INTERNAL REPORT #73

SRL PLOTTING PROCEDURES

by

Eric Holstege

Space Radiation Laboratory

California Institute of Technology
Pasadena, California

7/22/80

SRL Plotting Procedures

INTRODUCTION

This document describes the plotting software for the HP and VERSATEC. This software consists of two components: a set of Device-Independent-Graphics C subroutines, and three drivers. The DIG subroutines write coded plot insrtuctions on the standard output in the format described below (DEVICE INDEPENDENT REPRESENTATION). The three drivers read their standard input for these plot instructions and produce plots on specific devices. There are drivers for the HP2648a and for the Versatec.

DEVICE INDEPENDENT GRAPHICS SUBROUTINES

DEFINITIONS

Screen Coordinate:
Screen coordinates define locations on the physical plotting area. The values of each coordinate are integer. The device-independent dimensions of all plotting areas are 4096x4096 (coordinate values 0-4095). All screen cooordinates in this range are guaranteed to be visible. Most devices are not square, and one of the screen cooordinates can therefore be larger than 4095. 4096 is the length of the shortest dimension. The HP screen is 8192x4096; the Versatec is 5377x4096. The symbols MAX_X_VIEW and MAX_Y_VIEW represent the maximum lengths which will fit on all presently supported devices. At present, MAX_X_VIEW is 5377, MAX_Y_VIEW is 4096.

World Coordinate:
World coordinates are floating point numbers representing locations in the user defined logical plotting area. These are mapped using the windowing transformation to screen coordinates to get actual locations.

Viewport:
A viewport is a rectangular subarea of the screen. It defines the area beyond which no plotting will take place. Points outside will not be displayed, and lines going outside will be clipped. The pen position may go outside the viewport, but no plotting will.

Window:
A windowing transformation, or simply a window, is a special case of a mapping transformation which maps world coordinates onto screen coordinates. It involves only tranlations and scaling (no rotations), and is therefore particularly simple.

NOTATION

Variables starting with 'x' or 'y' are floating point numbers and represent world coordinates. Variables starting with 'ax' or 'ay' are integers, nominally in the range 0 to 4095 and represent screen coordinates. Upper case symbols (ex: NO_CHAR) refer to #define'd symbols from the include file "plot.h". In example calling sequences for the various functions, arguments enclosed in square brackets [ ] are optional.

PRIMITIVES

move(x,y)

setextsize(n)
Sets the default text size to n times normal. n is in the range (1-8).

setextdirection(n)
Sets the default text direction. n is one of TD_R0, TD_R90, TD_R180, TD_R270.

setextjust(n)
Sets the text justification. The text justification controls the interpretation of the pen location when a string is drawn with "print". The meanings of various values of 'n' are described in the following table:

3 TJ_UL	6 TJ_UC	9 TJ_UR
2 TJ_CL	5 TJ_CC	8 TJ_CR
1 TJ_LL	4 TJ_LC	7 TJ_LR

The #define'd symbols are all of the form TJ_yx, where y is one of L (lower), C (center), or U (upper), and x is one of L (left), C (center), or R (right). For a complete description of text justification, see the HP manual. Briefly, TJ_?C indicates that a text string will have its center at the pen position, TJ_?L causes text to have its leftmost character at the pen position, TJ_?R justifies relative to the rightmost character.

setplotchar(c)
Sets the plot symbol to c. If c is a printing character, the ascii character c is the plot symbol. If c is a non-printing character, special characters such as crosses, squares, etc. can be set. If c==NO_CHAR (0), no symbols are plotted. If another character is set, the specified symbol will be plotted on calls to plot. The presently implemented special characters are:

description	number	#define'd symbol
plus sign	1	C_PLUS
hollow square	2	C_SQUARE
hollow diamond	3	C_DIAMOND
asterisk	4	C_STAR
solid circle	5	C_SLD_CIRCLE
hollow circle	6	C_CIRCLE
solid square	7	C_SLD_SQUARE
diagonal cross (X)	8	C_CROSS
solid diamond	9	C_SLD_DIAMOND

The advantage of these characters is that they are centered on the pen location.

erase(n)
Erases the plotting area. On the HP, n is a function code:
n=0, Clear screen. (E_NOFCN)
n=1, Beep and wait for any character, then clear screen. (E_PAUSE)
n=2, Perform HARDCOPY operation, then clear screen. (E_HCOPY)
n=3, Beep and wait, then perform HARDCOPY, then clear screen. (E_PAUSE | E_HCOPY)
On the versatec, all the function codes perform the same operation: simulate a form feed.

