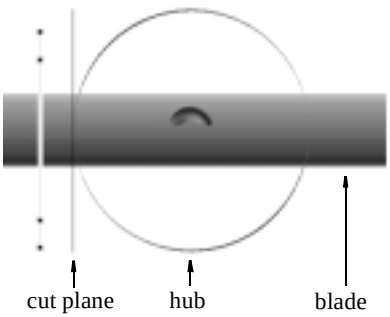
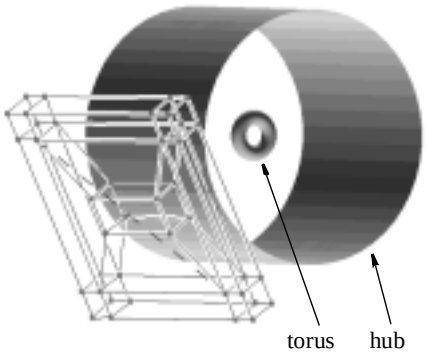


Figure 7.29



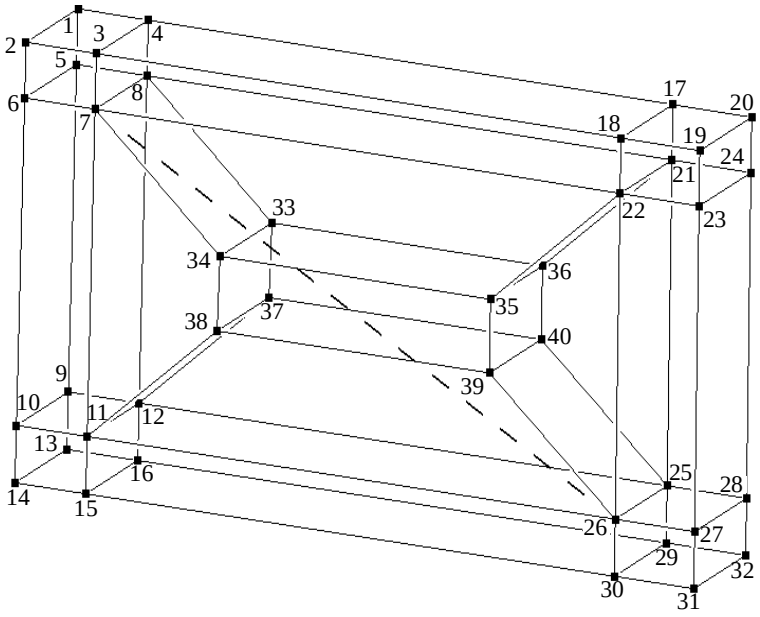
Now, we must duplicate the wireframe on the "back face", and link the corners of the two faces. To do so, we first rotate the drawing area so we can see the cut plane edge on, and move the cut plane slightly inward, to a point almost tangent to the hub. See figure 7.29. We place the wireframe in a group, push the [cp] button and select "+drop

Figure 7.30



back edges." Figure 7.30 illustrates the result. Our wireframe is now complete. Note that figures 7.29 and 7.30 hide most of the surfaces in order to improve visualization and to decrease screen clutter.

Figure 7.31



To complete our topology, we should now assign corners to surfaces. For convenience, we number the corners, as in figure 7.31. The dashed line across the front face signifies that the face is excluded; remember, we needed to exclude that face in order to obtain a one dimensional wrap. Figure 7.32 reveals which corners should be assigned to which surfaces. Let us first assign corners to all surfaces except the -xpolar surface. We expedite our surface assignments by using the [snap] button in conjunction with the appropriate [+] button. For example, by snapping the wireframe so that the blade is pointing out of the page, we can assign corners 1,2,5,6,9,10,13, and 14 with only one purple box. When assigning corners, we best remember which surface is current.

Figure 7.32

urface	*Corner numbers*
hub	1,4,5,8,9,12,13,16,17,20,21,24,25,28,29,32,33,36,37,40
shroud	2,3,6,7,10,11,14,15,18,19,22,23,26,27,30,31,34,35,38,39
plane	1,2,5,6,9,10,13,14
plane	19,20,23,24,27,28,31,32
blade	33,34,35,36,37,38,39,40

After completing the assignments specified in figure 7.32, we can assign corners to the -xpolar surface. Because the torus represents an -xpolar surface, the -xpolar surface is current when the color of the torus is sea blue. So, we toggle the current surfaces until the torus becomes sea blue. Now, remember that we must use the [p-bc] button, and not the [S.] or [+] buttons, to assign corners. Also, remember that we assign corners in pairs. These pairs are reflective. That is, one corner will map onto its partner when reflected about the periodic axis. Ggrid will perform this map - we must identify which corners are to be mapped.

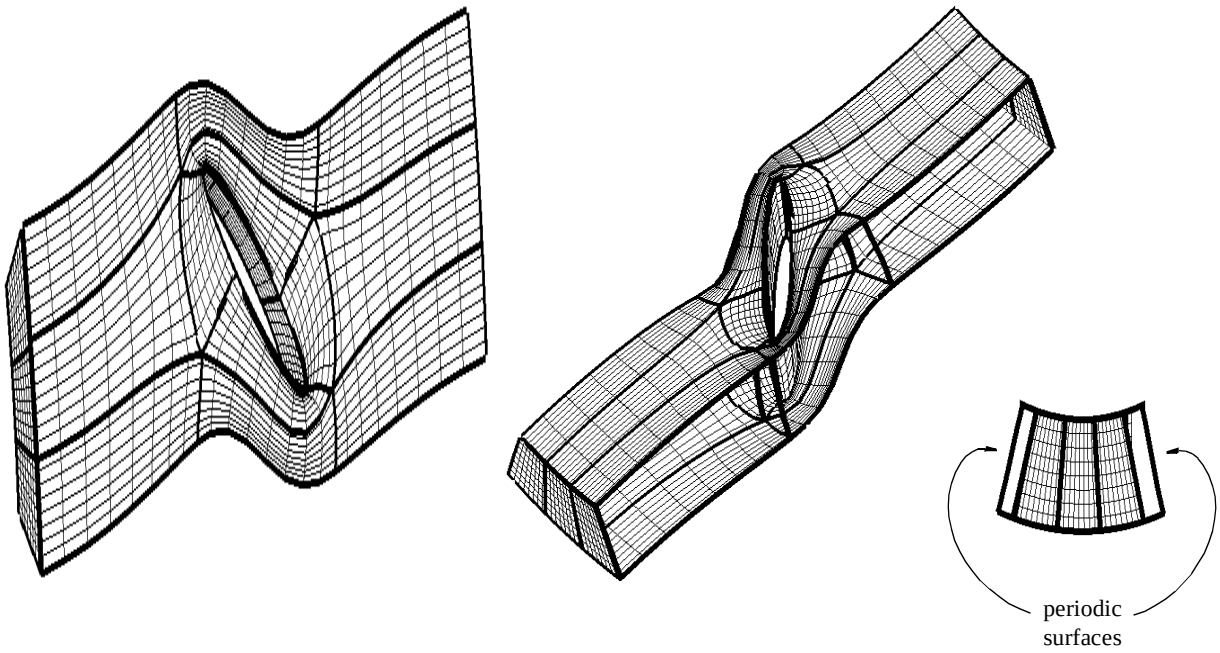
Figure 7.33

corner pairs
1-13
2-14
3-15
4-16
17-29
18-30
19-31
20-32

We do not need to pair all corners, only those on the periodic surface. These corners include the 8 topmost and 8 bottommost corners in figure 7.31. Figure 7.33 displays the pairs. Recall that in order to assign one pair of corners to a periodic surface, we open the [p-bc] box, click on both corners, and press apply.

After assigning all pairs, our topology is complete. We initiate the Ggrid process, switch to the "Grid Viewer" mode, and press [shell] in order to see our grid. Note that [shell] does *not* produce a grid outline on the faces of the periodic surfaces. Figure 7.34 displays three different views of our grid. In the snapped view, notice the periodic surfaces, and visualize how 12 such surfaces could combine and form a turbine with 12 blades. As mentioned before, we must use the "trf" utility and in order to complete our grid from our finished grid section. ***EXAMPLE PROBLEM #11 COMPLETED***

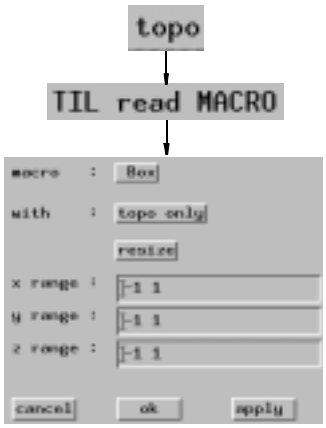
Figure 7.34



7.7 TIL Macros

GridPro recognizes the usefulness and heavy use of cylinders, planes and ellipsoids in many applications. Therefore, the program allows you to input these regions as surfaces and/or topologies without making you toil through the mathematics by using macros. To use a macro, select “TIL read MACRO” in the “topo” menu (on the top menubar). The box shown in figure 7.35 will appear.

Figure 7.35



The first three buttons in the macro box open up submenus; the first two represent default settings. The [box] submenu allows you to choose the surface and/or topology you want created. Each region has its own macro box, as shown in figure 7.36. The [topo only] submenu provides you with the option of letting GridPro produce the surface, topology, or both of a region. Keep in mind that a topology includes a wireframe and surface assignments. See figure 7.37. The [resize] submenu allows you to resize and/or fit your surface and/or topology.

Figure 7.36

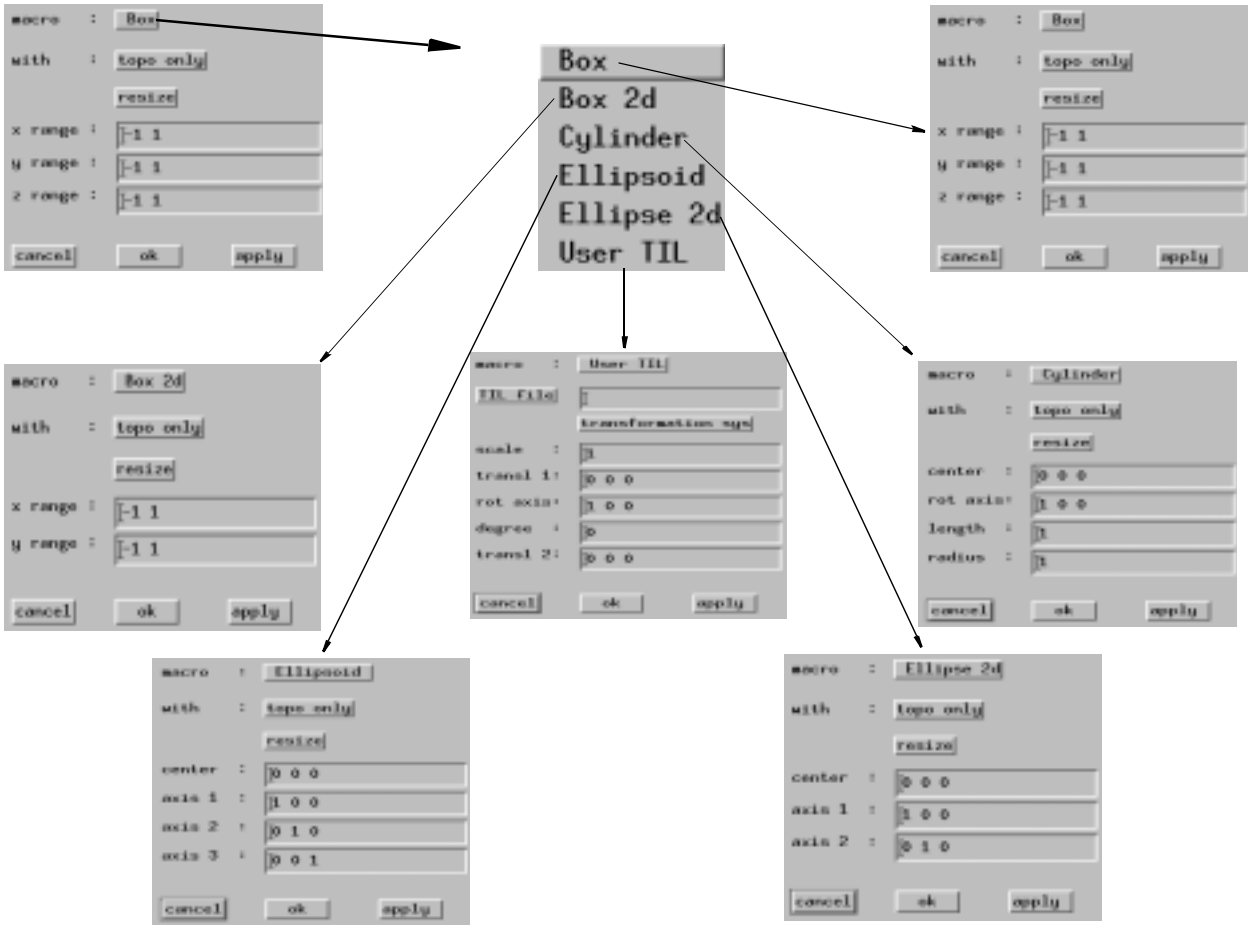
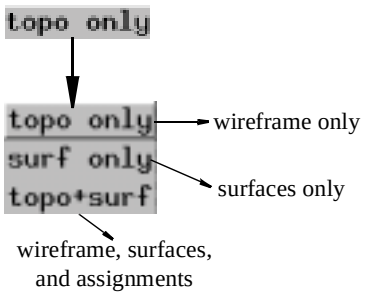
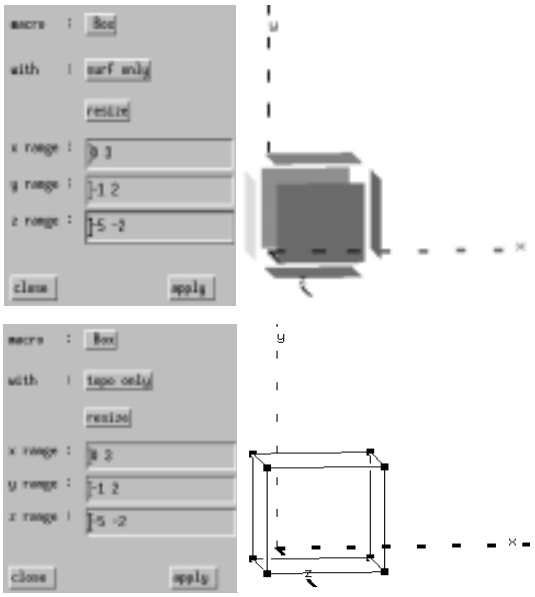


Figure 7.37



Each macro box allows you to specify properties of the surface and/or topology being created. The [box] selection allows you to create automatically a three-dimensional region bounded by 6 planes. The box 2d" selection allows you to create automatically a two-dimensional region bounded by 4 planes (you must be working on the plane z = 0). You can set the dimensions of these regions by changing the numbers in the "x range," "y range," and "z range" boxes from the default settings of -1 or 1. After you set your properties, you must press [apply]. Figure 7.38 provides two examples.

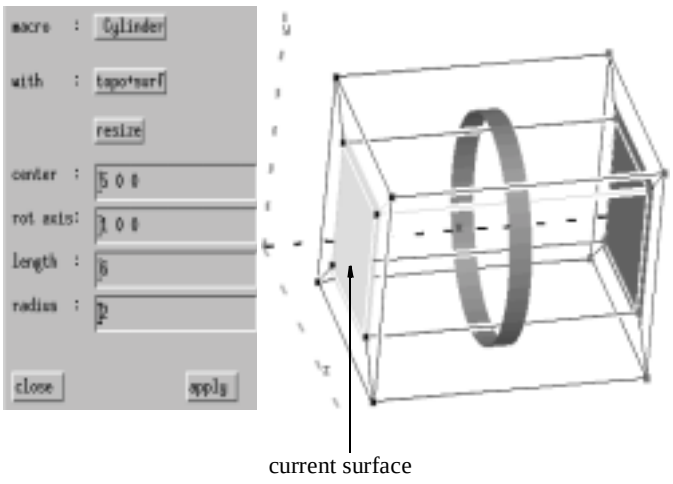
Figure 7.38



EXERCISE #18

Redo example problem #10; however, now use the "box" macro.

Figure 7.39



The "cylinder" box allows you to create automatically a cylinder with a center, radius, length, and rotation axis that you provide. Unlike the surface cylinders you create by extending one axis of an ellipsoid, this cylindrical region is not infinite. Two planes will automatically be positioned so as to bound a truly cylindrical region. The position of these planes depend upon the length of the cylinder. If you choose to create a surface out of this cylindrical region, the drawing area's representation of the cylinder is a fraction of that true length of the cylinder. This reduced representation allows you to observe and better manipulate around the intersections of the planes and cylinder. See figure 7.39. Notice in the figure that the [topo+surf] operation is chosen, and that the appropriate corners are assigned to the current surface.

EXERCISE #19

Grid the region bounded by a cylinder containing a unit cube. Use macros.

The “ellipsoid” box enables you to create an ellipsoidal region of any shape with any center, and the “ellipse 2d” box enables you to create an elliptical region of any shape with any center. To change the shape of the ellipsoid, change the numbers in the “axis 1,” “axis 2,” and “axis 3” boxes. These numbered axes represent the semi-u, semi-v, and semi-w axes of the ellipsoid.

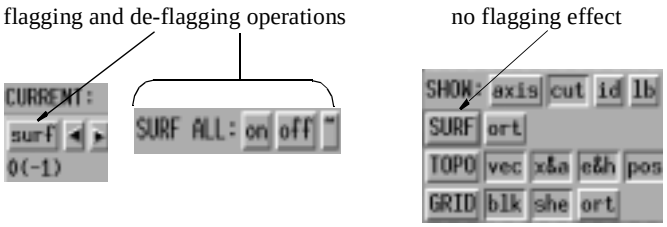
The “User TIL” box lets you modify an already existing TIL file. After you type in the name of the file in the “TIL file” box, you can choose a transformation system in the “transformation sys” submenu, and input transformation properties (as discussed in section 7.1). If you cannot remember the name of the TIL file you wish to transform, press the “TIL file” button to browse the directories. Note that the “User TIL” box deals only with wireframes, not surfaces.

The TIL macros are available solely to maximize your efficient use of GridPro. As you saw with exercises #18 and #19, you can save much time and effort by using macros. To complete exercise #18 without using a macro, you would need to load 6 planes separately, coordinate their orientations, and visualize their XYZ positions. With a macro, GridPro performs these tasks with the click of one button.

7.8 Surface Management

When you must deal with numerous surfaces, you will only want some displayed on the screen at the same time. You already know that you can hide the current surface, or all surfaces. But what if you just want to *see* the current surface, *and no others*? Do you have to toggle through every surface and hide it? GridPro answers this question by allowing you to flag surfaces.

Figure 7.40



The green, “SURF ALL” box, the lowest box in the “panel=T” panel, allows you to flag or de-flag surfaces. A flagged surface will be displayed whether or not it is current. A de-flagged surface will be displayed only when it is current. Pressing the [on] button in the “SURF ALL” box will flag all surfaces. Pressing the [off] button will de-flag all surfaces. Pressing the [~] button will reverse the flag classification of each surface. Also, locally hiding just the current surface (pressing the [surf] button in the green, “CURRENT” box) will de-flag it. However, globally hiding all surfaces (pressing the [SURF] button in the green, “SHOW” box) does *not* de-flag them.

