Adam Griffith
Rice University

ORNL/FED
August 21, 2002

Introduction to GuiRecess v 1.0 - The RANT3D Input Java Graphical User Interface

This document describes the GuiRecess RANT3D Java input program. Topics include code setup, compatibility, use of the code, features, and future improvements.

Authors: Adam Griffith[1], Kate White[2], Mark Carter[3], Phil Ryan[3], David Swain[3]
Keywords: GuiRecess, GUI, Java, RANT3D, antenna modeling, fusion

[1] Rice University, Houston, Texas
[2] University of Tennessee, Knoxville
[3] Oak Ridge National Laboratory, Fusion Energy Division

TABLE OF CONTENTS

Introduction 3
Java 3-6
What Do I Need and Where Can I Get It? 4
How Do I Use Java? 4-6
Getting Started With GuiRecess 6-11
Setting Up the Program 6-7
Navigation and Orientation 7-8
The Workspace 8
Recesses 8-9
Current Elements 9-11
Output 11
Exiting the Program 11-12
Using the WireFrame Viewer 12
Example of Use 13-14
Future Improvements to the Program 14-15
Figures 16-21
Acknowledgements 22
References 22
Resources 22-23

LIST OF FIGURES

Figure 1 - Startup Information 16
Figure 2 - Workspace Dialog 16
Figure 3 - Empty Recess Window 17
Figure 4 - Recess Dialog 18
Figure 5 - Current Dialog 1 18
Figure 6 - Current Dialog 2 19
Figure 7 - Full Recess Window 20
Figure 8 - JPEG File Output Dialog 20
Figure 9 - Output Files, 3D View, and Exit 21

Figure 10 - 3D Recess Viewer 21

INTRODUCTION

RANT3D is a physics code that is used to model RF antennas used in fusion tokamaks. The input for the RANT3D code primarily consists of two files, *recess_input* and *current_element_input*. The function of these files is to provide the geometry of a series of recesses and current elements or straps and their corresponding parameters that model an antenna. The code is quite complex, and the inputs can be quite tedious to write out for large, complex antennas such as the RF antenna in NSTX. In order to make the code easier to use, and therefore more accessible, the GuiRecess program was written. It is a Graphical User Interface (GUI) written in Java. It allows a RANT3D user to draw an antenna setup on any computer using only a mouse and a few keystrokes. The drawing and inputs are then formatted by the program into files that RANT3D can use. By being able to visualize the antenna on a screen, the likelihood of errors, time to create inputs, and frustration are all greatly reduced. GuiRecess is a relatively simple program to use and can be used on any computer. Although it cannot run RANT3D itself, the program makes it much easier for a user to set up a RANT3D test run. Keep in mind that, like all good programs, GuiRecess is always being upgraded and added to. Every effort will be made to make this guide as up-to-date as possible, but sometimes items in this guide may not be correct or may not apply to the current version of GuiRecess. The guide should be correct for GuiRecess version 1.0, Build 08.19.2002.

The notation used in this guide is intended to assist readers in learning how to use the program. While most of the text will appear in the font being used now, there will be two cases when it will not. When a filename is mentioned it will be in *italics*. For example, the GuiRecess program file is *GuiRecess.jar*. If a command to be typed in is being used, it will be in bold, 10-point **`Courier New`**. For example, the command to run GuiRecess when typed at the command prompt is **`java -jar GuiRecess.jar`**.

Questions may be directed to any of the authors.

JAVA

Java is a platform-independent programming language developed by Sun Microsystems. It allows software developers to write a single piece or package of code that can be run on any type of machine, regardless of the type of machine that the code was written on. For example,

code written by a developer using Linux can be run in Windows, Mac OS, or Unix. Although on occasion there is a situation that prevents a piece of code from being portable, these instances are rare and usually can be avoided by a skilled programmer. GuiRecess was written in Java due to its high portability, allowing RANT3D users to prepare input files on the machines that they are most comfortable with. Additionally, because Java is such a widely accessible language, someone with knowledge of Java can easily edit GuiRecess, perhaps to add or take away a feature. Although it would be preferred that GuiRecess be used as-is, to prevent the unwitting introduction of errors, the option of the user editing the code increases flexibility in the program. Sun regularly updates the Java standard, but backward compatibility is usually retained. For reference, this release of GuiRecess was written using the Java 2 Standard Edition (J2SE) v. 1.3.1, although Sun has now released a beta version of J2SE 1.4.1.

WHAT DO I NEED AND WHERE CAN I GET IT?

The only thing that is needed to run GuiRecess, other than the code itself, is the Java Runtime Environment (JRE). While Java code is platform-independent, the JRE is platform-specific. The JRE provides an interpreter between the code and the computer, which allows the same code to be used on different computers without differences in functionality. The JRE is available from Sun for Windows, Linux, Solaris/SPARC, and Solaris/Intel at http://java.sun.com. The Java platform comes pre-installed on Apple's Mac OS X. It is also available for OS X at http://www.apple.com/java and an older version of Java is available for Mac OS Classic (Version 8 and 9) at http://developer.apple.com/java/classic.html. It is advisable to use the most recent releases of the JRE that will work with your computer. Since the JRE is free to download, the only cost for using the most recent software is the time required to download and install it.

While the JRE is all that is needed to run GuiRecess, it might be advisable to download the Java Software Development Kit (SDK), which is also free from Sun's Java web-site. The JRE is packaged with the SDK, allowing Java code to be executed; however, the SDK itself allows the code to be edited and recompiled as well. Thus, if you know you will be editing the code, it is better to download the SDK from the start. The download for the SDK is large (~30 MB), but will not take long with a high-speed internet connection.

For installation and setup of the JRE or SDK, please visit http://java.sun.com.

HOW DO I USE JAVA?

Java is relatively simple to use. Generally, programs are run either as applets or as standalone programs. An applet is a program that runs as part of a web page, while a standalone program operates much like a normal non-Java program. GuiRecess is currently set up as a standalone program and, due to its nature, there are no plans to prepare an applet version of it.

The code for Java programs comes in three different forms. First, programs can come as source files. These files are named **.java* and cannot be executed. They contain the actual Java code as written by the developer. If you wish to see the code or edit the code, then the **.java* files are the ones to look at. Java programs can also come as **.class* files. These are output by a Java compiler, and there usually are one or two **.class* files for each **.java* file. Thus, if a program consists of 10 **.java* files, there may be 20 **.class* files, and maybe many more. There is usually one **.class* file that is responsible for the running of a program. Finally, Java programs may come as a single **.jar* file. This is a Java archive file, which is a compressed version of the whole program. While the command for executing this file is different from the execution of a **.class* file, the resulting program will be the same.

