

AD-A103 199

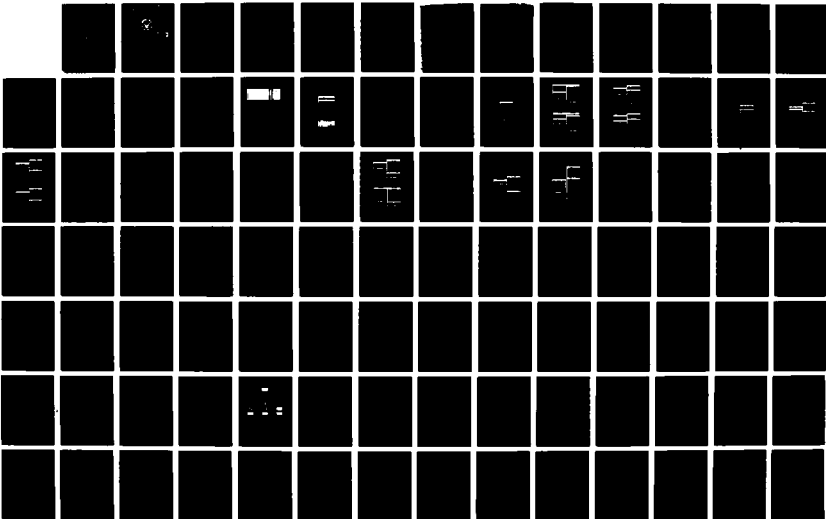
AN IMPROVED USER INTERFACE FOR AN INTERACTIVE GRAPHICS
FIGURE ILLUSTRATOR(U) NAVAL POSTGRADUATE SCHOOL
MONTEREY CA T J BEDA JUN 87

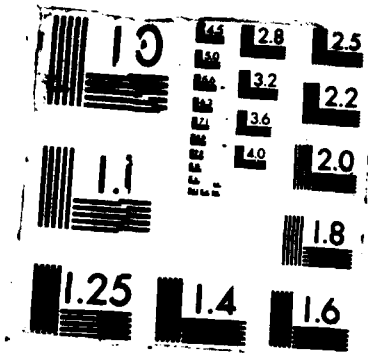
1/2

UNCLASSIFIED

F/G 12/5

NL





DTIC FILE COPY ②

NAVAL POSTGRADUATE SCHOOL

Monterey, California



AD-A183 199

DTIC
ELECTE
AUG 17 1987
S D
E

THESIS

AN IMPROVED USER INTERFACE FOR AN
INTERACTIVE GRAPHICS FIGURE ILLUSTRATOR

by

Thomas John Beda

June 1987

Thesis Advisor:

Michael J. Zyda

Approved for public release; distribution is unlimited.

unclassified

A183199

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION unclassified		1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY		3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE			
4 PERFORMING ORGANIZATION REPORT NUMBER(S)		5 MONITORING ORGANIZATION REPORT NUMBER(S)	
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b OFFICE SYMBOL (if applicable) 52	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000		7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	
8a NAME OF FUNDING/SPONSORING ORGANIZATION	8b OFFICE SYMBOL (if applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c ADDRESS (City, State, and ZIP Code)		10 SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO	PROJECT NO
		TASK NO	WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) AN IMPROVED USER INTERFACE FOR AN INTERACTIVE GRAPHICS FIGURE ILLUSTRATOR			
12 PERSONAL AUTHOR(S) Beda, Thomas John			
13a TYPE OF REPORT Master's Thesis	13b TIME COVERED FROM TO	14 DATE OF REPORT (Year, Month, Day) 1987 June	15 PAGE COUNT 160
16 SUPPLEMENTARY NOTATION			
17 COSATI CODES		18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP	
		computer graphics; interactive graphics figure illustrator; graphics; software maintenance; automated figure illustrator	
19 ABSTRACT (Continue on reverse if necessary and identify by block number) This study discusses briefly the history of technical figure illustration in the Computer Science Department of the Naval Postgraduate School. The single system experiencing the most usage is the fully automated interactive graphics figure illustrator--OZDRAW. During its short and active life, the need for perfective maintenance to include generalized documentation has been recognized. The result is a technical graphics figure illustrator with an improved user interface titled NPSDRAW and supporting documentation. <i>See also: Thomas J. Beda, NPSDRAW, NPS-87-001</i>			
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS		21 ABSTRACT SECURITY CLASSIFICATION unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL Prof. Michael J. Zyda		22b TELEPHONE (Include Area Code) (408) 646-2305	22c OFFICE SYMBOL Code 522k

Approved for public release; distribution is unlimited.