If you input more than five surfaces, toggling among them can become tedious. To make a surface current without toggling, merely hold down the <S> key and click on this surface.

Chapter 8 - Surface Preparation

8.1 Overview

Before gridding a region, you must often read in several surfaces. Because GridPro has a specific definition of a surface, you have to modify a read-in surface to adapt it for use on GridPro. Gridpro contains features in the form of CAD functionalities that allow you to perform this modification.

GridPro can read surfaces from roughly eight different formats. These conversions deal with a GridPro utility, and are described in the TIL manual. The Graphic Manager contains operations allowing you to manipulate these surfaces into GridPro-read surfaces. You can find most of these operations in the "Mini CAD" panel, accessed through the "panel=T" menu in the top menubar. The "Topology Builder" mode includes a couple other CAD functionality operations. Note that, by default, the "Panel=S" (Mini CAD) mode displays only surfaces with hidden line remove. Most of the time you spend in the "Mini CAD" panel will focus on clicking on surface nodes. When your cursor approaches a node, the slanted arrow should transform into a straight arrow. Only when the cursor is a straight arrow do you know for sure that the cursor is on a node. Figure 8.1 displays the switchable operations appearing under the "panel=S" menu.

Figure 8.1

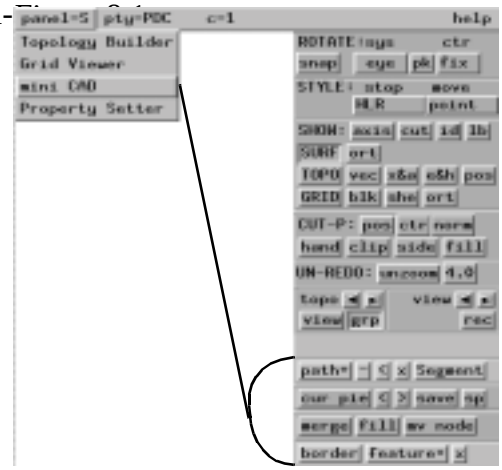
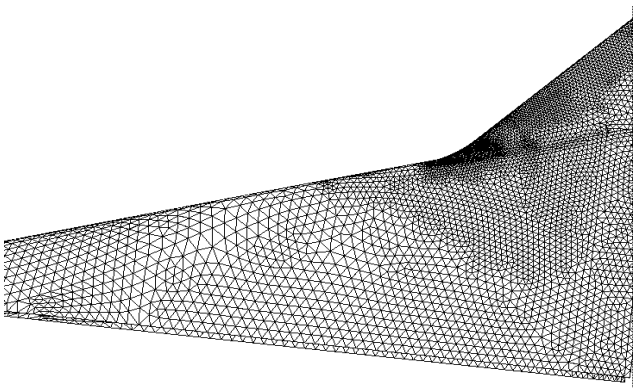
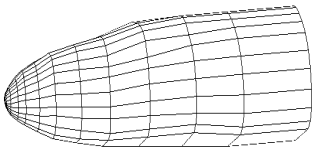


Figure 8.2



You can manipulate only -tria or -quad like surfaces in the "Mini CAD" panel. A -tria surface is a surface with three cells extruding from each node except border nodes. A -quad surface is a surface with four cells extruding from each node except border nodes. Figure 8.2 illustrates a -tria surface, and figure 8.3 illustrates a -quad surface.

Figure 8.3

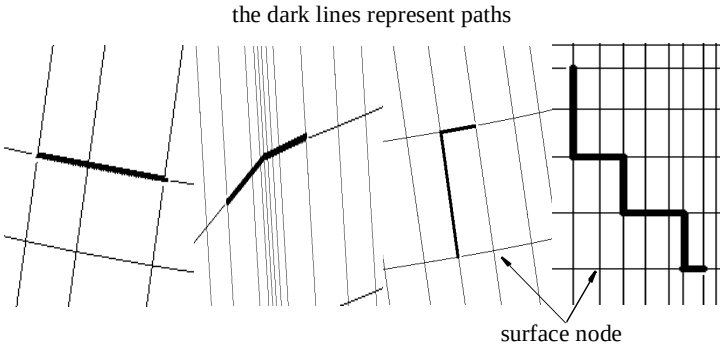


The extent of GridPro's CAD functionality includes creating, deleting, and refining surface parts or surfaces. GridPro allows you to split up surfaces, and fill the holes created. Also,

GridPro couples surface manipulation with topology, allowing transformations between surface nodes and corners.

8.2 Using Paths

Figure 8.4

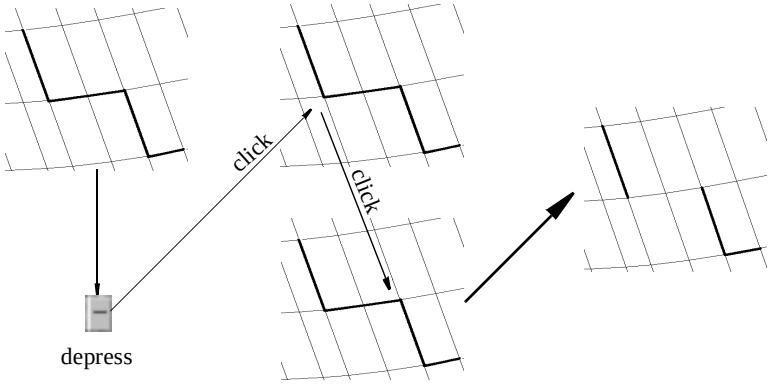


Creating paths is an essential tool in the surface manipulation process. A path is either a straight line between any two adjacent surface nodes, or a straight or bent line composed of paths between any two nodes. Paths appear as dark red lines. Each endpoint of a path appears as a white square. Figure 8.4 illustrates examples of paths. You can, and will often need, to create several paths.

In order to create a path, you must use the [path+] operation. After depressing the button, move the cursor to and click (with the left mouse button) on the two nodes that will act as endpoints for the path you wish to create. If the line connecting the two points is not along a surface cell, GridPro will automatically create a "zig-zag" path among the intervening cells. Notice that once used, the [path+] button will remain depressed (and activated) until you re-click it. Thus, after clicking two nodes, clicking a third will extend the path you just created. By repeatedly clicking nodes while [path+] is depressed, you can create as long and precise a path as you wish. Unlike inputting a string of topology edges among many corners, creating a string of paths (which is itself a path) requires no double clicks. Therefore, if you wish to create two disconnected paths, you must first create one path, *undepress the [path+] button, depress the [path+] button*, and create the second path. If you avoid the italicized steps, your paths would be connected.

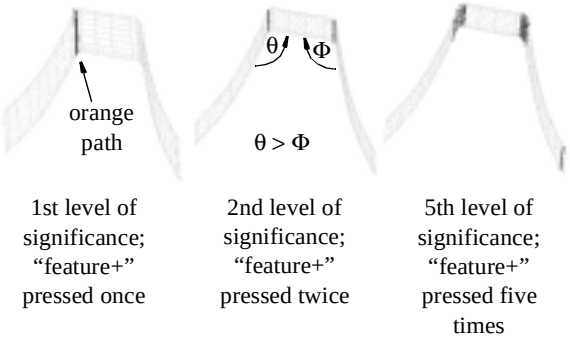
The three operations to the right of the [path+] button manage paths. The [-] operation deletes an arbitrary part of or whole path. To perform the deletion, depress the [-] button, and click on the two endpoints of the path or part of path you wish to delete. Notice that the [-] button remains depressed until you re-click it. See figure 8.5. The [<] operation acts as a

Figure 8.5



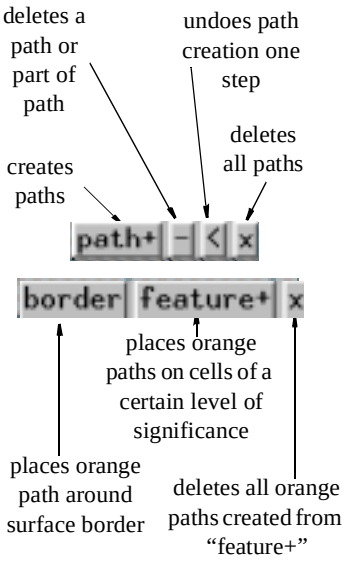
partial one-step undo. The operation will delete the path you just created, but not re-create the path or part of path you just deleted. The [x] button, when clicked, deletes all paths.

Figure 8.6



The operations in the bottom blue bar of the "Mini CAD" panel help you visualize your surface with orange paths. Orange paths are not recognized by GridPro as regular paths, and cannot be altered with the operations in the top blue bar of the Mini CAD panel operations. Essentially, orange paths are visual aids. Depressing the [border] button will place an orange path around the entire border of your surface. Un-depressing the same button will remove the orange path outline. The [feature+] operation illustrates surface cell's levels of significance. The angle between the normals of the two planes slanting off from a surface cell determine the cell's level of significance. The smaller the angle, the larger the significance level. Every time you press [feature+], all cells with the next (lower) level of significance will be highlighted with orange paths. Additionally, a white message will appear at the upper right portion of the drawing area. The angle given in the message is actually equal to the angle between the planes subtracted from 180^{o}. See figure 8.6. Notice in the figure that only one cell has the first and second level of significance. By pressing the [feature+] button enough times, all cells will be highlighted with orange paths. To delete the orange paths created by the [feature+] operation, click the [x] button to the right of the [feature+]. Figure 8.7 displays a quick summary of the operations presented in this section.

Figure 8.7

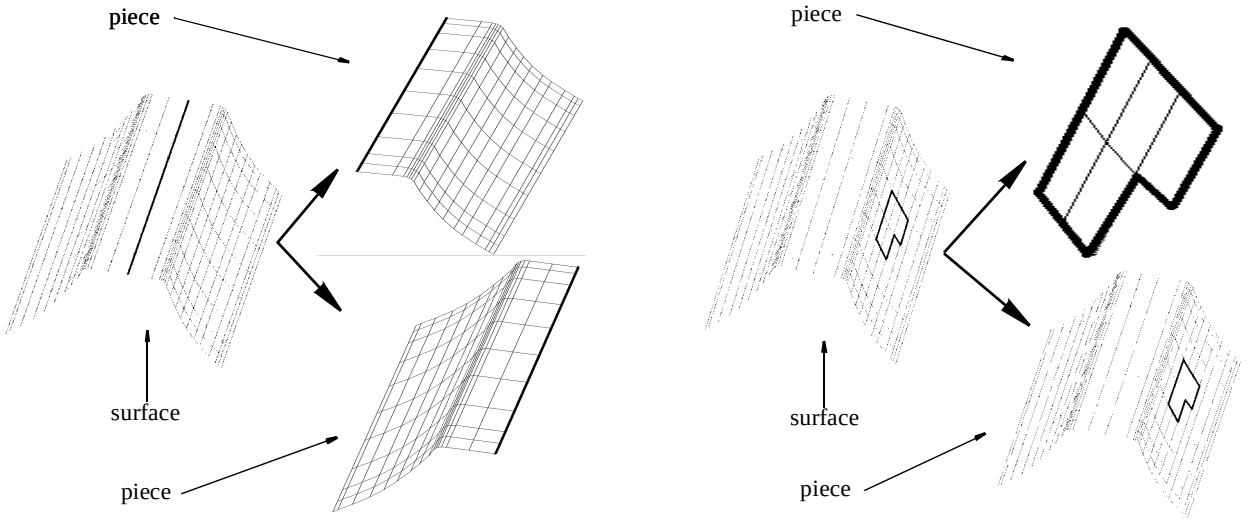


8.3 Segmenting Surfaces

When working with a complex surface, you will often want to create separate topologies for different parts of the surface. Or, you may not wish to grid the whole surface. In either case, you should segment your surface, than perform any necessary patch-up work.

In order to break up your surface, simply create a path at the segmentation points and press [segment] in the upper blue box. If your path does not completely cut the surface, the [segment] operation will fail. In other words, the endpoints of the path must intersect or lie on the border of the surface (which you can check by depressing [border]). Also, if you create at least two disconnected paths, the [segment] operation will fail. If [segment] fails, GridPro will say why in the upper left portion of the drawing area. Figure 8.8 illustrates examples of segmented surfaces.

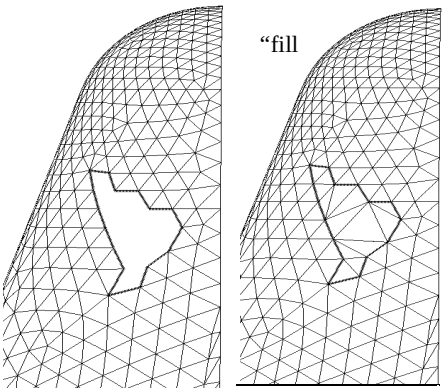
Figure 8.8



After successfully segmenting a surface, you will want to use a host of other operations. If [segment] succeeds, a message in the upper left of the drawing area will read “surf has 2 pieces,” and one such piece will become a different color (while one remains sea blue). You can continue segmenting pieces as much as desired. Like surfaces and sheets, pieces are either current or not current. The current piece is in sea blue. You can hide the current piece by clicking on the [cur pie] button in the single-row green box. To re-display the piece, re-click the [cur pie] button. You can scroll through pieces to make one current by pressing the [<] and [>] buttons found to the right of the [cur pie] button. Remember that you can scroll through hidden pieces. To save the current piece, click on the [save] button, located to the right of the scroll buttons. GridPro will automatically save the current piece as a surface file; specifically, ‘_surf.1’. Every new piece saved will have the same filename, except for the number proceeding the dot. If you want to rename the surface files, you should do so right after the file is created. Often, segmenting surfaces necessitates using many paths; saving these paths will guide you when you revisit the surface. The [sp] operation, found to the right of the [save] button, saves a path to the file name

'_path.tmp'. This file name is given to *every* path saved. Thus, if you wish to save more than one path, rename the file right after it is created.

Figure 8.9



GridPro allows you to merge segmented surfaces (pieces), as well as fill the holes created by segmentation. GridPro fills holes with -tria surfaces, even if the surface containing the hole is a -quad surface. To fill a hole in a surface or piece, make it current, then select [fill] in the single-row beige box. -Tria surfaces will automatically fill the hole. See figure 8.9. Filled holes are *not part of the surface until you press "merge,"* found to the left of the [fill] button. After pressing "merge," the surface is automatically saved. Aside from filling and merging, moving surface nodes help refine your surface. To move any surface node, click on the [mv node] button, and drag the node, with the left mouse button, to a new position. As the cursor approaches a node, the cursor will become a straight arrow:  Figure 8.10 illustrates the effect of moving a node. Moving the nodes in a filled hole may integrate it better with a surface. Figure 8.11 reviews the operations covered in this section.

Figure 8.10

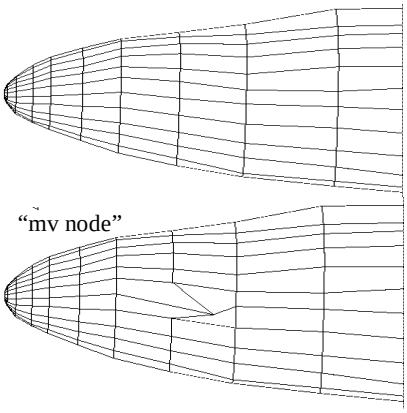
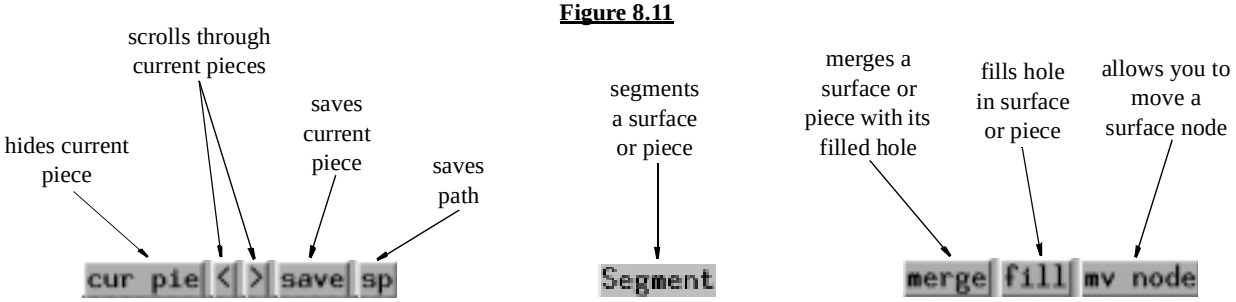
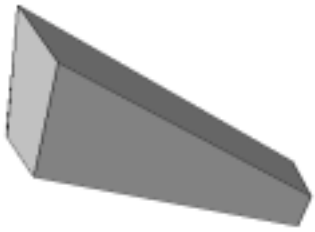


Figure 8.11



8.4 Adding Surfaces

Figure 8.12



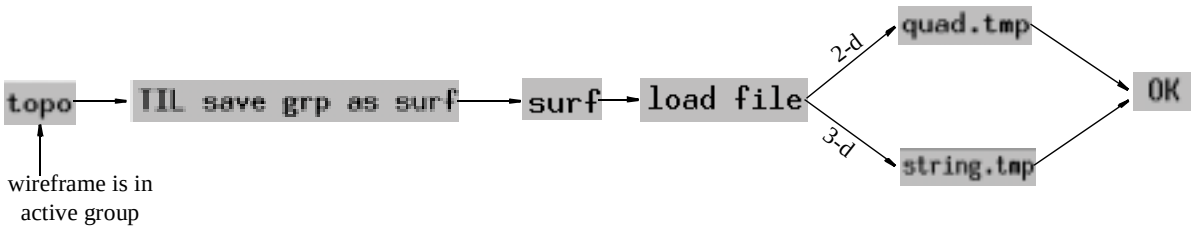
custom-made surface

You have used most of the operations in the "mini CAD" panel; now you will learn about CAD related functions found elsewhere. Often, you will want to procure an asymmetrical surface or topology, one GridPro has not built into its database. GridPro allows you to custom-make a surface based on a topology, and vice versa. One such custom-made surface is shown in figure 8.12. More specifically, GridPro can create a surface based on the form you ascribe to a wireframe. Conversely, GridPro can also create a wireframe based on the form you ascribe to a surface. To perform these transformations, you will use the "surf" and "topo" menus on the top menubar.

Creating a surface from a wireframe entails little more than setting up your wireframe correctly. Every corner will act as a vertex of the surface, and every wireframe edge will act as a surface edge. If the surface is to be two dimensional, the topology must be split up into quads. If the surface is to be three dimensional, the topology must be split up into hexes. After you construct the wireframe, you should place it in a group. Then, select the "TIL save grp as surf" operation in the "topo" menu on the top menubar. Suppose you are creating a three-dimensional surface. A small, white message in the upper-left corner of the drawing area should state, "group is written to 'quad.tmp' as a quad surface". If you are creating a two-dimensional surface, the white message will read the same except that "quad" is replaced by "string." Now, select "load file" in the "surf" menu, select "quad.tmp," and press [ok]. Your surface should appear on the screen. Of course, when dealing in two dimensions, you must select the 'string.tmp' file. Figure 8.13 illustrates the process. You should limit the number of corners converted to surface nodes to 10,000.

