SAND 93-1373
Unlimited Release
Printed July 1993

Distribution
Category UC-405

GENSHELL: A Genesis Database 2D to 3D Shell Transformation Program

Gregory D. Sjaardema
Solid and Structural Mechanics Department
Sandia National Laboratories
Albuquerque, New Mexico 87185

Abstract

GENSHELL is a three-dimensional shell mesh generation program. The three-dimensional shell mesh is generated by mapping a two-dimensional quadrilateral mesh into three dimensions according to one of several types of transformations: translation, mapping onto a spherical, ellipsoidal, or cylindrical surface, and mapping onto a user-defined spline surface. The generated three-dimensional mesh can then be reoriented by offsetting, reflecting about an axis, revolving about an axis, and scaling the coordinates. *GENSHELL* can be used to mesh complex three-dimensional geometries composed of several sections when the sections can be defined in terms of transformations of two-dimensional geometries. The code *GJOIN* is then used to join the separate sections into a single body. *GENSHELL* updates the *EXODUS* quality assurance and information records to help track the codes and files used to generate the mesh. *GENSHELL* reads and writes two-dimensional and three-dimensional mesh databases in the *GENESIS* database format; therefore, it is compatible with the preprocessing, postprocessing, and analysis codes in the Sandia National Laboratories Engineering Analysis Code Access System (SEACAS).

Intentionally Left Blank

Contents

1 Introduction **7**
1.1 SEACAS Mesh Generation Toolbox 7
1.2 Introduction to the GENESIS File Format 9
1.3 Overview of Capabilities in GENSHELL 9
1.4 Organization of Report 10

2 Command Input **11**
2.1 Mesh Transformation 12
2.2 Mesh Orientation 18
2.3 Modification or Creation of Material, Nodeset, or Sideset IDs 22
2.4 Information and Processing 22
2.5 Order of Transformation 23

3 GENSHELL Example Input and Output **25**

4 References **27**

A The GENESIS Database Format **29**

B Command Summary **31**

C GENSHELL Details **35**

Figures

Figure 1. Schematic of SEACAS Mesh Generation Process 7
Figure 2. Illustration of WARP POINT transformation 13
Figure 3. Illustration of WARP YAXIS MAP transformation 14
Figure 4. Illustration of WARP YAXIS VERTICAL transformation 15
Figure 5. Illustration of WARP ELLIPSE transformation 16
Figure 6. Illustration of SPLINE (SPHERICAL) transformation 19
Figure 7. Illustration of SPLINE (XSWEEP) transformation 20
Figure 8. Illustration of SPLINE (YSWEEP) transformation 21

Intentionally Left Blank

1 Introduction

GENSHELL is a mesh generation program that creates a three-dimensional shell mesh by transforming a two-dimensional mesh. Four transformations are available: (1) translating normal to the plane of the two-dimensional mesh, (2) mapping the two-dimensional mesh onto a spherical or ellipsoidal surface, (3) mapping the two-dimensional mesh onto a cylindrical surface, and (4) mapping the two-dimensional mesh onto a user-defined spline surface. *GENSHELL* is one of the mesh generation tools in the Sandia National Laboratories Engineering Analysis Code Access System (SEACAS) [1].

1.1 Sandia Engineering Analysis Code Access System Mesh Generation Toolbox

GENSHELL is typically used with the other SEACAS mesh generation codes *FASTQ* [2], *GEN3D* [3], *GREPOS* [4], *GJOIN* [5], and *Aprepro* [6]. Figure 1 shows the structure of the SEACAS mesh generation toolbox. The basic premise underlying this toolbox is that complicated geometries can be generated using a set of small specialized codes.

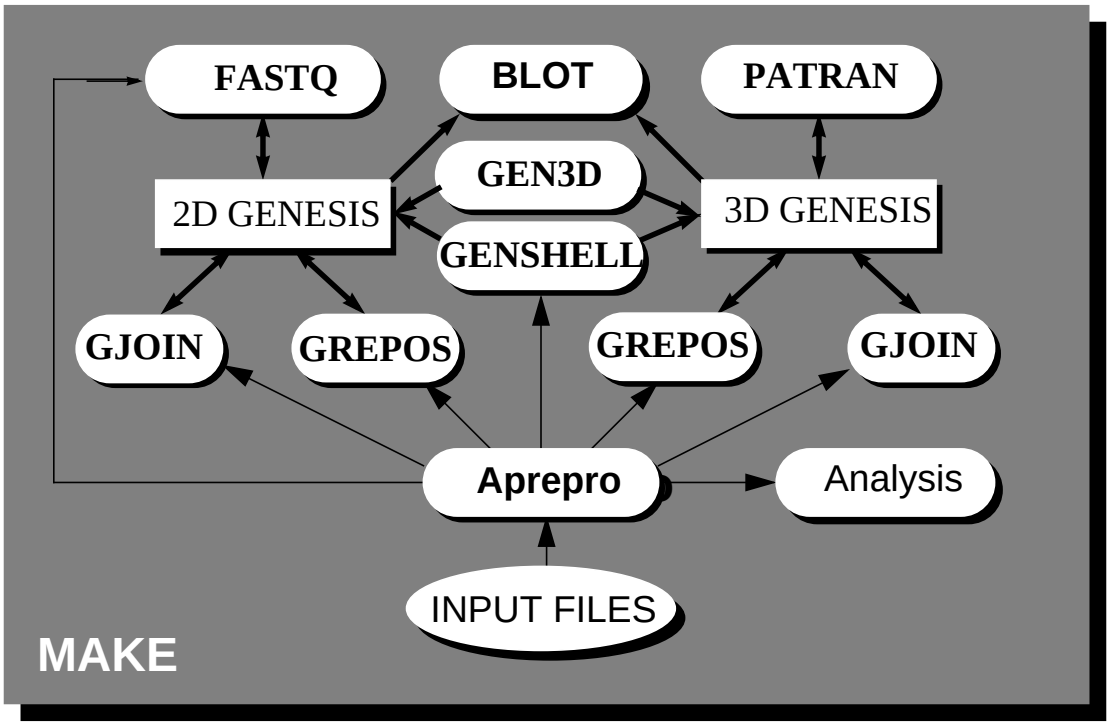

Figure 1. Schematic of SEACAS Mesh Generation Process

Each of these codes has a specialized purpose, a short synopsis of each code is given below, for more information consult the referenced documentation.

GEN3D Transforms a two-dimensional *GENESIS* database into a three-dimensional *GENESIS* database. Several

transformations are supported and additional transformations can be easily added.[3]

GREPOS Transforms the geometry of a *GENESIS* database by scaling, mirroring, offsetting, or rotating. It can also modify the database by deleting or renaming material blocks, sideset identifications, or nodeset identifications.[4]

GJOIN Join together two or more *GENESIS* databases into a single *GENESIS* database.[5]

FASTQ Interactive two-dimensional finite element mesh generation program. Includes several mesh generation options including paving.[2]

APREPRO An algebraic preprocessing program which is used to parameterize finite element analyses. Includes a unit conversion system and material database access routines.[6]

GENSHELL Transforms a two-dimensional *GENESIS* database into a three-dimensional shell *GENESIS* database. Several transformations are supported and additional transformations can be easily added.

NUMBERS Calculates several properties of an *EXODUS* file, including mass properties, timesteps, condition numbers, cavity volumes, and others.[7]

BLOT The primary graphical two-dimensional and three-dimensional postprocessing code. It includes deformed mesh plots, contour plots, shaded fringe plots, variable versus variable and time history plots, and distance versus variable plots. [8]

The SEACAS mesh generation toolbox is a simpler approach to three-dimensional mesh generation than the automatic and general-purpose programs that are available from commercial vendors. Many complicated three-dimensional geometries are composed of several primitives that can be defined in terms of transformations of two-dimensional geometries. Each of the primitives can be meshed using *FASTQ*, *GEN3D*, and *GENSHELL*, and then joined together using *GJOIN*.

This approach does; however, have some inherent difficulties. The biggest being managing and synchronizing several related files. For example, the meshes for some large problems can require more than one hundred files containing *FASTQ*, *GEN3D*, *GENSHELL, GJOIN*, and *GREPOS* input files; temporary *GENESIS* files; and parameter files. Manually building and modifying a mesh this complicated is obviously very difficult and time consuming. This problem has typically been minimized at SNL through the use of the *UNIX*[*] *make*[†] program and *Aprepro*. *Make* is used to build a set of dependencies between

* UNIX is a registered trademark of UNIX Systems Laboratories Inc.

the various pieces of the finite element model. The analyst can then change files as needed and simply type `make mesh` to generate the mesh. If the dependencies have been entered correctly, *make* will rebuild only those portions of the mesh that are affected by the changed file. The synchronization problem (that is, ensuring that all of the separate pieces have compatible dimensions and discretization) is typically solved by creating a few parameter files which contain key dimensions and discretization information. *Aprepro* is then used to preprocess the input files and insert the key dimensions and discretization information into the input files.

