A major purpose of the Technical Information Center is to provide the broadest dissemination possible of information contained in DOE's Research and Development Reports to business, industry, the academic community, and federal, state and local governments.

Although a small portion of this report is not reproducible, it is being made available to expedite the availability of information on the research discussed herein.

LA-UR -89- 2828

LA--UR--89-2828

DE89 016986

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

TITLE

TIKTEK—A TEKTRONIX EMULATOR BASED ON 2D GMR

AUTHOR(S)

Kenneth A. Van Riper
Radiation Transport Group
Applied Theoretical Physics Division

SUBMITTED TO

Apollo Domain User's Society
8th Annual Conference
September 10–13, 1989
New Orleans, Louisiana

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

TIKTEK—A Tektronix Emulator based on 2D GMR

Kenneth A. Van Riper

Radiation Transport Group
Applied Theoretical Physics Division
Los Alamos National Laboratory
X-6, MS B226
PO BOX 1663, Los Alamos, NM 87545
505 667 8104
kvr@lanl.gov

A. INTRODUCTION

Our group at Los Alamos set up an Apollo network in 1986. We have 15 monochrome DN3000s and a few DN320s. These workstations replaced Tektronix and VT640 terminals. The majority of the work in our group is to develop and maintain programs on the Laboratory CRAY computers; for most of our users, the workstations serve primarily as a fancier terminal for communicating with the CRAYs. The supercomputer codes we use and support include a number of graphics programs which are designed to talk to Tektronix terminals. We thus require a good Tektronix emulation capability on the Apollos. (The day may come when the workstations and the CRAYs are a fully integrated network, with the supercomputer manipulating huge data files and the workstations doing the graphics locally, but that is years away. Once we have the capability, there is still a lot of work to be done to convert existing programs, and we must still support users with terminals.)

We were unable to find a Tektronix emulation product with which we were really satisfied. Our communication to the supercomputers is over the network; we cannot use the SIO ports. That prevented use of Apollo's 4014 emulator. A east coast company provided a 4125 emulator for evaluation. The problems we encountered, together with the vendor's priority for looking at those problems, helped us decide to pursue other options. We purchased a copy of a 4014 emulator from a California firm. This works, but has some flaky behaviour. It can be run in a window, but only if the cursor does not leave the window (it then dies). Hardcopy is by a screen dump, which takes a long time to print. In addition, this product is no longer supported by the vendor. A 4014 emulator comes with X Window xterm client. The version with X11.1 was hopeless. The X11.2 version was better, but did have some flaws. We did not then have the X Window in a window product, and a new windowing system would have been too much of a change for our users.

The only thing to do was to write a Tektronix emulator ourselves. Some colleagues in our group did it for another brand of workstation, so why shouldn't we do it for the Apollos? I was assisted by Ken Koch, who figured out most of the communications and event handling and tutored me on UNIX programming.

I named the emulator TIKTEK, mostly because my fingers needed practice in typing words with repeated letters. The code is written in C. The graphics uses DOMAIN/2D GMR calls, and the user interface is implemented with DOMAIN/DIALOGUE. I attempted to

use UNIX system calls as much as possible, but reverted to Apollo system calls when I could not understand the UNIX documentation.

This paper describes some of the internal design features of the TIKTEK and places where Apollo specific features helped or hindered.

B. REQUIREMENTS

Tektronix Flavors. Tektronix model numbers are used synonymously for the terminal types and the protocol supported by that terminal type. The least sophisticated protocol with which we are concerned is 4014e. This is basic monochrome graphics with support for hardware characters and patterned lines. The "e" signifies 4096 x 3072 resolution. The 4105 protocol is this plus 4 color planes (16 colors). 4115 protocol adds more color planes and lots of bells and whistles.

Emulation. A Tektronix emulator must, of course, faithfully interpret and execute Tektronix graphics code (other features, such as the writing of straight text, need not be exactly reproduced). Since the Apollos in our group are monochrome and color is the only 4105/4115 feature taken advantage of by the graphics programs we use, it was sufficient to implement the 4014 protocol at first. Color was added later for the benefit of the privileged users with color nodes elsewhere in the laboratory.

Windows. The program must run in a Display Manager window of arbitrary size, with the full image showing. This naturally pointed to DOMAIN 2D GMR. More than one invocation should be able to run at once, allowing, for example, side by side comparisons of different problems.

Hardcopy. Output should be by a PostScript file containing actual PostScript graphics commands, not just a screen dump. This is quicker on the printer, and changes can be made to the plot file if desired.

C. WINDOWS

Text in the graphics window is, at best, difficult to read when written in a font emulating the size of the Tektronix hardware characters. It becomes nearly impossible by the time the graphics window is small enough so that two can be up at once. One also loses the ability to scroll back to any textual information in the graphics window. To solve this, TIKTEK runs with the original input pad/transcript pad window and creates a separate graphics window.

Figures 1, 2, and 3 show screen dumps of two instances of TIKTEK running. In Figure 1, the options menu is popped up in the left instance; most of the icons in this menu are for further popups to set parameters.

Window Positions. TIKTEK defines 3 sets (graphics window + transcript/input window) of window positions. Sets 2 and 3 are designed for side by side invocations. A choice among these positions can be specified on the command line. The user may override these settings in the `tiktekrc` options file. The `tiktekrc` file is also used to specify other user preference settings; it is either in the working directory (searched first) or the user's home directory.