An Improved User Interface For An Interactive Graphics Figure Illustrator

by

Thomas John Beda
Lieutenant Commander, United States Navy
B. S., Purdue University, 1974

Submitted in partial fulfillment of the
requirements for the degree of

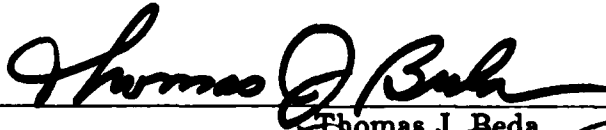
MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL

June 1987

Author:



Thomas J. Beda

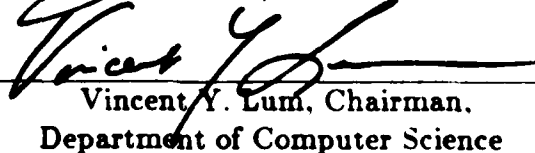
Approved by:



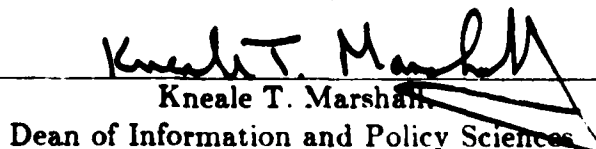
Michael J. Zyda, Thesis Advisor



C. Thomas Wu, Second Reader



Vincent Y. Lum, Chairman,
Department of Computer Science



Kneale T. Marshall,
Dean of Information and Policy Sciences

ABSTRACT

This study discusses briefly the history of technical figure illustration in the Computer Science Department of the Naval Postgraduate School. The single system experiencing the most usage is the fully automated interactive graphics figure illustrator--OZDRAW. During its short and active life, the need for perfective maintenance to include generalized documentation has been recognized. The result is a technical graphics figure illustrator with an improved user interface titled NPSDRAW and supporting documentation.

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

TABLE OF CONTENTS

I.	INTRODUCTION	8
	A. A NEED FOR AUTOMATED FIGURE ILLUSTRATORS	8
	B. SYSTEM DRAWBACKS	10
	1. PIC Graphics Language	10
	2. FIGURE Illustration Program	10
	3. OZDRAW Interactive Graphics Figure Illustrator	11
II.	NPSDRAW: SYSTEM OVERVIEW	13
	A. SYSTEM CAPABILITIES	13
	1. Primitives Supported by NPSDRAW	13
	2. NPSDRAW Figure Attributes	14
	3. The Screen Display	15
	4. The Printed Page	15
	5. The IRIS Mouse and Cursor	18
	B. MENU ORGANIZATION	18
	1. Main Menu	19
	2. Figure & Line Menus	23
	3. Single & Block Edit Menus	28
	4. Figure Attributes	30
	5. Clear Screen Menu	31

6. Read & Write File Menus	31
7. Line Style Menu	32
8. Line Width Menu	32
9. Font Source, Font Style and Font Sizes Menus	34
III. NPSDRAW - SOFTWARE IMPLEMENTATION DETAILS	36
A. BASIC GRAPHICS OPERATING SYSTEM	36
B. DATA STRUCTURES	37
1. Object	37
2. Drawing Struct	37
C. SOFTWARE IMPLEMENTATION	38
1. Initialization and General Functions	38
2. Figure Generation	41
3. Editing Functions	42
4. File Manipulation	47
5. Setting Attribute Values	49
6. Exiting The System	51
IV: NPSDRAW SOFTWARE MAINTENANCE	53
A. MAINTENANCE FACTORS	54
B. IMPROVEMENT OF DOCUMENTATION	55
C. PHYSICAL DIFFERENCES NPSDRAW / OZDRAW	56

VI. CONCLUSIONS	58
A. IMPROVED USER INTERFACE	58
1. Menu Presentation and Composition	58
2. Attribute Display Area	58
3. Font Selection	58
4. Texture Representation & Selection	59
5. Line Style and Width Representation & Selection	59
6. Figure Attribute Change	60
7. Geometric Figures Available	60
8. Smooth Line Generation	60
B. NPSDRAW LIMITATIONS	60
1. Font Representation	60
2. Scaling	61
3. Alignment	61
4. Text Placement	62
C. AREAS FOR FUTURE MODIFICATION	63
1. Scaling	63
2. Rotation	63
3. Fixed Point Drawing	64
4. Insertion of a Bitmap Image	64