1.2 Introduction to the GENESIS File Format

The *GENESIS* mesh database file format is the geometry definition portion of the *EXODUS* database file format used in the Engineering Sciences Center at Sandia National Laboratories. All of the mesh generation programs in the Engineering Sciences Center read and write files in the *GENESIS* format, which allows great flexibility in the choice of mesh generation, file translation, and graphical processing.

The *GENESIS* file contains the data to describe a finite element mesh including the location of the nodal points, the connectivity of the nodes that form each element, the material types of each element, and the boundary condition data which are used to specify load application points and nodal constraints. The reader is referred to References [9] and [10] for more information.

1.3 Overview of Capabilities in *GENSHELL*

With *GENSHELL*, the user can create a three-dimensional shell mesh by transforming a two-dimensional mesh; change the orientation and size of the resulting three-dimensional mesh; and change material block, nodeset, and sideset IDs. *GENSHELL* also updates the *EXODUS* QA and information records to help track the codes and files used to generate the mesh.

Transformations:
The three-dimensional shell mesh can be transformed by translation of a two-dimensional mesh, or by mapping a two-dimensional mesh onto a spherical, cylindrical, ellipsoidal, or user-defined spline surface.

Orientation:
The orientation of the mesh can be changed by revolving, offsetting, reflecting, zeroing, and scaling the nodal coordinates of the original mesh.

---

† See your UNIX documentation for more information on make. Typically this is done by entering the command `man 1 make`

Material Blocks, Sidesets, and Nodesets:
In the SEACAS system, material properties are input according to material block IDs. Similarly, boundary conditions are associated with sideset and nodeset IDs. The input material block IDs, sideset IDs, and nodeset IDs can be changed to a new unique ID (that is they can be changed, but not combined).

The user may also create sets of the nodes on the front or back of the three-dimensional mesh. The front or back nodeset contains all of the two-dimensional nodes. The front or back sideset also contains all of the two-dimensional nodes; however, the connectivity of the sideset is such that the right-hand-rule specifies the normal of the front sideset and the reverse holds for the back sideset.

Quality Assurance (QA) and Information Records:
A QA record for the *GENSHELL* program is added to the input QA record(s), and the file-name of the input file is added to the information records that are written to the output mesh. The information record is prefaced by *GENSHELL* to help in identification. These records, which are explained in Reference [9], are useful in tracing the evolution of a mesh during the mesh generation process.

Node and Element Numbering:
The node and element numbering of the generated three-dimensional mesh is the same as the input two-dimensional mesh.

Element Attributes:
The generated three-dimensional shell elements have a single element attribute which specifies the thickness of the shell. This attribute is set by the transformation commands to the user-specified thickness.

1.4 Organization of Report

The remainder of this report is organized as follows:

- Chapter 2 describes the command input and valid commands, and
- Chapter 3 presents a few short examples illustrating *GENSHELL* use.

Three appendices are contained in this report.

- Appendix A is a code segment defining the *GENESIS* binary database format,
- Appendix B is a summary of the commands in *GENSHELL*, and
- Appendix C describes the specifics of *GENSHELL* including executing it, obtaining it, compiling it, and quality assurance.

2 Command Input

The user directs the execution of *GENSHELL* by entering commands to set processing parameters. The commands are in free-format and must adhere to the following syntax rules.

- Valid delimiters are a comma or one or more blanks.
- Either lowercase or uppercase letters are acceptable, but lowercase letters are converted to uppercase.
- A "$" character in any command line starts a comment. The "$" and any characters following it on the same line are ignored.
- A command may be continued over several lines with an "*" character. The "*" and any characters following it on the current line are ignored and the next line is appended to the current line.

Each command has an action keyword or "verb" followed by a variable number of parameters.

An action keyword or verb is a character string matching one of the valid commands. It may be abbreviated as long as enough characters are used to distinguish it from other commands.

The meaning and type of the parameters depend on the command verb. Most parameters are optional. If an optional parameter field is blank, a command-dependent default value is supplied. Valid entries for parameters are:

- A numeric parameter may be a real number or an integer. A real number may be in any legal FORTRAN numeric format (e.g., 1, 0.2, -1E-2). An integer parameter may be in any legal integer format*.
- A string parameter is a literal character string. Most string parameters may be abbreviated.

The notation conventions used in the command descriptions are:

- The command verb is in **bold** type.
- A literal string is in all uppercase SANSERIF type and should be entered as shown (or abbreviated).
- The value of a parameter is represented by the parameter name in *italics*.
- A literal string in square brackets ("[ ]") represents a parameter option which is omitted entirely (including any following comma) if not appropriate. These parameters are distinct from most parameters in that they do not require a comma

* Arithmetic equations, variables, and control structures can be used in the input files if the -aprepro option is used. See the Aprepro manual for more information.

as a place holder to request the default value.

- The default value of a parameter is in angle brackets (“< >”). The initial value of a parameter set by a command is usually the default parameter value. If not, the initial setting is given in the command description.

2.1 Mesh Transformation

TRANSLATE *thickness* <1.0>

TRANSLATE sets the thickness of the 3D shell mesh to *thickness*. This command supersedes previous transformation commands. The following command sets the thickness of the generated shell elements to 1.5:

```
TRANSLATE 1.5
```

All of the X and Y coordinates remain the same as those in the 2D mesh; the Z coordinates are set equal to 0.0

WARP POINT *thickness* <1.0>, *radius* <no default>

WARP POINT causes the 2D mesh to be mapped onto a spherical surface to create the 3D mesh. The spherical surface has a radius of curvature equal to *radius*. The center of curvature is located on the z-axis, and it is a distance of *radius* above the X-Y plane. The total thickness is *thickness*. Note that *radius* must be greater than the maximum distance from the z-axis to the boundary of the 2D mesh. This command supersedes previous transformation commands. The mesh transformation is performed in two parts. First, the warped nodal positions (*xw*, *yw*, *zw*) are calculated by mapping the original 2D mesh onto a spherical surface with a radius of curvature equal to *radius*. The original x and y coordinates of the 2D mesh remain at the same values; the z coordinate is calculated such that the distance to the center of curvature is equal to *radius*.

$$x_w = x_{2D}$$

$$y_w = y_{2D}$$

$$z_w = radius - \sqrt{radius^2 - x_{2D}^2 - y_{2D}^2}$$

The warped nodal positions are projections parallel to the z-axis onto a spherical surface of radius *radius*. A square mesh warped with the command "WARP POINT 1 2" is shown in Figure 2.

WARP XAXIS [MAP|VERTICAL] *thickness* <1.0>, *radius* <no default>
WARP YAXIS [MAP|VERTICAL] *thickness* <1.0>, *radius* <no default>

This second form of the WARP command maps the 2D mesh to a cylindrical surface centered on either the X or Y axis to create the 3D mesh. The cylindrical surface has a radius of curvature equal to *radius*. The center of curvature is located

