September 1982

Plaster Users' Guide

Peter Karp
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-82-00.

Plaster Users' Guide

Abstract

Creating graphics for TV and storage tube displays is common and easy at most computer installations today. It is less easy to produce high-quality hard-copy of graphics for publication. This manual describes a program called Plaster which produces a raster representation of a picture created with the PlotIO graphics package for display on the Santec S700 printer. Plaster (PlotIO to Raster) thus allows one to obtain hard-copy of an image created with the prevalent and easy to use PlotIO language on the inexpensive but high quality Santec printer.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-82-00.

#150

Plaster User's Guide

Peter Karp

September 1982

Creating graphics for TV and storage tube displays is common and easy at most computer installations today. It is less easy to produce high-quality hard-copy of graphics for publication. This manual describes a program called Plaster which produces a raster representation of a picture created with the Plot10 graphics package for display on the Santec S700 printer. Plaster (Plot10 to Raster) thus allows one to obtain hard-copy of an image created with the prevalent and easy to use Plot10 language on the inexpensive but high quality Santec printer.

Plaster can accept input and produce output in several different forms.

The most common use of Plaster would be to feed it a file created by Plot10, and have it create a sequence of commands for the printer. These commands could either be sent immediately to the printer, or written to a file. The user could later send the file to the printer to print an image. In both of these modes Plaster really performs two distinct steps: it converts the Plot10 input file to a raster representation of the graphic image, and then converts the raster to the sequence of printer commands needed to display it. It is possible to both start and stop this process in the middle. That is, one can instruct Plaster to write its internal raster to a file, and one can instruct Plaster to read a raster from a file and then display it on the printer. This allows one to display the raster created by Plaster on another raster-oriented graphics device, or to display a raster from some other source on the S700 printer.

These possibilities are summarized below:

```
Plot10 file --> Printer image
Plot10 file --> Printer data file
Plot10 file --> Raster data file

Raster data file --> Printer image
Raster data file --> Printer data file
```

We will now consider each of the above components seperately and in more detail.

Creating A Picture From A Plot10 File

Plaster will print any picture you have created with Plot10 on the S700. There are no major limitations or restrictions on the pictures you can display. But there are a few hints and tricks which should be mentioned.

Most users of Plaster will probably have a Plot10 graphics device such as a Tektronix storage tube display (an alternate graphics device) available. Plot10 programs, like any others, will require some debugging before an acceptable result (the picture) is obtained. It is suggested that such picture debugging, in which an evolving picture must be viewed many times, be done with the alternate graphics device. It will most likely allow for faster display and require less computer time, noise, paper, and ribbon.

The integer device coordinate system addressable by Plot10 ranges from 0-1023 in the X direction and 0-1023 in the Y direction. However, some alternate graphics devices (such as the Tektronix 4010 display used by the author) are only able to display some subset of this range. For example, the 4010 only displays from 0-1023 along X and 0-767 along Y. The person who installed Plaster on your system should have included such hardware limitations into Plaster's assumptions about the size of your display. If you try to write vectors outside the range of your hardware's display capabilities, but within Plot10's software limits, Plaster's actions are unpredictable.

Plot10 allows you to define both a viewport (or region) of the screen for your entire picture to exist in, and a real coordinate system for you to draw your picture in. This is no problem for Plaster. Since you select a mapping in size from the screen (integer device coordinates) to the page, any viewport you select within the screen is automatically mapped to the corresponding region within the part of the page you have selected. Since Plot10 internally converts all real coordinates into integer device coordinates, Plaster never encounters the real coordinates.

Once you have created an acceptable picture, you must place the character command stream generated by Plot10 to display your picture on your alternate graphics device in a file for use by Plaster. Plaster will attempt to read this file through Fortran logical unit number 9. Thus, before you run Plaster you must associate this logical unit with your input file through your operating system.

Sending A Picture To The Printer

In the discussion that follows we assume that one of two hardware configurations is used to communicate with the printer: the printer is attached to the user's terminal through a printer port, or the printer is attached to the host through a seperate port. Thus, to communicate with the printer Plaster must communicate with either your terminal or the seperate line. Plaster is capable of using two different communication methods: the Fortran I/O system and a stream I/O system. Plaster is able to use the former on any computer. Using the latter requires that Plaster be specially modified to run on each different computer. It is important to know if your version of Plaster can use stream I/O. To communicate with the printer in its

ESL Plaster must be able to use stream I/O. If it cannot use stream I/O you are limited to using RCL, which presumably is not a serious limitation. If Plaster is able to use stream I/O it will do so by default for both ESL and RCL, regardless of which command language you select at run time. If Plaster is able to use stream I/O you need make no special arrangements with the operating system before running Plaster (unless you are instructed to do so by the person who installed Plaster at your site). If, on the other hand, you are using Fortran I/O to communicate with the printer, you must know that Plaster uses unit number 10 for printer output. Thus, when you run Plaster you must associate unit 10 with either your terminal or the seperate printer line through an operating system command.