Figure 8.13

creating a surface from a wireframe



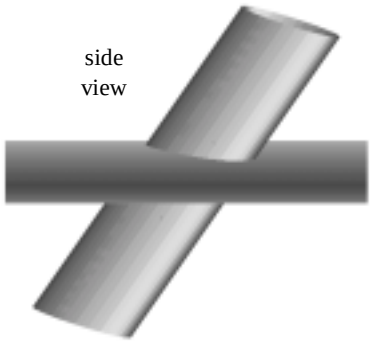
When you need to create a surface intersection, you need not visualize its shape and use the method described above. By using topology operations in the "Topology Builder" panel, you can create surfaces at intersections of other surfaces. As surfaces become increasingly complex, you can utilize similarly more complex operations that rely heavily on automation and little on user input (saving you time). Although you will be working in the "Topology Builder" panel, some of the operations described in this section do not involve the cut plane. Although you always must know the cut plane's position in order to input a corner, this section describes some operations which are exceptions to that rule. When you use these operations, GridPro enables the

position of the cursor *relative to the viewscreen* to override the position of the cursor on the cut plane.

Just as you have been inputting corners on the cut plane, you can input corners directly on surfaces. To do so, change the "corn=c" (corner on cut-p) mode on the top menubar to the "corn=s" (corner on Surf) mode. Now, by moving the cursor to any point on the *current* surface, depressing the keyboard <C> button, and clicking the left mouse button, you create a corner on that surface. Inputting corners in such a manner allows you to create a wireframe that fits well the contours of the region to be gridded, or to input a corner in hard to reach surface junctions.

EXERCISE #19
Redo exercise #14 in the "corn=s" mode (do not use the cut plane).

Figure 8.14



Often you will deal with multiple surfaces. Sometimes, these surfaces overlap, and you will need to create surfaces at the intersections to optimize the grid quality there. To create such a surface, you should first create strings of corners that fit the contours of your surface. GridPro can *automatically* construct these strings of corners. By pressing the [i0] button in the brown, "TOPO" box, then moving the cursor toward an intersection and clicking the left mouse button (in the "corn=s" mode), you automatically create a string of corners (all connected by edges) "hugging" the intersection. Every corner in this string is *automatically* assigned to each of the intersecting surfaces. Also, there are *no* restrictions on the type of intersections you can contour with corners. Figure 8.14 displayes a smaller, skewed cylinder intersecting a larger cylinder at a non-orthogonal angle. Figure 8.15 shows a top view, and figure 8.16 illustrates the simple process of creating corners at the intersection (the white square surface assignment indicators are not shown for clarity).

Figure 8.15

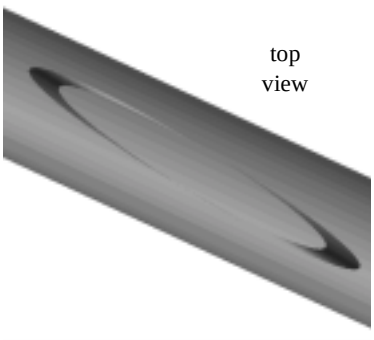
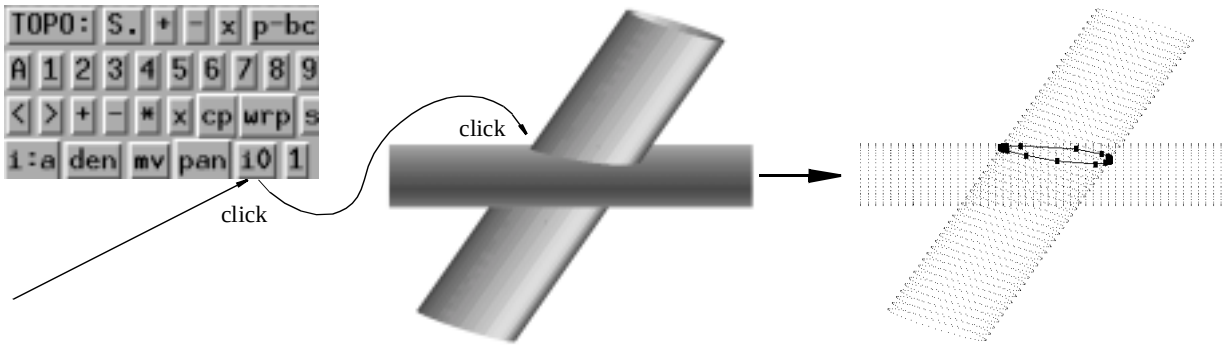


Figure 8.16



After using the [i0] button to create a string of corners around an intersection, you can use this topology string to create a detailed contour of the intersecting surface. By pressing the [1] button, found to the right of the [i0] button (not the group [1] button), you instruct GridPro to construct a surface contour based just on *the one string of topology.* GridPro computes the best fit contour for the intersection, often a cup-shaped intersection, and creates corners and links all composing the contour. If you do not use the [1] operation right after the [i0] operation, make sure whatever group containing the intersecting string is active before using the [1] operation. Figure 8.17 displays the result of the operation.

Figure 8.17

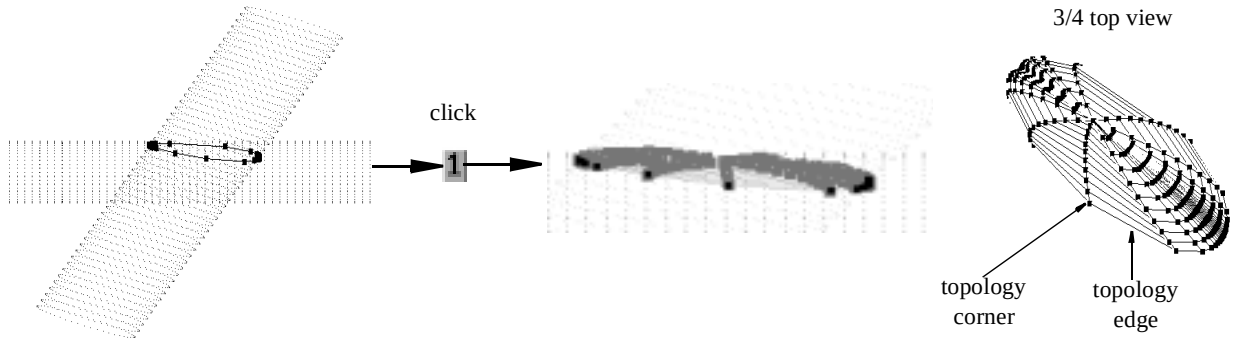
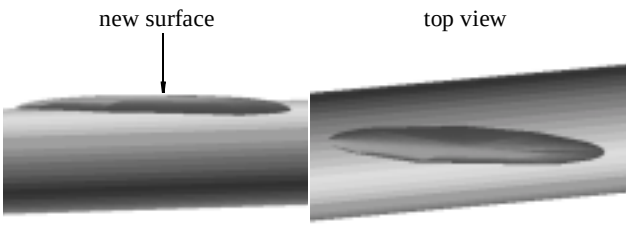


Figure 8.18



Now, you can make the contour into a surface. Following the process outlined in figure 8.13 to transform any wireframe into a surface, you come up with the desired surface intersection "cap" displayed in figure 8.18.

Typically, in the course of your work, you will encounter surfaces that appear, wholly or partly, *inside* the region to be gridded. As we mentioned in the last chapter, these surfaces are called internal surfaces. Remember, the orientation of internal surfaces must be double sided because the region on both sides of the internal surface will be gridded. To activate this orientation for a surface, first bring up the properties menu for the surface (make the surface current, and go to "reload current" in the "surf" menu). Then, select "2 sides" in the "orient" box. Most surfaces created using the [1] operation are internal surfaces.

You could benefit from internal surfaces even if there is no real internal surface in the region you wish to grid. Depending on your topology, sometimes changing the density around a sharp corner (or a hub, for example) does not affect the final grid. In these situations, the increased density does produce more mesh lines. However, these mesh lines are mathematically "shoved" to another part of the grid. To remedy this situation, place an internal surface in the subregion you would like refined. First, create a wireframe about that subregion. Then, by you assigning the appropriate corners to the internal surface, those corners are *forced* to create mesh lines about that surface (surface assignment overrides the mathematical "push" those corners are

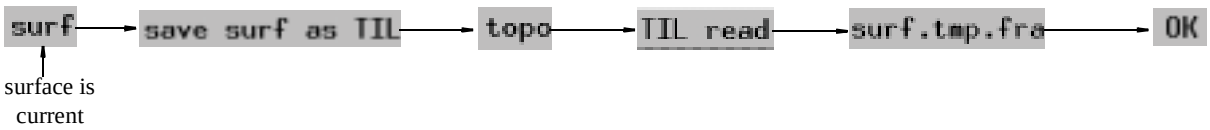
given). Now, you can change the density of the edges around the internal surface to the desired value.

8.5 Refining Surfaces

You have already learned to move surface nodes manually with the [mv node] operation; another way to refine surfaces is to transform them *into wireframes*, edit the wireframes, and transform the wireframes back into surfaces. Creating a wireframe from a surface is similar to creating a surface from a wireframe. However, the surface must be a quad surface. To convert it into a wireframe, first make the surface current. Then, select the "save surf as TIL" operation in the "surf" menu. A white message should appear on the upper left of the drawing area indicating that the surface is saved to the 'surf.tmp.fra' file. Now, select the "TIL read" operation in the "topo" menu. Click on 'surf.tmp.fra' and press [ok]. The wireframe should appear in the drawing area. Figure 8.19 illustrates the process.

Figure 8.19

creating a wireframe from a surface



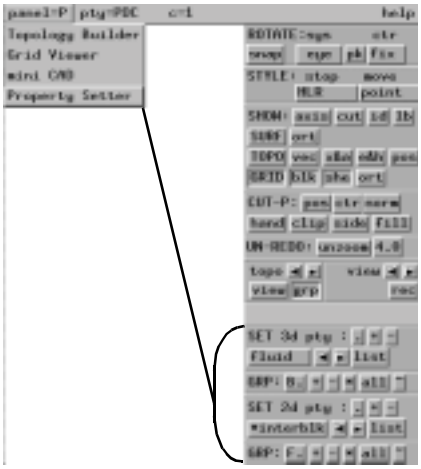
Note that if you wish to save more than one custom-made surface or topology, you must change their file names right after they have been given the default name.

Chapter 9 - Setting Grid Properties for Solvers

9.1 Overview

Before connecting grids to CFD solvers, you may want to set properties on these grids. GridPro enables you to set two dimensional and three dimensional properties. That is, you can assign properties to each face of each grid block, and to the volume of each grid block itself. You set properties in the "Property Setter" (panel=P) panel. Figure 9.1 illustrates the operations associated with the "Property Setter" panel. The default style in this mode is Hidden Line Removal. Also, note that the two green boxes ("SET 2d pty" and "GRP") deal with faces (in 2-d) and that the two beige boxes ("SET 3d pty" and "GRP") deal with blocks (in 3-d).

Figure 9.1



The "Property Setter" panel makes extensive use of skeletons for the purpose of efficient property assignment. Recall that a skeleton of a block is the lines connecting the indexed center of the block with the centers of the block's bounding edges (2-d) or faces (3-d). Each block skeleton is drawn in its own color, and represents the minimal amount of information required to show a broad representation of a multiblock grid.

Additionally, the "Property Setter" panel contains operations similar to those in the "Grid Viewer" panel. As you assign properties to blocks and faces, remember which components are activated, and which are hidden. You should be cognizant of what each operations does as well as what each operation displays on the drawing area.

9.2 Two-Dimensional Properties

This section focuses on assigning properties to faces of blocks. Remember, a property can be assigned to *each* face of *each* block, even if the faces intersect. Before learning how to assign surface properties, focus on the GRP green box, the one displayed in figure 9.2 and 9.3. This block contains all operations relevant to the activation and display of the *skeletons* of faces. If the center of a face skeleton is deactivated, it will be hidden. Conversely, an activated face skeleton center will cause the face to be displayed.

Figure 9.2
ACTIVATION METHODS

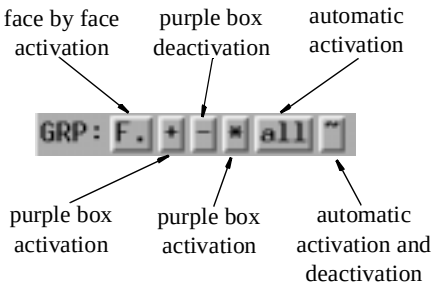
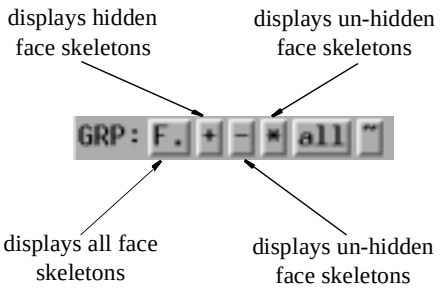
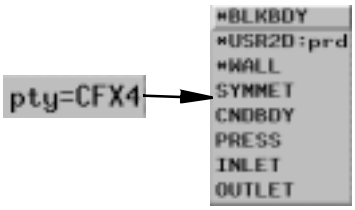


Figure 9.3
DISPLAY MECHANISMS



If you have read through and remember the latter part of Chapter 4 and all of Chapter 6, you do not need to read the following paragraph. The buttons in the "GRP" green box describe operations identical to those in the "Grid Viewer" panel. Depressing the [F.] button will: i) display the skeletons of all faces, ii) allow you to activate or deactivate one face by clicking (with the left mouse button) on the center of the face's skeleton. Depressing the [+] button will: i) display the skeletons of all hidden faces, ii) allow you to activate many faces by dragging a purple box (with the right mouse button) over the centers of these faces' skeletons. Depressing the [-] button will: i) display the skeletons of all displayed (un-hidden) faces, ii) allow you to deactivate many faces by dragging a purple box (with the right mouse button) over the centers of these faces' skeletons. Depressing the [*] button will: i) display the skeletons of all displayed (un-hidden) faces, ii) allow you to keep active only those faces whose skeleton centers you encompass in a purple box. Remember that the [*] buttons are "intersection" buttons; this particular [*] button allows you keep active the faces of whose skeleton centers are in the intersection of your purple box and the already active faces. Pressing the [all] button will activate every face, and the [~] button will switch the current activation status of each face. Of course, depressing one of the buttons in this box will un-depress all the others.

Figure 9.4



After determining what faces are activated, you can assign grid properties to them. First, you must choose a CFD solver format. Selecting "pty=PDC" from the top menubar will open a submenu containing CFD formats. Aside from the CFX4, GASP, and FIDAP formats, GridPro contains a default format, PDC (Program Development Corporation). Each format has unique properties. If you press the [list] button in the "SET 2d pty" box, a list of these properties will CFX4 format. Figure 9.4 provides an example of what properties under the [list] button are associated with a format. Appendix D describes how to incorporate user-defined formats.

Figure 9.5
ASSIGNMENT METHODS

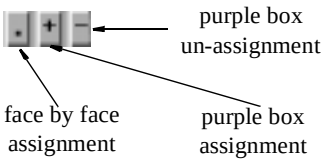
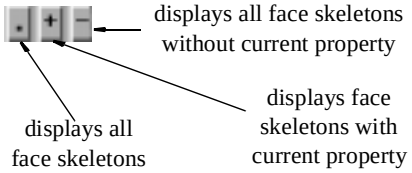


Figure 9.6
DISPLAY MECHANISMS



All operations in the green, "SET 2d pty" box deal with property assignment. When you select a property in the [list] button menu, that property becomes current. The three buttons in the top half of the box (the [.], [+], and [-] buttons) assign or "un-assign" the current property to faces. Assigning or un-assigning the cur-

rent property to a face involves the center of that face's skeleton. *Assigning a property to a face does not alter the shape of that face in any way.* Depressing the [.] button will: i) display the skeletons of all faces ii) allow you to assign the current property to any one face by clicking (with the left mouse button) on the center of the face's skeleton. By repeatedly clicking on the skeleton center with the [.] button depressed, you toggle between assigning the face to the current property and the previous one. Depressing the [+] button will: i) display the skeletons of all faces which do not contain the current property ii) allow you to assign the current property to many faces by dragging a purple box (with the right mouse button) over their skeletons' centers. Depressing the [-] button will: i) display the skeletons of all faces which contain the current property ii) allow you to un-assign the current property to many faces by dragging a purple box (with the right mouse button) over their skeletons' centers. When one button is depressed, the others un-depress. Figures 9.5 and 9.6 summarize how these operations function.

The other operations in the "SET 2d pty" box deal with scrolling through current properties. When depressed, the button containing the name of the current property will display only the faces that have the current property. When un-depressed, this button will hide the faces that contain the current property. GridPro remembers whether or not the button is depressed for each property when it becomes current. The [<] and [>] buttons will scroll though the current properties. For the faces you hid, the button containing the name of the property of those faces will remain un-depressed as you scroll though it.

9.3 Three-Dimensional Properties

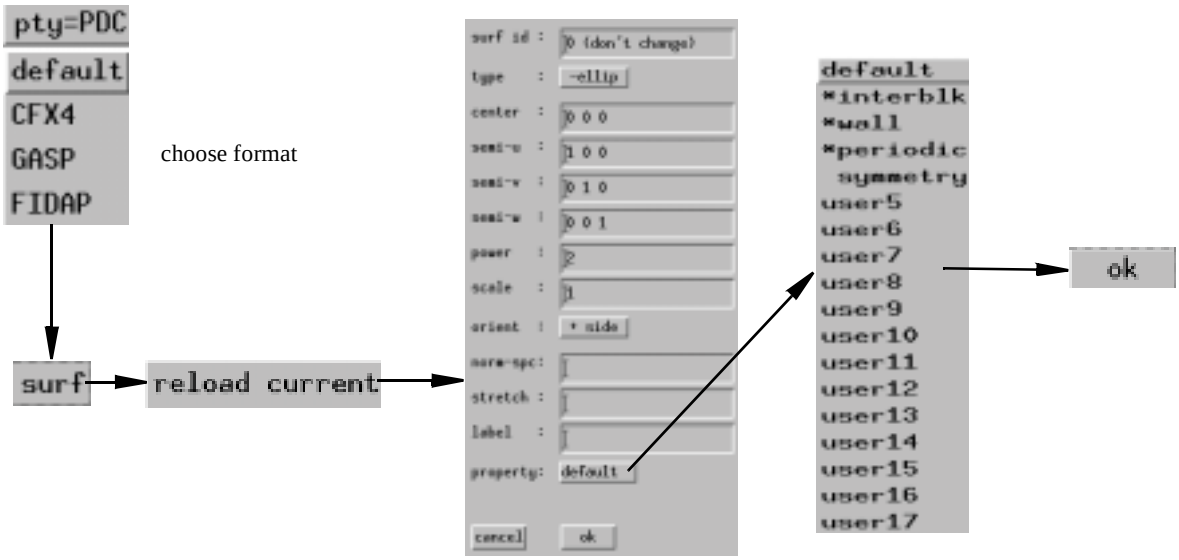
The process of assigning properties to blocks is identical to that of assigning properties to faces. Here, you must use the "SET 3d pty" and "GRP" beige boxes just as you used the "SET 2d pty" and "GRP" green boxes. You assign properties to blocks by clicking on their skeletons' centers, and hide and display blocks with the operations ([B.], [+], [-], [*], [all], and [~] buttons) in the "GRP" box. The property format you choose determines what 3-d properties are displayed when you press the [list] button in the brown "SET 3d pty" box. A format's 2-d properties and 3-d properties are independent of one another.

9.4 Inserting Properties on Surfaces

Surfaces bound the region to be gridded. In fact, the block faces of grids exist in the same space as the surface nodes which created those grids. Therefore, you need not wait to produce a grid before assigning properties to it; you can assign properties to the surfaces. Then, when a grid is produced, the properties you assigned to the surfaces will automatically be assigned to the corresponding parts of the grid. Because surfaces ultimately specify block faces, and not blocks, you can only assign 2-d properties to surfaces.