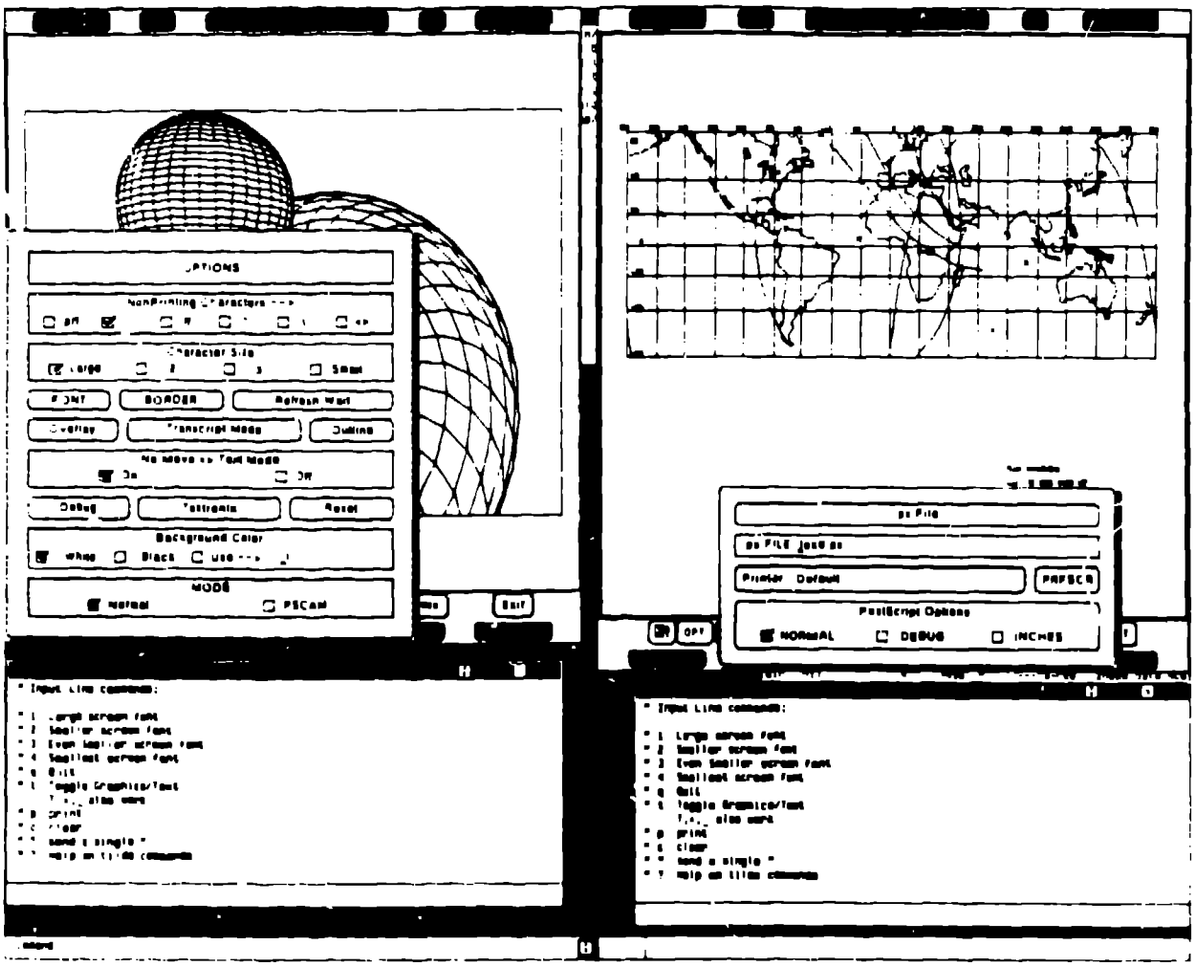


Figure 1. Screen Dump showing 2 instances of TIKTEK running in windows. The left TIKTEK has the options menu popped up. The icons in rows 4, 5, and 7 are for additional popups. The right TIKTEK has the print options menu popped up. The transcript pads show the TIKTEK commands which may be entered from the keyboard by using a escape

Keyboard Input. Typing is done in the original input pad; keystrokes are ignored in the graphics area (except in GIN—Graphics INput—mode). This decision was brought about by the refresh capability of 2D GMR version 1 —the entire graphics area must be refreshed. It is disconcerting to not see anything as you are typing, and more annoying when then entire graphics area blinks as each character is entered. Version 2 permits the refresh of only a rectangle (around each letter); typing in the graphics area may enabled in the future.

Textual Output. Information that is really text and not part of any graphics should go to the transcript pad —eg., messages from the code that just made the plot. The problem is how to decide when characters are not part of the plot. TIKTEK decides this by looking for a MOVE command preceding any text in the current buffer of information being processed. If a MOVE command is found, the text is considered to be part of the plot and placed in the graphics area; if not, it is directed to the transcript pad. This strategy is not perfect. Occasionally some real text gets in the graphics area. This is usually just a prompt, but may also be some statistics following a plot. Text which is really part of the plot may wind up in the transcript pad if the plot arrives in multiple buffers broken at just the right place; this text may also end up appearing as a prompt in the input pad since TIKTEK interprets short text strings without a carriage return as prompts. (The 4115 protocol allows for direction of textual information to one of several "dialog" areas; the enable/disable dialog commands will be used to decide the destination of data in a future version).

The user may turn off this business of looking for a MOVE in front of text by an entry in an options file or by a pop up menu during execution. All output is then directed to either the graphics area or transcript pad. The user may switch the target output area with the mouse or keyboard. TIKTEK automatically directs output to the graphics area when a graphics escape code is found. A text/graphics icon informs the user of the current destination.

Window Poping. The user may wish to size the graphics and transcript windows so that they overlap each other. A `pad_$pop_push_window` call ensures the graphics window is on top before each refresh and also at termination. The input/transcript window will pop to the top when the user types there because of the normal Display Manager handling of cooked windows. This can be annoying. Since one can type in a raw window and not have it pop to the top, TIKTEK has an option to turn the transcript window to raw mode. This needs a little more work there is trouble with recognizing and sending carriage returns and echoing what is typed.

D. INPUT HANDLING

TIKTEK must simultaneously monitor the keyboard, DIALOGUE events, input from the remote computer, and, when required, graphics events. Event Counts are used for this. This is pretty much straightforward, once you understand how to use event counts. Pipes are set up, with standard UNIX calls, to pass information to and from the shell (and thence on to the remote computer). The input pipe is monitored and read with STREAM calls. (IOS calls should be used, but the examples for using event counts used the stream calls.)

Input Buffering. Buffered reads are used for handling input from the pipe; the current buffer size is 1096. A buffer is read, processed, and further buffers from the pipe are

read and processed until the pipe is empty. Only then does TIKTEK check for other types of input. Thus an interrupt cannot be sent while a plot is being displayed. In addition to our internal buffering of the input pipe, the CRAYs do their own buffering of the plot commands.