5. Point Modification	64
D. CONCLUSION	65
LIST OF REFERENCES	66
APPENDIX A - THE QMS 1200 LASER PRINTER	67
APPENDIX B - FUNCTION AND FILE CROSS REFERENCE	109
1. FUNCTION TO FILE CROSS REFERENCE	110
2. FILE TO FUNCTION CROSS REFERENCE	118
3. FUNCTIONS : CALLED FROM & CALLS TO	123
APPENDIX C - PRODUCTS OF NPSDRAW	152
INITIAL DISTRIBUTION LIST	159

I. INTRODUCTION

A. A NEED FOR AUTOMATED FIGURE ILLUSTRATORS

In academic environments, graphical figures must be made available within a period of time equal to that of the time to complete the text. As word processors gain speed and can interact with assorted programs that check spelling, compute arithmetic values within text and even evaluate writing style, there is the need for a quick effective means of providing associated illustrations. The pressure upon the illustrator for completion of his product is increased each time new software and/or hardware provides faster output of a final text product. The only means by which the illustrator can keep pace with the word processor improvements is to utilize a system that is quick, easy to learn, powerful and uncomplicated to operate even for the most infrequent user.

In the Computer Science (CS) Department of the Naval Postgraduate School, the need for semi-automated or fully automated illustration systems has been served by a variety of systems such as the PIC Graphics Language, FIGURE Graphics Illustrator Program and most recently, OZDRAW, an interactive graphics figure illustrator. PIC is a procedural language, in which the user specifies the motions that one goes through drawing a figure. The language is the implementation of a "semi-automated" system, i.e. having no real-time computer

graphics representation during the drawing process. The user must maintain a mental or physical image of his proposed drawing, and utilize multiple drawing commands, variable names and values, positioning data and attributes to provide the input "coded text" before generation of his final product.

FIGURE is a departmentally produced semi-automated system that produces an illustration by using the users input of coded text similar to that of PIC. FIGURE requires only the primitive figure name and position to be drawn. Various line styles, line widths, fonts and fill patterns are simply identified as change is required. The text then is processed to generate the final product.

OZDRAW is the first fully automated system available for general usage within the CS department. The user has real-time feedback in the form of a drawing on a monitor of a graphics workstation. This system actually permits the user to do the actual drawing. The drawing is then converted to data acceptable by FIGURE. OZDRAW does not require the user to learn, nor even be concerned with the assignment of primitives, positioning data or attribute assignment format in the "coded text" file used by FIGURE. The user simply draws his illustration on the monitor, saves it to a file, and then prints out that illustration by invoking FIGURE.

B. SYSTEM DRAWBACKS

1. PIC Graphics Language

As with all semi-automated systems, the major deficiency of PIC is the lack of real-time feedback as the product is "laid out" or drawn before actual printing. This requires the user to have a detailed sketch or hand drawn illustration to work from and determine specific numeric values. The program then must be written to reflect the sketch and finally processed for printing. Though simple drawings are accomplished somewhat easily, modifications of any nature tend to increase the complexity of the code. Modification of figure size, attributes, positioning data, as well as, general picture size require a complete understanding of the system. PIC is powerful, but complex and sometimes difficult even for experienced users. Furthermore, PIC lacks any capability for fill patterns.

2. FIGURE Illustration Program

FIGURE experiences the same major deficiency of all semi-automated systems, i.e. the lack of real-time feedback. It was designed though for use by inexperienced and infrequent users. With each primitive figure's position specified in the text, simple modifications to positioning, attributes, etc. are accomplished through the simple editing of text. As an illustration increases in size and complexity, textual change becomes tedious and difficult. Modification to a single figure for repositioning, or attribute change can be accomplished with little difficulty, but movement of a block of figures requires extensive editing of the

text. As with all semi-automated systems, FIGURE and PIC require the user to be knowledgeable of the local text editor on the computer system being used. Additionally, the user needs acceptable typing skills to reduce the amount of compiler and system errors incurred during processing of coded text.

3. OZDRAW Interactive Graphics Figure Illustrator

OZDRAW is a real-time, interactive figure generation system. It has been designed primarily to run on the IRIS 2400 series of computer graphics workstations manufactured by Silicon Graphics, Inc of Mountain View, California. The output of the system is designed for any graphics capable laser printer. The primary laser printer for the CS department is a Quality Micro System (QMS) Lasergraphic 1200 .