To assign a property to a surface, first choose the format that contains this property. Remember, you can choose formats from the “pty=” menu on the top toolbar. Then, access the surface property box (by either creating a new surface or reloading the current one). On the bottom of this box, click on the button to the right of the “property” heading. A menu will appear containing the 2-d properties of the format you chose. After selecting a property, press the [ok] button to apply you assignment. Figure 9.7 outlines the process.

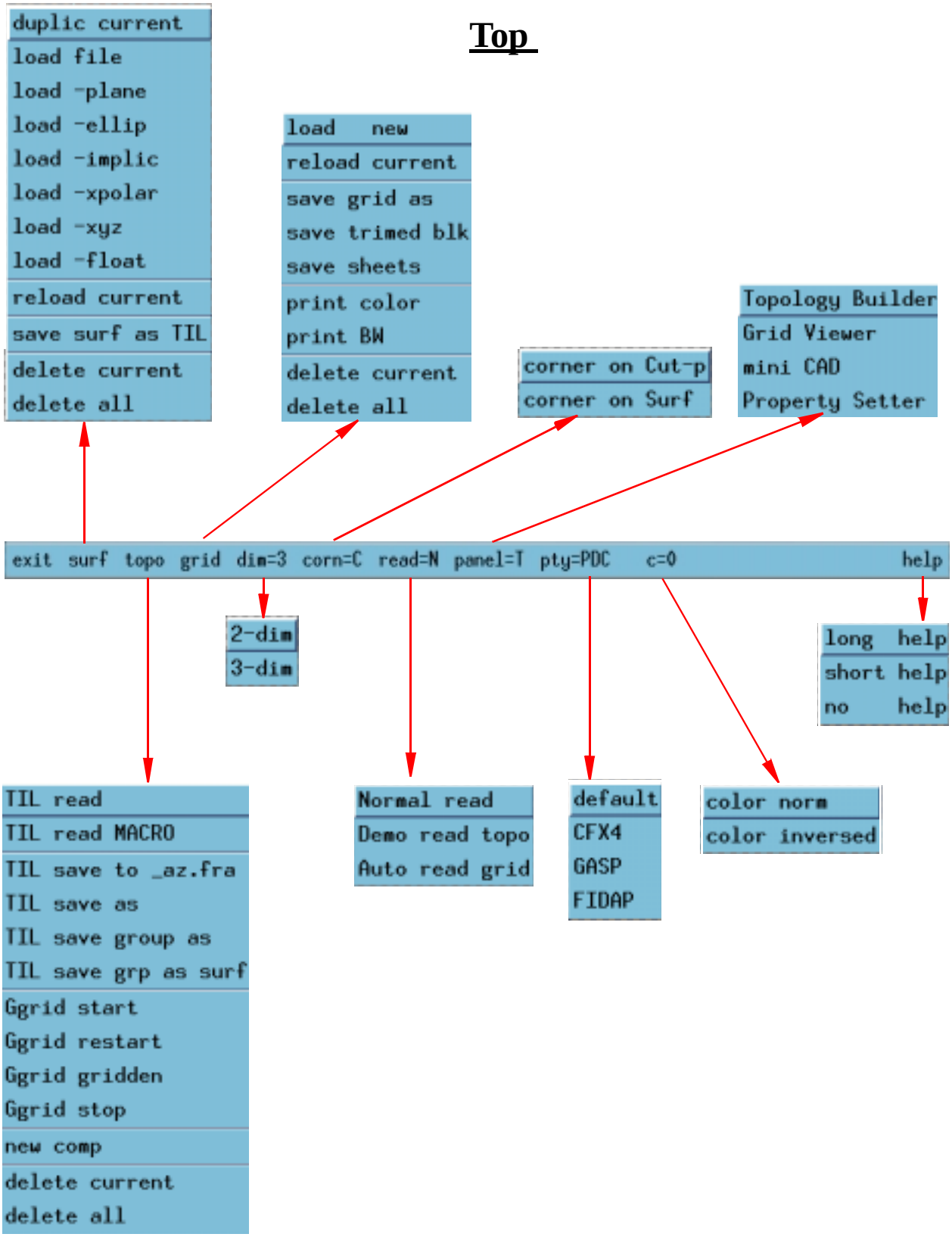
Figure 9.7



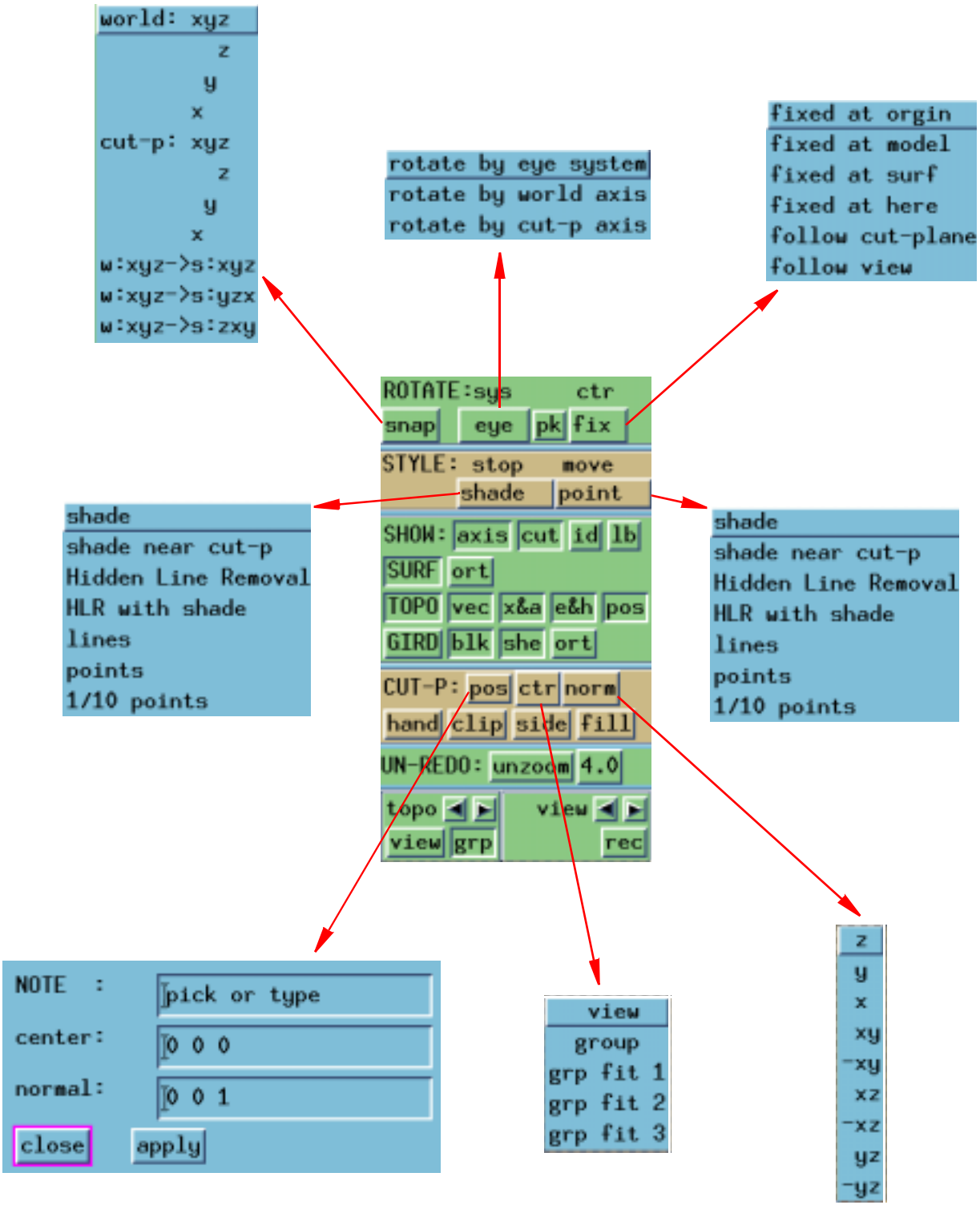
PART 3
APPENDICIES

Appendix A - Menu Navigation

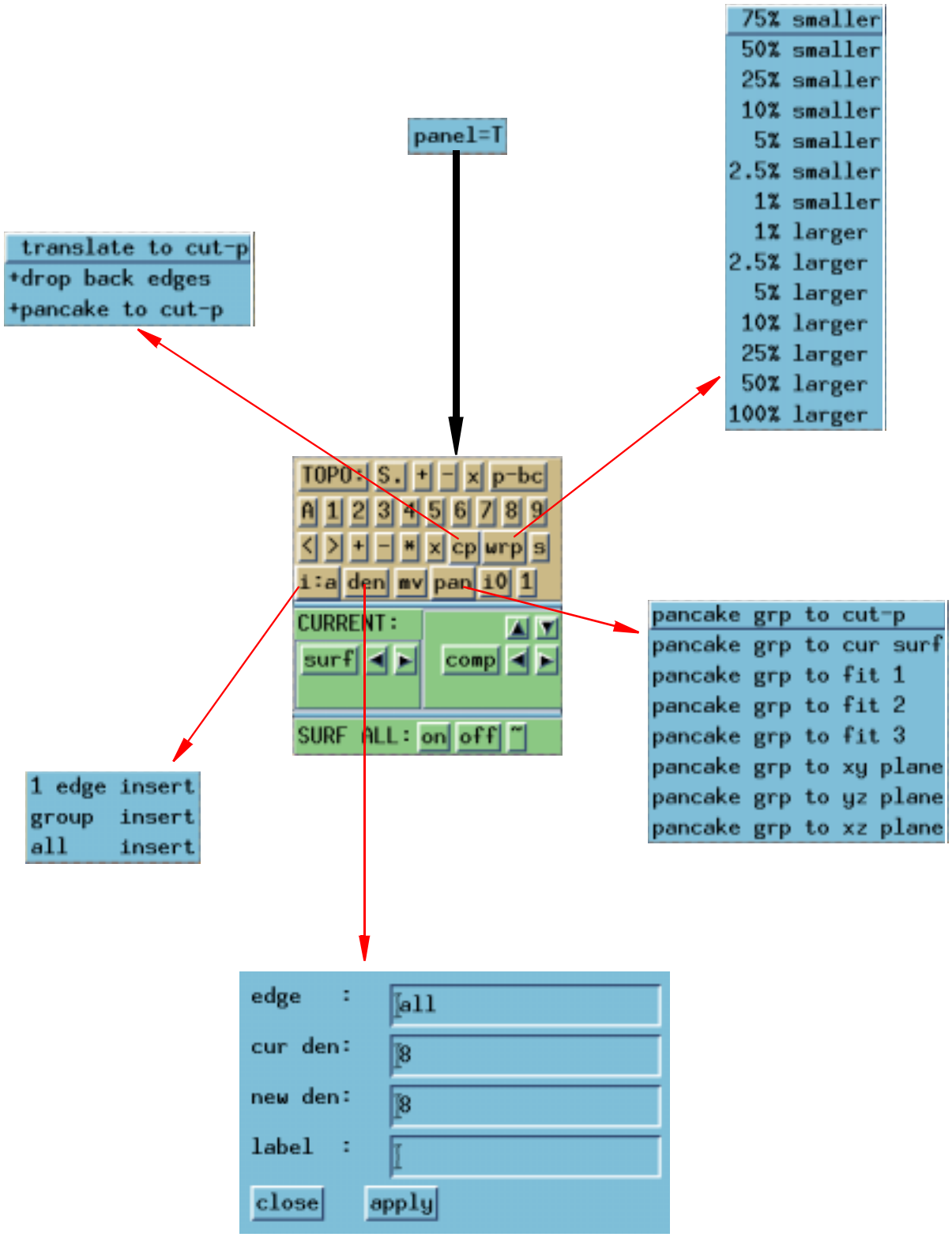
Top



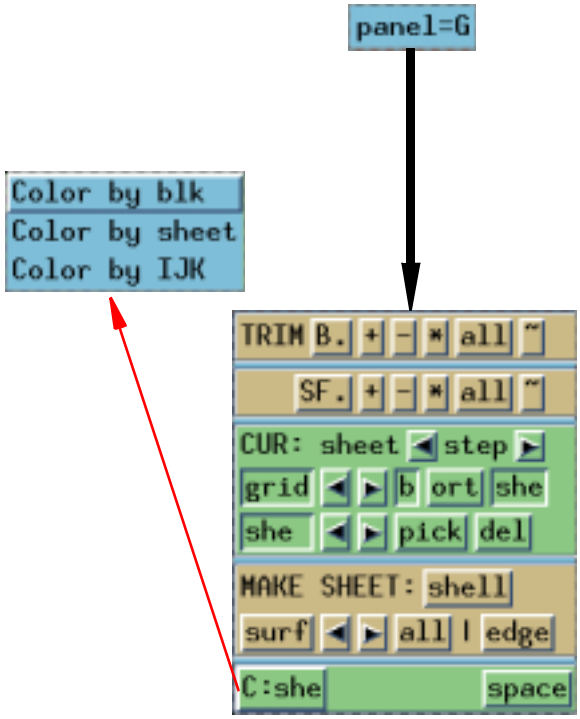
Fixed Operations



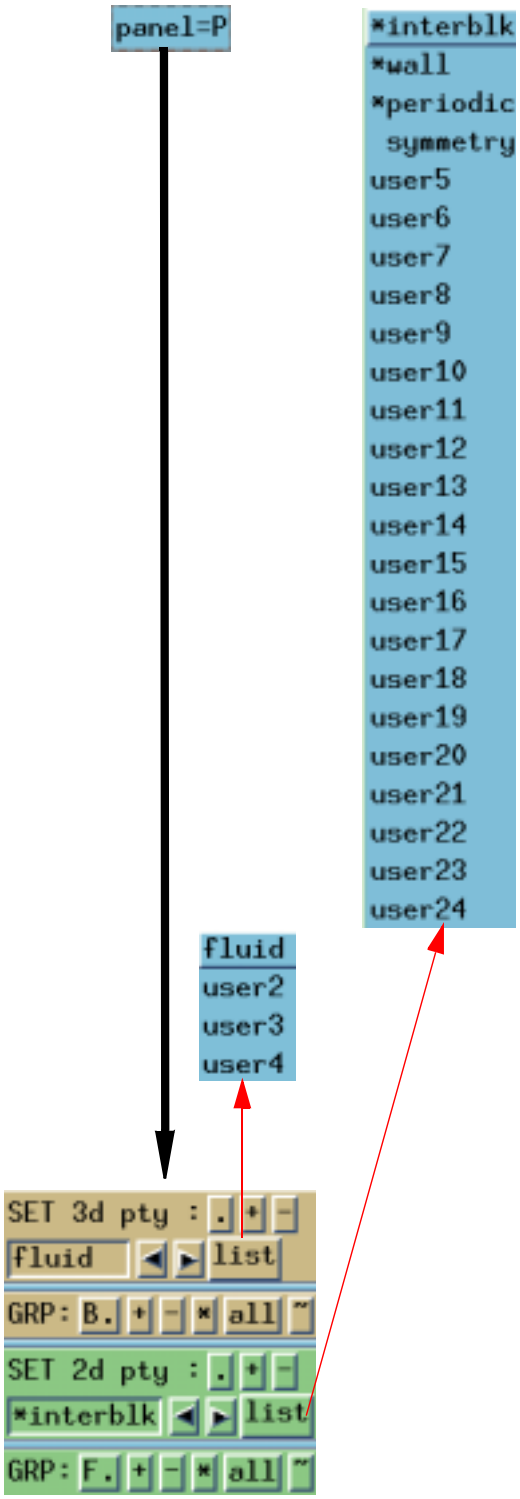
Panel T (Topology Builder) Operations



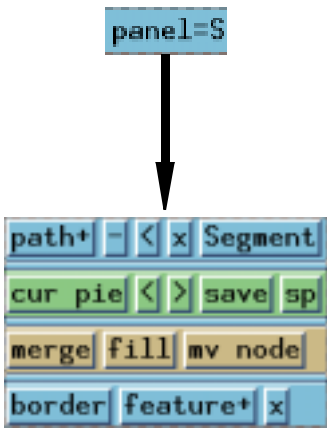
Panel G (Grid Viewer) Operations



Panel P (Property Setter) Operations



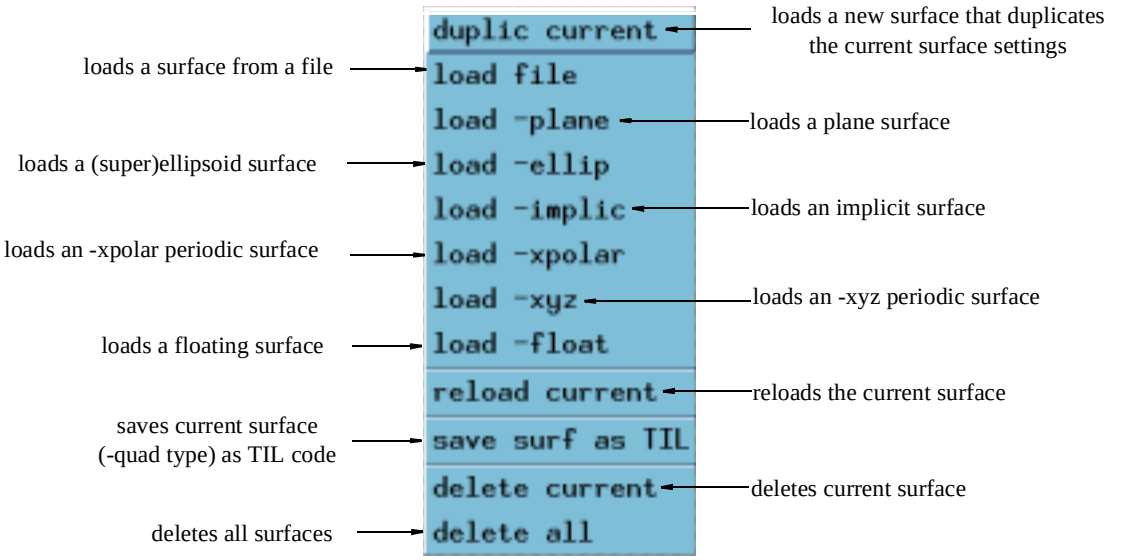
Panel S (Mini CAD) Operations



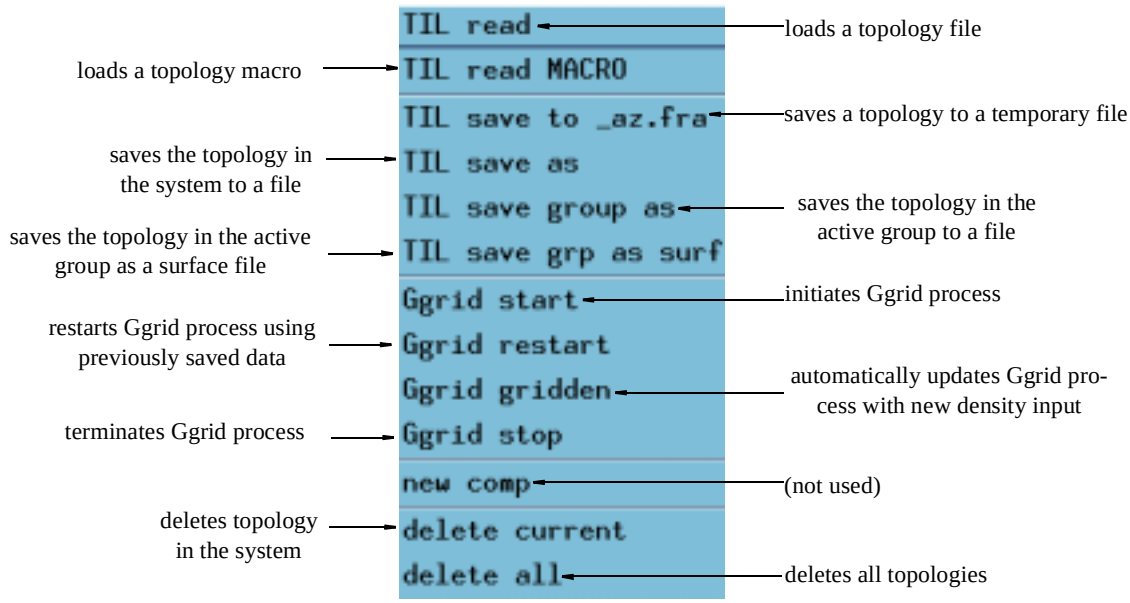
Appendix B - Summary of Operations

Top

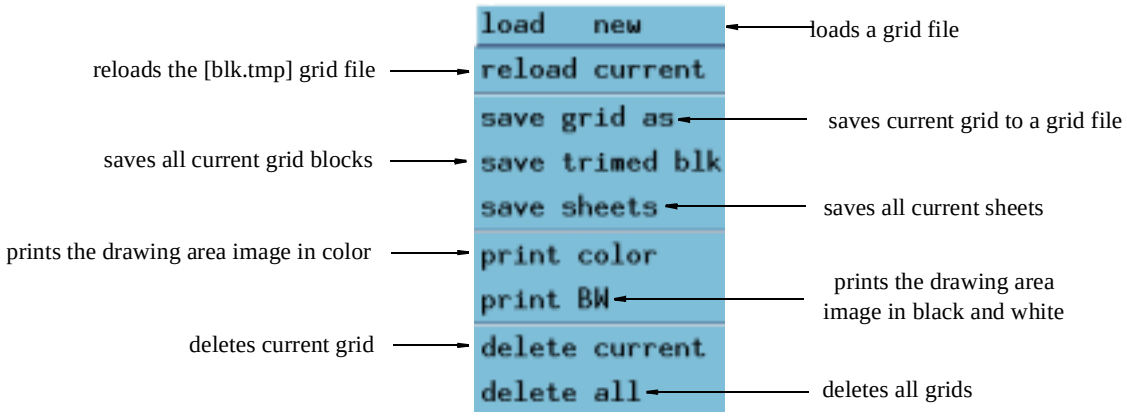
surf menu



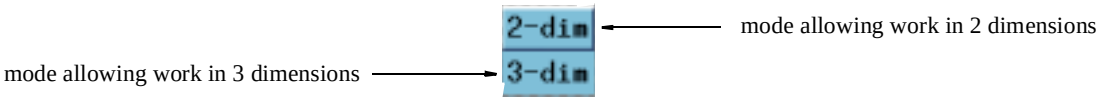
topo menu



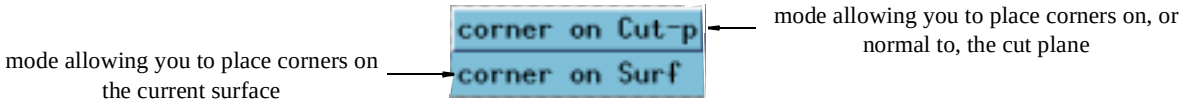
grid menu



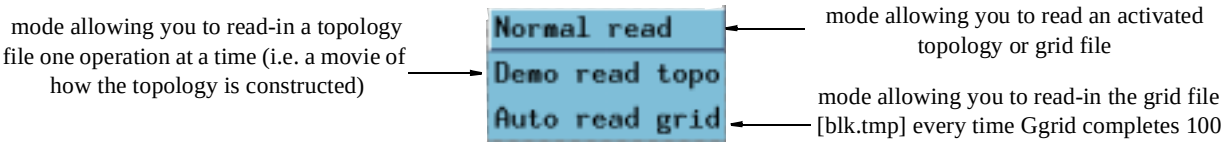
dimension toggle



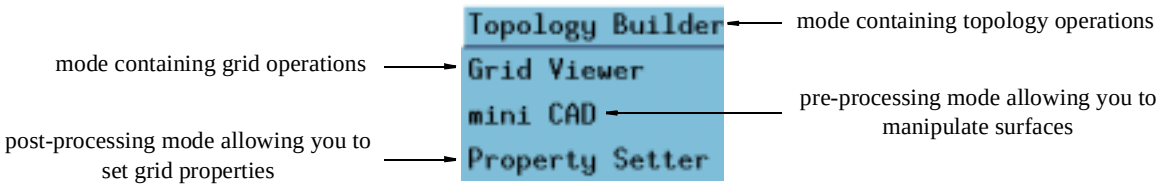