The buffering of the Tektronix code coming into TIKTEK can cause problems when the data is split in the middle of a command. The 4014e command for specifying an (x,y) pair, for example, comes in a variable number of characters, with the magnitude of the first characters serving as a flag as to whether more are coming. This information must be saved between the pipe buffers. The situation is more extreme with the 4115 protocol, where there are many commands with variable length data. Text which should be in the plot can appear in the transcript pad if the positioning command preceding the text is at the end of the previous buffer, as is described elsewhere.

Refreshes. How often should the graphics area be refreshed during transmission of a plot? It is inefficient and visually distracting to refresh the display every time a graphics command is received. TIKTEK does a refresh when it finds, after reading from the pipe, that the pipe is empty.

There is also an optional refresh between buffers—a refresh is done if the time since the last refresh is greater than the refresh wait time. The default is 5 seconds, but the user may change this in the `tiktekrc` options file or in the options menu during execution. A smaller value can be used if the user needs to be reassured that something is going on.

Lights. A RCV (receive) icon lights up when information is coming over the pipe and stays on until all of the data is processed; it does not care about the buffering. The left window in Figure 3 shows TIKTEK in the middle of receiving a plot. Information from the keyboard going the other way triggers a SND (send) "light".

E. ZOOMING AND TRANSLATION

It is desirable to be able to enlarge a portion of a plot when the image is in a small window; when only a portion of the image is showing, it is also desirable to be able to translate the view. Both of these features were easily implemented with DOMAIN 2D GMR; the speed of execution is very satisfactory. Figures 2 and 3 show the image(s) in Figure 1 zoomed and translated.

Zoom. To zoom, the user selects the zoom icon (magnifying glass). Another icon lights up to remind the user TIKTEK is looking for zoom input. The user selects a point with the mouse, and grows a box while holding down the button. The image is enlarged so that one dimension of the box fills the window; the image is not distorted. There appears to be some limit to magnification beyond which nothing will be drawn in the graphics window; I have yet to track down what is going on here.

Translation. Translation—up, down, right, and left—is done with either icons or the boxed arrow keys. The same keys, shifted, move the image by 1/10 the normal amount. The middle mouse button on one of the translation icons pops up a menu, shown in Figure 2, for amount to translate in fractions of the current view. An initial value for this fraction may also be set in the `tiktekrc` options file.

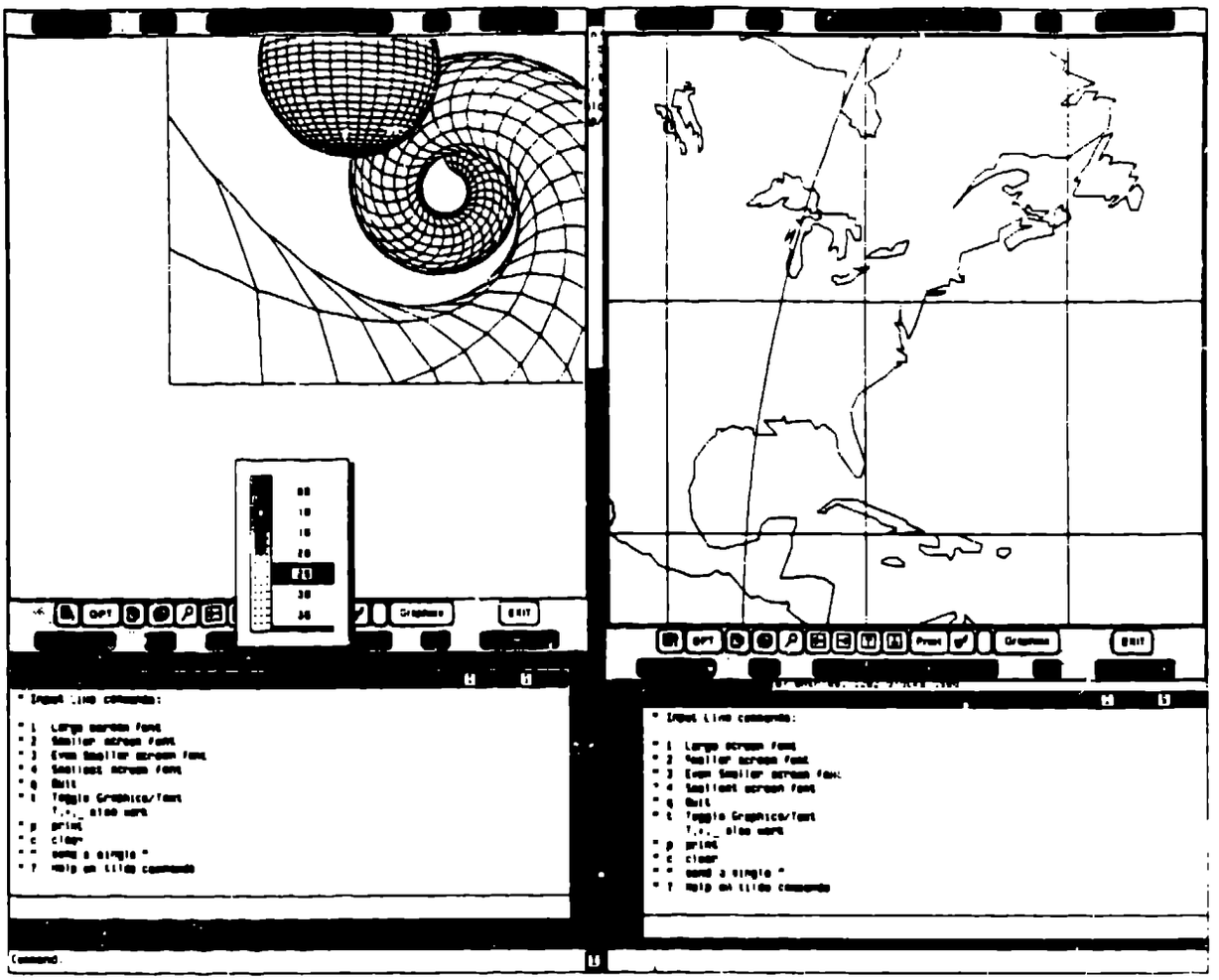


Figure 2. Screen dump of the same setup shown in Figure 1. The image in the left TIKTEK has been translated up and to the right. The popup menu shown is for selecting the fraction of the graphics window with which the image is moved in a translation operation (selecting or typing boxed arrow). The image in the right TIKTEK has been enlarged.