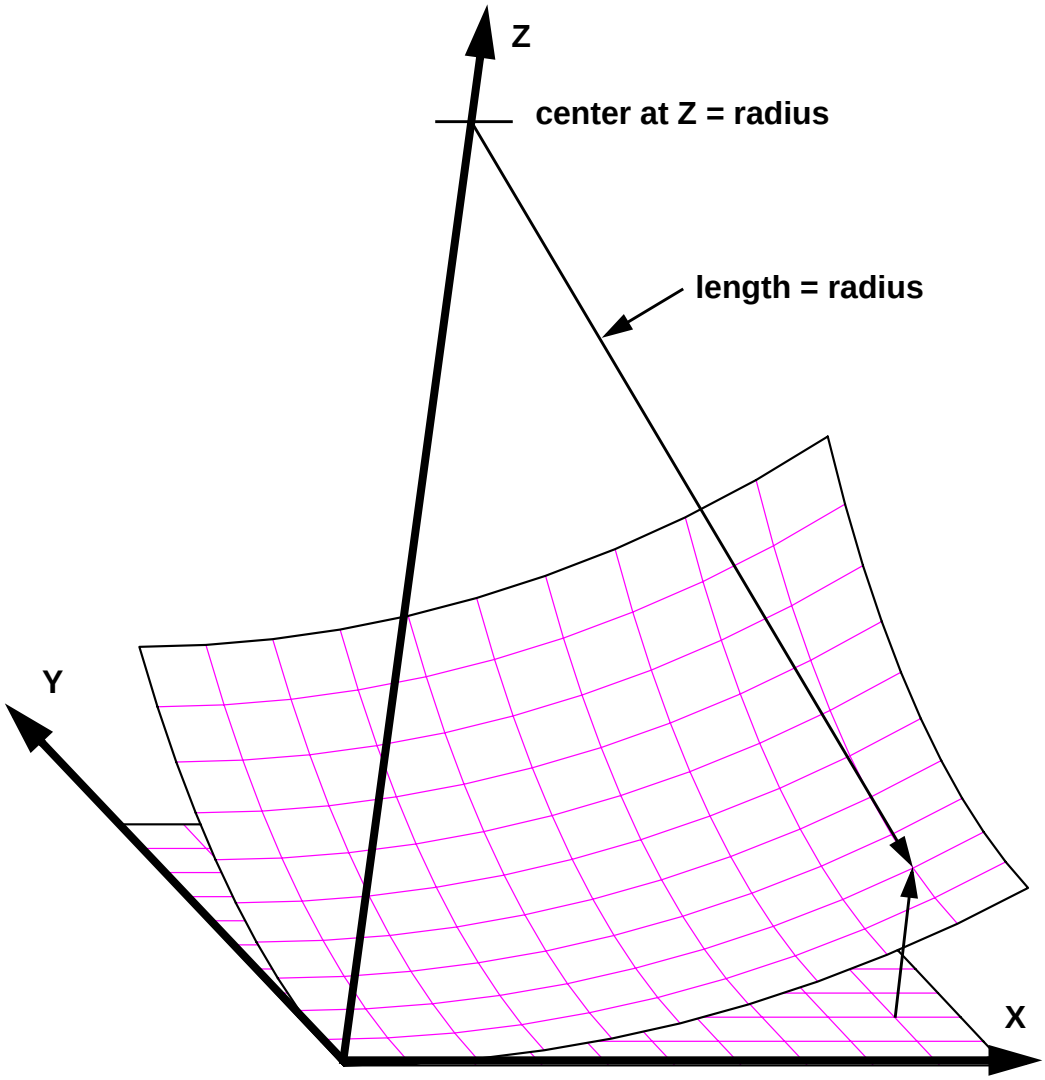

Figure 2. Illustration of WARP POINT transformation

a distance of *radius* above the X-Y plane. The total thickness is *thickness*. This command supersedes previous transformation commands.

If MAP is specified, the warped nodal positions (*xw*, *yw*, *zw*) are calculated by mapping the original 2D mesh onto a cylinder about the specified axis with a radius of curvature equal to *radius*. If XAXIS is specified, then the original X-coordinate remains at the same value. The generated Y and Z coordinates are calculated such that the distance from the generated node to the X-Z plane measured along the cylindrical surface is equal to the X coordinate of the node in the 2D mesh. If the Y-axis is specified, the X's and Y's are switched in the above discussion. The resulting 3D mesh will have an cylindrical angle of $x_{max}/radius$ radians if warped about the Y axis, or $y_{max}/radius$ radians if warped about the

X axis, where x_{max} and y_{max} are the maximum X and Y coordinates in the 2D mesh. A square mesh warped with the command "WARP YAXIS MAP 1 1" is shown in Figure 2.

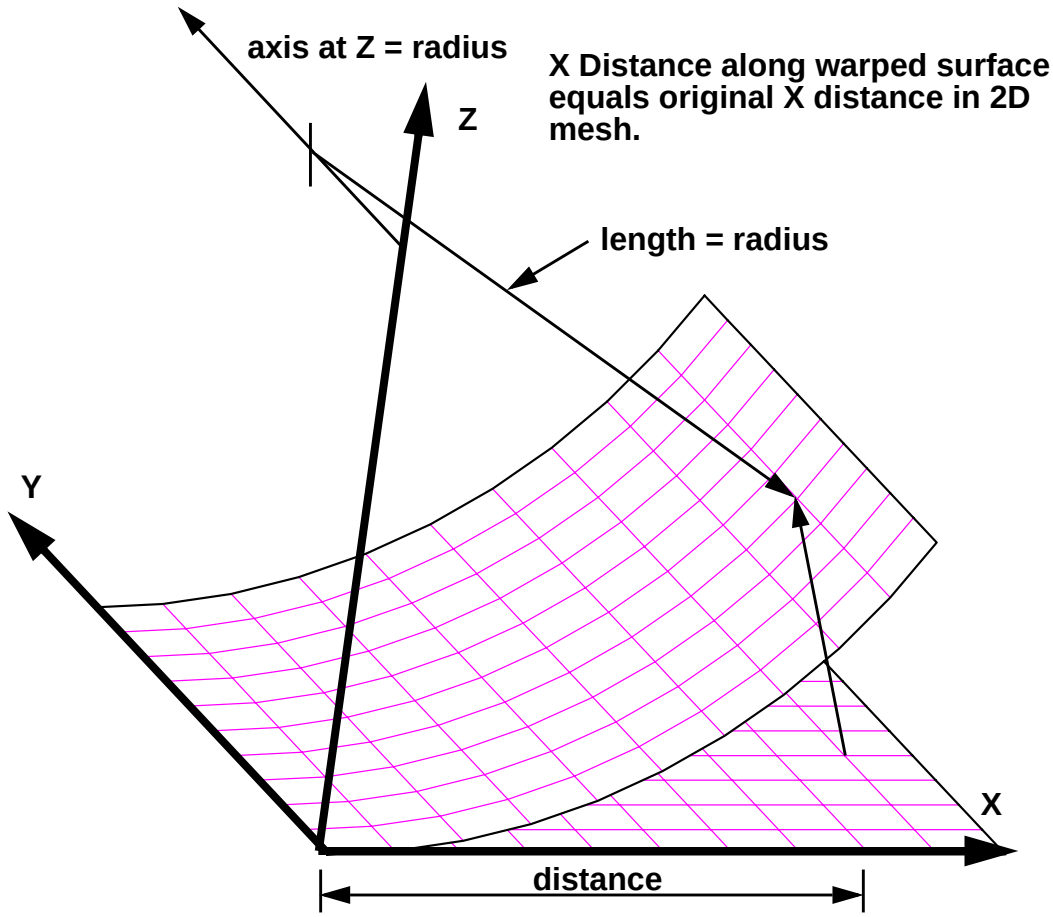

Figure 3. Illustration of WARP YAXIS MAP transformation

If VERTICAL is specified, the warped nodal positions xw, yw, zw) are again calculated by mapping the original 2D mesh onto a cylinder about the specified axis with a radius of curvature equal to *radius*. However, for this case, both the original X- and Y-coordinates remain at the same value and the Z-coordinate is calculated as:

$$z_w = radius - \sqrt{radius^2 - y_{2D}^2} \tag{1}$$

for an XAXIS warp. The *y* is replaced by *x* if a YAXIS warp is specified. A square mesh warped with the command "WARP YAXIS VERTICAL 1 1" is shown in Figure 2. Note that since the original X and Y coordinates remain the same in the transformed mesh, the elements elongate in the mapped mesh.