corner toggle



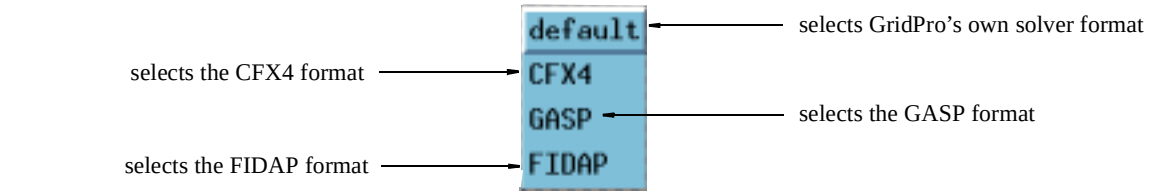
read-in menu



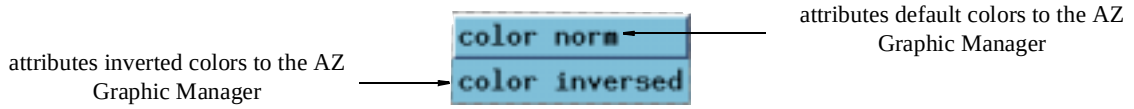
panel switcher



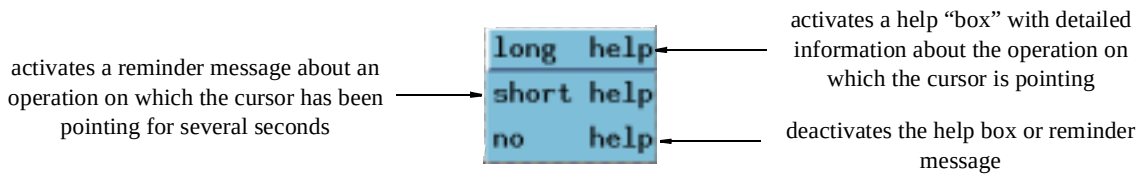
property menu



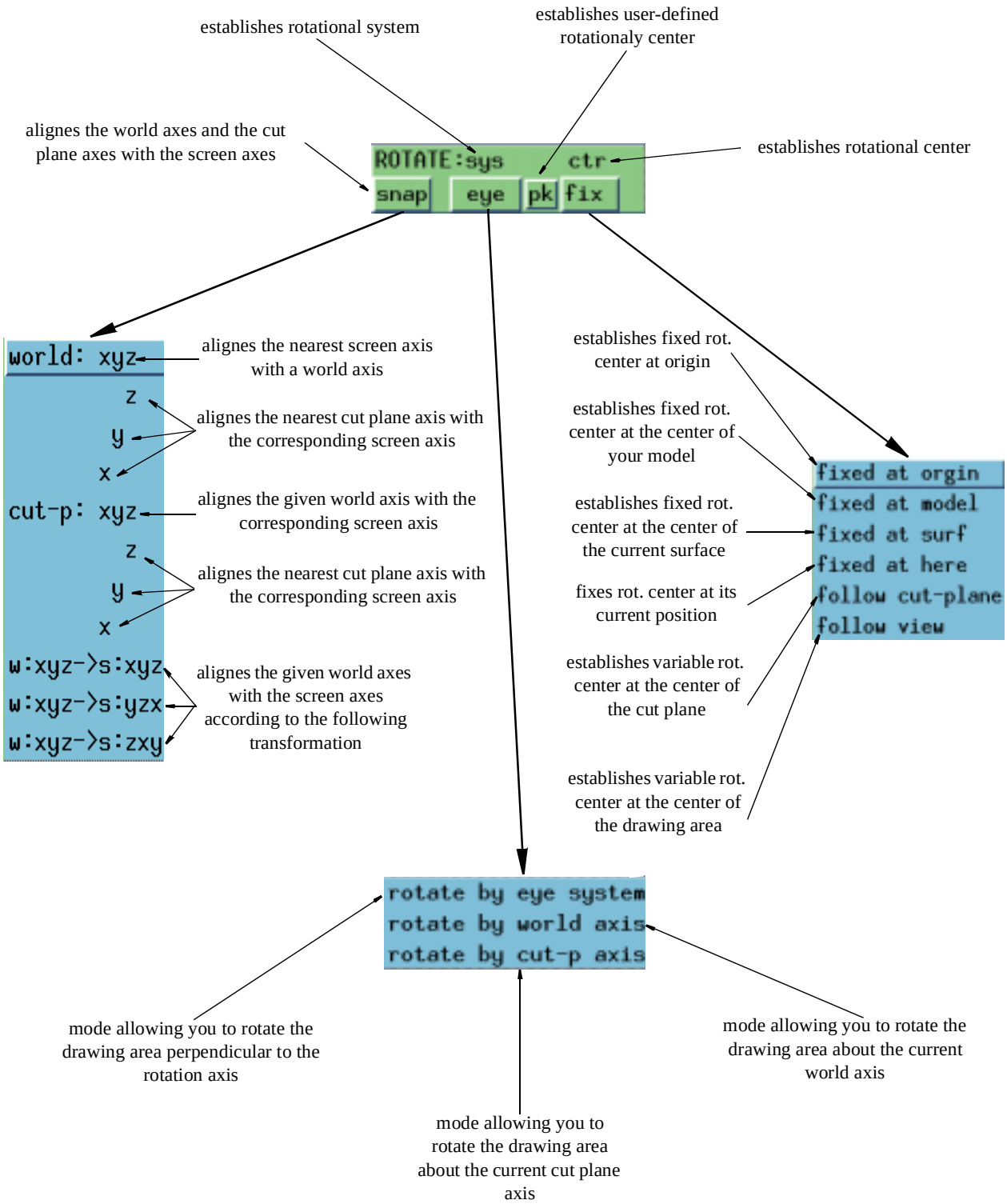
color changer

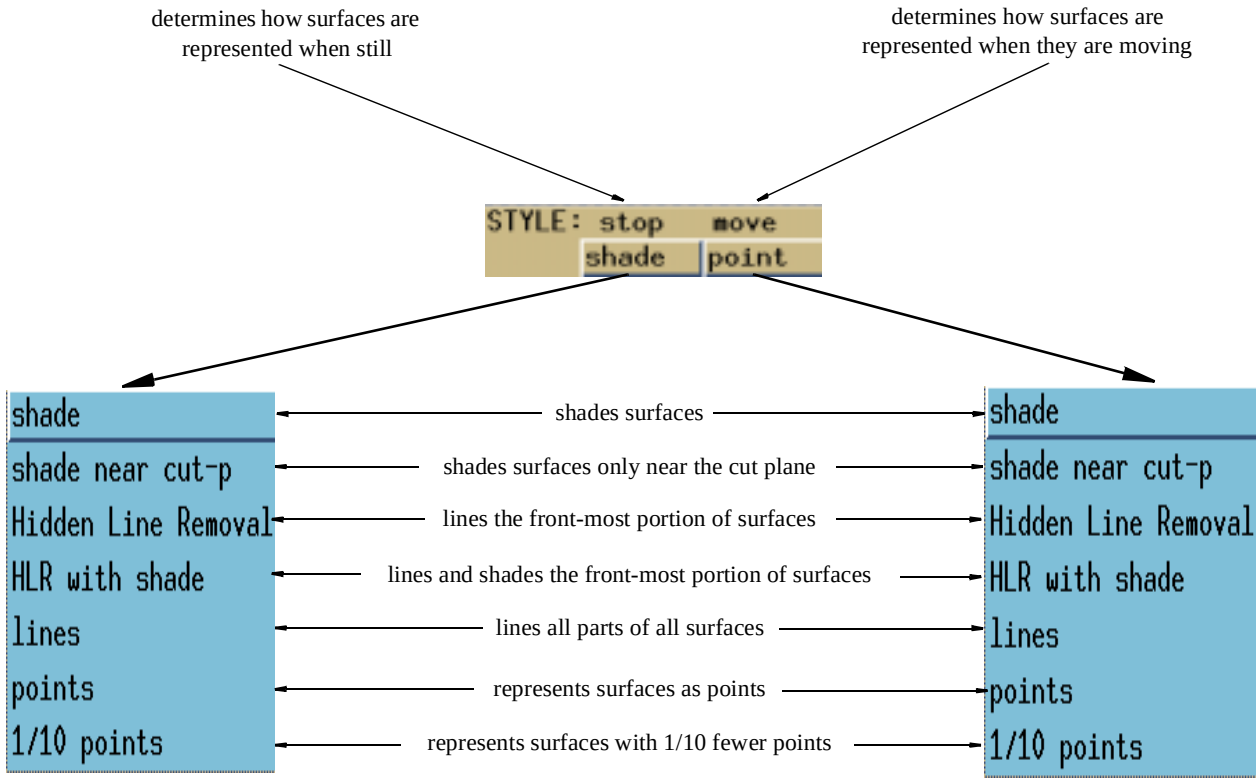


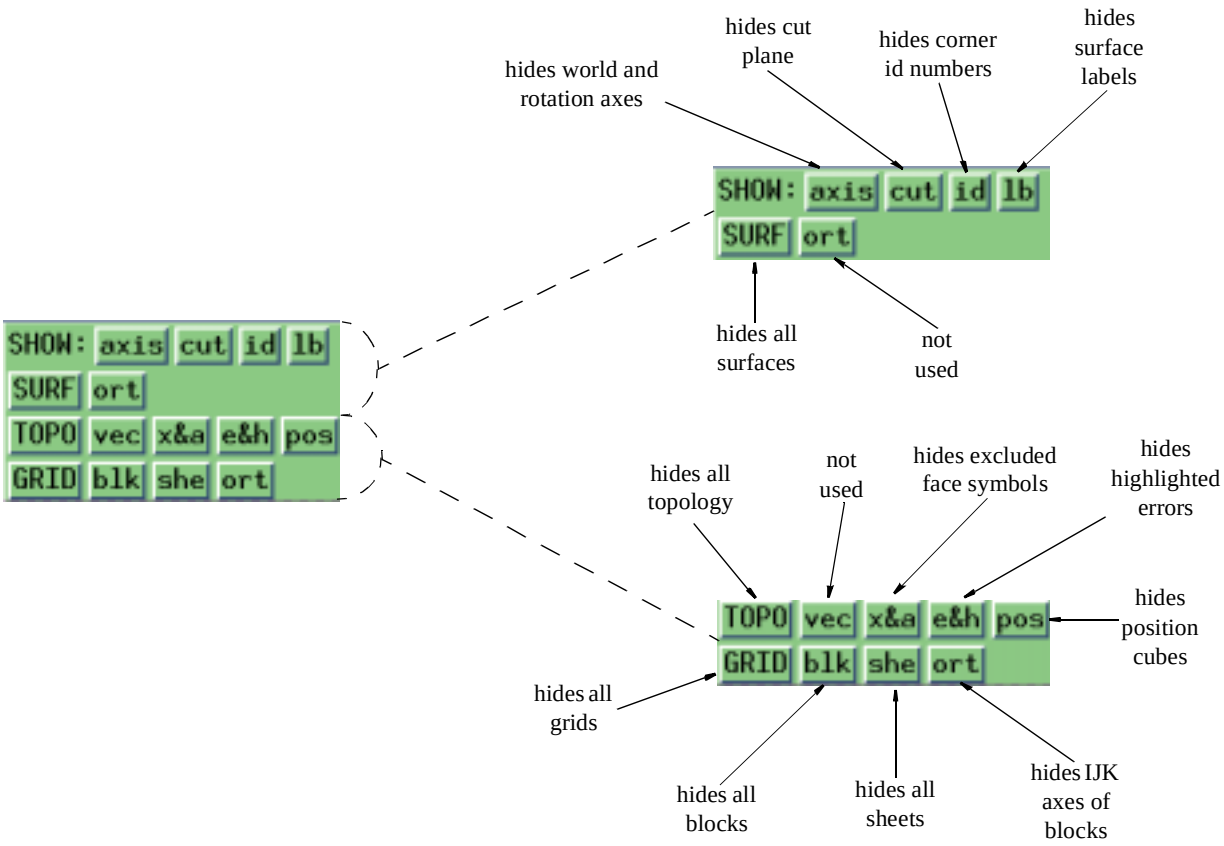
help menu

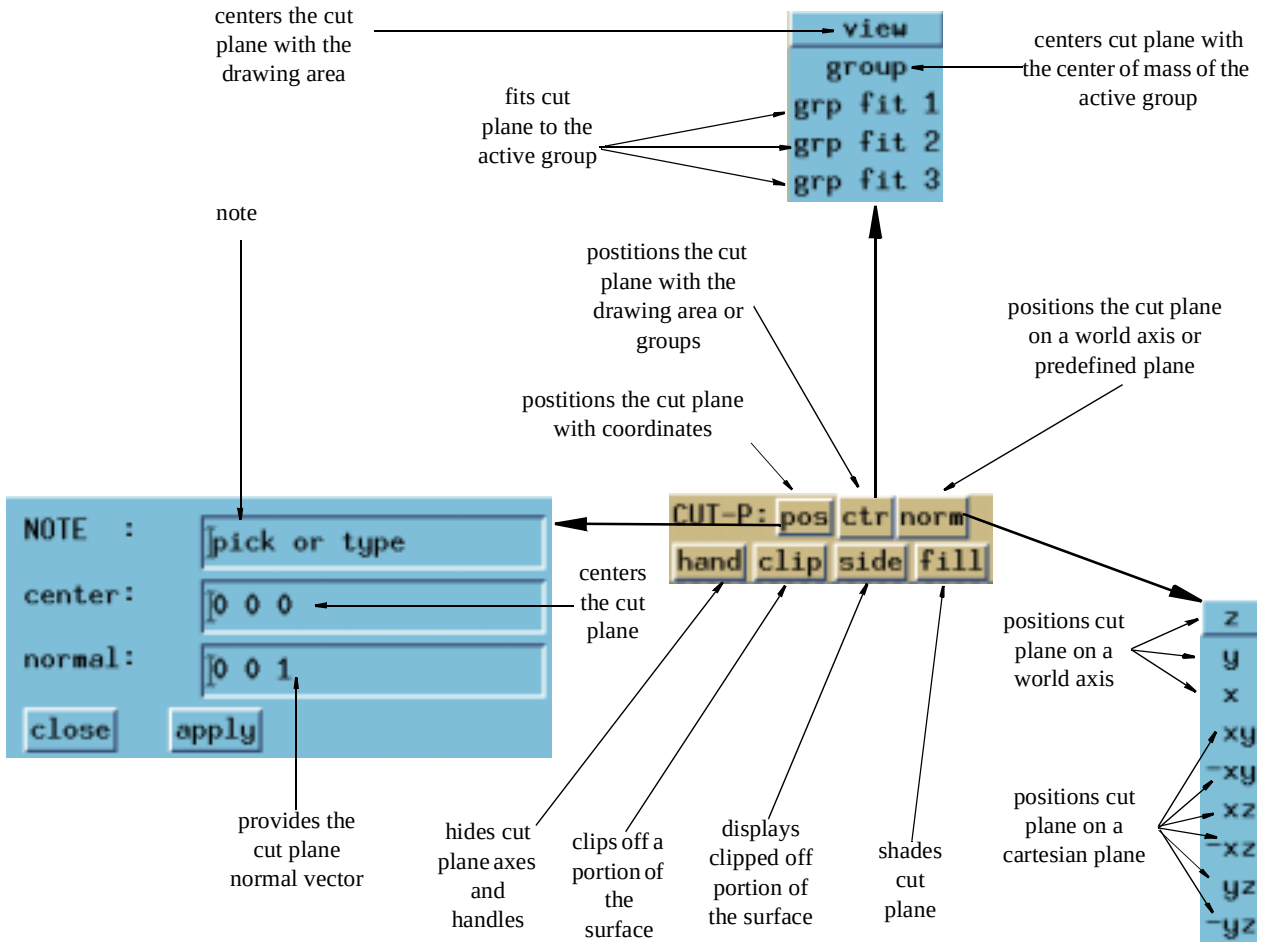


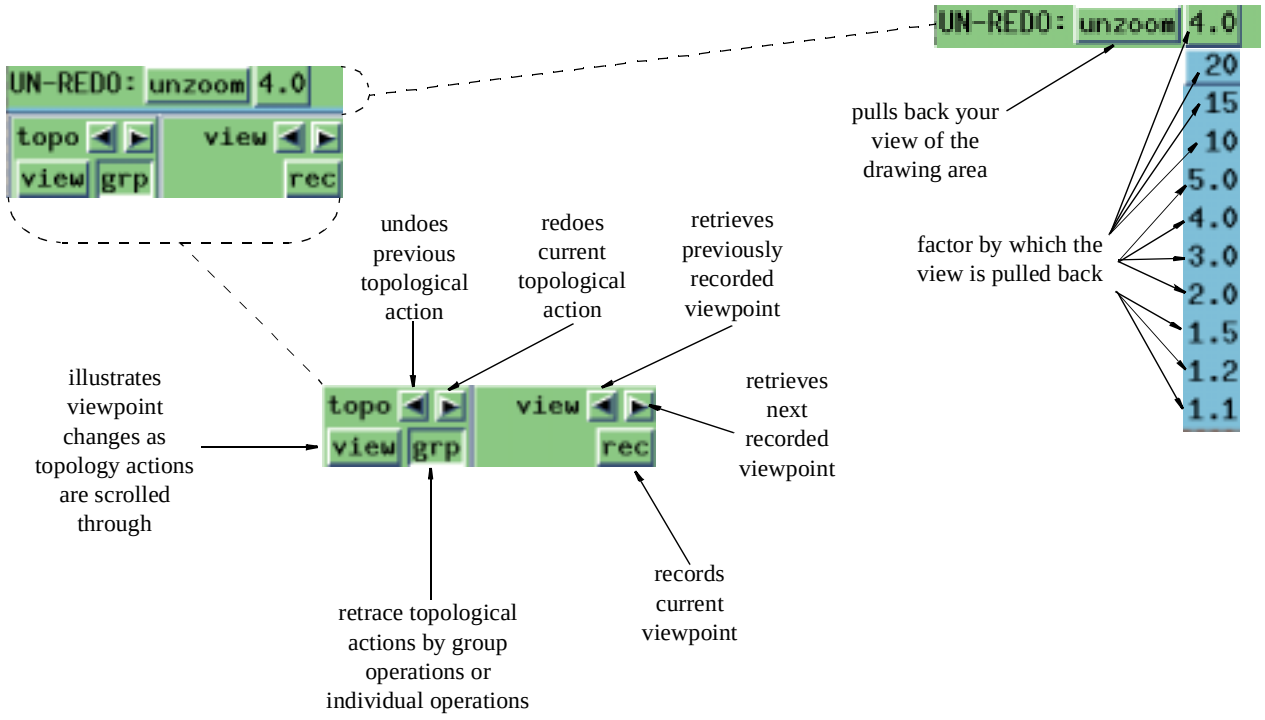
Fixed Operations



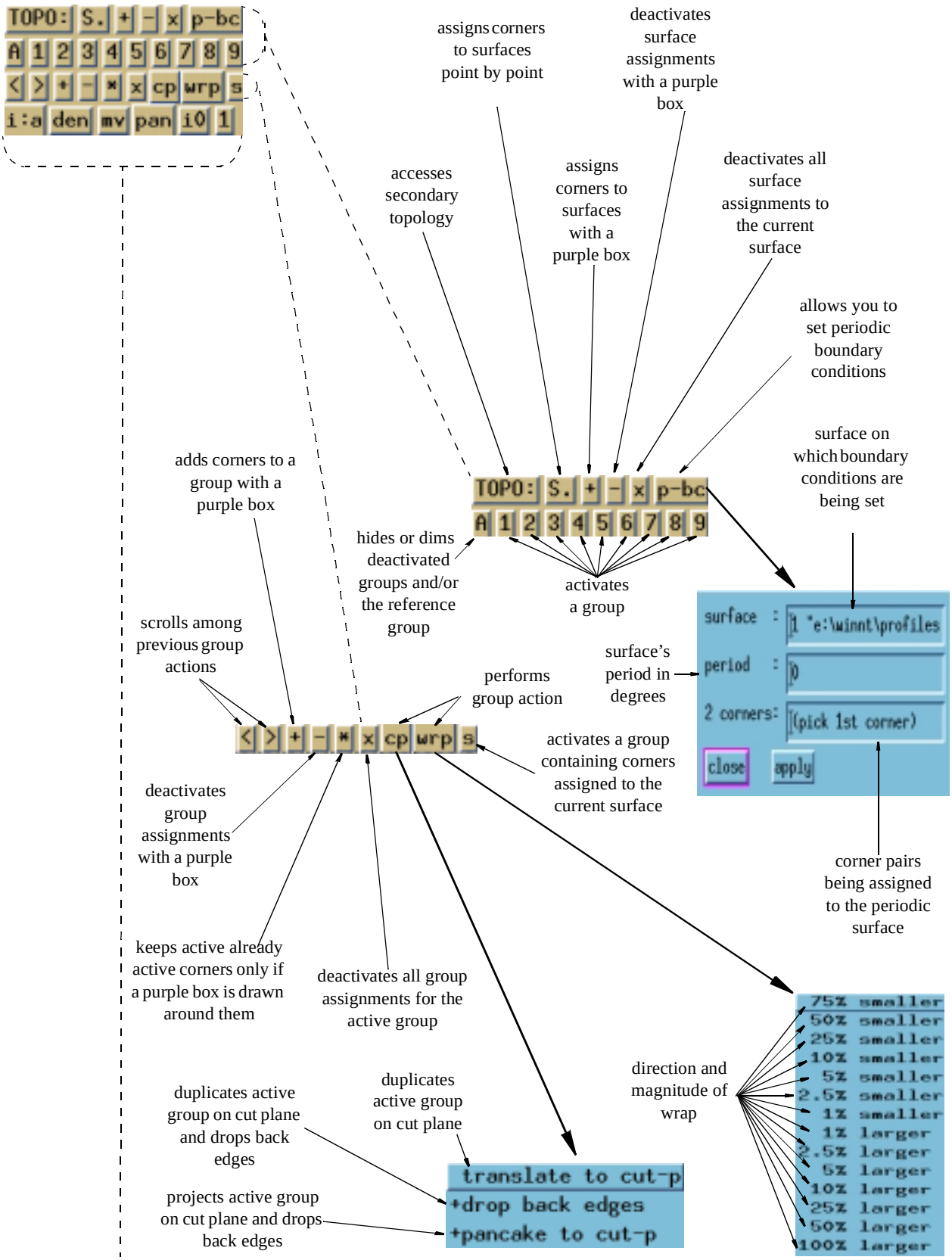

determines how surfaces are represented when still
determines how surfaces are represented when they are moving
STYLE: stop move
shade point
shade
shade near cut-p
Hidden Line Removal
HLR with shade
lines
points
1/10 points
shades surfaces
shades surfaces only near the cut plane
lines the front-most portion of surfaces
lines and shades the front-most portion of surfaces
lines all parts of all surfaces
represents surfaces as points
represents surfaces with 1/10 fewer points
shade
shade near cut-p
Hidden Line Removal
HLR with shade
lines
points
1/10 points

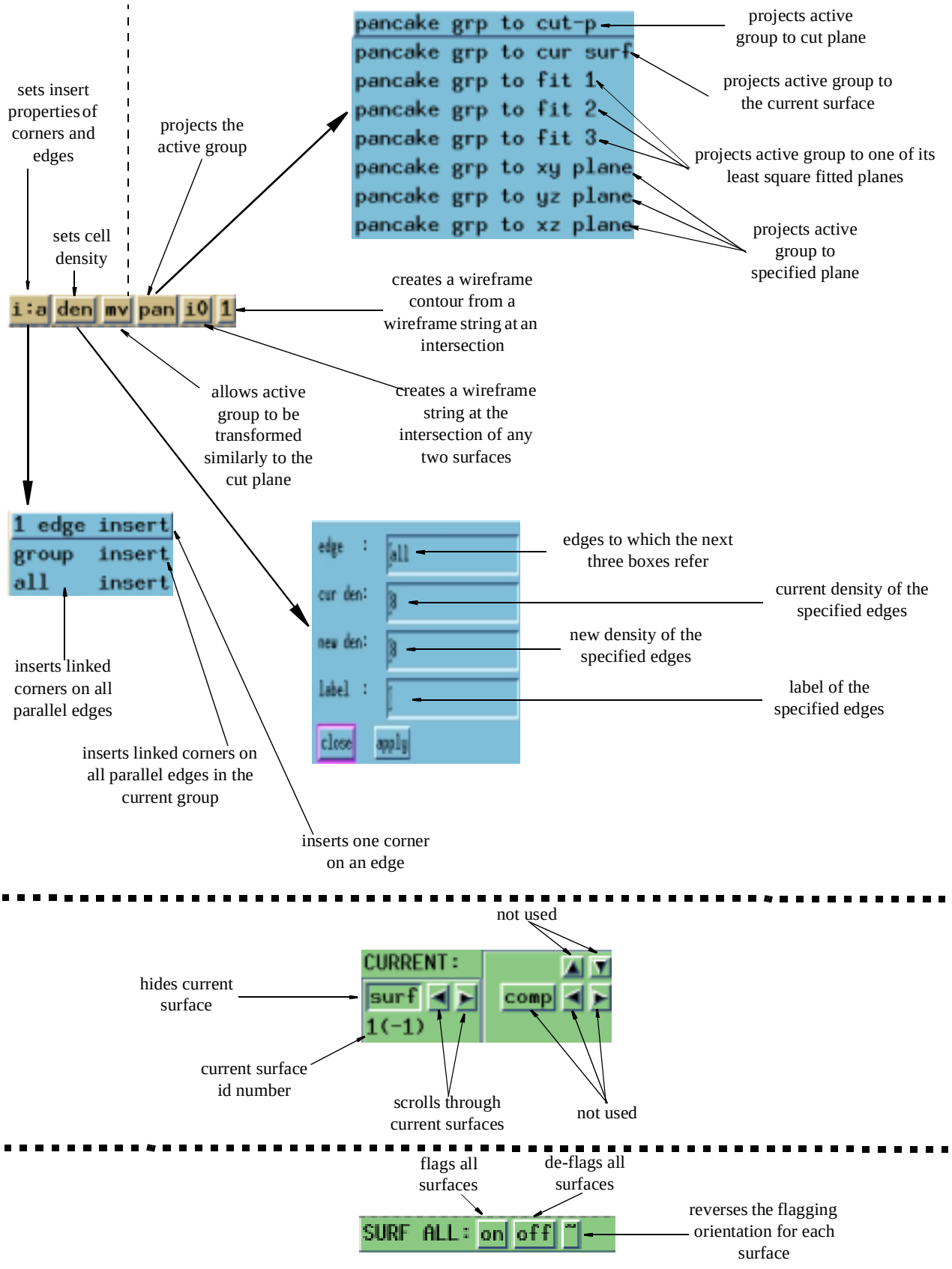

hides world and rotation axes
hides cut plane
hides corner id numbers
hides surface labels
SHOW: axis cut id lb
SURF ort
hides all surfaces
not used
SHOW: axis cut id lb
SURF ort
TOPO vec x&a e&h pos
GRID blk she ort
hides all topology
not used
hides excluded face symbols
hides highlighted errors
