Title
EGS_WINDOWS 2: An Enhanced Graphical Interface to EGS

Authors
Chatterjee, S.
Donahue, R.J.

Publication Date
1993-01-07

LBL-33429
UC-400

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

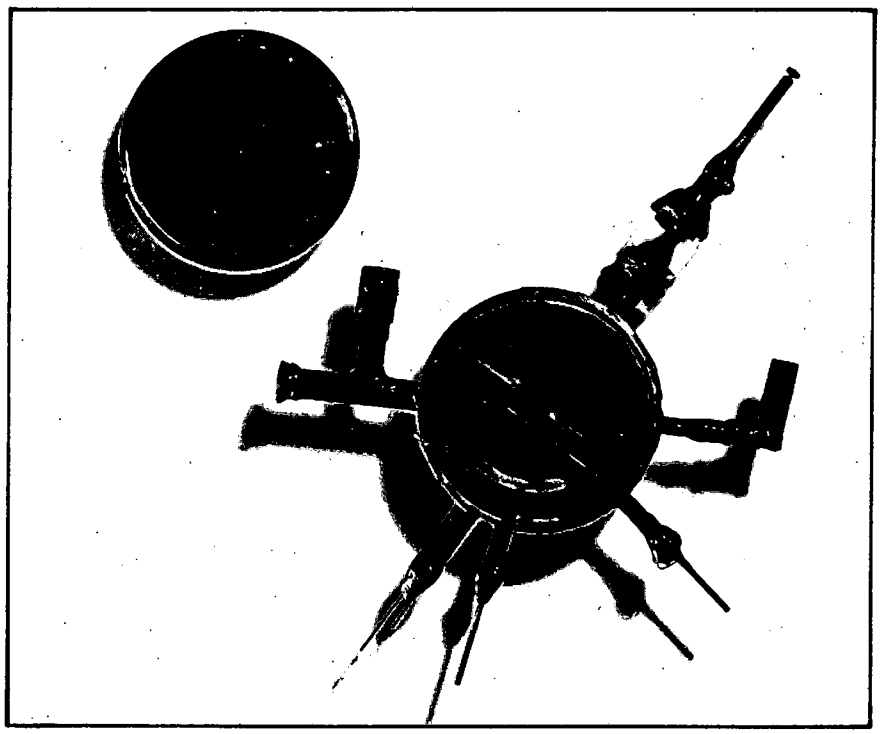
ENVIRONMENT, HEALTH AND SAFETY DIVISION

Presented at the International Conference on Monte Carlo Simulation in Nuclear and High Energy Physics, Tallahassee, FL, February 22–26, 1993, and to be published in the Proceedings

EGS_Windows2: An Enhanced Graphical Interface to EGS

S. Chatterjee and R.J. Donahue

January 1993



Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

DISCLAIMER

LBL-33429

EGS_Windows2: An Enhanced Graphical Interface to EGS

Sandeep Chatterjee

Department of Electrical Engineering and Computer Science
University of California
Berkeley, California 94720

and

Richard J. Donahue

Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

January 7, 1993

This work was supported by the Director, Office of Energy Research, Office of Basic Energy Sciences, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

1. Introduction

This report gives a general overview of the EGS_Windows2 program. The program is intended to be used to display individual particle histories of electrons, photons, and positrons generated by the EGS4 Code System[1]. Much of this work is based on the earlier version of EGS_Windows[2] from the Institute for National Measurement Standards at the National Research Council of Canada (NRCC). No attempt is made to modify their EGS subroutine WATCH, which is used to output the particle vectors to be read by EGS_Windows, or duplicate any of the documentation describing the subroutine[2]. EGS_Windows2 reads data in the same format as produced by WATCH. Users of EGS_Windows2 are encouraged to obtain the original EGS_Windows report from the NRCC in addition to this report as complete documentation of EGS_Windows2.
As in the original report, two levels of information are covered in this report: the user level, and the programmer level. The user level should be read by all users. It describes execution, operation, and capabilities. The programmer level should be read by those users who may want to modify the program or who would like a more detailed understanding. It describes briefly the routines and there function, as well as the major changes in the new release of the 3D graphics programming library Sun PHIGS 2.0.