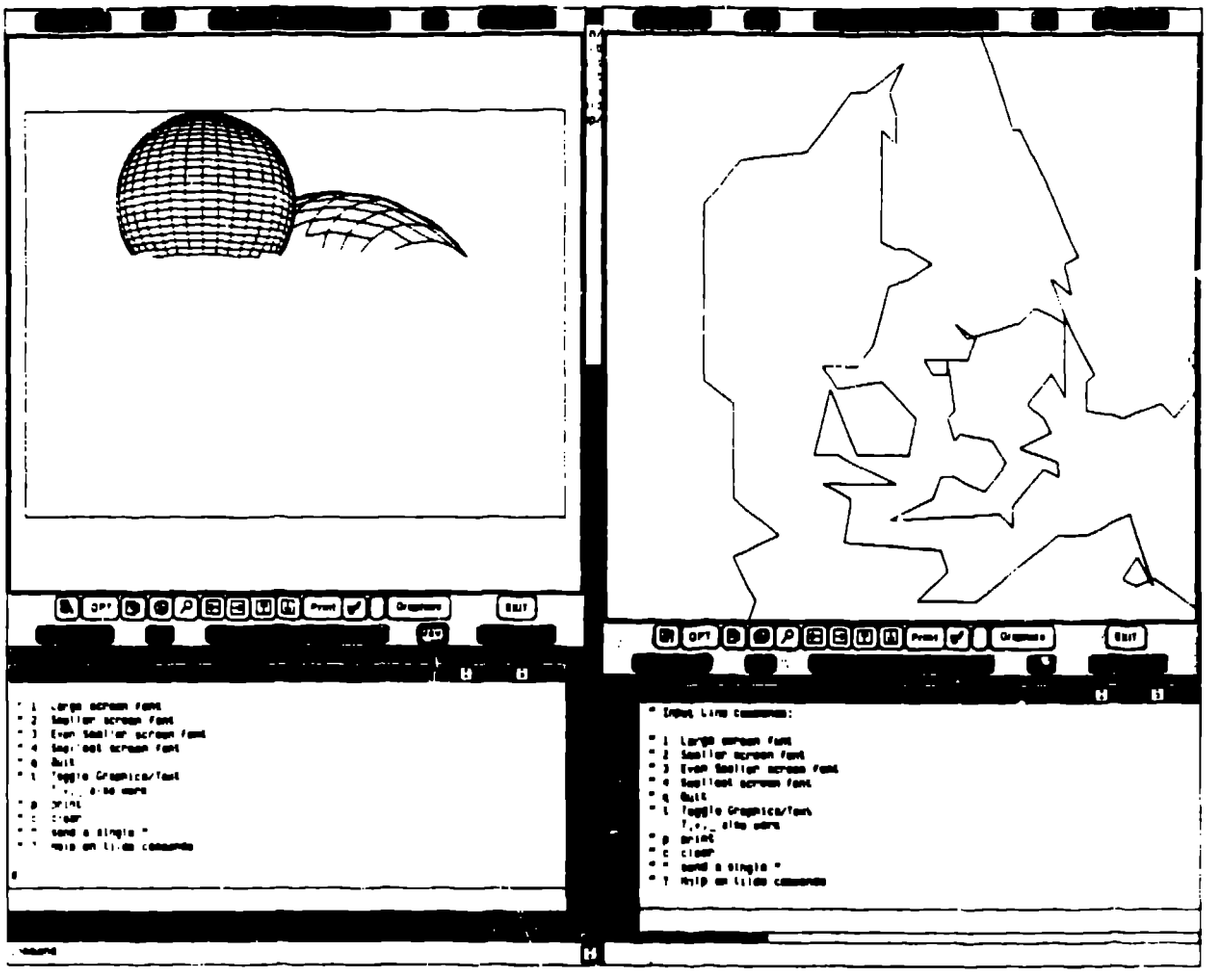


Figure 3. Screen dump of the same setup shown in Figures 1 and 2. The left TIKTEK is in the process of receiving a plot – note the receive light in the bottom icon row. The image in the right TIKTEK has been enlarged even more.

Reset. A reset icon cancels both the zooming and translations all the way back to the original image.

There are icons for refreshing the graphics area and for clearing it. I am working on a printing option to print only the portion of the plot shown.

F. PRINTING

Method. Hardcopy is done by making a PostScript file and sending it to a printer. The PostScript file is generated only when requested, rather than producing PostScript code on the fly for each plot displayed. When hardcopy or a PostScript file is requested, a routine reads through the gm metafile with `gm_$pick` commands and generates the appropriate PostScript code for each graphics command. This routine is based on a conversion to C of `debug_gmr.pas`, a debugging program distributed with 2D GMR version 2.

Reasons. The Apollo call `gm_$print_file` with the `gm_$postscript` option was not used for several reasons: I do not understand the units in its output, so it would have taken much trial and error to figure out how to rotate the image to landscape mode (the natural orientation for Tektronix plots) or to find the settings for placing 2 plots side by side. I wanted the option to use real inches as the units (TIKTEK 's default units are the Tektronix coordinates) in case a user wished to modify the PostScript file using a ruler and a text editor. I also wanted to feed the PostScript file to my GRAPHIT utility for display of the file and possible modification; GRAPHIT expects standard ADOBE comments, recognizes only a subset of PostScript commands, and cannot decode macro definitions (the macros I use are hardwired).

Print Icon. Hardcopy is made and sent to the printer by selecting the print icon or by typing "~ p". There is a noticeable, although acceptably short, wait. The PostScript generation and the issuing of the print command is not done in the background. This prevents the user from altering the metafile, and ensures a temporary PostScript file can be gracefully deleted.

The M2 button on the print icon pops up a menu, shown in the right window in Figure 1, which allows a choice of printer, generation of a PostScript file (which is not printed and stays around), and screen copies. The user may enter a name for the PostScript file, or except the default name, which is the next available in the series `tek[0,1,2,...].ps`. TIKTEK will complain if 99 or more such files are found. The printer menu contains the printer names which will follow the `-pr` option to `prf`.