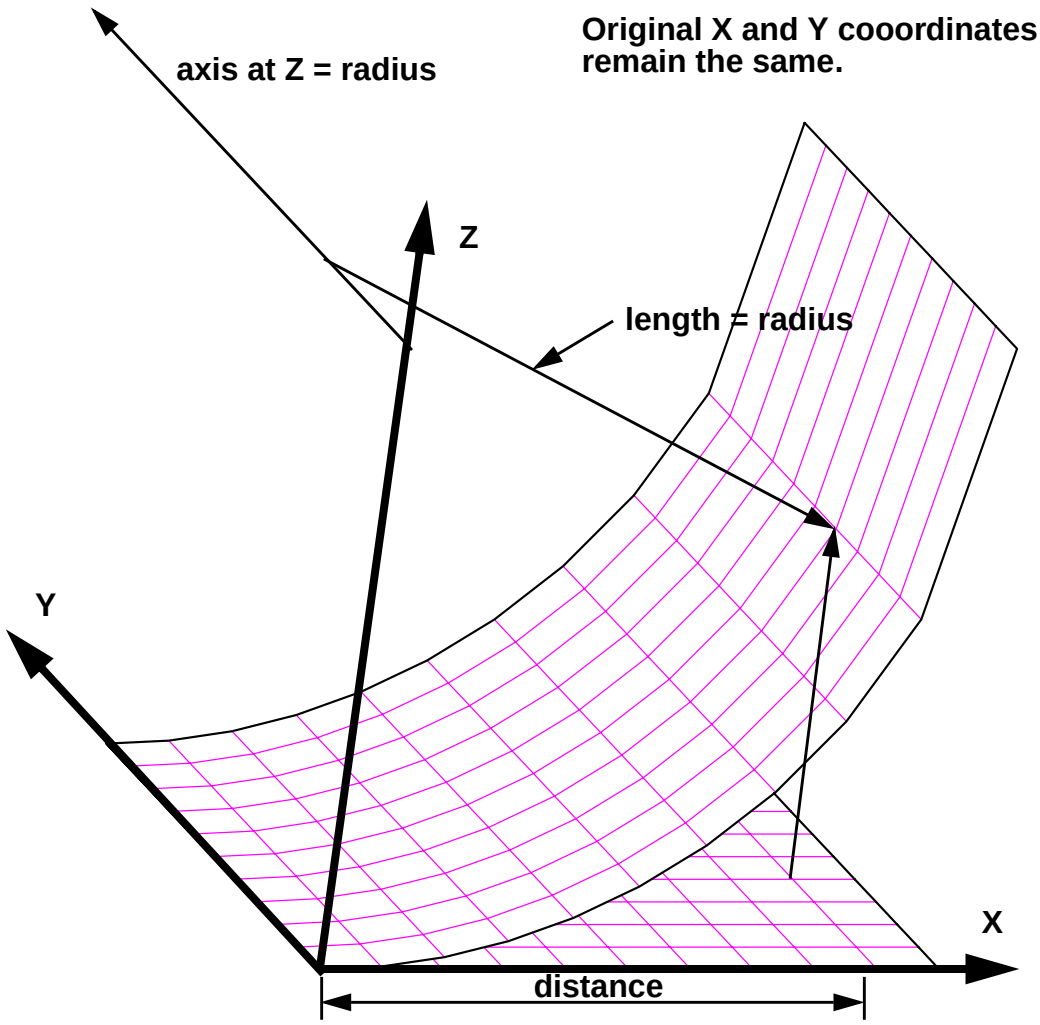

Figure 4. Illustration of WARP YAXIS VERTICAL transformation

WARP ELLIPSE, *thickness* <1.0>, *xfactor*, *yfactor*, *zfactor*

WARP ELLIPSE maps the 2D mesh onto an ellipsoidal surface defined by the equation:

$$\frac{x^2}{xfactor^2} + \frac{y^2}{yfactor^2} + \frac{z^2}{zfactor^2} = 1 \qquad (2)$$

The input X and Y-coordinates from the two-dimensional mesh remain the same and the Z-coordinate is calculated as:

$$z = zfactor \times \sqrt{1 - \frac{x^2}{xfactor^2} - \frac{y^2}{yfactor^2}} \qquad (3)$$

A square mesh warped with the command "WARP ELLIPSE 0.1 2 3 4" is shown in Figure 2. Note that xfactor and yfactor must be chosen with care to avoid calculating a negative square root in Equation 3.

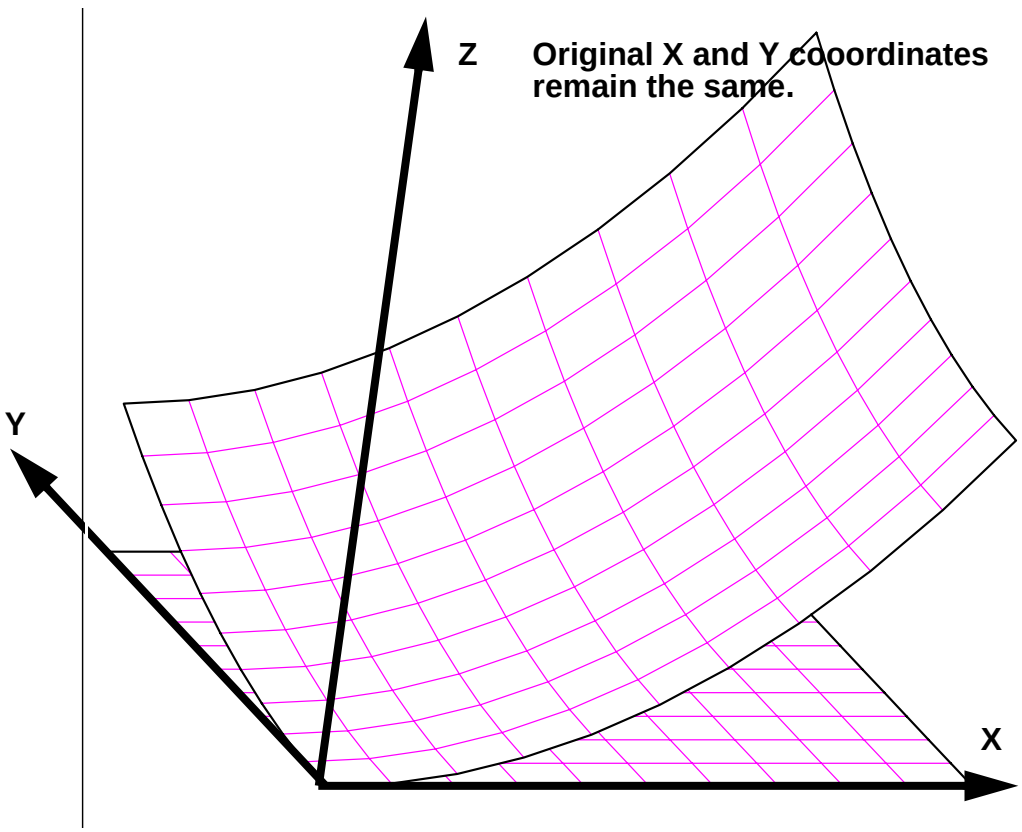

Figure 5. Illustration of WARP ELLIPSE transformation

SPLINE *thickness* <1.0>

SPLINE maps the 2D mesh onto a user-defined spline surface and sets the thickness of the generated mesh to *thickness*. The spline curved is defined in the X-Z coordinate plane. The spline surface is generated by (1) sweeping about the Z-axis if **SPHERICAL**, (2) sweeping along the Y-axis if **YSWEEP**, or (3) sweeping along the X-axis if **XSWEEP**. **SPHERICAL** is the default.

Following the SPLINE command line, *GENSHELL* enters the spline input mode in which the various spline options described below can be entered.

LINEAR—the spline data are input as Radius-Z data pairs, and the slopes at the end of the curves are linear slopes.

ANGULAR—the spline data are input as Theta(degrees)-Distance data pairs, where Theta is the angle between the X-Y plane and the line between the origin (X = Y = Z = 0) and the defined point, and Distance is the length of this line. The slopes at the end of the curves are relative to this curve.

FRONT—the curve data and slope specifications up to the next BACK, END, or EXIT command will be for the FRONT spline. The front surface Z values are greater (more positive) than the back surface Z values. NOTE: In *GENSHELL*, only the front spline data are used, the BACK keyword is retained so that spline definitions from *GEN3D* files can be used in *GENSHELL*.

BACK—In *GENSHELL*, only the front spline data are used, the BACK keyword is retained so that spline definitions from *GEN3D* files can be used in *GENSHELL*.

LEFT *slope*—the parameter *slope* specifies the slope of the spline curve at the LEFT end of the curve. The slope is measured in the same units specified in the ANGULAR or LINEAR command. If the slope is not specified, the end conditions of the curve will be set such that the second derivative is equal to zero which is the so-called natural cubic spline.

RIGHT *slope*—the parameter *slope* specifies the slope of the spline curve at the RIGHT end of the curve. The slope is measured in the same units specified in the ANGULAR or LINEAR command. If the slope is not specified, the end conditions of the curve will be set such that the second derivative is equal to zero which is the so-called natural cubic spline.

SPHERICAL—The spline surface is swept *about* the Z-axis to generate the surface onto which the two-dimensional mesh is mapped. This is the default.

XSWEEP—The spline surface is swept *along* the X-axis to generate the surface onto which the two-dimensional mesh is mapped. The X-coordinates in the generated mesh are the same as the X-coordinates in the original mesh. The spline surface is input as either Y-Z data pairs if LINEAR is selected, or as Theta(degrees)-Distance pairs in the Y-Z plane if ANGULAR is selected.

YSWEEP—The spline surface is swept *along* the Y-axis to generate the surface onto which the two-dimensional mesh is mapped. The Y-coordinates in the generated mesh are the same as the Y-coordinates in the original mesh. The spline surface is input as either X-Z data pairs if LINEAR is selected, or as Theta(degrees)-Distance pairs in the X-Z plane if ANGULAR is selected.

SCALE—Stretch the X-Y coordinates of the input mesh so that the X-Y extent of the output mesh matches the extent of the spline. That is, the 2D mesh will be stretched to fit the surface defined by the spline. This is the default.

NOSCALE—The X-Y coordinates of the 2D mesh are the X-Y coordinates of hte output 3D mesh. This option can only be used with the **LINEAR** spline.

EXIT or **END**—terminate spline input mode and return to general *GENSHELL*

command processing.

GENSHELL does not check the validity of the input spline curve; it is possible to generate a mesh with invalid elements if the spline curve crosses itself.

The following procedure is used to generate a three-dimensional shell mesh from a two dimensional mesh using the SPLINE option.