Programs are compiled and run using a command line in most operating systems. A few examples of possible commands are shown below. If a user has a Java program called HelloWorld, it can be compiled using:

```
javac HelloWorld.java
```

The command **`javac`** tells the Java SDK to compile the *HelloWorld.java* file. If that file has dependencies on other **.java* files, then those will be compiled as well. Additionally, to ensure that all files in a directory are compiled the command below can be used.

```
javac *.java
```

The **`javac`** command will cause several **.class* files to be produced. If the source files contain no errors, the program can be executed using the next command.

```
java HelloWorld
```

Notice that there is no file extension used in this command. Replacing **`HelloWorld`** with **`HelloWorld.java`** or **`HelloWorld.class`** will result in an error. The JRE will know what to do as long as the HelloWorld.class file is present. In fact, once the **.class* files have been produced, there is no real need to keep the **.java* files in the same directory. Keep in mind, however, that should the program need to be recompiled or edited, the **.java* files are necessary.

If a program has been compiled into **.jar* format, the command to execute the program differs. The command for executing *HelloWorld.jar*, for instance, would look like this:

```
java -jar HelloWorld.jar
```

The **`-jar`** tells the java command that a **.jar* file is being executed. Notice that executing a **.jar* file requires the presence of the **`.jar`** file extension. Viewing the contents of and extracting the contents of a **.jar* file are handled by the next two commands. These would be useful if you are given a **.jar* file but wish to know or get at the contents of the program.

```
jar tf jarfile.jar
jar xf jarfile.jar
```

The above three types of commands (**`javac`**, **`java`**, **`java -jar`**) should be enough to allow you to compile and run programs. For more information on working with Java, see Sun's Java website.

GETTING STARTED WITH GUIRECESS

SETTING UP THE PROGRAM

Setting up GuiRecess requires very little effort. The program will be delivered as a **.jar* file named *GuiRecess.jar*, along with a *README.txt* file. Place these files in a directory and the setup is complete. If desired, the program files can be extracted using the commands mentioned above. For reference, the source files required to recompile the code are listed below, with the resulting **.class* files:

- *CurrentDialog.java* → *CurrentDialog$1.class, CurrentDialog$2.class, CurrentDialog.class,*

CurrentDialog$DegChgListener.class, CurrentDialog$RadioListener.class, CurrentDialog$SliderListener.class

- *CurrentInfo.java → CurrentInfo.class*
- *EasyReader.java → EasyReader.class*
- *EasyWriter.java → EasyWriter.class*
- *FileDialog.java → FileDialog.class*
- *GRBox.java → GRBox.class*
- *GuiRecess.java → GuiRecess$1.class, GuiRecess$2.class, GuiRecess$3.class GuiRecess$4.class, GuiRecess$5.class, GuiRecess.class, GuiRecess$Recess.class, GuiRecess$Recess$drawPanel.class*
- *Matrix3D.java → Matrix3D.class*
- *RecessDialog.java → RecessDialog$1.class, RecessDialog.class*
- *RecessInfo.java → RecessInfo.class*
- *ThreeD.java → FileFormatException.class, Model3D.class, ThreeD.class*
- *WorkspaceDialog.java → WorkspaceDialog$1.class, WorkspaceDialog.class*
- *XlocDialog.java → XlocDialog.class, XlocDialog$SliderListener.class*

Two files are additionally included inside the *GuiRecess.jar* file. These files are the *mainClass* file used in compiling the *GuiRecess.jar* file, and *README.txt*, which gives information about the program and how to work with **.jar* files.

NAVIGATION AND ORIENTATION

GuiRecess follows the standard navigation format for most programs. In all windows, the TAB key on the keyboard cycles through all of the available buttons and fields in a window. SHIFT + TAB results in reverse cycling through items. To activate a button using the keyboard, cycle through to that button and press the SPACEBAR. Additionally, special keystrokes have been designated for each button. These keystrokes consist of ALT+"x", where "x" is a special letter for the command, which may be found underlined on the button. For example, a button labeled E<u>x</u>it may be activated by holding down the ALT key and pressing X. Of course, navigation can be handled with the mouse as well.

Orientation in GuiRecess and RANT3D can become complicated. In 2d Java applications, the x- and y-directions refer to right and up, respectively; thus this is the native

orientation used in GuiRecess. In RANT3D, however, x is the radial direction, y is the poloidal or up direction, and z is the toroidal or right direction. As much as possible, x and y terminology will be used to explain orientation in GuiRecess. The terms radial, poloidal, and toroidal will be used when discussing RANT3D orientation. Most users will not need to concern themselves with the details of the orientation unless they are concerned about specific asymmetries in the plasma response. The general orientation is one of looking at the antenna from inside the plasma.

THE WORKSPACE

When the program is first entered, two windows will open. One window controls saving the input files for RANT3D, as well as exiting GuiRecess. The other is an informational button. When the information button is closed, the Workspace Dialog Window will open.

The Workspace Dialog Window (Fig. 2) creates a general area in which to create an antenna. This dialog accepts different values for the depth, width, and height of the region, where depth is in the tokamak's radial direction, width is in the toroidal direction, and height is in the poloidal direction. There are also inputs for the x- and y-origin placement. This refers to the location of the lower, left-hand corner of the workspace. Finally, there is an input for a comment that describes the nature of the workspace. The FINISH button creates the workspace in a new window. RESET clears all input fields, and CANCEL exits the Workspace Dialog Window and kills the GuiRecess program.

RECESSES

Once the Workspace Dialog Window is closed, the workspace will appear in a new window. Although the workspace may be used in the same manner as a recess, it is *not* a recess and will <u>not</u> be written to output. Rather, this vacuum region between the plasma and the first wall will be automatically configured by RANT3D when the plasma impedance file, *impede.plt*, is processed.By default, a new workspace or a new recess window is initially in "Recess Mode", as indicated below the CHANGE MODE button. This means that the window is set to create a new recess. Creating a recess can be done in one of two ways. The SAME SIZE RECESS button creates a new recess attached to the back of the current recess with the width and height of the current recess. The second method of creating a recess is to draw it on the window's grid. Simply click where the upper left corner of the recess is to be, and drag to the point

where the lower right corner should be. As you draw, a green box will show the boundaries of the recess. If the upper left corner is not the first corner drawn, then a valid recess will not be produced. A third method of creating a recess will be implemented in a future version of GuiRecess that allows a recess to be created using a dialog that accepts keyboard input to gather the position, width, and height of the recess, rather than using the mouse for that information.