Commands. The commands for printing and doing a screen copy are read from a configuration file, as are the choices for printers. This allows for easy customization at different sites.

G. GM METAFILES

Each screen is a single metafile. Upon a page clear command—issued by the application or the user—the old metafile is pitched and a new one created. Everything is in a single segment. The 4115 protocol has segmentation capability analogous to that of 2D GMR, but emulation of that capability is not yet implemented.

Saved Metafiles. The user may save the current metafile. As with a saved PostScript file, the user may either accept the default name or type one in. I have made a separate program, GM_DISPLAY , to display metafiles. GM_DISPLAY provides the zooming, translation, printing, and some other capabilities of TIKTEK —they share some code. GM_DISPLAY can be invoked from TIKTEK by choosing the save icon. It presents the user with a menu of all files with a .gm suffix in the current directory. One of these may be displayed, or the user can add a file name to the menu.

Metafile Name. Since TIKTEK may be running in more than one window, a unique name must be chosen for the metafile. This name is `tek_temporary[0-99].gm`, where `0-99` is the first integer for which a file does not already exist. This file is deleted when TIKTEK terminates gracefully, but will be left around if TIKTEK crashes. A warning message is issued if TIKTEK detects more than 5 of these temporary files. It should be possible to query if these files are locked when looking for a free name, and delete those which are not in use.

H. DOMAIN 2D GMR VERSION 2

The basic graphics in TITKEK was developed using version 1 of DOMAIN 2D GMR. I had to move to version 2 when I began working on an F monitor (high resolution color).

Good Features. There are some useful improvements in version 2. The ability to refresh only a portion of the graphics area after writing a bit of text is very helpful. Specifying line thickness is necessary for emulating the corresponding 4115 command.

Fonts. Handling fonts through environment files has turned out to be a real problem. I wanted to provide a variety of fonts in order to get some feeling for what looked best. I made pixel font families based on the fWxH series, times-roman, helvetica, and the bold versions of these 3. Each family has about 10 members. I also wanted to try stroke fonts of varying complexity; since I had stroke fonts for gothic, script, greek, etc., I included these in the options just for fun. Under version 1, I would just exclude the current font family and include the new one in response to the font menu. It worked flawlessly.

Environment File. Under version 2, one must set up an environment file which references font families, and specify a font by setting the font family index. So I made a program, patterned after `/domain_examples/gmr2d/envfile.pas`, to build an environment file referencing 6 pixel font families and a dozen stroke font families. The program died after a few calls to `gm_$font_family_defn`. After some trial and error (I did not have documentation for version 2 that took a while - and I was working over a long weeken I - no one to help at the Response Center), I figured the best strategy was to make a separate environment file for each font family.

Not only did I have to carry around a dozen font family files for the stroke fonts, each of which contains a single line, but now there are 18 environment files to clutter up the directory structure. This seemed like a giant step backwards. I also could not figure out how the change from one environment file to another during execution. I gave up for a while and decided to wait and see if the documentation could give me any clues (it did not help a whole lot). I was also tired of trying to figure things out from the release notes, the examples, and the insert files.

Convert. The metafile format is different; my version 1 stroke font files would not work, according to the release notes. I tried regenerating them with version 2. My font creation program died in the middle for no obvious reason. I ran the `convert` program which comes with the version 2 distribution. That did work.

SR10. That was all under SR9.7 and 2D GMR version 2.2. I tried again under DOMAIN/OS 10.1, also with 2D GMR version 2.2. It got worse. Instead of programs just crashing when too many fonts were referenced, the node hung—completely dead with no option but the reset button. I did find out from the Response Center that there is a limit of 32 fonts, not font families, allowed in an environment file. If more than 32 fonts have been added, `gm_$font_family_defn` does not return a bad status; only after about 50 are added does the node get hung. The node also hung when I tried to close an environment file and open another. I tried putting more than one font family in an environment file, keeping under the 32 font limit. The node seemed to hang more often when more than a single font family was referenced per environment file. I have not yet tried to sort this out any further.

I reverted to a single environment file per font family. The font can not be changed once TIKTEK is running.

Line Patterns. Line and fill patterns are also defined in the environment file. The 4014 protocol has 5 fixed line patterns and no filling, so that is not a problem. The 4115 protocol does area fills, and lets the user define patterns as well as use predefined patterns. I have not yet implemented that capability.

My program for displaying PostScript files needs to define line patterns on the fly—the patterns are read from the PostScript code. It seems unnecessarily cumbersome to have to issue several commands (to add a pattern to an environment file and then a call to use it) where there was a single call before. Patterns can be reused, but it will take a lot of programming to remember if a specific pattern was encountered in the PostScript file before. For what I do, the extra space saved in the metafile by not having patterns defined there does not seem worth it all.

I. GRAPHICS FEATURES

GIN Mode. Tektronix terminals can pass graphics coordinates back the host computer. In response to a command to enter GIN (Graphics INput) mode, cross hairs appear. The user positions these with thumbwheels and types a key, sending that character and the coordinates. The host may also silently enquire the current cursor position.

Cross Hairs. TIKTEK draws cross hairs using the 2D GMR rubberband mode. The cross hair figure must be a single connected line, and no segments of the line can overlap

(else they cancel each other and nothing shows up). The figure is a box with two extended sides as shown in Figure 4.

Once the desired position is chosen, the coordinates and a character are sent when a button or key is pressed. The left mouse button sends an "a" (most applications accept any character), the right button sends a "." (ends GIN mode in a lot of applications), and a keystroke sends that character.

Users would prefer the cross hairs to reappear at the position sent in the previous GIN response. This may be possible, but it would not be easy. The position is set by the cursor position, which can only be controlled with global display manager commands rather than commands specific to the graphics area which already know about the segment coordinates.

Outline. A rectangle outlining the Tektronix graphics area is shown in the graphics window. The user may control whether the outline is there with the `tiktekrc` options file or from the options menu at runtime. The space between the Tektronix graphics area and the edge of the graphics window can also be set in the options menu.