2. User Level

2.1. The EGS_Windows2 Directory

Assuming that the EGS_Windows2 directory is installed in $EGS_Windows2_HOME, one should find the following directories:

```
$EGS_Windows2_HOME/src/
$EGS_Windows2_HOME/man/
$EGS_Windows2_HOME/examples/
$EGS_Windows2_HOME/screendumps/
```

The src subdirectory contains the source code for the EGS_Windows2 files, applicable header files, and the executable file `EGS_Windows2`.
The man subdirectory contains the man pages for EGS_Windows2 and this report, in PostScript format. To see the man page for EGS_Windows2, type `man EGS_Windows2`, assuming the MANPATH has been set to include $EGS_Windows2_HOME/man.
The examples subdirectory envelops further subdirectories that contain sample outputs from EGS4 to be used by EGS_Windows2. These files have been taken from the original EGS_Windows program and adhere to that program's naming conventions: the .egs4gph files are the phase space input files while the .egs4geom files are the geometry input files.
The screendumps directory contain screendumps of the respective phase space and geometry input files in the $EGS_Windows2_HOME/examples subdirectory. Each screendump file can be viewed individually by typing:

```
xloadimage filename
```

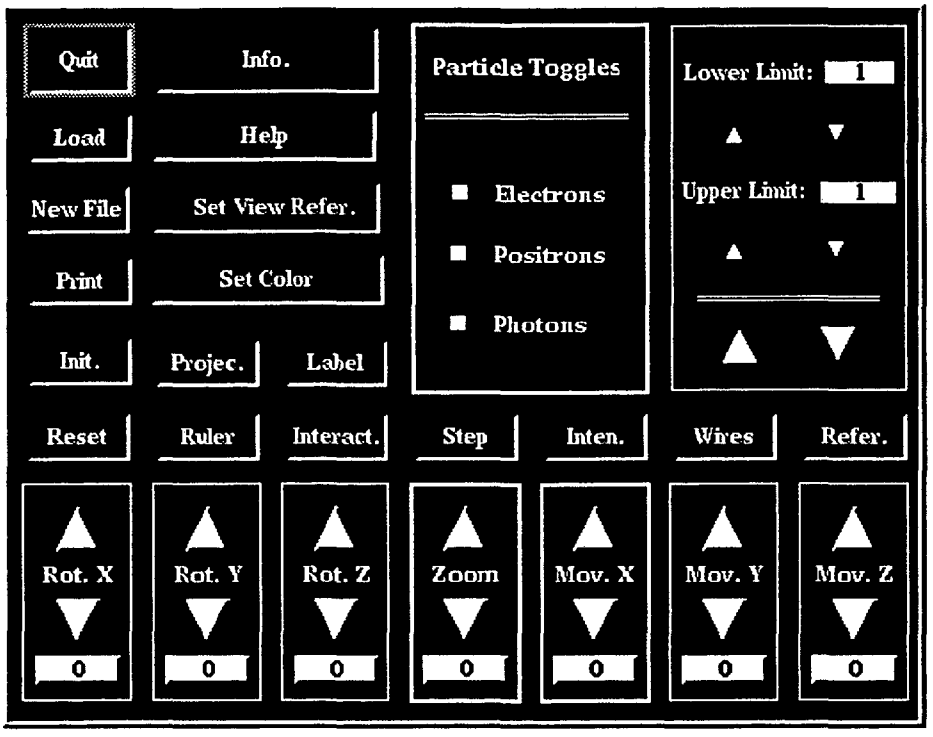


Figure 2.1: EGS_Windows2 main menu interface.

Pressing the space bar within the screendump window will terminate the screendump. All of the screendumps can be viewed sequentially by executing the `slides` script. After displaying a screendump file, the `slides` script will wait for approximately three seconds before displaying the next screendump.