1. Generate the cubic spline parameters to fit the data entered by the user.

2. Calculate the cumulative chord lengths d_i of each segment of the spline. The chord length of each segment is calculated as the straight line distance between each of the entered data points. This is a good approximation for smooth curves, but it underestimates the distance for non-smooth curves. If the spline does not begin at the Z axis, the minimum distance of the spline is equal to the distance from the Z axis. Note that the minimum distance is equal to d_1, and the maximum distance is equal to d_{NS} where *NS* is the number of points defining the curve.

3. Determine the minimum and maximum extent of the input two-dimensional mesh. If SPHERICAL is specified, the extents are radii measured form the point X=Y=0; if XSWEEP or YSWEEP are specified, the extents are simply the minimum and maximum coordinates. If the spline curve starts at 0.0, the minimum extent of the two-dimensional mesh is also set to 0.0

4. The two-dimensional mesh is mapped onto the spline surface such that the maximum mesh extent (m_{max}) is mapped to the outside edge of the spline surface, and the minimum mesh extent (m_{min}) is mapped to the inside edge of the spline surface. This mapping defines a coordinate transformation where the two-dimensional mesh is uniformly stretched or shrunk onto the spline surface. Note that the distance is measured along the spline surface. If SPHERICAL is specified, both the X- and Y-coordinates are transformed; however, the generated three-dimensional node will have the same polar angle in the X-Y plane as the original two-dimensional node. If XSWEEP is specified, only the Y-coordinates are transformed; and if YSWEEP is specified, only the X-coordinates are transformed.

Figures 2, 2, and 2 show a square mesh mapped onto SPHERICAL spline surface, an XSWEEP spline surface, and a YSWEEP spline surface, respectively.

THICKNESS *thickness <1.0>*

THICKNESS sets the thickness of the generated shell elements to *thickness*. A mesh transformation command must be entered prior to this command. Subsequent mesh transformation commands override this value.

2.2 Mesh Orientation

MIRROR $axis_1$, $axis_2$,… <No Default>

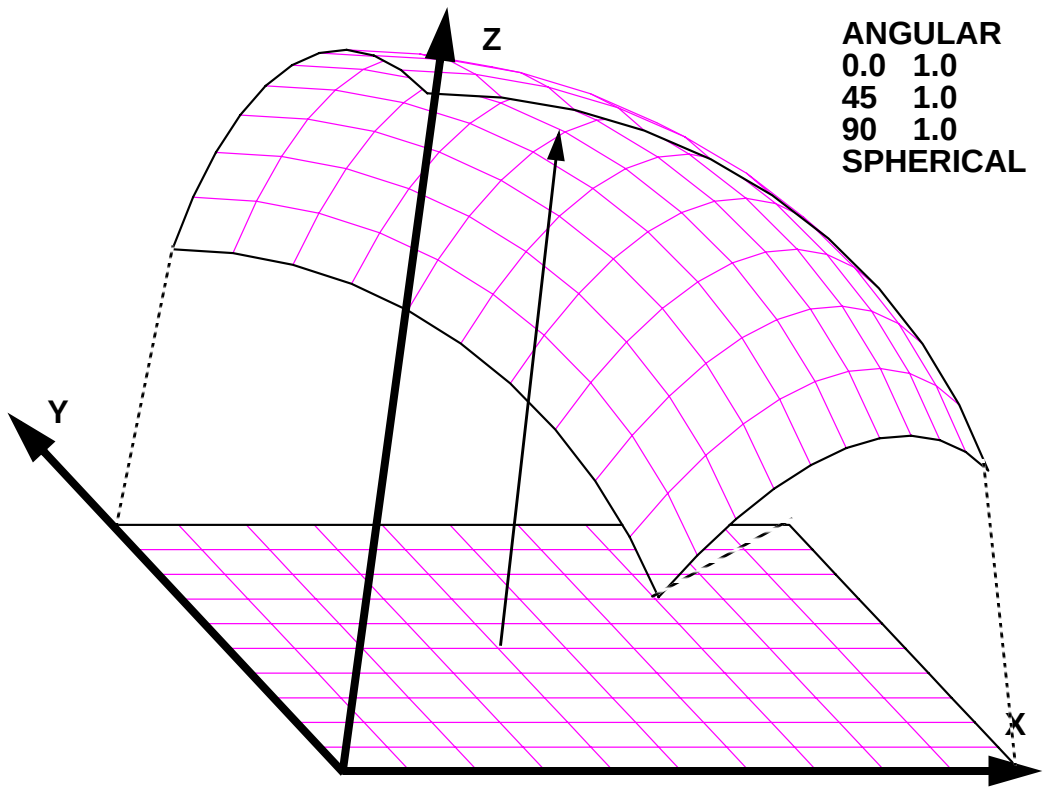

Figure 6. Illustration of SPLINE (SPHERICAL) transformation

MIRROR RESET <no reflections>

MIRROR causes the mesh to be reflected about a coordinate plane. Each *axis* parameter specifies which coordinates (X or Y or Z) will be modified. Reflections are performed after the mesh has been repositioned by the REVOLVE and OFFSET commands.

The MIRROR RESET command resets to no reflection.

Reflections are not cumulative, that is, if MIRROR X Y Y X is entered, only one X and Y reflection will be performed. The element connectivity and the sideset face numbering will be correctly reordered.

OFFSET [**ADD**] $axis_1$, $offset_1$, $axis_2$, $offset_2$, …
OFFSET [**ADD**] **ALL** *offset* <0.0>
OFFSET RESET <initial condition>
OFFSET *xoff* <0.0,> *yoff* <0.0,> *zoff* <0.0>

OFFSET specifies offsets to be added to the coordinates. If a REVOLVE command has been issued, the mesh is rotated before it is offset. The last form of the offset