Sending A Picture To A File

The sequence of commands which instruct the printer to produce a given picture can be stored in a file rather than being sent immediately to the terminal. To store commands in a file, they must be sent through the Fortran I/O system. Thus, since printer commands in ESL can only be sent through the Stream I/O system, it follows quite logically that a sequence of ESL commands cannot be stored in a file.

If your version of Plaster is able to use Stream I/O it will do so by default. Thus, to force Plaster to use the Fortran I/O system you must override this default through use of the USE_DEFAULT_IO_SYSTEM parameter. Plaster uses Fortran logical unit number 10 for printer output, so you should assign this logical unit to the file you wish to store the printer commands in with the operating system before running Plaster. If your printer is hung on a terminal printer port, you print the picture stored in the file by listing the contents of the file on the terminal and turning on the printer port. Sending a file of commands to a printer on a seperate line may be more difficult and is highly site-specific.

Storing The Internal Raster In A File

Enabling the SAVE_RASTER option will cause Plaster to write the internal raster to a file. The file will be written through Fortran logical unit number 13, so you should assoicate this unit with the file you wish to write to through the operating system before executing Plaster. Note that this option does not affect whether or not the picture is sent to the printer - that option is controlled by

the PRINT parameter. See Appendix I for a description of the raster file format.

Reading A Raster File From A File

Enabling the READ_RASTER option causes Plaster to read a raster from a raster data file. The file will be read through Fortran logical unit number 13, so the file to be read should be assoicated with this unit. See Appendix I for a description of the raster data file format. Note that Plaster assumes that the structure of a given raster file is the same as the current setting of the ROW_MAJOR option. If you try to write files with one type of organization and read them with the other (e.g., write them in row-major order and read them in column-major) you will get a garbled picture.

The way Plaster functions with the READ_RASTER option in effect is significantly different from its operation in other circumstances, so it is worthwhile to spend some time discussing these differences. First it is important to mention why this feature was provided: it allows one to read a raster of a given size from a file, and then print it with any spacing desired between the fixed number of rows and columns in the raster. This makes it possible to display rasters from other graphics devices on the S700, and to adapt to the different possible resolutions on these devices by changing the relative X and Y dot spacing of the image on the S700.

In this mode the Plot10 file interpreting process is totally bypassed - the input to Plaster is the raster file, not a Plot10 file. Thus any parameters which affect the conversion of graphic data from a Plot10 file to a raster file have no meaning now. Examples are character rotation, image rotation, width, and page aspect ratio. It is important to note that the latter two parameters of an image are now determined by the dimensions of the raster read, and the resolution at which the raster is printed. In the descriptions of the parameters below, any parameters which have no meaning in READ_RASTER mode are marked with an asterisk.

Description Of Parameters

Plaster was designed to be simple and fast to use by a novice, yet to be flexible enough to meet an advanced user's needs. For this reason the program assumes default values for all its parameters so the user can produce a picture without being concerned about the value

of every parameter. In fact it is reasonable to create each picture the first time with the default parameters, and then go back and modify the appropriate parameters if you would like to change the picture's appearance. In the discussion below, any parameter marked with an asterisk (*) has no meaning when the READ_RASTER option is in effect. Here is a list of the various parameters and their functions:

Parameter: WIDTH *
Legal Range: 1. - 13. inches
Default value: 8. inches

Selects the width of the region of the page to which the picture is mapped, in the current unit system (inches or centimeters). Note that if the picture is rotated to one side (through the IMAGE_ROTATE parameter) this parameter still determines the width of the page image.

Parameter: PAGE_ASPECT_RATIO *
Legal Range: .2 - 5.
Default value: 1.

Selects the ratio of Y units to X units for the page image. These first two parameters define the final size of the picture on the page. A page aspect ratio of 1 will not distort the picture. A ratio of less than 1 will compress the picture in height, and a ratio of greater than 1 will stretch the picture vertically.

Parameter: HORIZONTAL_DOT_SPACING
Legal Range: 1 - 100
Default value: 8

Specifies the horizontal resolution of the image - the number of heps (1 hep = 1.041667 mils) between dots in the raster. Different resolutions require different numbers of passes at different printing speeds, and require different amounts of computer time, thus printing times vary. A spacing of 8 heps only requires one pass and gives relatively good resolution. A spacing of 16 heps requires one pass at high speed and thus is quite fast with a decrease in resolution. All spacings of more than 8 heps require one pass - the resolutions mentioned here are in a sense optimal since they provide good resolution at high speed.