The grid in the window is such that one pixel corresponds to one millimeter, which is true for all measurements of distance on the screen in GuiRecess. There are two pairs of numbers at the bottom of the window that can be used to monitor the cursor location at any time. The first pair shows the local coordinates, the coordinates based on the lower left corner of the window being (0,0). The second pair shows the "true" coordinates. These coordinates are relative to the workspace's lower left corner, which need not be (0,0). For instance, if the workspace's lower left corner, as input in the Workspace Dialog, is (-100,-100), then the point on the workspace 250 pixels to the right and 350 pixels up actually has the "true" coordinates (150,250). The upper left corner of the recess or workspace window will display the "range", or depth, of the workspace/recess. When a recess or current element is being drawn in the box, numbers will appear showing the true position of the upper left corner of the box, the cursor position, which is also the lower right corner of the box, the box width, and the box height. These are here to assist in drawing the box to the proper size.

When SAME SIZE RECESS has been pressed or a recess has been drawn on the grid, a new window, the Recess Dialog, will appear. The Recess Dialog gathers the remaining input necessary to create a new recess. It has fields for a comment to distinguish the recess from others in the output, for the depth of the recess, and for the poloidal (mmax) and toroidal (nmax) modes. All numbers must be positive integers. Pressing FINISH will close the dialog and a new recess window will appear. The green box that was drawn in the previous recess window or workspace will remain. If SAME SIZE RECESS was pressed, then a green box should be visible around the previous window's edges. At this point it will be possible to continue drawing recesses or current elements.

To remove a recess, press the REMOVE RECESS button. This will remove not only the specified recess, but also will remove any recesses and current elements that have been drawn in the specified recess. Make sure of which recesses you wish to remove before using REMOVE RECESS.

CURRENT ELEMENTS

Current elements can be drawn into any recess window, including the workspace. Because recess windows are in "Recess Mode" by default, the CHANGE MODE button must be pressed in order to draw current elements. When this button is pressed, the label saying "Recess Mode" will be changed to say "Current Mode". Once this is done, a current element may be made. Drawing in the window is currently the only method for making a current element; however, a future version of the program will allow a dialog-based input as well.

Current elements are drawn in the same way as recesses, except that the box shown while drawing will be red, not green. Drawing the current element will cause a small dialog to be displayed. This dialog provides a slider bar that selects the radial positioning of the current element. The range of possible values is the same as the range of depth for the recess that the current element is drawn in. Thus, if a current element is drawn in a recess with depth 5 to 10, the strap can be given a radial position from 5 to 10. With this version of the code, drawing a second current element inside the same recess will automatically set the depth of the new current element to be the same as the first current element. Note that multiple current elements in any single recess must all have the same radial position. Different radial positions for current elements can be specified by creating other recesses with the appropriate radial range, either by drawing the recesses or using the SAME SIZE RECESS button.

When the radial position has been selected and the dialog has been finished, a new dialog will be opened. This dialog allows the user to input a comment describing the current element, the amount of current in amps, the phase of the current in either radians or degrees, the phase velocity along the element, and the poloidal location of ground on the element. The dialog also displays the width, height, poloidal (yc) and toroidal (zc) location of the center of the element, the selected phase as converted to radians, if necessary, and the tilting of the element in degrees. A tilted element will not appear to be tilted in the GuiRecess interface, but the rotation will be performed about the center of the current element when the RANT3D code processes the resulting *current_element_input* file. To input the phase of the current element,

first select the button marked "radians" or "degrees", and then input the desired phasing in the field to the right. As the phase is typed in, the value in radians or degrees will be converted to radians and displayed in the field below. The location of ground may be selected by using the slider bar at the bottom of the dialog, with values ranging the entire height of the current element.

When FINISH is pressed in the second dialog, it will close and the current element that was drawn in the recess window will be filled with an orange/copper color. The recess window provides a REMOVE CURRENTS button. This button will remove *all* current elements from the active window, but it will not remove current elements in other windows. In this version of the program, there is no way to specify individual current elements to remove.

OUTPUT

GuiRecess is capable of providing four outputs. The first output creates a JPEG image of a specified recess area. If an image of a particular recess is desired, simply press the JPEG IMAGE CAPTURE button on that recess's window. A dialog will appear requesting the name to save the image under. The dialog will add the ".jpg" file extension automatically, so do not enter it in the window. This will save the file in the local directory. A future version of the program will allow more flexibility in the file destination and image type.

The next forms of output are the true purpose of the program. When the program was started initially, a window was created that has two buttons marked WRITE RECESS_INPUT and WRITE CURRENT_ELEMENT_INPUT. These buttons convert the recesses and current elements that have been created in the program into the two input files used by RANT3D and save them to the local directory. Once the files have been saved, they should be ready to use; however, it is possible to edit them in any text document if fine-tuning is desired. When *recess_input* and/or *current_element_input* are written, a *recess.obj* object file is also written to the active directory. This file can be used with the WireFrame program that comes both as a demo with the Java SDK and packaged with GuiRecess; however, the GuiRecess program also uses this file to display a 3D representation of the recesses and current straps. These files, *recess_input*, *current_element_input*, and *recess.obj*, can only be written once during each execution of the program. If a user saves one of these files and then wishes to edit it, the editing cannot be done inside the same instance of GuiRecess. Additionally, each time GuiRecess is started,

these three files are overwritten. Therefore, **make sure to back up these files if you wish to keep them before you start GuiRecess**.

EXITING THE PROGRAM

Exiting GuiRecess can be performed in three different ways. The first method is to remove the workspace, either by pressing REMOVE RECESS in the workspace window or canceling the Workspace Dialog. The second method is to close the main window that writes the RANT3D input files. Finally, there is an EXIT button on the same window that closes the program. Keep in mind that if the program is closed before the *recess_input* and *current_element_input* files are written, all work will be lost.

USING THE WIREFRAME VIEWER

The Java 2 Standard Edition SDK comes with a demo wireframe viewer. This viewer has been incorporated into GuiRecess so that the final geometry may be observed. The viewer may be activated from the same window that writes the *recess_input* and *current_element_input* files. There are two buttons below the buttons that write the files. The first is 3D VIEW ON; the second is 3D VIEW OFF. These buttons may be used once one or both of the input files have been written. Keep in mind that once the 3D viewer has been turned on, no additional recesses or currents can be added to the *recess.obj* file. This means that a user cannot press the write the *recess_input* file, view the recesses, and then write *current_element_input* and view both the currents and recesses. Hopefully a later version of the code will allow for more flexibility.

Pressing the 3D VIEW ON button causes a window to appear. This window will have a wireframe representation of the recesses and/or current elements in the program, depending on which of these has been written. Recesses are represented by boxes; current straps are represented by cross-hatched rectangles with short tabs representing the feeders. Clicking and dragging the mouse will cause rotation of the object. Because this is a simple wireframe, it is possible for optical illusions to occur that make the image look wrong. If this happens, spin it around some to reorient your vision. If it still appears wrong, this may be an indication that the recesses and currents you have drawn are incorrect. However, this function was implemented in a "quick and dirty" fashion and is not intended to be a true representation of the antenna. It is simply intended to provide a quick check on the geometry. Compare the 3D view to the output files to be sure if you think that the 3D view is incorrect.