TOPO vec x&a e&h pos
GRID blk she ort
hides position cubes
hides all grids
hides all blocks
hides all sheets
hides IJK axes of blocks


centers the cut plane with the drawing area
view
group
centers cut plane with the center of mass of the active group
fits cut plane to the active group
grp fit 1
grp fit 2
grp fit 3
note
postitions the cut plane with the drawing area or groups
positions the cut plane on a world axis or predefined plane
postitions the cut plane with coordinates
NOTE :
pick or type
center:
0 0 0
normal:
0 0 1
close
apply
CUT-P: pos ctr norm
hand clip side fill
centers the cut plane
positions cut plane on a world axis
z
y
x
xy
-xy
xz
-xz
yz
-yz
positions cut plane on a cartesian plane
provides the cut plane normal vector
hides cut plane axes and handles
clips off a portion of the surface
displays clipped off portion of the surface
shades cut plane

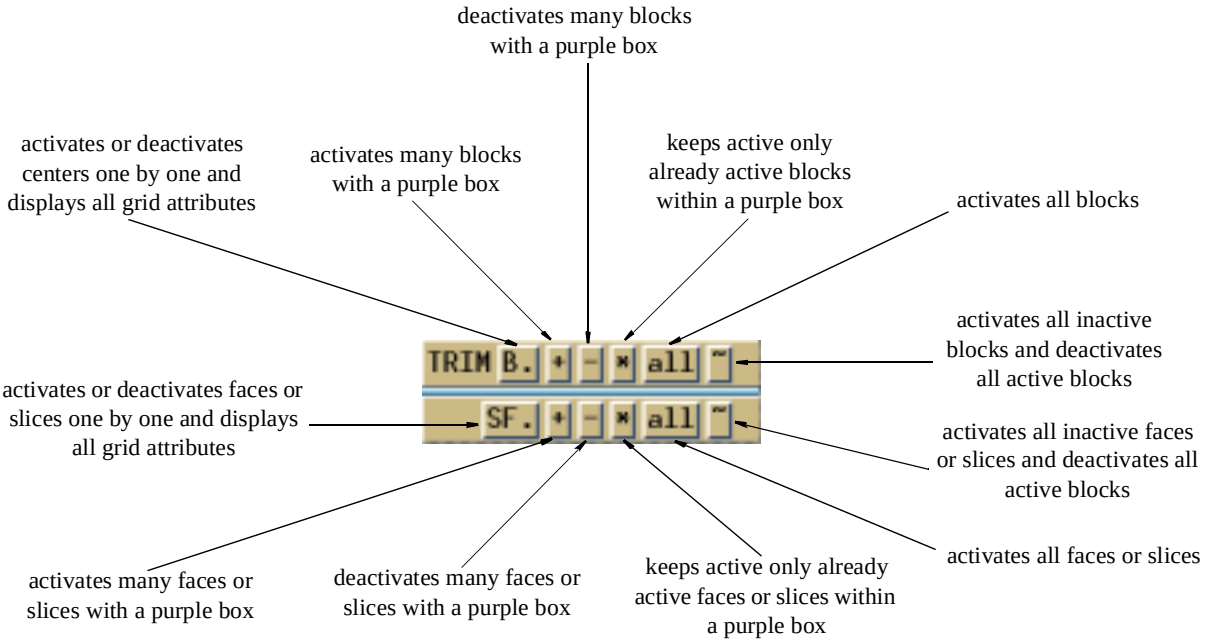

UN-REDO: unzoom 4.0
topo
view
view grp
rec
pulls back your view of the drawing area
20
15
10
5.0
4.0
3.0
2.0
1.5
1.2
1.1
factor by which the view is pulled back
undoes previous topological action
redoes current topological action
retrieves previously recorded viewpoint
illustrates viewpoint changes as topology actions are scrolled through
retrieves next recorded viewpoint
retrace topological actions by group operations or individual operations
records current viewpoint