2.2. Executing EGS_Windows2

EGS_Windows2 can be started by typing:

```
$EGS_Windows2_HOME/src/EGS_Windows2
```

This operation will display and start the EGS_Windows2 menu system.
A Window Manager must be running in order to properly display the EGS_Windows2 menu. Although the menu was written using OSF/Motif Version 1.1, the Motif Wondow Manager (`mwm`) need not be running. In order not to need Motif and its run-time libraries, EGS_Windows2 has been compiled and linked statically with the -Bstatic option of the C compiler. However, if Motif and its run-time libraries are absent, starting the EGS_Windows2 menu and displaying any pop-up boxes will result in numerous warnings being printed in the shell where the EGS_Windows2 file was executed. This will not detract in any way from the operation of the program.

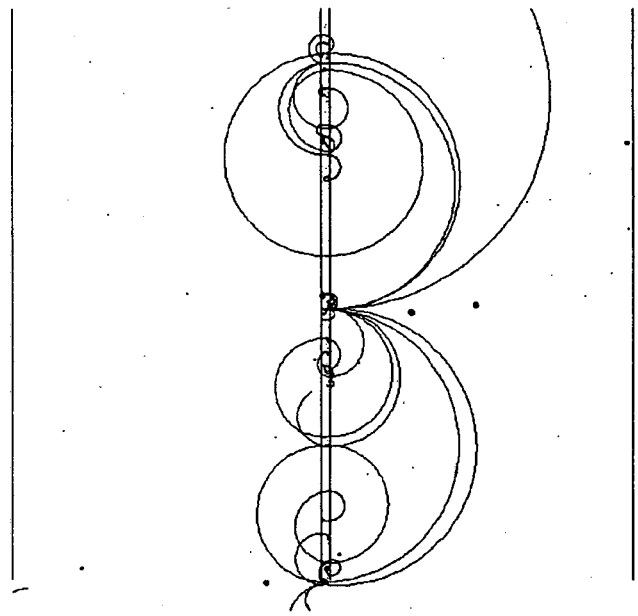


Figure 2.2: Sample graphics output showing 10 - 1GeV γ's striking $\frac{1}{2}$ radiation length lead plate inside liquid hydrogen bubble chamber.[2] Only charged particles are shown.

2.3. Capabilities

All of the capabilities of EGS_Windows2 stem from the menu. The following is a discussion of these capabilities and a detailed description of all functions.

To start the EGS_Windows2 program, type `$EGS_Windows2_HOME/src/EGS_Windows2`. The EGS_Windows2 menu system is started and displayed. Initially, except for the Load and Quit buttons, all buttons are inoperative. Upon selecting the LOAD button, an interactive file selection dialog box, along with a message box prompting for the input of the phase space file are shown. The file selection box contains a filter, a directory listing, a file listing and a selection listing. The filter can be used to search a directory for filenames with a particular pattern. The default for the filter is all files, symbolized by a wildcard (*). To enter a filter search pattern, position the pointing device within the boundaries of the filter, click the first button, enter the search pattern placing wildcards (*) where applicable, and then press the FILTER button. All files matching the new pattern are displayed in the file listing box.

To change to another directory, position the pointing device over the name of the directory in the directory listing, click the first mouse button and then press the FILTER button. The files in this new directory matching the pattern of the filter are displayed.

To select a file, position the pointing device over the name of the filename in the files listing, click the first button and then press the OK button, or double-click on the filename. The

selection listing box is updated to reflect this choice and the file is loaded. To enter a file name manually by keyboard, position the pointing device over the selection listing, enter the filename and press the OK button.
Upon entering the phase space filename, the message box prompting for its entry is removed and is replaced with a message box prompting for the entry of the geometry input filename. The geometry is selected by adhering to the above conventions. The CANCEL button discards all files selected and exits the file selection box.
Depending on the size of the input files and CPU speed, loading may take from a few seconds up to a few minutes. Because of limited memory array space, some files may even be too large to be read completely. The following limits must be adhered to in the phase space and geometry input files:

1. The phase space input file must be fewer than 200,000 lines long and must contain fewer than 1,000 histories.
2. The maximum number of steps before a change in stack is set to 100,000.
3. The number of ROTZ geometries in the geometry input file must not exceed fifteen (15).
4. The number of CYLZ geometries in the geometry input file must not exceed ten (10).
5. The number of QUAD geometries in the geometry input file must not exceed ten (10).
6. The number of PARL geometries in the geometry input file must not exceed ten (10).
7. The number of DISK geometries in the geometry input file must not exceed twenty (20).
8. The number of COLX geometries in the geometry input file must not exceed twenty (20).
9. The number of COLY geometries in the geometry input file must not exceed twenty (20).
10. The maximum number of fields per line in the geometry input file must not exceed twelve (12).
11. The maximum number of rings in a ROTZ geometry in the geometry input file must not exceed five (5).

If the above limits are exceeded, an error describing which of the above arrays was (were) overflowed will be printed in the shell where EGS_Windows2 was started. The error will not halt execution of the program. The program will continue; however, any data read after exceeding that particular array's limit will be ignored.
After loading is complete, the first history read in will be displayed in the EGS_Windows2 main window. The current history indices range is labeled LOWER LIMIT and UPPER LIMIT. The lower limit of the history range can be changed by using the arrow buttons directly beneath the LOWER LIMIT label. The upper limit of the history can be changed by using the arrow buttons directly beneath the UPPER LIMIT label. Both the lower and upper limits of the history can be changed simultaneously by using the larger arrow buttons located at the bottom of the history box. The upper and lower history limits can be changed manually with the keyboard by first pressing on the corresponding limit label push button and then entering the numerical history limit at the prompt.

The particle history structure at startup includes all electrons, positrons, and photons. These individual particles can be set to on or off by the respective toggles in the particle toggles box.
The image view can be changed about the xy, yz, or xz axes by selecting the SET VIEW REFERENCE pop-up menu and then selecting the appropriate view.
The rotation of the image about the xy, yz, and xz axes can be changed by using the appropriate ROT. X, ROT. Y, and ROT. Z arrow buttons. The increment is initially set to 10 degrees. The current rotation about the respective axes is shown beneath the down arrow button. The degree increment can be changed by selecting the degree label push button and subsequently entering the numerical increment value.
A cube is located at the upper left corner of the main EGS_Windows2 window which rotates in accordance with the rotation of the image. Three faces of the cube are labeled X, Y, and Z to provide a reference orientation.
The magnification of the image can be changed by using the appropriate ZOOM arrow buttons. The magnification of the image above 100 percent is displayed below the down arrow button. The magnification increment factor is initially set to 10%. It can be changed by selecting the magnification label push button and subsequently entering the numerical increment value.
The displacement of the image from the initial center point can be changed by using the appropriate MOV. X, MOV. Y or MOV. Z arrow buttons. The current displacement is shown below the appropriate down arrow button. The displacement increment factor is initially set to 10 cm. This can be changed by selecting the displacement label push button and subsequently entering the numerical increment value.
The point about which the image is zoomed and rotated can be changed by using the REFERENCE button. To change the center point, select the REFER. button. Place the mouse arrow within the EGS_Windows2 window. Choose the X and Y coordinates of the center point and press the second mouse button. An orthogonal view is subsequently displayed. Now select the Z coordinate and press the second mouse button.
Below is a brief description of most of the EGS_Windows2 window buttons:

- The button labeled INTERACT toggles the display of circles which indicate every discrete interaction and every point where a particle is terminated.
- The button labeled STEP toggles the display of crosses (x) which indicate every step.
- The button labeled RESET eliminates any transformations (rotations and magnifications) which have been performed on the image, and displays the original image in the current view reference.
- The button labeled INIT. resets the center point of the image to its initial values.
- The button labeled RULER toggles the display of a line segment and a numerical value delineating the length of a segment in the units of the input file - usually centimeters. This value can be used as a key to gauge the dimensions of the image in the EGS_Windows2 main window. When the ruler is on, the image is set to a parallel view so as to portray accurate dimensions.
- The button labeled PROJ. toggles between parallel and perspective projection views. To choose a perspective view, press the PROJ. button and enter the

numerical distance of the center point of the image to the eyepoint. To choose parallel projection, press the PROJ. button again and the image will default to the parallel projection view.