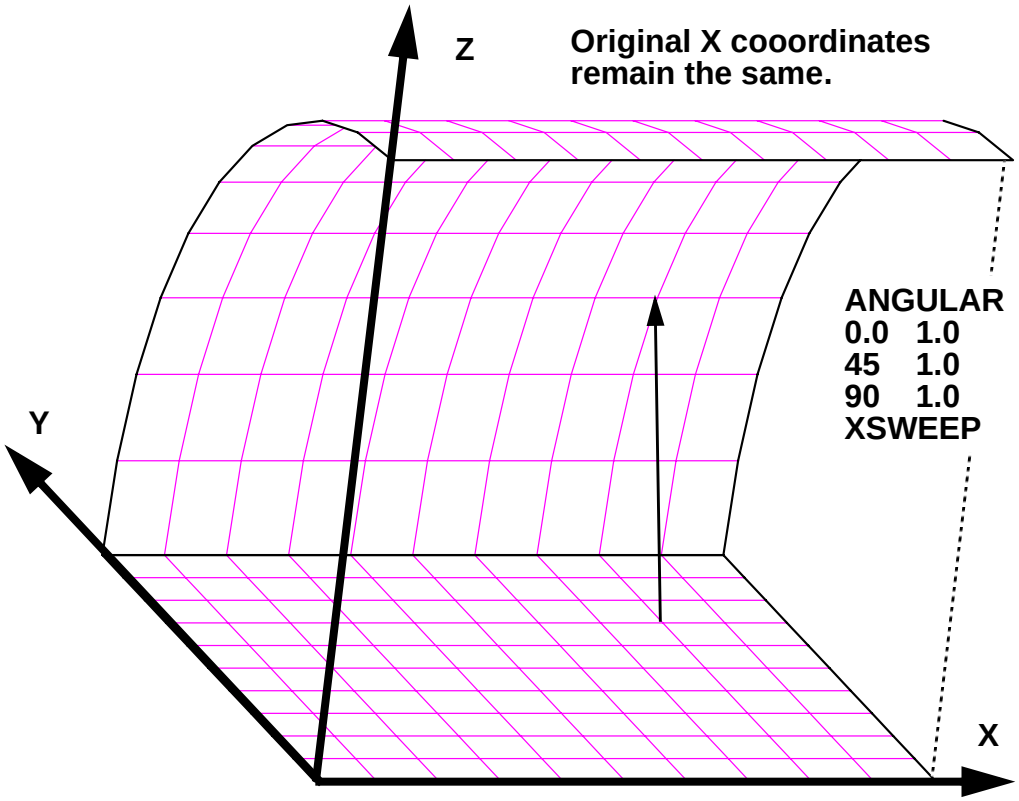

Figure 7. Illustration of SPLINE (XSWEEP) transformation

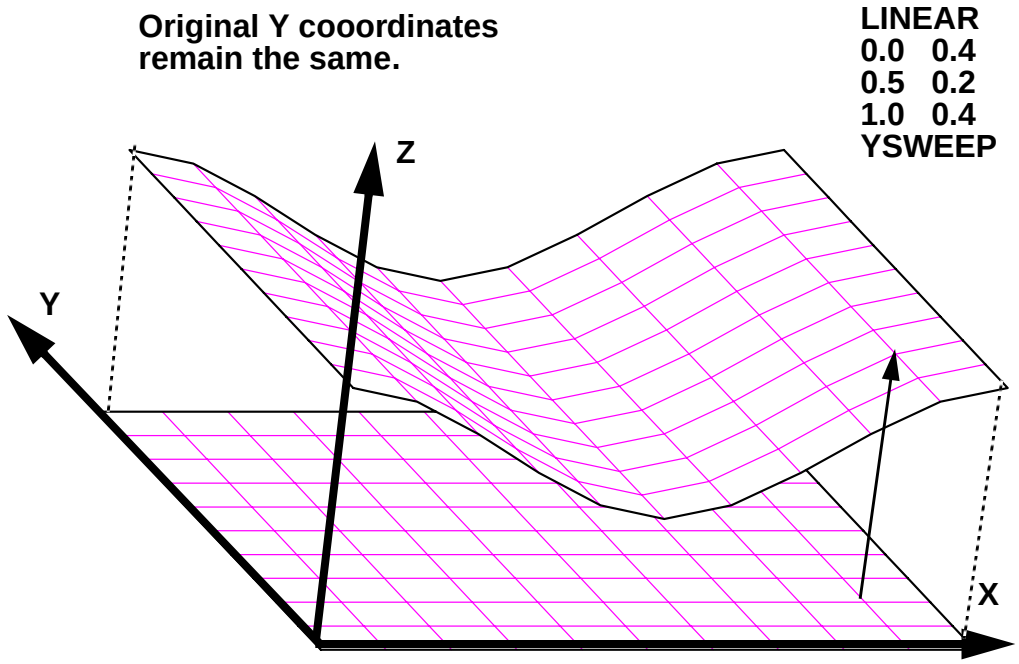

Figure 8. Illustration of SPLINE (YSWEEP) transformation

command is included to maintain compatibility with the offset command in *GEN3D.*

OFFSET ALL offsets all of the coordinates by the specified offset, and OFFSET RESET resets the offsets to zero.

If the optional keyword ADD is specified, offsets are cumulative, that is, if OFFSET ADD X 0.5 X 1.0 is entered, the X coordinates will be offset by 1.5. If ADD is omitted from the above line, the X coordinates will be offset by 1.0.

RANDOMIZE *axis*$_1$, *magnitude*$_1$, *axis*$_2$, *magnitude*$_2$, …
RANDOMIZE ALL *magnitude* <0.0>
RANDOMIZE RESET <initial condition>

RANDOMIZE specifies random offsets to be added to the coordinates. If a REVOLVE command has been issued, the mesh is rotated before it is offset. The specified coordinates are offset by a random amount in the range from -*magnitude* to +*magnitude*.

RANDOMIZE ALL randomly offsets all of the coordinates by the specified magnitude, and RANDOMIZE RESET cancels the offsets.

If there are contact surfaces in the mesh description, the nodes on both surfaces will be moved using a different random offset; therefore, if the two surfaces are initially in contact, it is almost certain that they will overlap after they are randomized.

REVCEN *xcen* < min X> *ycen* < min Y> *zcen* <min Z>

REVCEN sets the center of revolution for the REVOLVE command to the point (*xcen*, *ycen*, *zcen*).

REVOLVE *axis*$_1$, *ndeg*$_1$, *axis*$_2$, *ndeg*$_2$, … <last selection >
REVOLVE RESET <initial condition>

REVOLVE causes the mesh to be rotated. Each (*axis*$_i$, *ndeg*$_i$) parameter pair specifies an axis (X or Y or Z) and the number of degrees to rotate. The rotations are according to right-hand rule. The center of the rotation is specified by the REVCEN command.

Revolutions are cumulative and order-dependent; however, only one center of revolution (see REVCEN) may be specified. The REVOLVE RESET command resets to no rotation.

SCALE *axis*$_1$, *scale*$_1$, *axis*$_2$, *scale*$_2$, …
SCALE ALL *scale_factor* <1.0 >
SCALE RESET <initial condition>

SCALE causes the specified coordinates of axis *axis*$_i$ to be multiplied by the

scaling multiplier $scale_i$. The scaling multiplier must be greater than zero; the MIRROR command must be used with the SCALE command if a negative scaling multiplier is required. For example, SCALE X 2.54 followed by MIRROR X will scale the X coordinates by -2.54.

SCALE ALL multiplies all of the coordinates by the specified multiplier, and SCALE RESET resets to no scaling.

Scalings are cumulative, that is, if SCALE X 0.5 X 0.6 is entered, the X coordinates will be scaled by 0.3.

SHIFT [**ADD**] $axis_1$, $shift_1$, $axis_2$, $shift_2$, …
SHIFT [**ADD**] **ALL** *shift* <0.0>
SHIFT RESET <initial condition>
SHIFT *xoff* <0.0,> *yoff* <0.0,> *zoff* <0.0>

SHIFT is simply a synonym for the OFFSET command. See the description of the OFFSET command for more information.

ZERO $axis_1$, min_1, $axis_2$, min_2, …
ZERO ALL *min*
ZERO RESET <no automatic zeroing>

ZERO sets all $axis_i$ coordinates with an absolute value less than min_i equal to zero. If ALL is specified, the minimum value is applied to all three coordinates. The ZERO RESET command resets to no automatic zeroing. This command is used to zero nodal coordinates that should be equal to zero, but due to roundoff errors they have slightly nonzero values.

2.3 Modification or Creation of Material, Nodeset, or Sideset IDs

NODESETS|NSETS FRONT|BACK <no default>, set_id_1, set_id_2,…

NODESETS or NSETS defines front or back node sets with the given identifiers. The identifiers must be unique from existing node set identifiers and previously defined front and back node set identifiers. Back sets cannot be defined on a 360-degree rotation.

SIDESETS|SSETS FRONT or BACK *<no default>*, set_id_1, set_id_2,…

SIDESETS is equivalent to the NODESETS command except that it defines side sets.

CHANGE MATERIAL|NODESET|SIDESET|NSETS|SSETS *old_id new_id*

CHANGE changes the identification number *old_id* of a material block, nodeset, or sideset to *new_id*. The *new_id* must be unique, that is, CHANGE can only be used to change the identification number; it cannot be used to combine or delete identification numbers (use *GJOIN* to combine material blocks, sidesets, or

nodesets).

2.4 Information and Processing

END|EXIT

END or EXIT ends the command input and starts the mesh transformation.

HELP *command* <general help>

HELP displays information about the program command given as the parameter. If no parameter is given, all the command verbs are displayed. This command is system-dependent and may not be available on some systems.

LIST *option*

LIST displays information about the requested *option*. Valid options are: VARIABLES, VARS, and COMMANDS.

LIST {VARS|VARIABLES}

displays a summary of the database. The summary includes the database title; the number of nodes, elements, and element blocks; the number of node sets and side sets; and the number of each type of variable.

LIST COMMANDS

displays a list of the valid commands.

QUIT

QUIT ends the command input and exits the program immediately without writing an output database.

SHOW *command* <no parameter>

SHOW displays the settings of parameters relevant to the *command*. For example, the command SHOW REVOLVE displays information about the currently defined revolution.

SUMMARY

SUMMARY displays a short list of the command syntax for the mesh transformation, orientation, and modification commands.

2.5 Order of Transformation

Although *GENSHELL* commands may be entered in any order, the mesh transformation command is performed first followed by processing the orientation commands in the following order:

REVOLVE, OFFSET, MIRROR, RANDOMIZE, ZERO, SCALE

Processing of mesh transformation and orientation commands does not occur until an EXIT or END command is entered.

Unlike the transformation and orientation commands, the CHANGE operations are performed in the order they are entered, and at the time they are entered.

Intentionally Left Blank

3 GENSHELL Example Input and Output

A few examples of *GENSHELL* commands are shown below. In the examples below, lines beginning with the string "**GenShell>**" show the user input and the lines following are the *GENSHELL* output. Lines in *Italic* type indicate comment about the command or the output and will not appear during a run of *GENSHELL*.

The following example generates the mesh shown in Figure 2.

GenShell> spline 0.1

```
Valid Commands:
  LEFT      RIGHT     ANGULAR   TOP       FRONT     BOTTOM*   BACK*
  END       EXIT      LIST      HELP      SPHERICA  XSWEEP    YSWEEP
```

Spline Option > linear
Spline Option > front
Spline Option > 0.0 0.4
Spline Option > 0.5 0.2
Spline Option > 1.0 0.4
Spline Option > left 0.0
Spline Option > right 0.0
Spline Option > xsweep
Spline Option > show
Spline Option > end

```
Spline mesh, thickness =  .1000
```

GenShell> end

```
Database:  tst.g
Square mesh for genshell examples
Number of coordinates per node       =     3
Number of nodes                      =   121
Number of elements                   =   100
Number of element blocks             =     1

Number of node sets                  =     0
Number of side sets                  =     0

Total length of top spline=  1.07703E+00, Proportion=  1.07703E+00
```

The next example shows a simple translation followed by several reorientation commands.