As mentioned earlier, the viewer is also available as a Java demo from Sun. To use the *recess.obj* file with the Java demo, edit one of the *example*.html* files provided in the demo. Change the tag

`<param name=model value=models/object.obj>`

where **`object.obj`** is the name of an object file, to read

`<param name=model value=(directory)/recess.obj>`

where **`(directory)`** is the directory where the *recess.obj* file is relative to the *example*.html* file.

EXAMPLE OF USE

Below is a script of an actual use of GuiRecess that demonstrates the general commands needed to run the program. The script comes from the program's use on a machine running Red Hat Linux 7.2. You will notice a slight departure from the normal formatting. All of the text is in 10 point `Courier New` to indicate that it is displayed on the computer screen. Items in ***bold and italics*** were input by the user. Items not in `bold and italics` were displayed by the computer.

```
Script started on Tue Aug 20 08:20:21 2002
[griffith@feddual griffith]$ cd GuiRecess/
[griffith@feddual GuiRecess]$ ls
GuiRecess.jar  README.txt
[griffith@feddual GuiRecess]$ java -jar GuiRecess.jar &
[1] 1817
[griffith@feddual GuiRecess]$
[1]+  Done                    java -jar GuiRecess.jar
[griffith@feddual GuiRecess]$ ls
adamversion  backup  current_element_input  GuiRecess.jar  README.txt
recess_input recess.jpg recess.obj
[griffith@feddual GuiRecess]$ head current_element_input
#
enter feeder type (0 constant, 1 standing, -1 stop input
and traveling wave parameter (0.0 gives half standing wave)
0 0.00
enter distance from plasma to strap in meters, plasmad
0.0200
enter amp, tphaz, yc, zc, theta (theta in degrees)
1.0000 0.0000 0.2960 0.1980 90.0000
enter width, height, phazi, phazs
0.2360 0.4440 0.0000 0.0000
[griffith@feddual GuiRecess]$ head recess_input
# this is recess number 1
enter range for this recess, closest x value first (a,b)
0.0050 0.0550
enter height (poloidal) and width (toroidal) of cavity, (ay,bz)
```

```
0.6000 0.8000
enter periodic recess coordinates for lower left corner (qcy, qcz)
0.0000 0.0000
enter boundary condition type for poloidal direction (yboundary)
enter either waveguide or periodic
waveguid
[griffith@feddual GuiRecess]$ head recess.obj
v 0.0050 0.0000 0.0000
v 0.0050 0.0000 0.8000
v 0.0050 0.6000 0.8000
v 0.0050 0.6000 0.0000
v 0.0550 0.0000 0.0000
v 0.0550 0.0000 0.8000
v 0.0550 0.6000 0.8000
v 0.0550 0.6000 0.0000
f 1 2 3 4
f 5 6 7 8
[griffith@feddual GuiRecess]$
Script done on Tue Aug 20 08:23:35 2002
```

What is going on in this script? The user changes his active directory to the directory containing GuiRecess by using the **cd** command. The name of the directory is not important; what is important is the location of the *GuiRecess.jar* file. The user then lists the contents of the directory using the **ls** command. This shows that the only files in the directory are the *GuiRecess.jar* and *README.txt* files. Nothing else is needed! GuiRecess is then run using **java -jar GuiRecess &**. You may notice that this is slightly different than described above. The use of **&** allows the user to run GuiRecess in the background. What this means is that the terminal/command line is not tied up in running GuiRecess. Unfortunately, the **&** is a UNIX command and requires a UNIX/Linux environment to use. Once GuiRecess has been completed, the user again lists the contents of the directory. This shows that the four output files have been created in the directory. These files are the *current_element_input*, *recess_input*, *recess.jpg*, and *recess.obj* files. Finally, the user checks to make sure that data was written to the files using another UNIX command, the **head** command. The user could have used the **cat** or **more** UNIX commands as well. Keep in mind that some commands are the same in UNIX/Linux, Windows/DOS, and Mac OS, and some commands are different. This has been a display of using the program in Linux, so things may look different for Windows and Mac OS users. Additionally, it should be noted that the pictures of the actual program shown later may differ depending on operating system and display manager (the pictures were produced on Linux using the KDE desktop and gimp graphics program).

FUTURE IMPROVEMENTS TO THE PROGRAM

Below is a list of improvements that are intended for future versions. These include both functional improvements and bug fixes.

- Recess zoom in/out feature
- Dialog based creation of recesses and current straps
- Dialog based fine-tuning of recesses and straps before completion
- Allow recesses and straps to be drawn in any direction (from any corner)
- Implementation of tilting for straps
- Bug fix for error when a recess or current element is drawn outside the usable area
- Option to display all sub-recesses and sub-current straps on a specified window, a merging of layers and corresponding de-merging of layers
- Improved 3d viewing of antenna, currently uses a gray, generic 3d wireframe viewer
- Ability to read recess_input and current_element_input files and create the corresponding recesses and current elements in GUI form
- Prevent overlapping recesses
- Optionally allow multiple current strap depths in same recess
- Ensure proper rounding of numbers
- Implement a naming scheme for the recesses to aid in organization, perhaps by labeling each window with the name of its parent window followed by whether it was the first, second, third, etc. recess drawn in its parent. For example, the second recess drawn in the workspace would have the label "1.2". The first recess drawn in recess "1.2" would be "1.2.1".
- Allow individual removal of current elements
- Provide a window that will give information on any specified recess or current element
- Copy and paste recesses inside another recess
- New and improved view (more program-like)
- Buffered image paint method so that the drawing looks smoother
- Green/red lines for recesses and current straps disappear when the dialog box is dragged over them because they are not part of the permanent paint.