- The button labeled INTEN. toggles between a solid color display of the histories and a display whose intensity variation is indicative of the kinetic energy of each history.
- The button labeled WIRES toggles the display of the geometry wire frame. To remove the geometry, press the wires button and subsequently press the NO WIRES button in the pop-up window. To change the number of wires enveloping the geometry press the WIRES button and then enter the numerical number of wires.
- The button labeled LABEL toggles the display of a user entered label. To render a label, press the LABEL button and enter the alphanumeric label. Then position the mouse in the EGS_Windows2 main window and choose the position of the label. Press the first mouse button to display the label. To permanently place the label, press the second mouse button. The label is removed during any transformations of the image.
- The button labeled SET COLOR allows a user to specify a RGB (Red Green Blue) value for the electrons, positrons, photons, geometry, and background. To set a color, press the SET COLOR button and press the appropriate object button in the popup window. Then enter the RGB values on a scale of 0 being zero brightness and 255 being full brightness.
- The button labeled PRINT outputs an Encapsulated PostScript file of the current image in the EGS_Windows2 main window. The output is the file EGS_Windows2.ps. Print also outputs a screendump of the PostScript image and places it in the file EGS_Windows2.dump. To view the screendump, type:

  ```
  EGS_Windows2_HOME/src/xwud -in EGS_Windows2.dump
  ```

  or

  ```
  EGS_Windows2_HOME/screendumps/xloadimage EGS_Windows2.dump
  ```
- The button labeled HELP is an on-line help menu. To obtain help, press the HELP button and then select the appropriate item where help is needed.
- The button labeled INFO. gives information about the program and an email address where comments can be sent, on a handsome backdrop.
- The button labeled NEW FILE terminates the current EGS_Windows2 main window and displays the file selection box and prompts for new input files. Before terminating the main window, a popup window verifying the action is displayed.
- The button labeled QUIT terminates the EGS_Windows2 main window and exits the menu. Before terminating the main window, a popup window verifying the action is displayed.

EGS_Windows2 has been equipped with a reliable way to output hardcopy. The PRINT push button in the EGS_Windows2 menu takes the image in the main window and converts it to both a full color PostScript format and to a color screendump format. The PostScript format of the image is placed in the EGS_Windows2.ps file and the screendump format is placed in the EGS_Windows2.dump file. It is important to note that the EGS_Windows2

main window must be completely unobstructed to work properly. All images enclosed by the periphery of the EGS_Windows2 window are captured by the PRINT option and then converted to PostScript and screendump formats.
Both of the EGS_Windows2.ps PostScript and the EGS_Windows2.dump screendump files are placed in the current directory. To print the EGS_Windows2.ps file to an output device, type:

```
lpr -P<Printer Name> EGS_Windows2.ps
```

To view the screendump file, print:

```
xloadimage EGS_Windows2.dump
```

or

```
xwud -in EGS_Windows2.dump
```

The screendump files may also be added to the EGS_Windows2_HOME/screendumps subdirectory. The file must first be renamed to <filename>.dump and then moved to the EGS_Windows2_HOME/screendumps subdirectory. If renamed and placed properly in the screendumps subdirectory and depending on its position in the screendumps subdirectory, this file is also displayed during execution of EGS_Windows2_HOME/screendumps/slides.
The EGS_Windows2 PRINT option has been built using standard X11 software. To ensure the proper execution of PRINT, all relevant files have been included in the EGS_Windows2 distribution. If any X11R5 software are not properly installed or if any problems are encountered, point the PATH environment variable to EGS_Windows2_HOME/src.