GenShell> translate .01

```
 Translate mesh, thickness = 10.00E-3
```

GenShell> revolve x 30 y 30

```
 Rotation matrix for generated mesh:
      .8660     .0000    -.5000
      .2500     .8660     .4330
      .4330    -.5000     .7500
```

GenShell> randomize z .01 x .001

```
 Coordinate random factors =  1.00E-3   .00E-3 10.00E-3
```

GenShell> scale all 2.0

```
 Coordinate scale factors = 2.000 2.000 2.000
```

GenShell> mirror x

```
    New X = - Old X
```

GenShell> offset x 10 z -1

```
 Coordinate offsets = 10.00   .00 -1.00
```

GenShell> change material 1 10

```
 *** Material 1 changed to Material 10
```

GenShell> sideset front 100

```
 IDs for INPUT side sets:
 IDs for FRONT side sets:             100
 IDs for BACK side sets:
```

GenShell> nodeset back 10

```
 IDs for INPUT node sets:
 IDs for FRONT node sets:
 IDs for BACK node sets:               10
```

GenShell> zero x 1e-5 y 1e-5

```
 Minimum nonzero coordinates = 10.00E-6 10.00E-6   .00E-6
```

GenShell> end

4 References

[1] G. D. Sjaardema, “Overview of the Sandia National Laboratories Engineering Analysis Code Access System,” Technical Report SAND92–2292, Sandia National Laboratories, Albuquerque, New Mexico, January 1993.

[2] T. D. Blacker, “FASTQ Users Manual, Version 2.1,” Technical Report SAND88–1326, Sandia National Laboratories, Albuquerque, New Mexico, July 1988.

[3] A. P. Gilkey and G. D. Sjaardema, “GEN3D: A GENESIS Database 2D to 3D Transformation Program,” Technical Report SAND89-0485, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

[4] G. D. Sjaardema, “GREPOS: A GENESIS Database Repositioning Program,” Technical Report SAND90-0566, Revision 1, Sandia National Laboratories, Albuquerque, New Mexico, June 1993.

[5] G. D. Sjaardema, “GJOIN: A Program for Merging Two or More GENESIS Databases,” Technical Report SAND92-2290, Sandia National Laboratories, Albuquerque, New Mexico, December 1992.

[6] G. D. Sjaardema, “Aprepro: An Algebraic Preprocessor for Parameterizing Finite Element Analyses,” Technical Report SAND92–2291, Sandia National Laboratories, Albuquerque, New Mexico, December 1992.

[7] G. D. Sjaardema, “NUMBERS: A Collection of Utilities for Pre- and Postprocessing Two- and Three-Dimensional EXODUS Finite Element Models,” Technical Report SAND88–0737, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

[8] A. P. Gilkey, “BLOT—A Mesh and Curve Plot Program for the Output of a Finite Element Analysis,” Technical Report SAND88–1432, Sandia National Laboratories, Albuquerque, New Mexico, June 1989.

[9] W. C. Mills-Curran, A. P. Gilkey, and D. P. Flanagan, “EXODUS: A Finite Element File Format for Pre- and Post-processing,” Technical Report SAND87–2977, Sandia National Laboratories, Albuquerque, New Mexico, September 1988.

[10] L. M. Taylor, D. P. Flanagan, and W. C. Mills-Curran, “The GENESIS Finite Element Mesh File Format,” Technical Report SAND86–0910, Sandia National Laboratories, Albuquerque, New Mexico, May 1986.

[11] J. R. Red-Horse, D. P. Flanagan, and W. C. Mills-Curran, “SUPES Version 2.1: A Software Utilities Package for the Engineering Sciences,” Technical Report SAND90-0247, Sandia National Laboratories, Albuquerque, New Mexico, May 1990.

[12] B. Berliner, “CVS II: Parallelizing Software Development,” in *Proceedings of the Winter 1990 USENIX Conference*, 1990.

[13] “American National Standard Programming Language FORTRAN,” Technical Report ANSI X3.9–1978, American National Standards Institute, Inc., New York, 1978.

Intentionally Left Blank

A The GENESIS Database Format

The following code segment reads a *GENESIS* database.

```
C   --Open the GENESIS database file
      NDB = 9
      OPEN (UNIT=NDB, ..., STATUS='OLD', FORM='UNFORMATTED')
C   --Read the title
      READ (NDB) TITLE
C      --TITLE - the title of the database (CHARACTER*80)
C   --Read the database sizing parameters
      READ (NDB) NUMNP, NDIM, NUMEL, NELBLK,
     &   NUMNPS, LNPSNL, NUMESS, LESSEL, LESSNL
C      --NUMNP - the number of nodes
C      --NDIM - the number of coordinates per node
C      --NUMEL - the number of elements
C      --NELBLK - the number of element blocks
C      --NUMNPS - the number of node sets
C      --LNPSNL - the length of the node sets node list
C      --NUMESS - the number of side sets
C      --LESSEL - the length of the side sets element list
C      --LESSNL - the length of the side sets node list
C   --Read the nodal coordinates
      READ (NDB) ((CORD(INP,I), INP=1,NUMNP), I=1,NDIM)
C   --Read the element order map (each element must be listed once)
      READ (NDB) (MAPEL(IEL), IEL=1,NUMEL)
C   --Read the element blocks
      DO 100 IEB = 1, NELBLK
C      --Read the sizing parameters for this element block
         READ (NDB) IDELB, NUMELB, NUMLNK, NATRIB
C         --IDELB - the element block identification (must be unique)
C         --NUMELB - the number of elements in this block
C         --   (the sum of NUMELB for all blocks must equal NUMEL)
C         --NUMLNK - the number of nodes defining the connectivity
C         --   for an element in this block
C         --NATRIB - the number of element attributes for an element
C         --   in this block
C      --Read the connectivity for all elements in this block
         READ (NDB) ((LINK(J,I), J=1,NUMLNK, I=1,NUMELB)
C      --Read the attributes for all elements in this block
         READ (NDB) ((ATRIB(J,I), J=1,NATRIB, I=1,NUMELB)
 100 CONTINUE
C   --Read the node sets
      READ (NDB) (IDNPS(I), I=1,NUMNPS)
C      --IDNPS - the ID of each node set
      READ (NDB) (NNNPS(I), I=1,NUMNPS)
C      --NNNPS - the number of nodes in each node set
      READ (NDB) (IXNNPS(I), I=1,NUMNPS)
C      --IXNNPS - the index of the first node in each node set
C      --   (in LTNNPS and FACNPS)
      READ (NDB) (LTNNPS(I), I=1,LNPSNL)
C      --LTNNPS - the nodes in all the node sets
      READ (NDB) (FACNPS(I), I=1,LNPSNL)
```

```
C      --FACNPS - the factor for each node in LTNNPS
C   --Read the side sets
      READ (NDB) (IDESS(I), I=1,NUMESS)
C      --IDESS - the ID of each side set
      READ (NDB) (NEESS(I), I=1,NUMESS)
C      --NEESS - the number of elements in each side set
      READ (NDB) (NNESS(I), I=1,NUMESS)
C      --NNESS - the number of nodes in each side set
      READ (NDB) (IXEESS(I), I=1,NUMESS)
C      --IXEESS - the index of the first element in each side set
C      --   (in LTEESS)
      READ (NDB) (IXNESS(I), I=1,NUMESS)
C      --IXNESS - the index of the first node in each side set
C      --   (in LTNESS and FACESS)
      READ (NDB) (LTEESS(I), I=1,LESSEL)
C      --LTEESS - the elements in all the side sets
      READ (NDB) (LTNESS(I), I=1,LESSNL)
C      --LTNESS - the nodes in all the side sets
      READ (NDB) (FACESS(I), I=1,LESSNL)
C      --FACESS - the factor for each node in LTNESS
```

A valid *GENESIS* database may end at this point or after any point described below.

```
C   --Read the QA header information
      READ (NDB, END=...) NQAREC
C      --NQAREC - the number of QA records (must be at least 1)
      DO 110 IQA = 1, MAX(1,NQAREC)
         READ (NDB) (QATITL(I,IQA), I=1,4)
C         --QATITL - the QA title records; each record contains:
C         --   1) analysis code name (CHARACTER*8)
C         --   2) analysis code qa descriptor (CHARACTER*8)
C         --   3) analysis date (CHARACTER*8)
C         --   4) analysis time (CHARACTER*8)
  110 CONTINUE
C   --Read the optional header text
      READ (NDB, END=...) NINFO
C      --NINFO - the number of information records
      DO 120 I = 1, NINFO
         READ (NDB) INFO(I)
C         --INFO - extra information records (optional) that contain
C         --   any supportive documentation that the analysis code
C         --   developer wishes (CHARACTER*80)
  120 CONTINUE
C   --Read the coordinate names
      READ (NDB, END=...) (NAMECO(I), I=1,NDIM)
C      --NAMECO - the coordinate names (CHARACTER*8)
C   --Read the element type names
      READ (NDB, END=...) (NAMELB(I), I=1,NELBLK)
C      --NAMELB - the element type names (CHARACTER*8)
```

B Command Summary

Mesh Transformation (page 12)

TRANSLATE *thickness*

translates the 2D mesh to create the 3D shell mesh.