- Write output files only when buttons pressed, currently files are written at start to ensure new files not appended to old
- Allow choice of destination directory for output files, destination name for 3d object file, more flexible write/save options

FIGURES

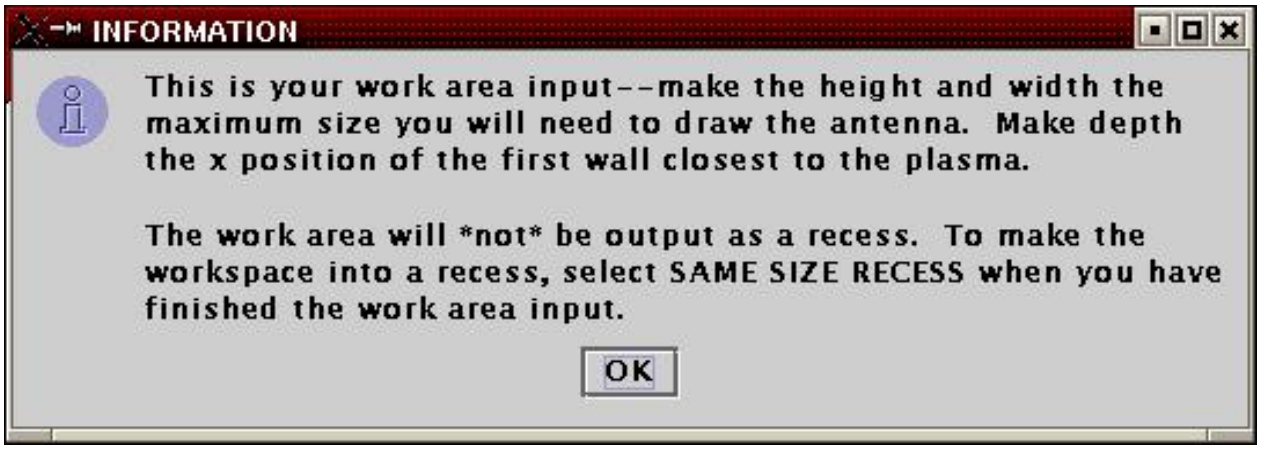

Figure 1 - Startup Information

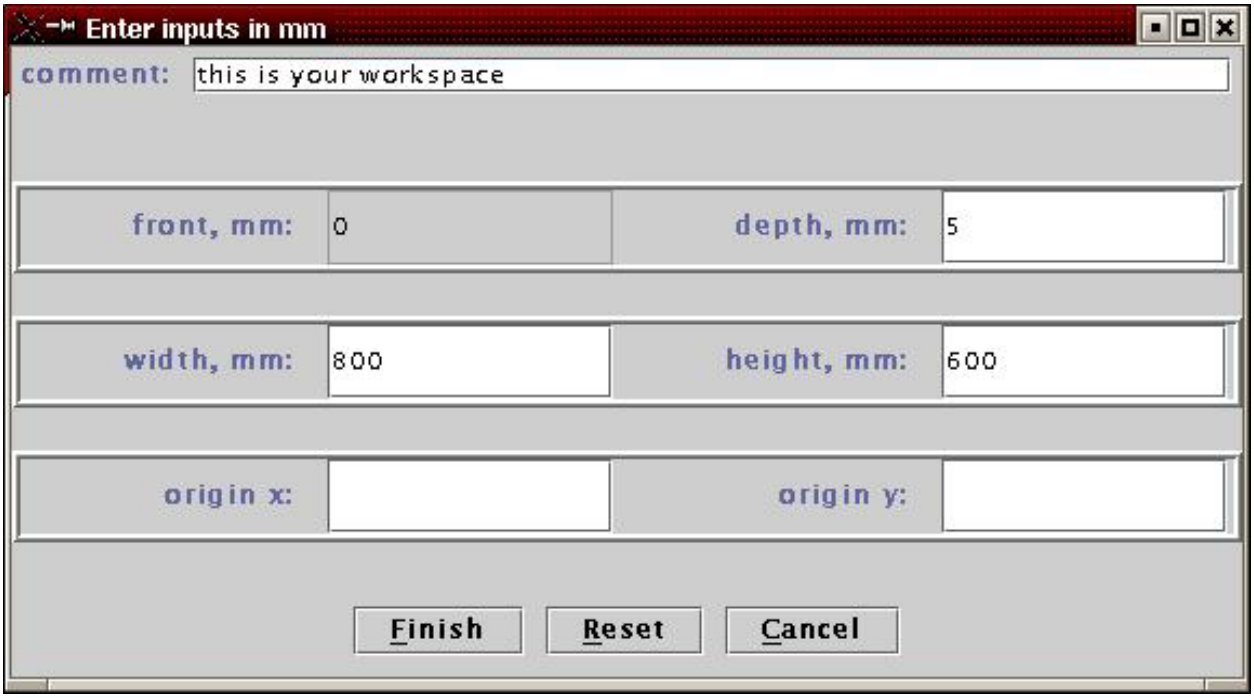

Figure 2 - Workspace Dialog

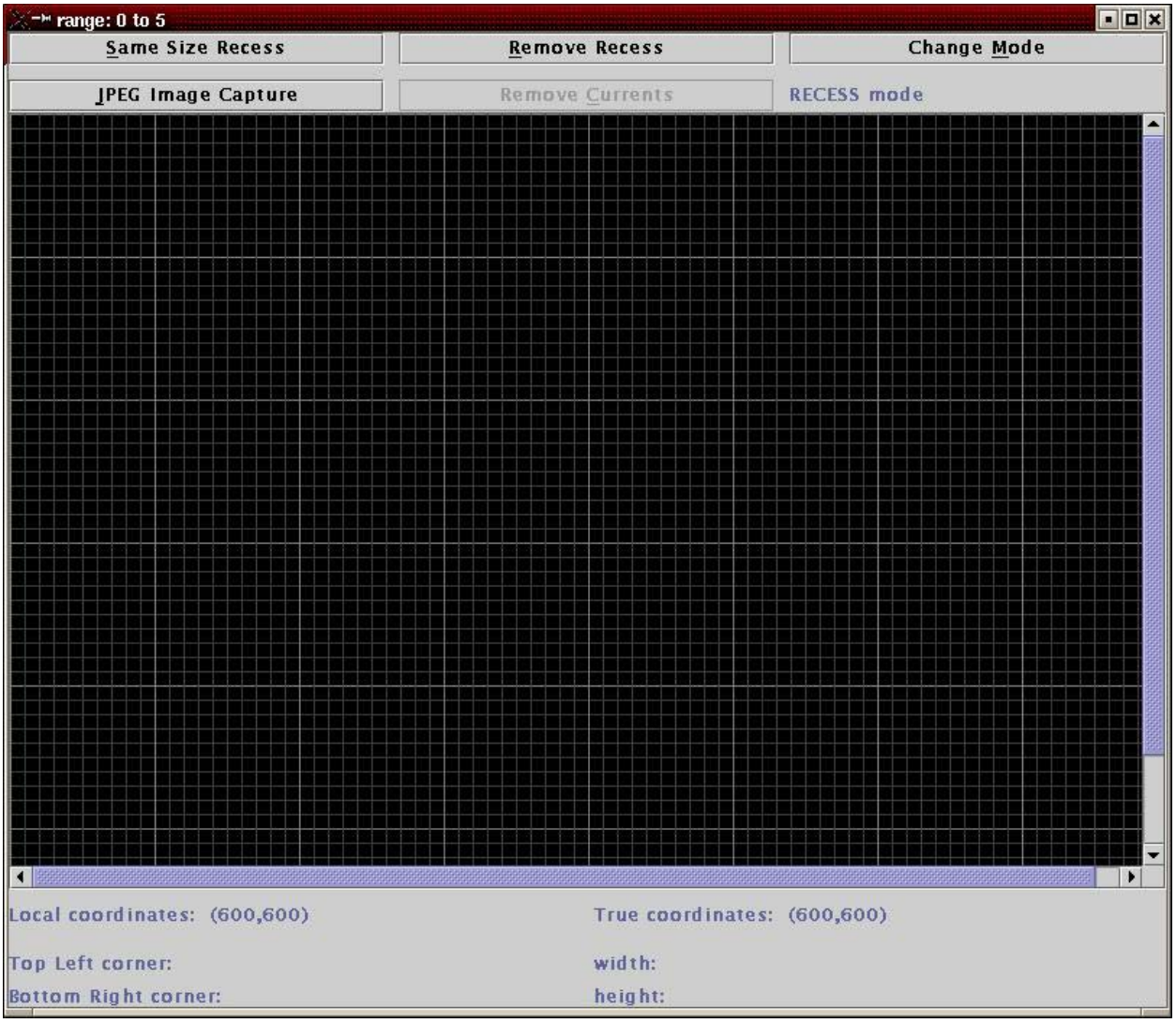

Figure 3 - Empty Recess Window

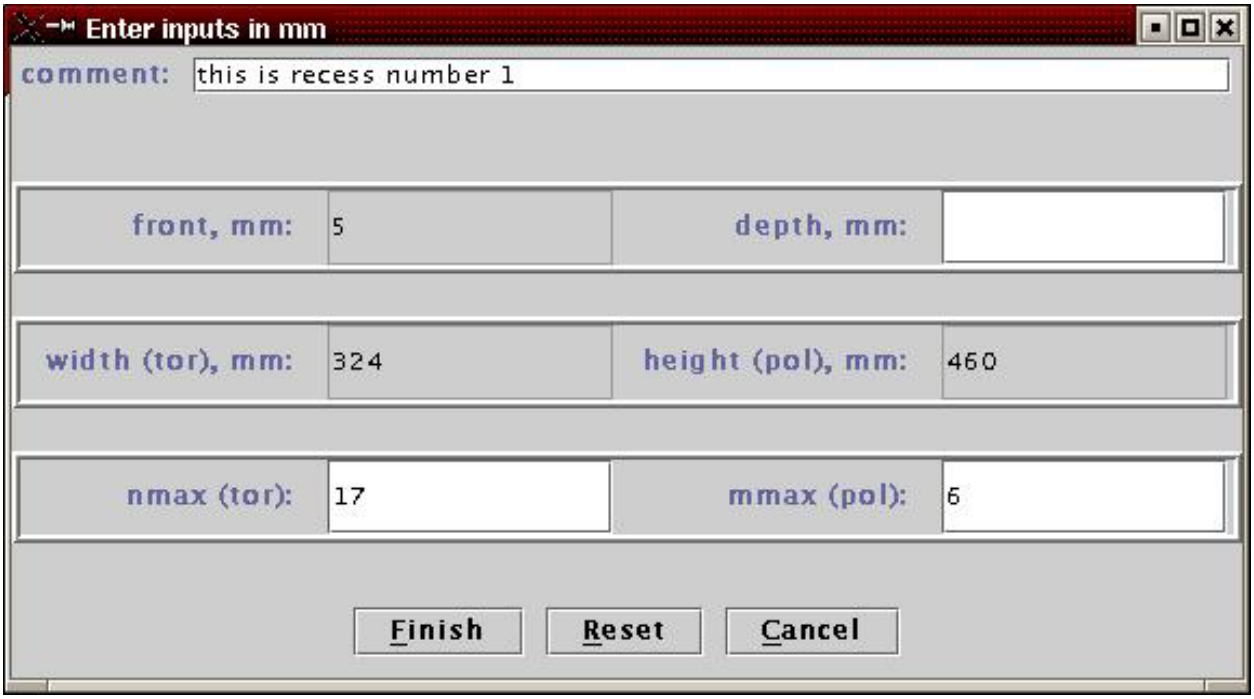

Figure 4 - Recess Dialog

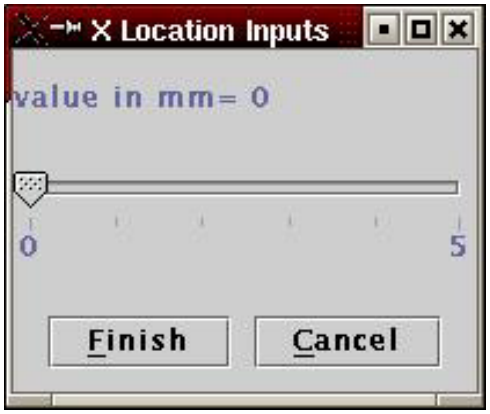

Figure 5 - Current Dialog 1

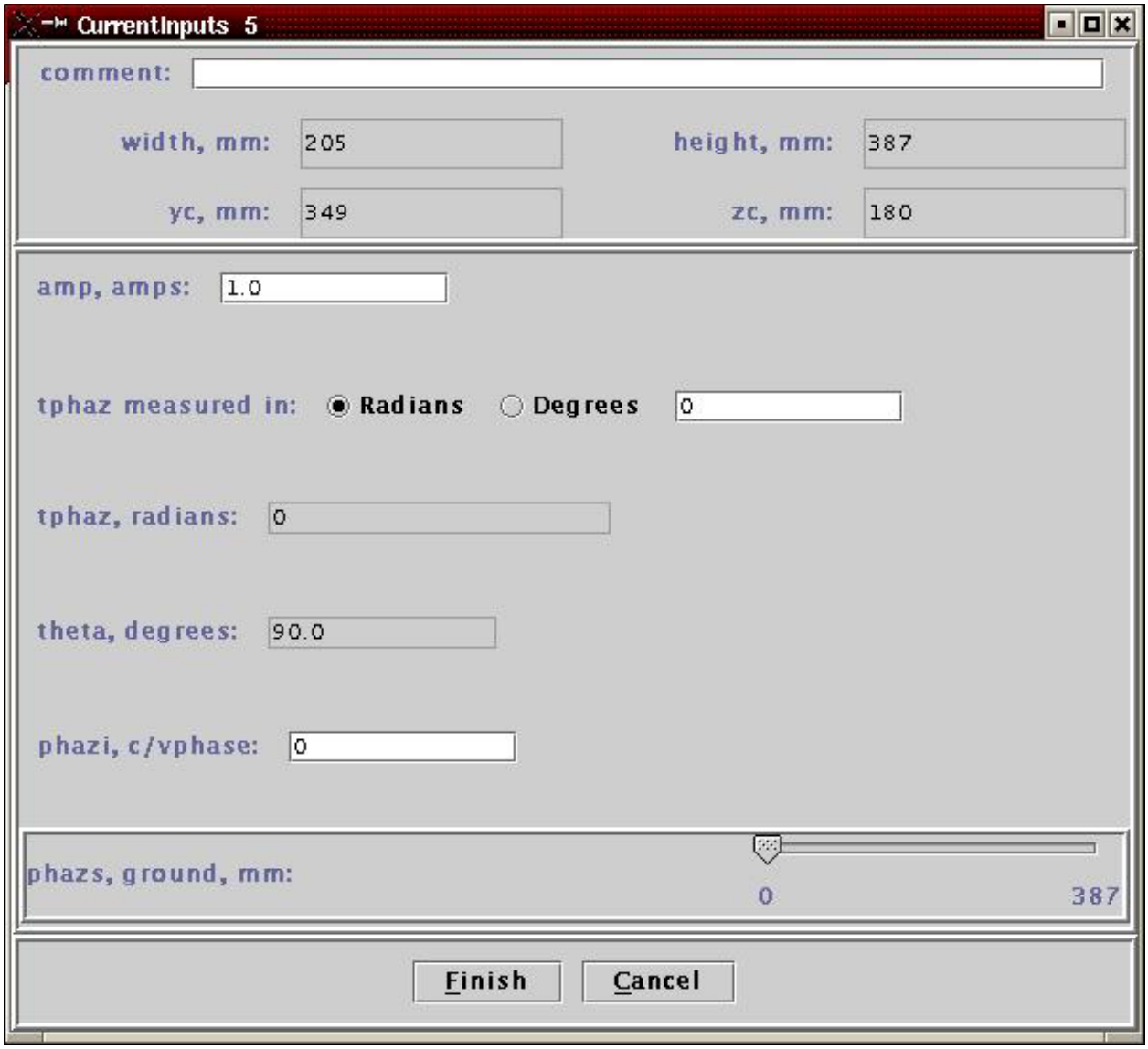

Figure 6 - Current Dialog 2

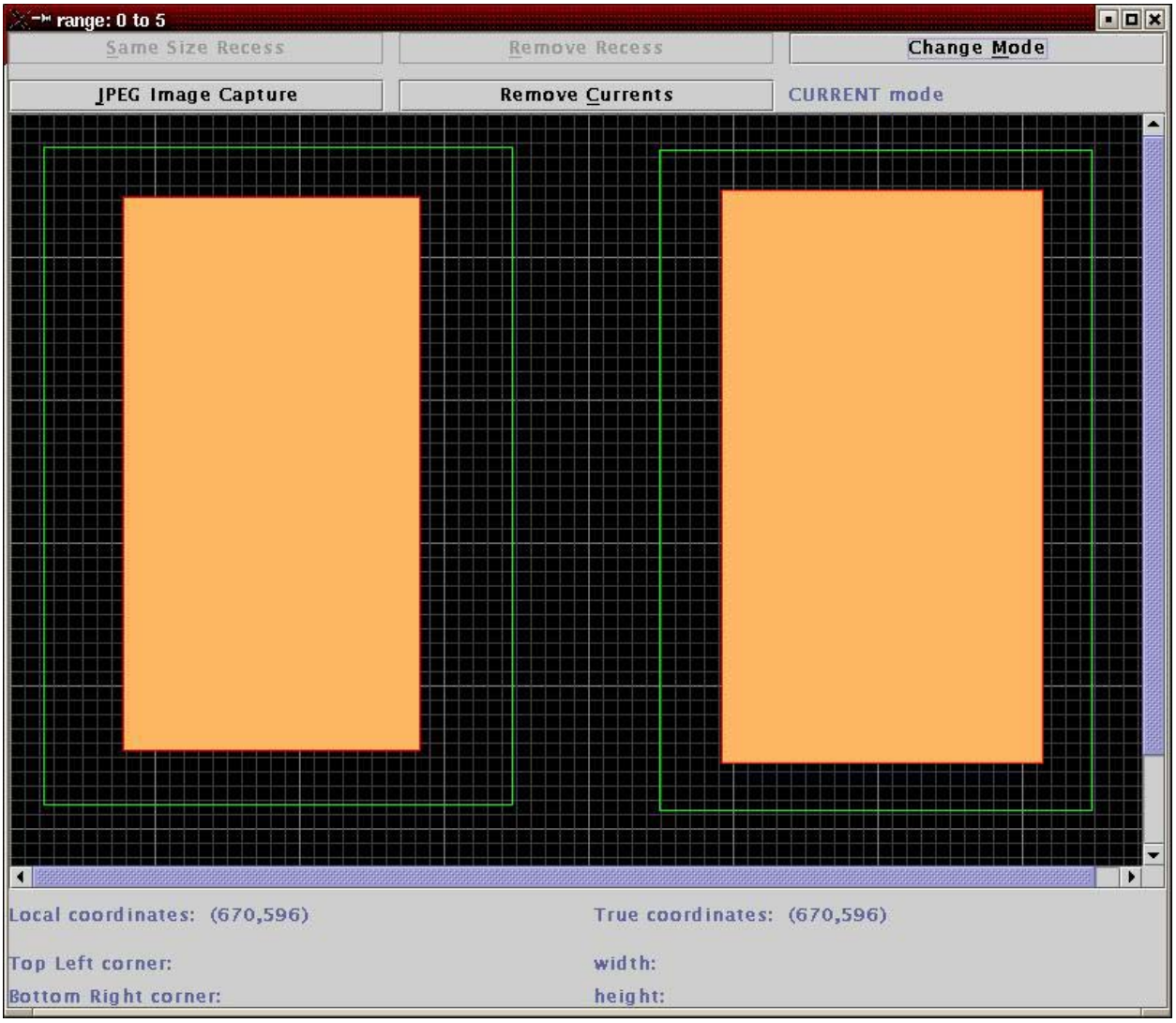

Figure 7 - Full Recess Window

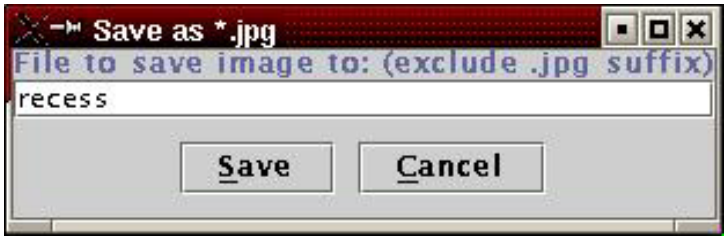

Figure 8 - JPEG File Output Dialog