3. Programmer Level

EGS_Windows2 is written in both ANSI standard Fortran and Standard C, with the main driver loop and menu system written in C and the ancillary subroutines written in Fortran 77. Familiarity with C, Fortran 77, Fortran bindings for SunPHIGS 2.0, and the C bindings for Motif is required for modifying and understanding EGS_Windows2.

3.1. Compiling and Linking EGS_Windows2

Assuming that EGS_Windows2_HOME is the directory where EGS_Windows2 is installed, the $EGS_Windows2_HOME/src subdirectory contains the following files:

```
main-c.c
creation-c.c
callbacks-c.c
dialog-c.c
filedialog-c.c
helpdialog-c.c
colortext-c.c
newtry.f
help.h
show.h
EGS_Windows2
```

`Makefile`

The `main-c.c` file initializes the Motif root window and returns the top-level window to use as the parent window to the EGS_Windows2 menu system.
The `creation-c.c` file builds the initial EGS_Windows2 menu structure and directs the response of all menu buttons upon their use.
The `callbacks-c.c` file handles many of the callback routines which are referenced when a button in the EGS_Windows2 menu is pressed. Most of the functions in `callbacks-c.c` reference the appropriate ancillary subroutine in `newtry.f`.
The `dialog-c.c` file initializes and opens all popup dialog prompt boxes and message boxes. The callback routines for the buttons within these dialog boxes are self-contained within the `dialog-c.c` file.
The `filedialog-c.c` file initializes and opens the file dialog box used for file input. The callback routines for this dialog are also contained within this file.
The `helpdialog-c.c` file initializes and opens the help popup message boxes. This file references the predefined help texts in `help.h`.
The `colortext-c.c` file initializes and opens the color dialog boxes. Upon receiving the proper input, this file references the `newtry.f` file to set the color of the appropriate object.
The newtry.f file contains subroutines which are referenced by the EGS_Windows2 menu to modify the displayed image in the EGS_Windows2 main window.
The file `help.h` contains predefined help texts to be used by the `helpdialog-c.c` file. Similarly, the `show.h` file contains predefined information used by the `newtry.f` file. The EGS_Windows2 is the statically linked executable file.
A Makefile is supplied if the EGS_Windows2 code is to be recompiled. **Compilation requires Motif and SunPHIGS 2.0 libraries and include files.** To link the C and Fortran 77 files correctly, all files must be compiled with the `-c` option and then later linked with the `-Bstatic` and `-lF77` options.

3.2. SunPHIGS 2.0

The original EGS_Windows program has been ported from SunPHIGS 1.2 to SunPHIGS 2.0 to take full advantage of the added functionality and flexibility of the newer release. Although there have been numerous changes in SunPHIGS, only changes relevant to the EGS_Windows2 program are discussed. We first begin with an overview of SunPHIGS.
PHIGS (Programmers Hierarchical Interactive Graphics System) is a library of functions which support graphics application programs. Its particular usefulness stems from the fact that it is almost completely devoid of hardware dependencies. To further blur these dependencies, PHIGS implements its own input devices.
The SunPHIGS 2.0 release contains many changes from SunPHIGS 1.2. Although both C and Fortran language bindings are supported by SunPHIGS, only the Fortran language bindings are discussed herein as it is the language of `newtry.f` – the file that contains all of the routines which utilize SunPHIGS 2.0 to appropriately transform the graphical image.
The following list outlines changes to SunPHIGS 2.0 relevant to the EGS_Windows2 program. Many of these changes are later discussed in greater detail.

- Use of ANSI Fortran bindings
- PHIGS Tool Workstation Type replaced by the new X Tool Workstation Type
- Changes to input devices
- Workstation connection ID specifies X display location
- Changes to acceleration and double-buffering
- Requires the use of Open Windows Version 3
- Requires the SunOS 4.1.1 with 4.1.1-GFX Rev. 2 and the XGL 2.0 run-time libraries.