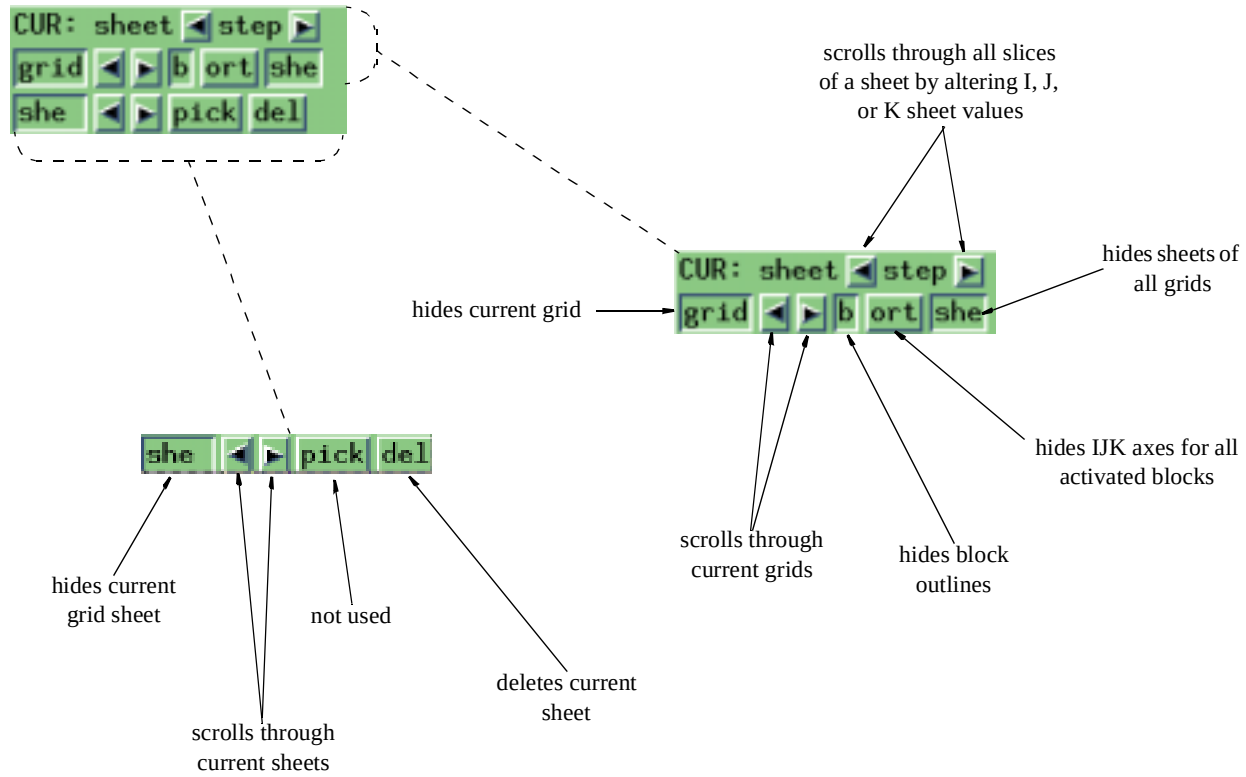
Panel T (Topology Builder) Operations

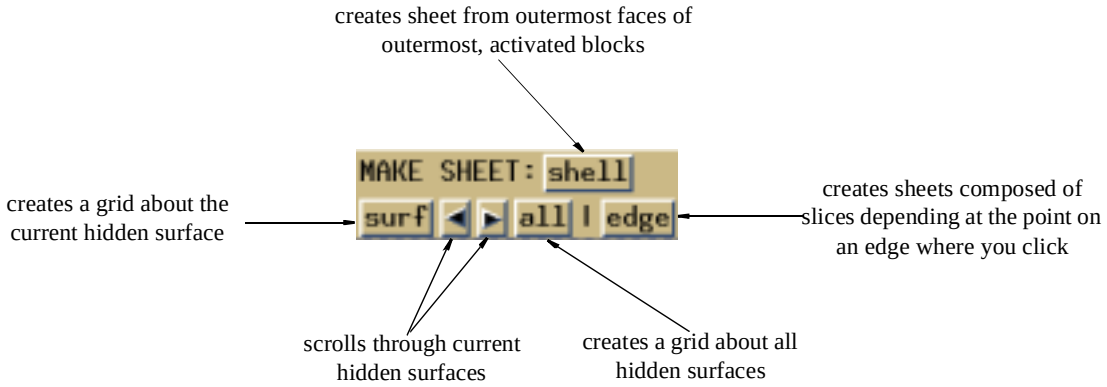


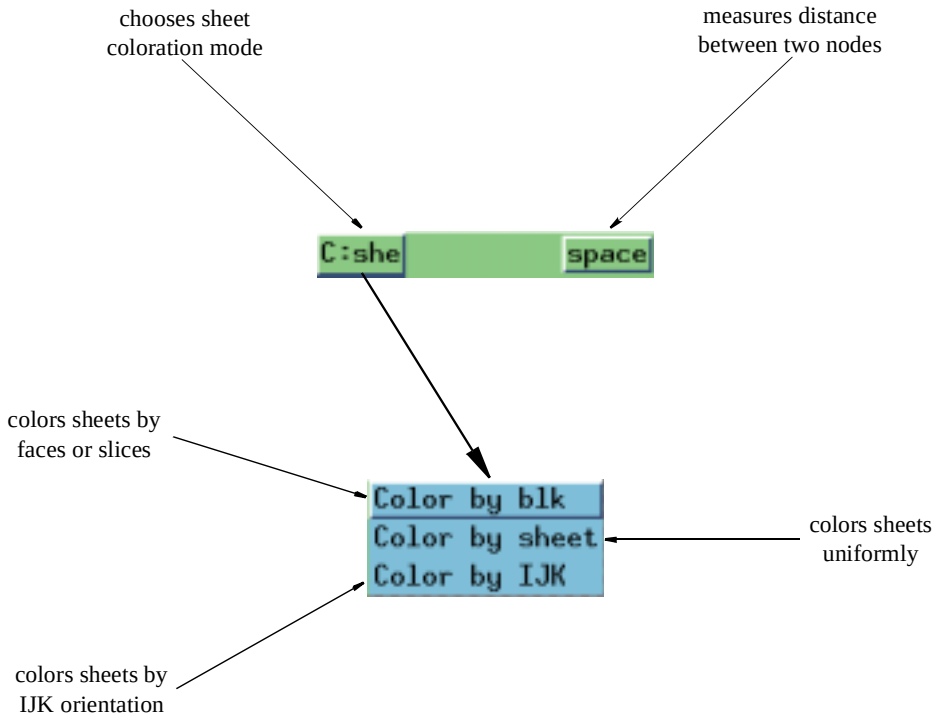

pancake grp to cut-p
pancake grp to cur surf
pancake grp to fit 1
pancake grp to fit 2
pancake grp to fit 3
pancake grp to xy plane
pancake grp to yz plane
pancake grp to xz plane
projects active group to cut plane
projects active group to the current surface
projects active group to one of its least square fitted planes
projects active group to specified plane
sets insert properties of corners and edges
projects the active group
sets cell density
i:a
den
mv
pan
i0
1
creates a wireframe contour from a wireframe string at an intersection
allows active group to be transformed similarly to the cut plane
creates a wireframe string at the intersection of any two surfaces
1 edge insert
group insert
all insert
inserts linked corners on all parallel edges
inserts linked corners on all parallel edges in the current group
inserts one corner on an edge
edge :
all
cur den:
8
new den:
8
label :
close
apply
edges to which the next three boxes refer
current density of the specified edges
new density of the specified edges
label of the specified edges
not used
CURRENT:
surf
comp
1(-1)
hides current surface
current surface id number
scrolls through current surfaces
not used
flags all surfaces
de-flags all surfaces
SURF ALL:
on
off
reverses the flagging orientation for each surface

Panel G (Grid Viewer) Operations

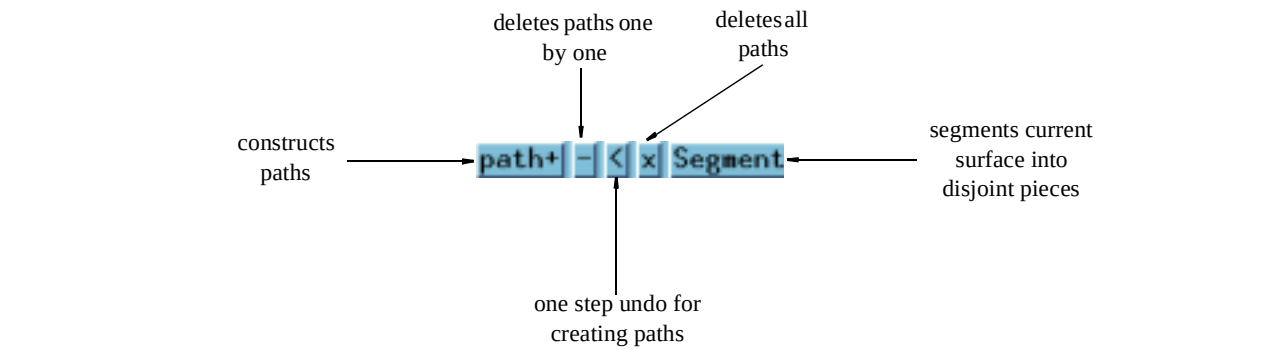


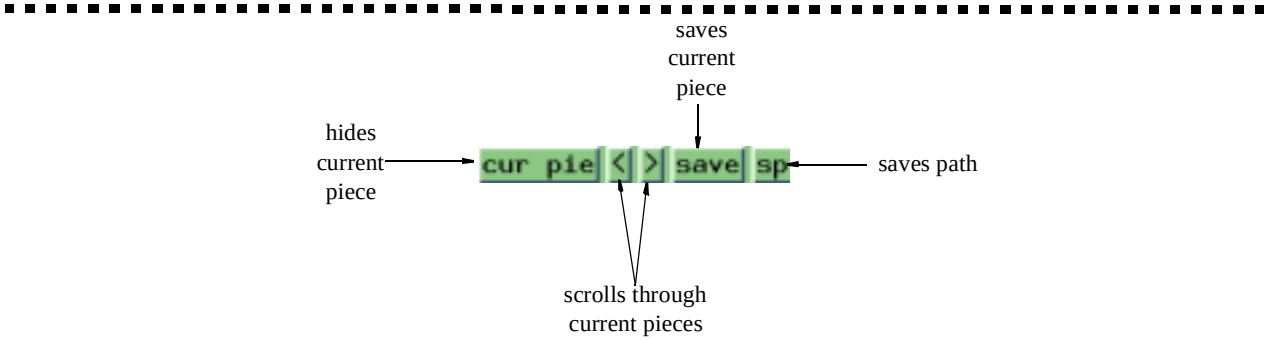


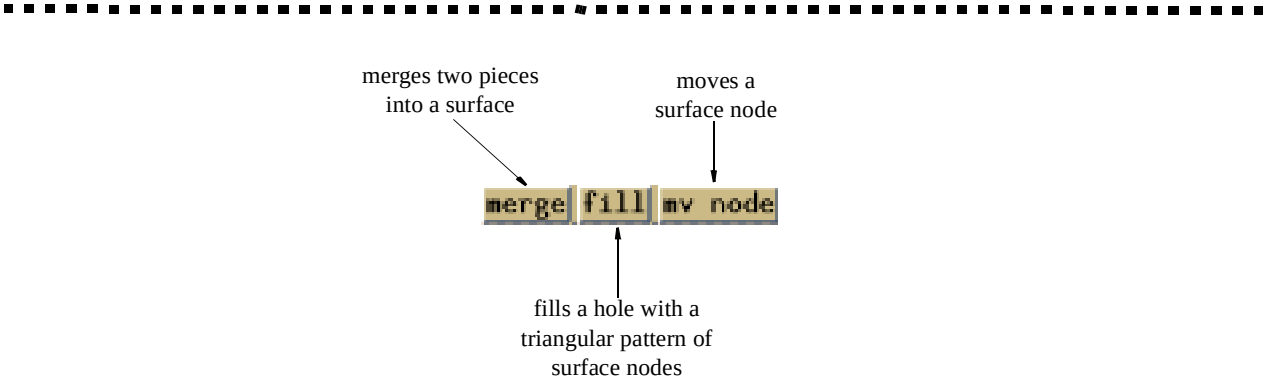

creates sheet from outermost faces of outermost, activated blocks
MAKE SHEET: shell
surf
all
edge
creates a grid about the current hidden surface
creates sheets composed of slices depending at the point on an edge where you click
scrolls through current hidden surfaces
creates a grid about all hidden surfaces

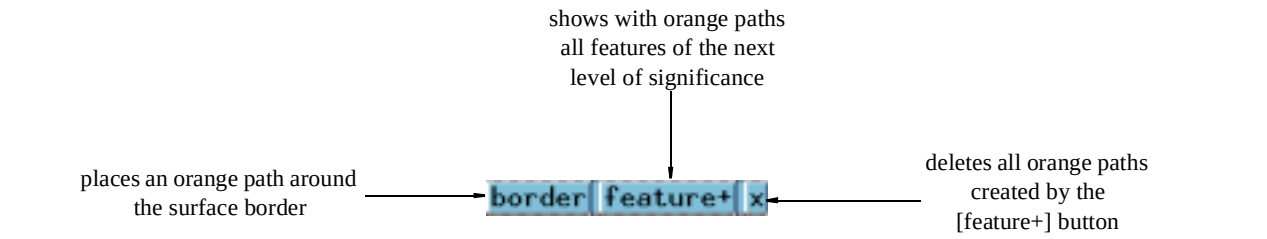

chooses sheet coloration mode
measures distance between two nodes
C:she
space
colors sheets by faces or slices
Color by blk
Color by sheet
Color by IJK
colors sheets uniformly
colors sheets by IJK orientation

Panel S (Mini CAD) Operations

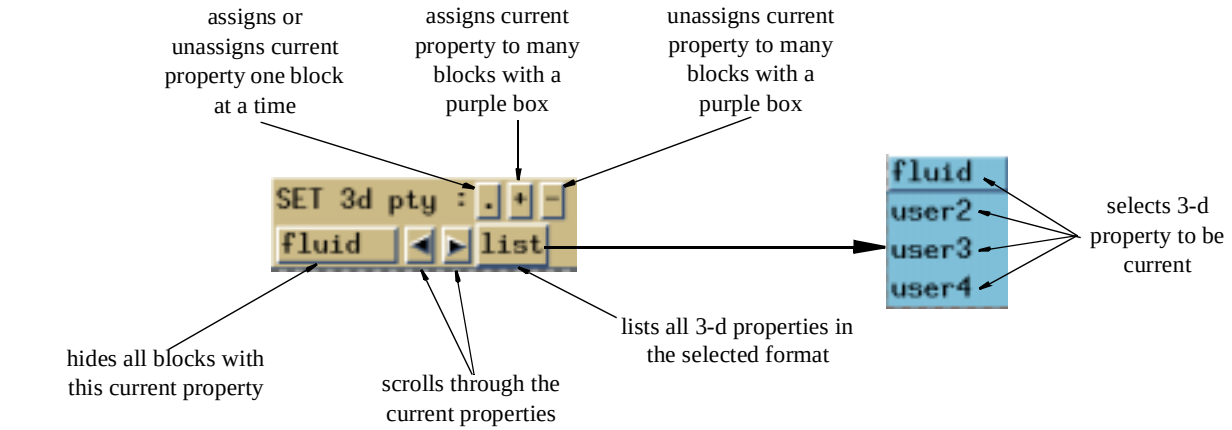


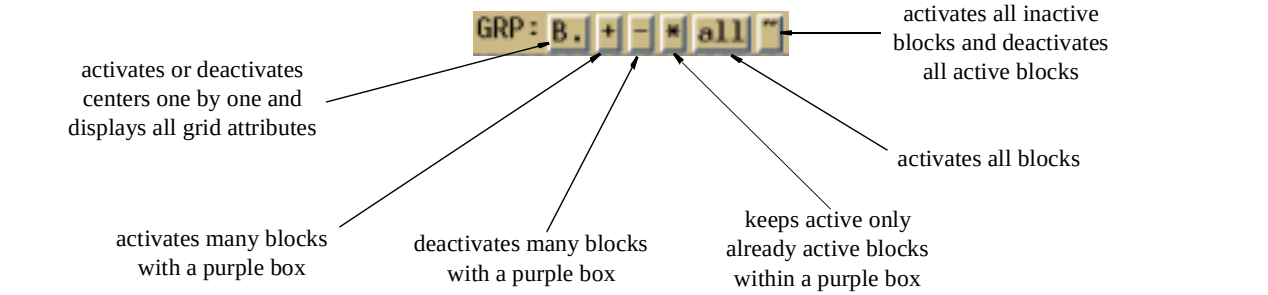


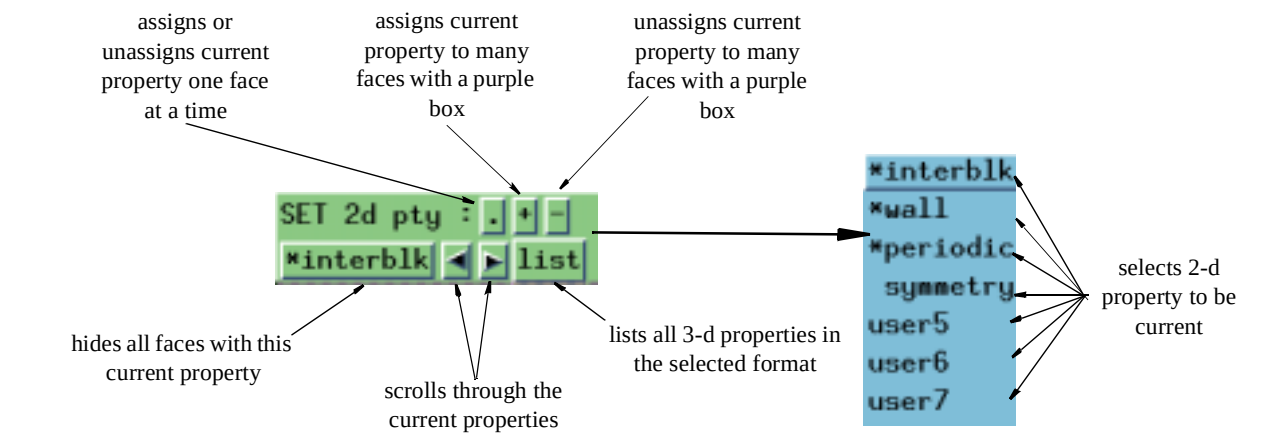


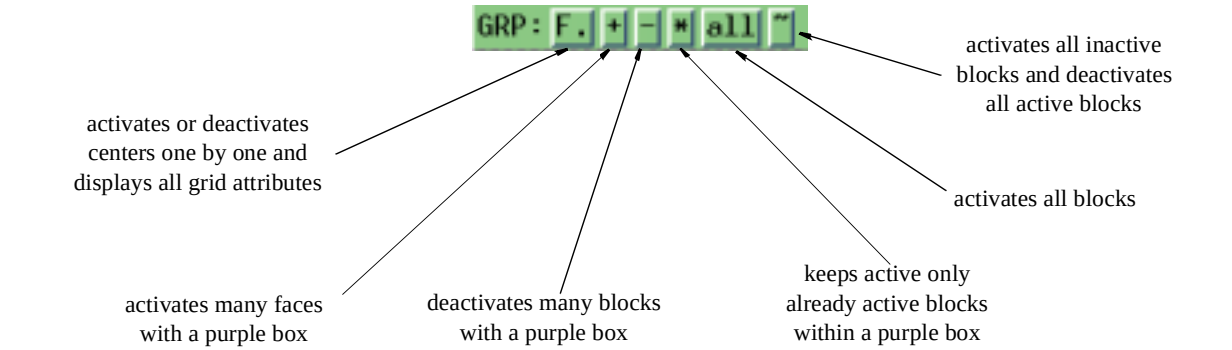


Panel P (Property Setter) Operations









Keypad Operations

C - allows you to input corners with the left mouse button

E - allows you to create an edge

F - allows you to remove a face by clicking diagonal corners

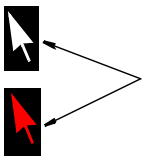
I - allows you to input a corner on an edge or many corners at once

O - displays the coordinates of the cursor on the cut plane

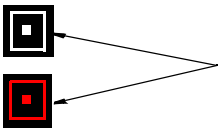
R - allows you to remove corners and edges

S - allows you to assign a corner to the current surface

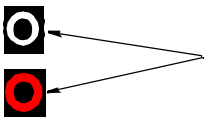
Cursor Shapes



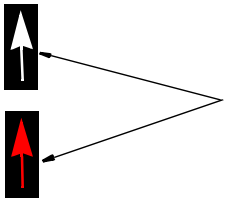
-pushes buttons
-selects menu items
-inputs topological objects



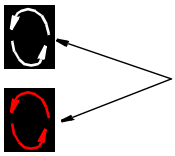
-selects cut plane handle
-makes a surface current



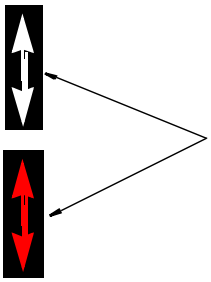
selects corners



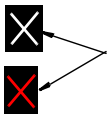
selects surface nodes



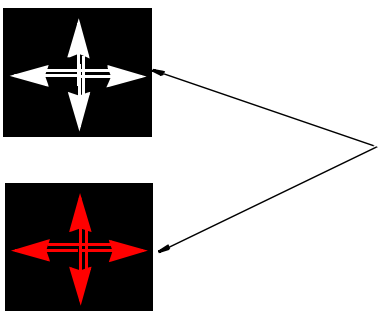
-rotates drawing area
-picks rotation axis



-selects topological edges
-selects grid edges



removes face exclusion indicators



drags corners

Appendix C - Guidelines for Building GridPro Topology

Note: If your question is not answered here, see Appendix D - Frequently Asked Questions

I. General Process

1). Load surfaces that leaklessly bound the region to be gridded.
2). Create topology that is a coarse hex decomposition of the region with proper wrap structures.
3). Logically assign corners to intended surfaces.
4). Assign grid densities to edges.
5). Launch Ggrid process to debug and generate a grid.
6). View and 'qchk' the grid.