Parameter: VERTICAL_DOT_SPACING
Legal Range: 1 - 50
Default value: 4

Specifies the vertical resolution of the image - the number of veps (1 vep = 3.472222 mils) between dots in the raster. Different resolutions require different numbers of passes and different quantities of computer time, thus printing times vary. A spacing of 4 veps (or a multiple of 4) only requires one pass and provides

relatively good resolution.

Parameter: CHAR_SET *
Legal Range: STICK, SCRIPT, ROMAN, GOTHIC
Default value: STICK

Selects the Plaster-generated character set in which text encountered in the Plot10 input file will be sketched on the page. The fancier sets will probably not have acceptable quality unless they are scaled above their default Plot10 size. Note that one image can only contain characters from one character set.

Parameter: CHAR_SCALE *
Legal Range: .1 - 10.
Default value: 1.

Selects a scale factor which is applied to characters generated by Plaster. This allows one to change the size of characters relative to the image. The default size is approximately equal to that generated by Plot10 displays. Note that you may have to adjust the placement of characters in the picture with Plot10 Move commands as their size changes. All charaters in one image are the same size.

For various complicated reasons related to character scaling Plaster will not deal with every possible use of character data. You should avoid issuing the following series of Plot10 commands: drawing a character string, issuing a move command to a new point whose Y coordinate is the same as that of the first string, and then drawing more characters. That is, you should not write a character string, then re-position the cursor to a point on the same line as that string, and draw another string. Because this is a very specific case it should not prove to be a very serious restriction. If you do need to draw two strings of text on the same line, simply insert other graphics commands (moves and/or draws) between the commands to draw the text. It is the specific sequence above that is ambiguous to Plaster.

Parameter: CHAR_ASPECT_RATIO *
Legal Range: .2 - 5.
Default value: 1.

A scale factor which is the ratio of width to height for characters generated by Plaster. Again, one aspect ratio is applied to all characters in one image.

Parameter: CHAR_ROTATION *
Legal Range: -90. - +90.
Default value: 0.

The number of degrees by which every character in an image will be rotated by Plaster. If you write a line of text the entire line will be rotated, making the page image look quite a bit different from the image you get on an alternate graphics device.

Parameter: IMAGE_ROTATION *
Legal Range: -1, 0, +1
Default value: 0

Will flip an image on its side for presentation "the long way" on a rectangular page. -1 flips left, +1 flips right, and 0 does not flip the image.

Parameter: INCHES *
Legal Range: YES, NO
Default value: YES

Specifies whether the number enterred for the WIDTH parameter is interpreted in inches or centimeters.

Parameter: PRINT
Legal Range: YES, NO
Default value: YES

Specifies whether or not a page image is to be generated from the raster and sent to the printer through the default I/O system.

Parameter: SAVE_RASTER *
Legal Range: YES, NO
Default value: NO

Specifies whether or not the raster produced is to be saved in a raster file.

Parameter: READ_RASTER
Legal Range: YES, NO
Default value: NO

Specifies whether or not Plaster should read a raster to be displayed from a raster file rather than displaying the contents of a Plot10 file.

Parameter: ROW_MAJOR
Legal Range: YES, NO
Default value: NO

Specifies whether Plaster should store its raster internally in row-major or column-major order. This parameter also determines how Plaster interprets a raster it reads from a raster file.

Parameter: PASS_ALGORITHM *
Legal Range: ALTERNATING, MULTIPASS
Default value: ALTERNATING

Specifies the algorithm Plaster should use in passing over the Plot10 input file. The Alternating method will probably be faster in most cases, though the Multipass algorithm may be faster for pictures which consist mainly of long vertical vectors.

Parameter: COMMAND_LANGUAGE
Legal Range: RCL, ESL
Default value: RCL

Specifies whether Plaster sends commands to the S700 in its Readable Command Language or its Escape Sequence Language. Note that to communicate in ESL Plaster must have been generated on your system with stream I/O capability. If Plaster is able to communicate through the stream I/O system then this will be the default I/O system.

Parameter: USE_DEFAULT_IO_SYSTEM
Legal Range: YES, NO
Default value: YES

If Plaster is able to communicate through stream I/O it will do so regardless of whether it is using RCL or ESL. If, however, you wish to send Plaster's output to a file rather than directly to the printer, Fortran I/O must be used. Thus, if Plaster would by default use stream I/O, but you want it to use Fortran I/O, disable the default (stream) I/O system. Of course, if Plaster is not able to use stream I/O, Fortran I/O will be the default.