SunPHIGS 2.0 includes much of the functionality of SunPHIGS 1.x releases, but in order to accomodate and conform to the above changes, many major and some minor modifications have been made. Several new functions have been added while other functions have been removed. The following paragraphs discuss these changes in detail and how they have been remedied in the EGS_Windows2 program.

One of the most important changes to SunPHIGS 2.0 was its implementation of logical input devices. Because SunPHIGS' implementation of these input devices almost remove all hardware dependencies, the original EGS_Windows program exhaustively utilized these devices in its program. However, these devices incurred some limitations with regard to their capabilities, control and probably most important, aesthetics.

One of the biggest limitations of these input devices can be seen by the way they must be initialized before they can be displayed. The initialize valuator subroutine `pinvl`, for example, is used to set and/or reset the initial value, the two delimiting values, the label, the graphical appearance and other properties of a valuator device (slider box). In SunPHIGS 2.0, this function loses some of its arguments to a data record which is formed with a call to `Pack Data Record`. This data record is later referenced by the initialize valuator subroutine. This allows the SunPHIGS user to have a sort of data abstraction which is independent of the actual call to initialize the valuator device.

This function also controls the appearance of the valuator device on the workstation. This is controlled by setting the `Prompt/Echo Type (PET)` to a certain integer value. SunPHIGS defines a limited number of these `Prompt/Echo Types` and thus, greatly limits the capabilities of valuator slider boxes and all other logical input devices. It may seem feasible that the user choose a `PET` that is most conducive to his/her application. However, any attempt to change the appearance and capabilities of the device is not simply remedied by modifying the integer value of the `PET`, but instead requires major reorganization to each data record. The choice of string input devices has incurred the same deficiency.

As Motif gains more acceptance, we felt that it would soon become a standard and therefore, decided to base our control panel on it and abandon most of the SunPHIGS logical input devices. Although we discarded the valuator (slider box), choice (button box) and string input devices, we remained steadfast to the locator input device. This is because SunPHIGS' implementation of the locator input device remains to be the most simple and effective method for obtaining the coordinates of the location of a pointing device within the PHIGS window.

The locator input device is currently used to obtain the coordinates of the center point during the REFERENCE operation and to identify the point where a label should be displayed. However, the valuator input devices are no longer used and their functions have

been usurped by arrow buttons in the EGS_Windows2 menu. Similarly, the choice of input devices' operations are now controlled by push buttons.
The set color representation subroutine is used to set and/or modify an enrty in a workstation's color table. Most drawing primitives, such as the ones used to draw the geometries and histories, use this subroutine to set a certain color and then later reference it as the actual structure is traversed. The arguments to this function change in order to allow different methods of modifying the color table's RGB values. In SunPHIGS 2.0, the user must specify the number of elements being set and the name of an array containing the individual color intensities.
In accordance with the changes to the workstation types in SunPHIGS 2.0, there have been numerous modifications, additions, and deletions to the workstation attributes which determine every workstation specification from double buffering to window dimensions. The EGS_Windows2 program has been upgraded to use the newer X Tool workstation type. Many of the workstation attributes such as some cursor types have been removed and have no clear mapping or have been modified in some way but still remain in SunPHIGS 2.0.

3.3. EGS_Windows2 Menu System Control

A Motif based interactive menu system has been added to EGS_Windows2 to further the menuing capabilities of the original EGS_Windows program, to increase clarity and organization and, of course, to aesthetically enhance the program. The Motif menu is the main control loop of the entire program. Upon execution of the program, each push button, toggle button and arrow button is assigned a callback routine. When the particular button is activated and/or deactivated, its assigned callback routine is referenced. Most callback routines are placed in the `callbacks-c.c` file while others reside in `dialog-c.c`, `colortext-c.c`, `filedialog-c.c`, and `helpdialog-c.c`. When the callback routine is referenced, control is passed to that routine and any other buttons pressed in the EGS_Windows2 menu are queued in an input buffer for later execution. That is, the present operation must be completed and control must return to the menu before any other operations are executed. The only exception to this is the INFORMATION button. The job is backgrounded and any other operations may be executed before terminating the image.