HIGHER LEVEL FUNCTIONS

xaxis (x,y,length,tic_count [,tic_length,tic_dir])
yaxis (. . . . .)

default will be used.

Proposed:

```
drawaxes(xtype,xtics, xleft,xcenter,xright,
         ytype,ytics, yleft,ycenter,yright);
```

Draws, labels, and puts grids on a set of axes. The axes are placed on the edges of the present viewport. xtype and ytype are one of LINEAR, LOG. x(y)left, ... are titles to be placed on the lower x axes in direction TD_R0, and the left y axes in direction TD_R270. The axes will be in LT_SOLID, the grids in LT_DOTTED. The formats for the numeric labels will be the defaults as specified in label. The tics will be IN. This proposed do-it-all routine will hopefully provide a quick way of specifying the most commonly used axes formats. (Not yet implemented).

THE DRIVERS

The routines described above write out device-independent code sequences on the standard output. To display these, the output must be piped into one of the device-dependent drivers "vplot", "qvplot", or "hpplot". The output can also be saved in a file for later plotting:

```
% yourprog | driver
:           Plot.

% yourprog > plotfile
:           Save device-independent plot.

% driver < plotfile
:           Plot a previously saved plot.
```

The three drivers are described in detail below, in short. hpplot plots on the HP, vplot and qvplot on the Versatec.

HPPLOT

The driver "hpplot" is used to produce output on an HP terminal, which must be the controlling terminal. Several HP specific functions are subsumed within the "erase" routine function code (see above).

Note that:

```
% yourprog | hpplot > file ; cat file
```

will not work, because hpplot engages in an answerback conversation with the HP to drive it at the maximum rate possible. The preceding command sequence will hang, because the pipe will not answer back.

VPLOT

Vplot is used to produce full-page output on the Versatec. It produces a bit map of the Versatec page in the file /jnk/raster, and when an "erase" or a "closeplot" is done, if anything has been plotted, this file is sent to the Versatec (device /dev/vc).

The plottable area of the Versatec screen is 1560 by 2048 pixels, since this is not in the ratio 1x2, as is the HP screen, plots which fill the HP screen will be truncated on the right when sent to the Versatec. To deal with this problem, vplot supports an "HP compatibility mode", invoked with the -h switch:

```
% yourprog | vplot -h
```

In this mode, the plottable region is a 1024-by-2048-pixel, centered region, and HP plots will fit untruncated. Note that this wastes about 1/3 of the Versatec page, and results in a corresponding loss of resolution.

QVPLOT

```
    openplot();
    setviewport(200,200,3800,7900);   /* leave a little space */
                                      /* for labels          */
    setwindow(0.0,0.0, 1.0,1.0);
    xaxis(0.0,0.0, 1.0, 10,80,IN);    /* 10 tics of length 80 */
    label();                          /* use defaults         */
    title("X Axis");                  /* Centered title       */
    yaxis(0.0,0.0, 1.0, 5,80);        /* tics IN by default   */
    label();
    title("LYaxis","CYaxis","RYaxis",TD_R270);
    hgrid();
    vgrid();                          /* use defaults         */
    xaxis(0.0,1.0, 1.0);              /* no tics              */
    yaxis(1.0,0.0, 1.0);
    setlinetype(LT_DOTPLOT);          /* want to do dot plots */
    for (n=1000; n; --n)              /* for a while ...      */
        draw(ran(),ran());            /* make DOTS            */
    setextjust(TJ_CC);                /* set center text just */
    setextsize(2);
    move(0.5,0.8);
    print("Random Plot");
    closeplot();
}
```

To run this program:

```
% cc prog.c -lP -lS -o prog
    - then -
% prog | hpplot
    - or -
% prog | vplot
    - or -
% prog | vplot -h
    - or -
% prog | qvplot
```

GENERAL NOTES

* The plotting routines are not yet in a library; the compiling command is:

```
% cc prog.c /usr/eric/Plot/basic.o /usr/eric/Plot/axes.o¢
           /usr/eric/Plot/label.o -lS -o prog
```

If no labeling is to be done, label.o need not be linked; similarly for axes.o.

* The #include file "plot.h" is not on /sys/include; programs must use:

```
    #include "/usr/eric/Plot/plot.h"
```

instead of:

```
    #include <plot.h>
```

* HPPLOT ignores linewidth changes. The axes routines set the line width to 2 for axes; this looks better on the Versatec, but would not be good on the HP. QVPLOT also ignores linewidth changes; it doubles everything anyway.

* VPLOT and QVPLOT do not know about vertical text justification. They treat all text justifications perpendicular to the text direction as centered.

* "Print" calls are not clipped.