Overlay. An overlay setting chosen from the options menu tells TIKTEK to ignore Page (Page Erase) commands from the application program (the local CLEAR command still works). The user can then plot one graph on top of another. The hardcopy will show all overlayed plots.

Character Size. A Tektronix 4014 has 4 sizes of hardware characters. TIKTEK lets the user set the current size used from the options menu or from the keyboard. The application program may reset the size at any time.

Reset. An application may leave the graphics state in an undesirable condition. A reset menu is provided to set the line style to solid, the line width to a single pixel, and text and draw colors to black (or white, depending on the background color). The resets affect only subsequent graphics commands.

J. OTHER FEATURES

Error Popup. In the event of a system call returning an error status, a popup appears for a second or two with an error message. The last popup may not go away until the user puts the cursor in it and moves out. Error messages also appear in the transcript pad. Most errors will, of course, be caused by improper user input, failure to read the documentation, or downright abuse. A highly visible error report may make the user more likely to fix the problem or seek help.

Character Filter. Control characters may come down the pipe. TIKTEK has a filter to look for and do something with these before writing to the transcript pad or graphics area. The default is to suppress any non printing characters. The user may choose from a menu to suppress form feeds as well, not to suppress anything, or to print representations of control characters with one of several formats.

Debug Menu. A debug set menu is available to turn on diagnostic prints. Debugging is available for the 4014, 4105, and 4115 commands, color map changes, events looked at by the event count manager, and the coarse program flow. Any or all may be on at once. (The print menu has its own debug choice). In addition to diagnostics on TIKTEK itself,

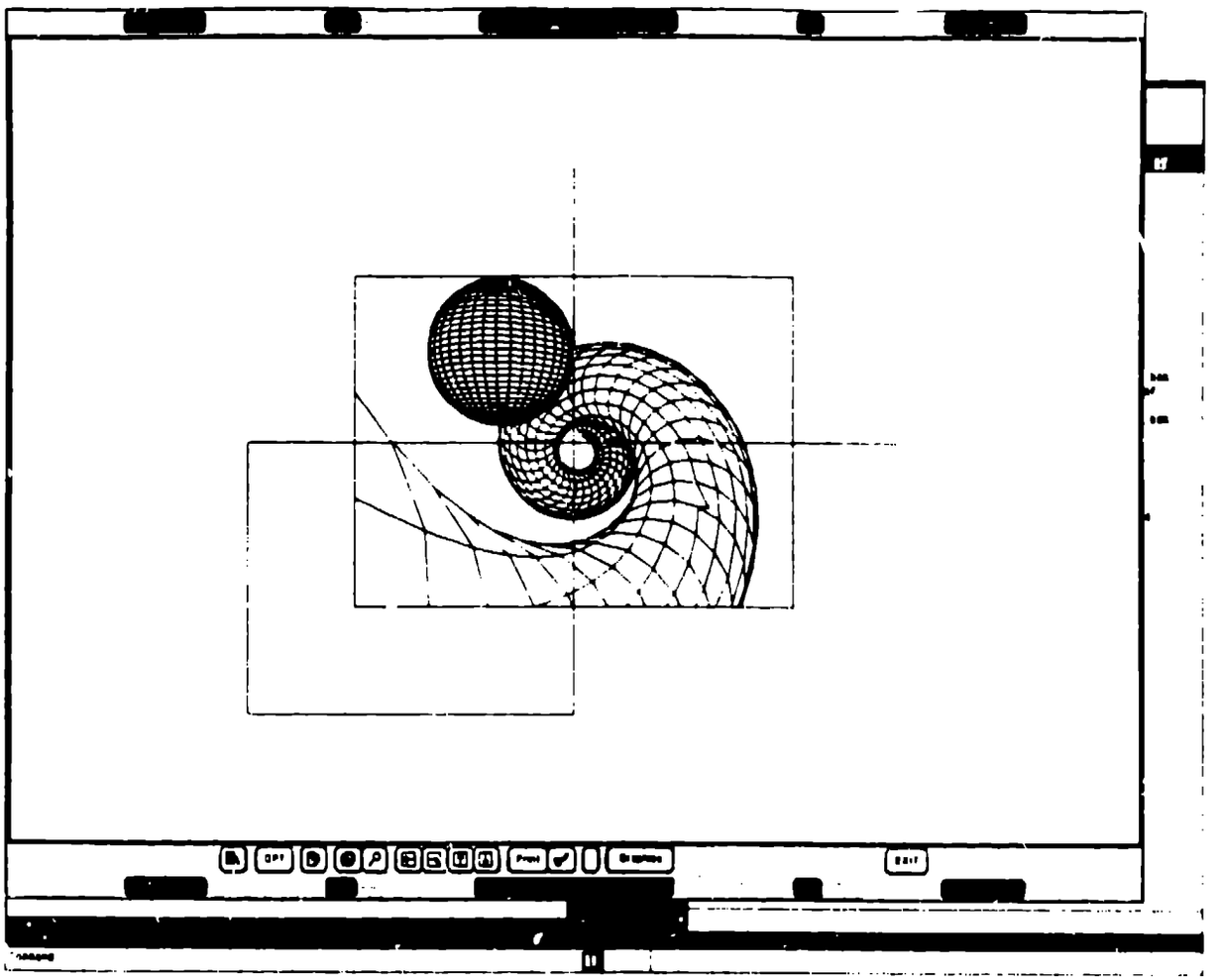


Figure 4. The Cross Hairs figure used for GIN mode. The point at which the 2 lines cross follows the cursor; the bottom left point of the figure is stationary. The left and bottom segments of the figure do not ordinarily appear; the image was shrunk by changing the shape of the graphics window several times.

the debugging features are useful for tracking down problems in the application program issuing the Tektronix commands.

K. CONNECTIONS TO OTHER HOSTS

The Tektronix commands interpreted by TIKTEK are usually generated on a remote computer. The `tiktek_remotes` configuration file permits automatic connections to these remote machines. The available machines and the command to get to them is specified in this file. (Available printers and the commands to print a PostScript file and do a screen dump are also specified in `tiktek_remotes`.)

In addition to the actual command used to connect to the remote host, this file contains the machine name and short description which appear in icons at the periphery of the graphics window, an ID which is used as a machine specifier on the TIKTEK command line, and a character corresponding to an icon for the remote computer in the font file `/usr/tiktek/remote_icons`.