Figure 9 - Output Files, 3D View, and Exit

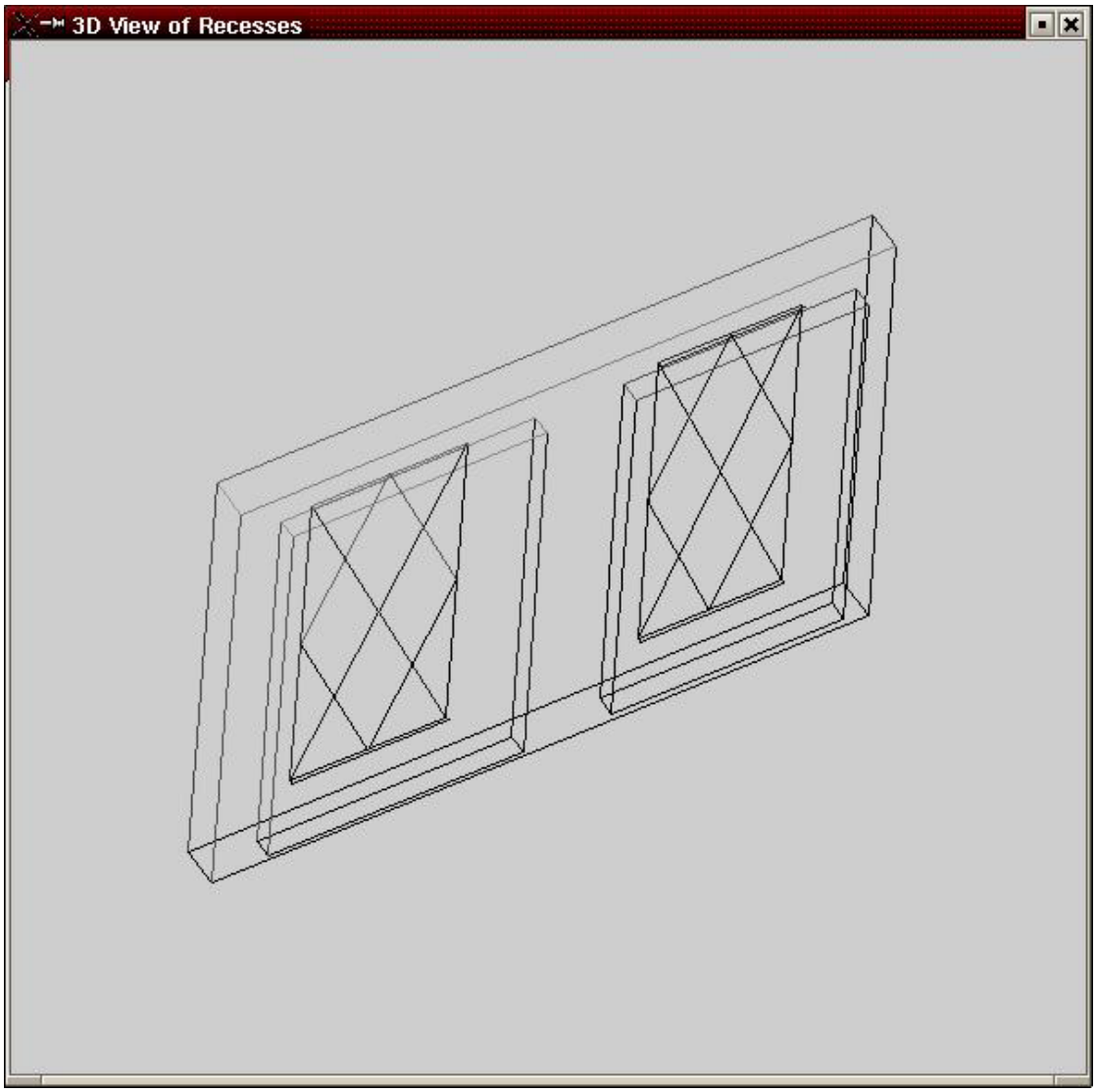

Figure 10 - 3D Recess Viewer

ACKNOWLEDGEMENTS

This document was written by Adam Griffith (argriffi@rice.edu *) with assistance from Kate White (kate@utk.edu) and under the direction of Mark Carter (cartermd@ornl.gov), and Phil Ryan (ryanpm@ornl.gov). Adam Griffith is an undergraduate student at Rice University, Houston, Texas, who participated in the Higher Education Research Experiences at Oak Ridge National Laboratory 2002 Summer session. Kate White is and undergraduate student at the University of Tennessee, Knoxville, who participated in the DOE Energy Research Undergraduate Laboratory Fellowship 2002 Summer session. Both Adam and Kate were assigned to the Fusion Energy Division for their terms. Mark Carter and Phil Ryan both work at the Fusion Energy Division of the Oak Ridge National Laboratory.