Modifying Parameters

When first executed Plaster will issue a prompt for a command line. On this line you can specify a set of parameters which you wish to modify, along with their new values. Enter these specifications in the form of assignments of values to keywords as shown below. You may enter one or more assignments per line, seperated by commas, and you may spread several assignments out over several lines if you so desire. Indicate you are done modifying parameters by typing carriage-return on a line by itself (do this for the first line if you don't wish to enter any parameters). Space characters function as seperators on a line, and you can enter characters in upper or lower case.

Form of a command line:

<parameter name> = <value> [, <param> = <value> ....]

Anywhere you type the name of a parameter, or an option which is a word (such as a character set name) you may abreviate that name/word to its shortest unique abreviation (parameter names must be abreviated to at least two characters). For example, all of the following assignments are legal:

```
PRINT= YES
Prin = yE
pr=y
```

A sample set of command lines is:

```
width=6, horiz=4, ve=1
char_set=rom, char_rot = -45.
```

Error Messages

When Plaster detects an error condition it issues an error message describing that condition. Error messages have three componenets and have the following form:

```
xxxxx-Y-zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
```

where xxxxx is usually the name of the routine where the error condition was detected, Y is an error severity code, and zzzz is a phrase describing the error. Y can be one of: F (fatal), W (warning), and I (internal). Fatal messages were probably caused by the user and can be avoided by having the user change his selection of parameters or his input file (or occasionally by having more powerful version of the program created). Internal errors should never occur. They are the result of either bugs in the program or improper program installation. Warning messages inform the user that the program is perfectly able to create a picture from its input, but that it may not appear as the user expected for some reason. The routine names in the message should make it relatively easy to tell what Plaster was trying to do when it encountered the error. If a user encounters an internal-coded error message he should report it to the person who installed Plaster on his system, along with a detailed description of what he did to cause the error.

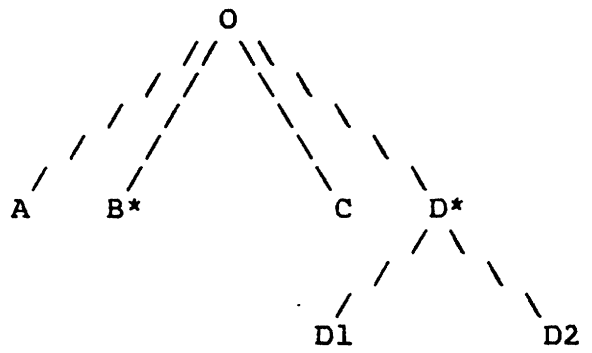
Appendix I - Raster Data File Format

Raster files are accessed by Plaster through Fortran logical unit number 13. These files are unformatted (binary), and are accessed sequentially. Three classes of data are stored within a raster file: a description of the raster, the raster itself (broken up into chunks, or bands), and comments or data specific to an individual user's application (which Plaster ignores). When Plaster creates a raster data file it normally writes only the first two types of data. However, if the user creates a raster file himself, Plaster is capable of ignoring extraneous (to it) data which the user may include.

The raster is bit-oriented - each pixel in the raster is represented as one bit. 1-bits are printed (black), 0 bits are whitespace. As mentioned the raster is stored as a set of seperate bands. Each band is a set of adjacent rows from the raster. Bands are stored in the raster file in order from the top of the raster to the bottom. The purpose of using bands is to reduce the memory requirements of the program. Each band consists of a string of bytes, where each byte contains a string of bits which lie sequentially in the raster. Obviously the term "sequentially" can have two different meanings: bits in adjacent rows or columns could be considered sequential. Thus it is important that organization of bits in each band be the same, and that it correspond with the way Plaster will interpret the data on input - which is determined by the ROW_MAJOR option.

For example, imagine a band which is 20 bits wide and 25 bits high. Such a raster requires 500 bits - and hence 63 bytes - of storage (with the last 4 bits in the last byte unused). If the organization is row-major, then the first 3 bytes contain all of the first row and part of the second.

The following is a description of the records in the file and what information each record should contain. All of the descriptive integers in this file are written in the default word size of the machine. The bands are arrays of bytes written with implied do loops.



The above is a tree diagram which represents the structure of componenets within the file (the node O is empty). Any component marked with an asterisk can be repeated a number of times. The components are as follows:

A - One record which contains one integer: the number of records to be skipped (A1).

B - A group of A1 records, where each record consists of a one word followed by a string of bytes, where the word specifies how many bytes follow.

C - One record containing two integers: the number of raster bands stored in the file (C1), and the width of each band.

D - The D component consists of two records. There are C1 D components in the file. The first record in each component (D1 above) contains two integers: the height of the band in the second record, and the number of bytes in the second record. The second record (D2 above) contains one band.