II. Surface Preparation and Usage

1). Smoothness: make sure surfaces are smooth. Other qualities such as element shape are less important.
2). Details: use enough elements to resolve the curvature regions.
3). Do not over specify surface with too many elements. Do not specify a surface too much beyond the intended region.
4). Do not trim the surface part that is bounded by intersections with other surfaces.
5). Whenever possible, use build-in implicit surfaces over other types and use -tube type over digitized types.
6). Intersection: If two surfaces are supposed to intersect (e.g. a corner assigned to both sur faces), specify the surfaces slightly beyond the intersection.
7). Intersection: If two surfaces are supposed to intersect, make sure on any intersection point the normals from the two surfaces are not parallel. You can either use additional internal surface on the intersection or make the two surfaces into one to solve the problem.

III. Topology Construction:

1). Avoid singular edges on surface: One can always use wrap to move singular edges into volume.
2). Avoid the use of high degrees of singularities.
3). Place topology on the convex side of the intended surface near the high curvature regions.
4). Run less dense grid first to test out the topology. But the density should be enough to resolve the curvatures. This is a trail and error process.

IV. Scale Separation:

1). Use internal surfaces to separate regions of different scales.
2). Define a blade shaped surface as two surfaces, one for each side of the blade. Then use an internal surface to define the separation of the two halves.

V. Most Encounted Problems:

1). Causes of Topology Errors:
   a). wrong surface assignments.
   b). wrong surface sideness (1-sided vs 2-sided).
   c). isolated or hanging corners or edges.
2). Causes of Bad Grids.
   a). surfaces do not intersect.
   b). topology cuts surface.
   c). surface orientation wrong.
   d). not enough grid densidy for curvature.
   e). need more sweeps.
   f). surface assignments wrong.
   g). singular edges on surfaces.
   h). no internal surface for scale separation
   i). bad topology design.

Appendix D - Frequently Asked Questions

Drawing Area

Why can't I find my surfaces even though I have translated the viewscreen extensively?

If you created your surfaces on GridPro/AZ3000, did you press "apply" after setting the surface properties? Are you in the "topology builder" mode? Is the "surf" button in the green, "SHOW" box depressed? Have you dilated the viewscreen way too many times? Pressing the "Q" key and moving the cursor should give you an indication of what scale you are working in. Depending on your viewscreen scale and the size of your surfaces, you may have to zoom in or unzoom many times in order to see your surfaces.

I cannot find the world axes even though the "axis" button is depressed. How can I find them?

You have dilated the viewscreen too many times. You must zoom or unzoom repeatedly. From a distance, the axes look like small orthogonal segments, or even just a point.

I cannot find the cut plane even though the "cut" button is depressed and I have translated the viewscreen. How do I find it?

You have dilated the viewscreen too many times. You must zoom or unzoom repeatedly. If you have enlarged the viewscreen, you are most likely looking inside the cut plane rectangle; pressing the "fill" button will let you know for sure.

I cannot find the grid even though I am in the "Grid Viewer" mode. Where is it?

Did you remember to load your grid from the [blk.tmp] file after the Ggrid process was complete? If so, is the "MB" button in the green, "SHOW" box depressed? Maybe you need to zoom in or zoom out because your grid is very large or very small.

Why does GridPro/AZ3000 rotate my surfaces so slowly?

Most likely, the "shade," "shade0," "HLR," "HLR with shade," or "line" operation is accessed in the "move" menu found in the brown, "STYLE" box. Although the speed of your computer factors into this problem, these operations tend to slow down any computer. Therefore, change the operation in the menu to "point" or "1/10 point."

Why does my three-dimensional object look like a two-dimensional object?

You are looking at the object head on. Rotate the viewscreen.

How do I remove screen clutter?

See what you can hide. Surfaces and the position cube often must not always be displayed. Also, under the "stop" heading in the brown, "STYLE" box, change the surface representation to "point" or "1/10 point."

Why does my three-dimensional object look like a two-dimensional object?

You are looking at the object head on. Rotate the viewscreen.

Topology

GridPro/AZ3000 claimed that my topology was complete and ready for gridding. However, the resulting grid has a strikingly unexpected shape. Is something wrong?

Yes. One or more of your surface orientations are incorrect. As a result, surfaces either folded in on themselves or tried to stretch to infinity. You must go back to the "topology builder" mode and check the surface properties for all surfaces.

Why can't I create corners in two dimensions?

Remember, you can only create corners on the cut plane. Therefore, the axes of the cut plane are probably not aligned with the screen axes. The easiest solution is to exit and reenter the az manager.

Why won't the cursor manipulate or recognize the topology that is on the screen?

Most likely, a group is activated and the topology is dimmed. Deactivate the group.

Two corners are very close together. How can I drag a purple box over just one of them?

Zoom in on both corners; they will separate.

I know my wireframe is correctly constructed. Why won't GridPro grid the wireframe?

Your surface assignments are either incorrect or nonexistent.

Grids

Why won't my grid obey my instructions?

There might more than one grid present in the grid viewer; maybe you forgot to eliminate your previous grid before starting a new one.

Appendix E - Incorporating User-Defined Solver Formats

E.1 Overview

The AZ-Graphic Manager has a command panel, the Property Setter (panel=P) panel, for assigning properties (boundary conditions) to blocks and block faces. Grid data and associated information can be outputted in any solver format that has already been implemented in the AZ-Graphic Manager.

The AZ-Graphic Manager implements, as a plug-in script or plug-in executable, functionalities for outputting a grid in any given solver format. That is, a user can add to the AZ-Graphic Manager the capability of outputting a grid in a given solver format without the requirement of knowing, changing, or recompiling the source code of the AZ-Graphic Manager.

The incorporation of a new solver grid format involves:

1). Adding entries into two existing files:
-- GridPro/az_mngr/gridfmt.menu
-- GridPro/az_mngr/ptymap.menu

2). Creating a property map file that maps GridPro property ids to solver specific literal names. These property maps are loaded into the AZ-Graphic Manager when the given solver format is selected from the menu button “pty=*” on the top menubar.

3). Creating one or two UNIX or PC (.bat) script files that are called when the grid is saved in the given solver format. A user-supplied executable, and other GridPro utilities, such as “trf”, “mrgb”, and “chfmt,” can be used in the scripts.

4). Creating an executable that reads: a) the GridPro grid (including grid data, connectivity and property assignments) and the corresponding property format b) the grid output in the given solver format. The execution of this executable is part of the tasks included in the script discussed.

The grid files in the GridPro format are solver independent in the sense that no solver specific names appear in these files. The linkage between the grid in the GridPro format and a particular solver is through the property map file, where the numbers (property ids) are mapped to the solver specific literal names.

For the purpose of explaination, we will call the new solver “MYSLV”.

E.2 Adding an Entry to ’GridPro/az_mngr/gridfmt.menu’

The file, 'GridPro/az_mngr/gridfmt.menu', controls the solver list in the format line of the file dialog box when a grid is to be saved. Each solver format takes one line in the file. The line for a new solver can be inserted in any position in the file. Each solver format line has four items separated by the symbol '&'. For example,

MYSLV &MYSLV & outE_myslv.script & outM_myslv.script

The first item, 'MYSLV', is the solver name that appears in the format list when a grid is saved. It should not be longer than 10 characters. The second item should be identical to the first except if the solver name exceeds seven characters. In this case, the second item should be an abbreviation of the first. The third item is the script or executable file which will run when the grid is saved with elementary block data. The fourth item is the script or executable file which will run when the grid is saved with super block data. The file names in items 3 and 4 are not important for UNIX systems. For the PC, a script file should end with the '.bat' extension. Either the third or the fourth item can be the reserved name 'unused'.

Internally, when a grid is saved, say, with the elementary block data, the AZ-Graphic Manager always first saves the corresponding '.pty' file, then executes the following system call:

system("outE_myslv.script grid_file_name output_file_name");

where,

outE_myslv.script -- name of the file to run as specified in the solver line above.
grid_file_name -- the file name of the current grid.
output_file_name -- the output file name typed in during the file dialog.

Figure E.1 illustrates the result.

Figure E.1

The current 'gridfmt.menu' file is listed below:

```
#----  FILE: gridfmt.menu ----
#menu lbl & head & elem b script to run & super b script to run
CFX4   &CFX4test & outE_cfx4.script  & outM_cfx4.script
GASP   &GASP   & outE_gasp.script  & outM_gasp.script
plot3d &plot3d & outE_p3d.script   & outM_p3d.script
FIDAP  &FIDAP  & outU_fidap.script & unused
Nastran&Nastran& outU_nast.script  & unused
Patran &Patran & outU_patran.script& unused
Fluent &Fluent & unused            & unused
pdc    &pdc    & outE_pdc.script   & outM_pdc.script
pdc uns&pdc uns& outU_pdc.script   & unused
#----  END OF FILE ----
```

E.3 Writing the Output Script

As mentioned in section E.2 with the "MYSLV" example, when a grid is saved, the AZ-Graphic Manager makes a system call to execute the script 'outE_myslv.script' (or 'outM_myslv.script'). The user is responsible for writing the script, although one may copy an existing script of another solver in the GridPro/az_mngr/ directory, and do some editing from there. This script should be placed in the GridPro/az_mngr/ directory. This script should read in the GridPro elementary block grid (grid data, '.conn' file and '.pty' file), and the property map file, 'ptymap.myslv,' which will be discussed later. Also, the script should output the grid in the MYSLV format. One can use a combination of GridPro utilities (including "trf", "mrgb" and "chfmt") in the script to accomplish some of the tasks involved.

However, in general, one needs to write at least one C or FORTRAN program to finish the final conversion. Typically, this program will read GridPro files (i.e., 'blk.grd', 'blk.grd.conn', optionaly 'blk.grd.pty' for elementary blocks, and 'blk.grd blk.conn_n' for super blocks).

An example script for GASP is listed below:

```
#!/bin/csh -fe
#---- FILE: outM_gasp.script ----
mrgb $1 -s 1 -gasp
mv $1.tmp.inp $2.inp
chfmt $1.tmp -f p3d
```

```
mv $1.tmp.tmp $2.p3d
\rm $1.tmp
exit
#---- END OF FILE ----
```

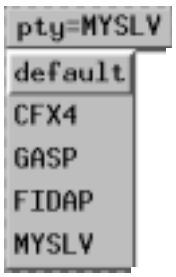
E.4 Adding an Entry to 'GridPro/az_mngr/ptymap.menu'

The property maps map the internally used number property id to solver specific literal names. Each solver has its own property map file, which is also a part of the corresponding solver entry in the file 'GridPro/az_mngr/ptymap.menu'.

In 'GridPro/az_mngr/ptymap.menu', each solver takes one line. And each solver line has three items separated by the symbol '&'. Again, using MYSLV as an example, we need to add one entry as follows:

```
MYSLV &pty=MYSLV & ptymap.myslv
```

Figure E.2



The first item is the new solver name appearing in the menu item list for both the button [pty=*] on the top menubar and the property line in the surface parameter dialog box. Figure E.2 illustrates the result. The second item is the name label appearing on the two menu buttons once the property is selected from the menu item list. The first two items should not be longer than 10 characters each.

The third item is a file name containing the property maps for the given solver, which will be discussed in the next section. This file is solver specific and user supplied.

The current 'ptymap.menu' file is listed below:

```
#----  FILE: ptymap.menu ----
default&pty=PDC  & ptymap.default
CFX4   &pty=CFX4 & ptymap.cfx4
GASP   &pty=GASP & ptymap.gasp
FIDAP  &pty=FIDAP& ptymap.fidap
#----  END OF FILE ----
```

E.5 Writing the 'ptymap.*' File

Recall that the *user* must create the file 'ptymap.myslv'. The literal property names in the file will appear in the AZ-Graphic Manager. The user sees these literal property names when assigning properties to grids with the involved solver format.

Note that, the GridPro grid is solver independent in the sense that the properties are just numbers (with possibly GridPro-defined names). The file, 'ptymap.myslv', makes the connection from the numbers to the liternal names that are specific to the solver.

To create 'ptymap.myslv', one can copy and edit either the file 'ptymap.template,' or the file 'ptymap.*', for another solver.

Let us use the CFX4 format as an example. The file is listed below:

```
#---- FILE: ptymap.cfx4 ---
10    # 2d-ptys.
#pdc-id& pdc-name & mapped_name &label #comments
1     & INTERBLK & BLKBDY  &*BLKBDY   #grid generic basic
2     & PERIODIC & USER2D  &*USR2D:prd#grid generic basic
3     & BOUNDARY & WALL    &*WALL     #grid generic basic
4     & SYMMETRY & SYMMET
5     & user5    & unused
6     & user6    & CNDBDY
7     & user7    & PRESS
8     & user8    & INLET
9     & user9    & OUTLET
10    & user10
#---------------
7     # 3d-ptys.
#pdc-id& pdc-name &mapped-name  &label #comments
1     & BULK     &default &*FLUID  #grid generic
2     & user2    &SOLID   &
3     & user3    &SOLCON  &
4     & user4    &POROUS  &
5     & user5    &USER3D  &
6     & user6
7     & user7
#---- END OF FILE ----
```

As shown above, there are two sections of maps, one for the 2-d maps and another for the 3-d maps. The sections of maps have the same syntax: The first line is the number of properties

listed, followed by the property map list. Each line here defines the map for one property.

There are four items for each map line. They are:

1). pdc-id: must be >=1 and <= the length of the list, and can appear in any order.

2). pdc-name: generic property names. can be renamed. However, (2d)id=1..3 (3d)id=1 have special roles when the pty of surfaces, faces and blocks are initialized. The following rules are applied to the pty initialization:

Rule for surface:

If a surface has a default pty, the surface is initially assigned a pty-id=1(pdc-name=INTERBLK), 2(pdc-name=BOUNDARY) and 3(pdc-name=PERIODIC) for a 2-sided surface, a 1-sided surface and a periodic surface respectively.

Rule for faces:

a). If the face is on a surface, the face is reassigned with the surface pty.
Otherwise,
b). If the face attaches to two blocks, the face is reassigned the pty id=1(pdc-name=INTERBLK).
For anything else,
c). The face is reassigned the pty id=2(pdc-name=BOUNDARY).

Rule for overlap ptys:

If two pty ids are mapped to the same mapped-name, then the faces with larger id's are reassigned with the lower pty id.

3). mapped-names: solver specific names. Routines for a particular solver format may use these names. A blank mapped_name means unused. Due to the pty initialization rules, the mapped-names for (2d)id=1..3 should have the same corresponding meaning as the pdc-names.

4). label: (<= 10 chars) is what is shown on the button, can have spaces.

5) &: delimiter. A blank item should use '& &' instead of '&&'

For a PDC pre-defined pty map file, a user...

1). can change the order of the lines;
2). can change the labels;
3). can change pdc-name for id >= 4(2d) and id >= 2(3d);

4). needs to understand the initialization rules.

For a user defined pty map file, the user needs to map INTERBLK correctly since “mrgb” will merge blocks through it. In the Property Setter (Panel=P) panel, when you press the “list” button, and your format is selected, your custom-made properties will appear. Figure E.3 displays the final result.

Figure E.3

Also, as mentioned in chapter 9, you can assign properties to surfaces. These properties are inherited by the grid. To assign your custom-made properties to these surfaces, select your format, reload one of your surfaces, and press the button to the right of the “property” heading. Your properties should appear in the submenu. Figure E.4 summarizes the process.

Figure E.4

<$symbols<$numerics<$alphabeticsGridPro

<$symbols<$numerics<$alphabeticsTIL (Topology Input Language)

<$symbols<$numerics<$alphabeticsAZ Graphic Manager

<$symbols<$numerics<$alphabeticssurface

<$symbols<$numerics<$alphabeticstopological wireframe

<$symbols<$numerics<$alphabeticswireframe

<$symbols<$numerics<$alphabeticstopology

<$symbols<$numerics<$alphabeticsgrid

<$symbols<$numerics<$alphabeticsCAD (Computer Aided Design)

<$symbols<$numerics<$alphabeticssurface assignment

<$symbols<$numerics<$alphabeticscorners

<$symbols<$numerics<$alphabeticsGgrid

<$symbols<$numerics<$alphabeticssurface id (identification)

<$symbols<$numerics<$alphabeticscurrent surface

<$symbols<$numerics<$alphabeticsscreen axis

<$symbols<$numerics<$alphabeticsworld axis

<$symbols<$numerics<$alphabeticscut plane axis

<$symbols<$numerics<$alphabeticsrotation axis

<$symbols<$numerics<$alphabeticssnap

<$symbols<$numerics<$alphabeticscut plane

<$symbols<$numerics<$alphabeticscut plane rectangle

<$symbols<$numerics<$alphabeticshandle

<$symbols<$numerics<$alphabeticssnap

<$symbols<$numerics<$alphabeticsclip

<$symbols<$numerics<$alphabeticsgeneral position

<$symbols<$numerics<$alphabeticsfull face matching

<$symbols<$numerics<$alphabeticswrap

<$symbols<$numerics<$alphabeticsedge

<$symbols<$numerics<$alphabeticscorner

<$symbols<$numerics<$alphabeticsedge

<$symbols<$numerics<$alphabeticssurface assignment

<$symbols<$numerics<$alphabetics/[_az.fra/]

<$symbols<$numerics<$alphabeticssurface orientation

<$symbols<$numerics<$alphabeticsinternal surface

<$symbols<$numerics<$alphabeticsrun

<$symbols<$numerics<$alphabeticssweep

<$symbols<$numerics<$alphabeticsresidual

<$symbols<$numerics<$alphabeticsfold count

<$symbols<$numerics<$alphabetics[blk.tmp]

<$symbols<$numerics<$alphabeticsmesh

<$symbols<$numerics<$alphabeticsblock

<$symbols<$numerics<$alphabeticscenter

<$symbols<$numerics<$alphabeticsoutline

<$symbols<$numerics<$alphabeticsskeleton

<$symbols<$numerics<$alphabeticsgroup

<$symbols<$numerics<$alphabeticsactive group

<$symbols<$numerics<$alphabeticssecondary wireframe

<$symbols<$numerics<$alphabeticspancake

<$symbols<$numerics<$alphabeticsdensity

<$symbols<$numerics<$alphabeticsutility

<$symbols<$numerics<$alphabeticsface

<$symbols<$numerics<$alphabeticsslice

<$symbols<$numerics<$alphabeticssheet

<$symbols<$numerics<$alphabeticsshell

<$symbols<$numerics<$alphabeticsface sheet

<$symbols<$numerics<$alphabeticsslice sheet

<$symbols<$numerics<$alphabeticspartial shell

<$symbols<$numerics<$alphabeticsnode spacing

<$symbols<$numerics<$alphabeticsEuler grid

<$symbols<$numerics<$alphabeticsNavier-Stokes grid

<$symbols<$numerics<$alphabeticsboundary conditions

<$symbols<$numerics<$alphabeticsoff-wall

<$symbols<$numerics<$alphabetics–xpolar surface

<$symbols<$numerics<$alphabetics–xyz surface

<$symbols<$numerics<$alphabeticsperiodic axis

<$symbols<$numerics<$alphabeticsflag

<$symbols<$numerics<$alphabeticsquad.tmp

<$symbols<$numerics<$alphabeticsstring.tmp

<$symbols<$numerics<$alphabeticssurf.tmp.fra

Index

S

T

V

W