GuiRecess v 1.0 was originally written by Kate White. Additional modifications to that code were performed by Adam Griffith. The code was developed under the guidance of Mark Carter, with input from Phil Ryan and David Swain. GuiRecess v 1.0 was written following the Java 2 Standard Edition version 1.3.1 API.

* This email will be changed to argriffi@alumni.rice.edu shortly after Adam's graduation in May, 2003. If argriffi@rice.edu does not work, try argriffi@alumni.rice.edu.

REFERENCES

Griffith, A.R., "Introduction to RANT3D - The Three-dimensional Recessed Antenna Model," Aug. 2002 (unpublished).

RESOURCES

The following is a list of resources on Java available on the Internet. Be advised that the locations of sites change often, which may cause some of these resources to become unavailable.

Sun Microsystems' Java Site - http://java.sun.com/

Java 2 Platform Standard Edition (J2SE™) - http://java.sun.com/j2se/

J2SE™ v 1.4.0 SDK and JRE download - http://java.sun.com/j2se/1.4/download.html

J2SE™ v 1.3.1 SDK and JRE download - http://java.sun.com/j2se/1.3/download.html

J2SE™ v 1.4.0 Application Programming Interface (API) - http://java.sun.com/j2se/1.4/docs/api/

J2SE™ v 1.3.1 API - http://java.sun.com/j2se/1.3/docs/api/

Apple's Mac OS X Java Site - http://www.apple.com/java/

Apple's Mac OS X Java Developer Site - http://developer.apple.com/java/

Apple's Mac OS Classic Java Developer Site - http://developer.apple.com/java/classic.html

The Java Tutorial - http://java.sun.com/docs/books/tutorial/

Sun's Java Developer Site - http://developer.java.sun.com/

JavaWorld - http://www.javaworld.com/