3.4. Fortran and C

The EGS_Windows2 menu system and its immediate callback and popup dialog routines are all written in standard C. The `newtry.f` file, which controls the initial loading of both input files and all transformations on the images, is written with the Fortran bindings of SunPHIGS 2.0. The blending of Fortran code with C code has resulted in many complexities in the program.

The most fundamental of these complexities is related to referencing functions from one language to another. Fortran programs automatically attach an "_" (underscore) to the end of all subroutines and function names. C, however, does not. Therefore, when referencing a Fortran subroutine from a C function, an underscore must also be attached. There are further complexities that must be remedied when a Fortran subroutine references a C function. To avoid these further issues, the EGS_Windows2 program only references Fortran subroutines from C functions and not its converse. Fortran subroutine names have also been made lower case to suppress problems with C's naming conventions.

The methods that C and Fortran use to pass arguments to functions presented another problem. Fortran passes arguments by reference. That is, Fortran subroutines pass the address of the arguments to other subroutines. C, on the other hand, passes arguments by value. C makes a copy of the argument value and then passes that copy to the function. Therefore, if a C function is to pass an argument to a Fortran subroutine, it must pass the address of the argument to Fortran. The following segment of code, excerpted from the EGS_Windows2 program, illustrates C functions calling and passing arguments to Fortran subroutines.

```
.
.
int i;
char labelstr[200];
 .
 .
projection_(&i);
label_(labelstr);
 .
 .
```

3.5. Motif

To most efficiently use limited time, portions of the EGS_Windows2 code have been generated by the Builder Xcessory (bx). The graphical layout of the EGS_Windows2 menu and some buttons' attributes and callbacks have been set by bx. However, because of bx's limited capabilities in generating proper code for more complex menu structures and finely detailed customized popup boxes, we migrated to Motif. The EGS_Windows2 menu background was built with bx, but everything on the menu is controlled by Motif.

We found The Motif Programming Manual[3] an invaluable resource for Motif programming. Not only is the book a clear and concise reference manual, it is fraught with numerous working examples to further elucidate the material. This book and its examples were particularly helpful in writing the code for popup dialog boxes in EGS_Windows2. Some

portions of the example dialog code have been used, modifying them where applicable, to build and ameliorate larger, more complex popup boxes.

4. Known Bugs

1. In perspective projection mode, appending a label to an image may result in erratic label placement.
2. In Open Windows Version 3, the destruction of widgets in the menu system may not always be complete.

5. Future Developments

1. Port entirely to ANSI standard C and remove inherent complexities of jointly referencing and linking Fortran 77 and C.
2. Dynamic loading of the phase space file so that histories are read and stored only when needed. This would necessitate an interpreter between EGS4 geometry and PHIGS geometry. Perhaps a more universal geometry description file (CAD drawing) could be utilized.
3. A more advanced printing feature to recognize different output devices and printing formats.
4. Ability to save changes and transformations to phase space and geometry input files for later reuse.
5. A ruler mechanism which would display an accurate length for any arbitrary distance in the main view menu.
6. A more advanced and interactive method to choose and set colors.

Acknowledgement

We which to thank Christian Mangin for his original version called SHOW_HIST and the subsequent version, EGS_Windows, by Alex F. Bielajew and Paul E. Wiebe. The example graph and geometry files have been taken from EGS_Windows.

6. References

1. W. R. Nelson, H. Hirayama, and D. W. O. Rogers, "The EGS4 Code System", SLAC-265, December (1985).
2. A. F. Bielajew and P. E. Wiebe, "EGS_Windows: A Graphical Interface to EGS", National Research Council of Canada, PIRS-0274, May (1991).
3. Dan Heller, The Motif Programming Manual, O'Reilly & Associates Inc. (1991).

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
TECHNICAL INFORMATION DEPARTMENT
BERKELEY, CALIFORNIA 94720