WARP POINT *thickness, radius*

maps the 2D mesh onto a spherical surface.

WARP XAXIS [MAP|VERTICAL] *thickness, radius*
WARP YAXIS [MAP|VERTICAL] *thickness, radius*

maps the 2D mesh onto a cylindrical surface.

WARP ELLIPSE *thickness, xfactor, yfactor, zfactor*

maps the 2D mesh onto an ellipsoidal surface.

SPLINE *thickness*

maps the 2D mesh onto a spline surface. Commands available are:
LINEAR|ANGULAR
FRONT|BACK
LEFT *slope*
RIGHT *slope*
SPHERICAL|XSWEEP|YSWEEP
EXIT|END

THICKNESS, *thickness*

sets the thickness of the generated 3D mesh.

Mesh Orientation (page 18)

MIRROR $axis_1$, $axis_2$, …
MIRROR RESET

causes the mesh to be reflected about the specified axes, or resets the mesh to no reflections.

OFFSET [ADD] $axis_1$, $offset_1$, $axis_2$, $offset_2$, …
OFFSET [ADD] ALL *offset*
OFFSET RESET
OFFSET *xoff, yoff, zoff*

specifies the coordinate offsets for the mesh, or resets the mesh to no offsets. Synonym for SHIFT command.

RANDOMIZE $axis_1$, $magnitude_1$, $axis_2$, $magnitude_2$, …
RANDOMIZE ALL *magnitude*
RANDOMIZE RESET

randomly offset nodes.

REVCEN *xcen*, *ycen*, *zcen*

sets the center of rotation for the REVOLVE command.

REVOLVE $axis_1$, $ndeg_1$, $axis_2$, $ndeg_2$, …
REVOLVE RESET

causes the mesh to be rotated, or resets the mesh to no rotations.

SCALE $axis_1$, $scale_1$, $axis_2$, $scale_2$, …
SCALE ALL *scale_factor*
SCALE RESET

causes the mesh to be scaled, or resets the mesh to no scaling.

SHIFT [ADD] $axis_1$, $offset_1$, $axis_2$, $offset_2$, …
SHIFT [ADD] ALL *offset*
SHIFT RESET
SHIFT *xoff, yoff, zoff*

specifies the coordinate offsets for the mesh, or resets the mesh to no offsets. Synonym for OFFSET command.

ZERO $axis_1$, min_1, $axis_2$, min_2, …
ZERO RESET

sets all $axis_i$ coordinates with an absolute value less than min_i equal to zero, or resets the mesh to no automatic zeroing.

Modification or Creation of Material, Nodeset, or Sideset IDs (page 22)

NODESETS|NSETS FRONT|BACK *set_id,…*

define nodesets on front or back of generated 3D mesh.

SIDESETS|SSETS FRONT|BACK *set_id, …*

define sidesets on front or back of generated 3D mesh.

CHANGE MATERIAL|NODESET|SIDESET|NSET|SSET *old_id, new_id*

change material, nodeset, or sideset identification numbers.

Information and Processing (page 23)

END|EXIT

terminate command input and start mesh processing.

HELP *command*

display information about the program command.

LIST *option*

display information about the program option.

QUIT

terminate command input and quit the program immediately.

SHOW *command*

display the settings of the parameters relevant to the specified command.

SUMMARY

display a short list of the command syntax for the mesh transformation, orientation, and modification commands.

Order of Orientation Processing (page 23)

Orientation commands are processed in the following order: REVOLVE, OFFSET, MIRROR, RANDOMIZE, ZERO, SCALE.

Intentionally Left Blank

C GENSHELL Details

Version:

The current version of *GENSHELL* is 1.6

Execution:

To execute *GENSHELL* on a *UNIX** system (with SEACAS), type:

genshell [-options *option*] *input_database output_database*

Input_database is the filename of the input *GENESIS* database.
Output_database is the filename of the output *GENESIS* database

Valid *options* are:

-executable = *alternate-executable*
to specify running a different version of *GENSHELL*,

-aprepro to pipe the input through the program aprepro [6],

-command=*single-line-command* to run grepos with just a single command given on the command line instead of interactively or in an input file.

-help to get a usage synopsis,

-VMS to indicate that the *EXODUS* file is in VAX/VMS binary format†, and

-IEEE to indicate that the *EXODUS* file is in IEEE binary format‡.

User input is read from the terminal keyboard (unless redirected using '<').
User output is directed to the terminal.

Execution Files:

The table below summarizes *GENSHELL* file usage.

Description	Unit	Type	File Format
User input	std input	input	ASCII
User output	std output	output	ASCII
GENESIS database	9	input	*GENESIS*
GENESIS database	10	output	*GENESIS*

All files must be connected to the appropriate unit before *GENSHELL* is run. Each database file is opened with the name retrieved by the EXNAME routine of the *SUPES* [11] library.

* UNIX is a registered trademark of UNIX Systems Laboratories Inc.
† Cray Unicos systems only
‡ Cray Unicos systems only

Source Code:

The *GENSHELL* source code is maintained in the SEACAS system which is managed by the Concurrent Version System (cvs) [12]. *GENSHELL* is written in ANSI standard FORTRAN-77 [13] with the exception of the following extension:

- Include files are used.

GENSHELL uses the following software package:

- the *SUPES* [11] package which includes dynamic memory allocation, a free-field reader, and FORTRAN extensions.
- the *SUPLIB* package which is an undocumented internal development library which includes *GENESIS* reading and writing routines, command parsing, string utilities, and other miscellaneous useful functions.

Availability:

GENSHELL and all other SEACAS codes are available on a licensed basis. The license agreements for these codes stipulate that (1) the software is to be used solely for internal purposes, (2) the codes are not to be distributed or transferred to any person without written permission, (3) the codes are to be used at a single site and should be copied only for necessary maintenance, development, or backup purposes, and (4) there should be a procedure, or site plan, in place for protecting the provisions of the license agreements.

For more information on obtaining *GENSHELL* or other SEACAS codes, contact:

Marilyn K. Smith
Computational Mechanics and Visualization Department
Department 1425
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-5800
(505) 844-3082, FAX: (505) 844-9297

Distribution

1	1400	E. L. Barsis
1	1401	J. R. Asay
1	1402	S. S. Dosanjh
1	1403	G. S. Davidson
1	1404	J. A. Ang
13	1425	J. H. Biffle & staff
50	1425	M. K. Smith
1	1431	J. M. McGlaun
1	1431	K. G. Budge
1	1431	J. S. Peery
1	1432	W. T. Brown
15	1434	D. R. Martinez & staff
1	1500	D. J. McCloskey
1	1501	C. W. Peterson
1	1502	P. J. Hommert
1	1511	J. S. Rottler
1	1511	D. K. Gartling
1	1511	M. W. Glass
1	1511	P. L. Hopkins
1	1511	M. J. Martinez
1	1511	P. A. Sackinger
1	1511	P. R. Schunk
1	1511	J. D. Zepper
1	1512	A. C. Ratzel
1	1513	R. D. Skocypec
1	1513	R. G. Baca
1	1513	R. E. Hogan, Jr.
1	1513	J. L. Moya
1	1551	W. P. Wolfe
1	1552	C. E. Hailey
1	1553	W. L. Hermina
1	1554	W. H. Rutledge
15	1561	H. S. Morgan & staff
14	1562	R. K. Thomas & staff
10	1562	G. D. Sjaardema
1	1832	J. M. Ramage
1	2565	S. T. Montgomery
1	6313	J. Jung
1	6411	A. S. Benjamin
1	6423	J. F. Dempsey
1	6513	D. S. Oscar
1	6522	J. D. Miller
5	7141	Technical Library
1	7151	Technical Publications
10	7613-2	Document Processing for DOE/OSTI
1	8523-2	Central Technical Files
6	8741	G. A. Benedetti & staff
1	8742	M. R. Birnbaum
1	8742	J. J. Dike
1	8742	L. I. Weingarten
5	8743	M. L. Callabresi & staff