TIKTEK may be invoked with a command line option asking for a connection to another computer. TIKTEK does not check if the user is allowed to log on to the other machine, nor does it not check to see if a connection was successfully made. The user may also invoke a local shell (`/bin/csh`) instead.

Switching Hosts. The user may switch to a different remote machine by selecting the CONNECT icon which sends a <ctrl D> (alternative logoff commands can be implemented if required). The icon changes to the disconnected icon (plug out of socket). The M2 button pops up the remote machine menu from which a new connection can be chosen. The appropriate command is be set upon this selection. While connected (CONNECT icon showing plug in socket), M2 sends a <RETURN>.

L. COLOR

Color Map. The Tektronix 4105 is a 4 plane terminal. The 16 predefined colors are put into the DM color map starting at slot 17 on a node with more than 4 color planes; the window colors are thus not disturbed. A 4 plane node gets its color map reloaded. TIKTEK's 4115 emulation is for 8 color planes. At startup, the first 240 colors are loaded into the upper DM color slots. If the application calls for a color index beyond 240, I load the remaining 16 colors into the first 16 DM color slots. This gives some awful looking windows with the default 4115 colors. I may implement a strategy of mapping the color slots to fix this. In both the 4105 and 4115 cases, the original color map is saved at startup and reloaded at termination. A fault handler should be implemented, which would reload the original color map as well as delete temporary files.

Emulation Mode. The number of color planes in the workstation is queried to set the Tektronix model to be emulated. A monochrome node defaults to 4014 emulation, a 4 plane node to 4105, and an 8 plane node to 4115. The user may switch the emulation mode from a menu.

Background Color. Text and draw values and the background color are monitored to see if the program is trying to draw black on black or white on white. If so, it changes the text or draw value to white on black (or black on white). No changes are made if the background color is other than black or white. Under DOMAIN/OS, the background color can be set to black, white, or any color slot. This can be specified in the `tiktekrc` options file, or from a menu during execution. The SR9 user is stuck with a black background. A monochrome display shows black on white or white on black depending on the INV setting.

M. INSTALLATION

In addition to the executable, TIKTEK requires 2 icon files, a 4115 color file, and a host of font family, stroke font, and environment files. Some sites may also wish to install the documentation. Ideally, the system administrator would be able choose the location of the TIKTEK directory. It would be possible to allow dynamic specification of paths in the C routines (reading, for instance, a shell environmental variable). There might be problems with the font family pathnames specified when building a 2D GMR environment file; I have not looked into that. But DIALOGUE requires a fixed path name for the icons it uses. Because of this limitation, I have not programmed flexible path names in the other parts of the TIKTEK code.

The TIKTEK directory *must* be `/usr/tiktek`. This at least keeps it out of the / level. Besides, we already have a `/usr/apollo` directory.

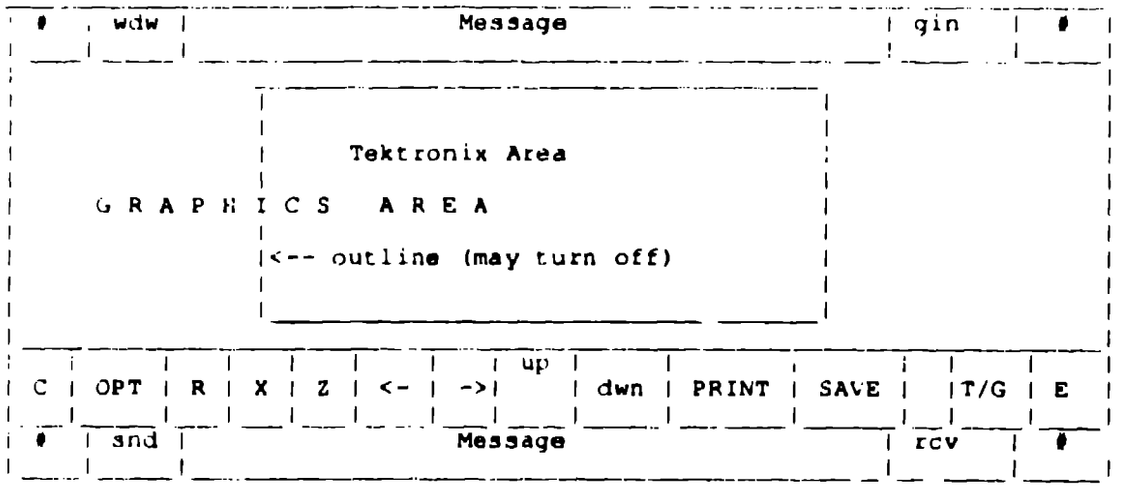
The system administrator should also set up a `/usr/tiktek/tiktek_remotes` file. This file contains information on remote computers and how to get to them, available printers, and the commands used for printing and screen copying. The user may override this by having a `tiktek_remotes` file in the working directory (searched first) or home directory.

N. SELECTED USER MANUAL PAGES

The next 6 pages are from the TIKTEK User Manual. The first 3 are a description of icons in the main graphics window. These are followed by a description of the command line options and samples of the `tiktek_remotes` and `tiktekrc` configuration files.

DISCLAIMER

THE GRAPHICS WINDOW



ID of the machine to which you are connected, or 'S' for local shell. This is set in the tiktek_remotes file. Display only, can not be selected.

Message Message, specified in the tiktek_remotes file, corresponding to the remote computer. Display only, can not be selected.

wdw Lets you know if a local zoom command needs to be completed. Display only, can not be selected.

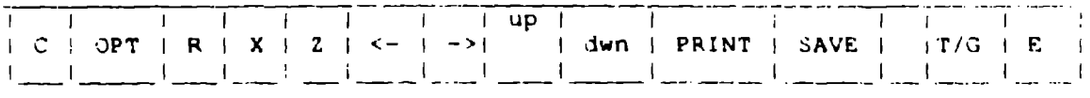
qin Lets you know if GIN (Graphics INput) mode is on. Cross Hairs appear in GIN mode. Left Mouse sends an 'a' to the CRAY; Right Mouse sends ','. Display only, can not be selected.

snd Lets you know when TIKTEK is sending information to the CRAY. Display only, can not be selected.

rcv Lets you know when TIKTEK is receiving and processing information from the CRAY. Display only, can not be selected.

Tektronix Area All graphics appear in this area. You may turn a box surrounding this area on and off. The size of the Tektronix Area adjusts to fit in the window while maintaining the proper perspective. The minimum BORDER between the Tektronix area and the edge of the Graphics Area may be changed (OPT icon).

Changing the window size (Grow or changing the special mode) will cause the Tektronix area to shrink. It will reset to the proper size on the Page or Clear operation.



C Lets you know if you are (plug is in) or are not (plug is out) connected.
Left Mouse Dis-Connects if connected (sends <ctrl>D),
Connects (same machine) if not connected.
Middle Mouse Sends <RETURN> if connected,
Lets you change CRAY if disconnect.

OPT OPTIONS POPUP---see below.

R REFRESH Refresnes the graphics. Clears spurious things like pieces of cross hair. Use if graphics goes away when window changes size or is covered.

X RESET (Cancel Magnify) Cancels local zooms and translations and restores view of full Tektronix Area .

Z ZOOM (Magnify) Lets you zoom in on a portion of the graphics display:

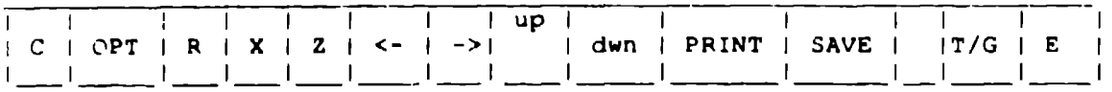
Select the icon.
Position the cursor at a corner of the area you wish to enlarge.
Press AND HOLD Left Mouse.
Move the cursor. A box appears outlining the area upon which to zoom.
Release the Left button.

More than you selected will be displayed as required to preserve the perspective (same X and Y scaling).
If you attempt to zoom in too much, nothing will appear.
RESET cancels ALL zooms, not just the latest.
The CRAY knows nothing about local zooms.
A CLEAR or PAGE (new page command from CRAY) cancels zoom.

<- -> up dwn SHIFT buttons
Shift the graph in the direction chosen (use the left arrow if you wish to move something currently at the left of the screen towards the center.

The <boxed-arrow> keys perform the same function.
The Shifted arrow keys move the view by 1/10 of the the Shift Amount.

SHIFT AMOUNT : The fraction of the graphics area by which the display moves may be set in the tiktekrc default file or by a menu called up by the Middle Mouse button. The amount chosen applies to all directions.



PRINT — LEFT MOUSE makes the image in the Tektronix Area appear on either your default printer or a another printer if you desire. The current graph is closed; nothing further may be added to it.

MIDDLE MOUSE Pops up a menu to set Print parameters and to make a PostScript file (not sent to printer) See below.

SAVE — Pops up a menu for making .GM files.

_ — CLEAR icon. CLEARS the Graphics Area.

T/G — TEXT / GRAPHICS
Output from Cray (or shell) goes to the Transcript Pad (text) or to the Graphics Area.
F8, F9 (while in graphics window) and ~T, ~t, ~*, ~_ on the input pad toggle this setting.
TIKTEK selects Graphics upon seeing a Tektronix command to enter graphics state.

COMMAND LINE OPTIONS

-W0 Use set 0 of window positions. This set is intended for a single invocation of TIKTEK.

-W1 Use set 1 of window positions.
-W2 Use set 2 of window positions.
Sets 1 and 2 are intended for side by side invocations of TIKTEK.

All default window positions may be overridden by settings in an initialization file (tiktekrc).

-M# Connect to the remote computer specified by the single character ID # (see description of tiktek_remotes file).

-c Do not connect, run a shell (/bin/csh).

-d DEBUG flag. Not meant for the general user.

tiktek_remotes FILE --- SAMPLE FILE

```
# File specifying remote machines and how to get to them
#
# ID is name on window border
# Name is what is used on tiktek command line
#
#           ID  Name  Icon-Char
r           XMP1  1     1
connect 1
R           XMP5  5     5
connect 5
5-Message
R            Y     y     y
connect y
UNICOS
#
printer ps3
printer ps2
printer ps52
#
command print
/usr/tiktek/prfps
#
command screen_copy
/usr/tiktek/prfscr
```

DEFAULT FILE --- tiktekrc

TIKTEK looks first in the local directory, and then in your home directory for a file name tiktekrc (lower case please). This file may contain lines like:

```
                        | <--  comments for documentation.
                                                  Do NOT put in file --> |

graphics top          1              position of graphics window
graphics left         1                  No number selected with -0
graphics width      500                     use for single invocation
graphics 0 height   400              '0' optional for set 0
transcript top      410              position of transcript/input window
transcript left       1
transcript width    500
transcript height   550
graphics 1 top        1       (etc.)       '1' selected with -1
transcript 1 top    400       (etc.)
graphics 2 top        1       (etc.)       '2' selected with -2
transcript 2 top    400       (etc.)        use 1 and 2 for 2 invocations at once
background            1       value (color) for graphics window background.
shift               .10       fraction of screen to shift image (box-arrow)
printer               6       printer for hard copy
pscan time            2       time limit for pscan     (LANL only)
pscan priority        5       priority for pscan       (LANL only)
outline           on/off      box around Tektronix graphics area
refresh               2       max time to wait between refreshes on parsing pipe.
```

Here are the DEFAULTS:

```
graphics    0 top          1
graphics    0 left         1
graphics    0 width     1000
graphics    0 height     700
transcript  0 top        702
transcript  0 left         1
transcript  0 width      600
transcript  0 height     270

graphics    1 top          1      graphics    2 top          1
graphics    1 left         1      graphics    2 left       602
graphics    1 width      600      graphics    2 width      600
graphics    1 height     520      graphics    2 height     520
transcript  1 top        522      transcript  2 top        522
transcript  1 left         1      transcript  2 left       602
transcript  1 width      600      transcript  2 width      600
transcript  1 height     468      transcript  2 height     468

background          1 (white) [0 (black) under SR9.7]

shift                 .25
printer           default
outline             on
refresh             5       (seconds)
```