The aim of OZDRAW is to provide a powerful, user-friendly figure generation system to enhance the technical figures required by the Computer Science Department at the Naval Postgraduate School [Ref. 1]. Since its production, OZDRAW has been utilized by a large number of thesis students and faculty. It has become the most popular figure illustration system in the department. With the system's popularity, it has become apparent that some of the initial design features are a hinderance to the experienced user. Various users have expressed a desire to see features expanded, as well as new capabilities added to the original version. The following are often cited areas of OZDRAW design deficiencies:

- o Inefficient attribute specification and display features.
- o Menu organization requires traversal through numerous levels to accomplish related system functions.
- o Limited font specifications / descriptions.
- o Lack of texture representation during real-time feedback of drawing.
- o Insufficient line width selections.
- o Tedious attribute change mechanics.
- o Unavailability of common geometric figures.
- o Inability to scale individual figures and blocks of figures.
- o Lack of figure alignment.
- o Foreign figure data utilization / conversion.

To date, there have been two implementations of OZDRAW put into daily use with the differences between the systems being only in the method of selecting menu options. This study revolves around the actual maintenance of OZDRAW originally written by Steve Firth in 1985/86 [Ref. 1] and examines the addition of new features to OZDRAW, retitled NPSDRAW. These changes are compatible with the OZDRAW and FIGURE file formats. Primary focus of attention is on the overall design and implementation of that new system.

II. NPSDRAW: SYSTEM OVERVIEW

A. SYSTEM CAPABILITIES

The program used as a basis for NPSDRAW was originally written by Steve Firth in a 1985 master's thesis titled "OZDRAW - An Interactive Graphic Figure Illustrator". Our goal was to provide an improved user interface for that system. Additionally, new features as well as system enhancements were included.

1. Primitives Supported by NPSDRAW

Most figures that are needed in technical figure illustration can be derived from the primitives supported by NPSDRAW. The available primitives are listed in Table 2.1.

Primitives	Variations
Polygons	Rectangles, Squares, Diamonds, Triangles and Arrowheads
User Formed Polygons	
Circles & Ellipses	
Arcs	
Straight Lines	Vertical, Horizontal, Single and Multiple Lines
Smooth Curvilinear Lines	
Seed Points	Used to provide a fill pattern (texture) for concave figures
Text	Single lines of Text

2. NPSDRAW Figure Attributes

Each figure generated with NPSDRAW is assigned specific attributes that define its appearance. The attributes are for line width, line style, texture (fill pattern) and font. Specific attribute values available are dependent on the current graphics printer installed. Table 2.2 shows the default settings and some of available settings for the QMS 1200 LaserGraphic Printer currently installed. A complete listing available attributes with examples can be found in Appendix A.

The IRIS workstations provide only one predefined font. All text is displayed using that one font. To show size differentials of the actual font as displayed on the laser printer, NPSDRAW generates a red rectangle around the text. This rectangle represents the actual printed size of the character string. Note: this rectangle can be either larger or smaller than the text characters

TABLE 2.2 FIGURE ATTRIBUTES FOR NPSDRAW		
Attribute	Default Value	Selections
Line Style	Solid	Solid, Large Dash, Medium Dash, Dotted
Line Width	1 pixel (screen), 0.01" (printed)	0.01" to 0.10" (printed)
Texture	Clear	25 possible including, Clear, Solid, Shaded, Horizontal, Vertical & Oblique Lines, and various patterns
Fonts	Roman 6-LPI, 10-CPI	Over 100 possible: Roman, San Serif, Typewriter, Slanted, Bold, Special Symbols, with sizes ranging from 6-10 to 8-15

displayed. Textures are accurately shown with each figure when drawn. A pallet of the available textures is on screen at all times.

3. The Screen Display

The primary screen layout of NPSDRAW is shown in Figure 2.1. NPSDRAW does not display the entire page but only a clipped version (8.5" x 8.5") of the page at any one time. The image on the screen is approximately 25% larger than the printed image. In horizontal page alignment, the entire height is seen, but not the full width. When in vertical alignment, the full width is seen but not the height. The user can move about the page when in editing or drawing functions via the Arrow (cursor) keys of the keyboard.

Initially, NPSDRAW provides a quarter-inch grid overlaying the drawing area. Discussion of this grid is covered in section B.1. Superimposed on the drawing area is a permanent red dashed rectangle. This rectangle represents the thesis margins used at the Naval Postgraduate School. There are no limits though on where the user can draw, even to the boundaries of the page.

The right side of the screen provides instructions for the currently selected functions. The current settings of the attributes are also displayed (Figure 2.2). Lastly a pallet with all possible textures (fill patterns) is maintained for user reference and selection of textures in changing attributes (Figure 2.3).

4. The Printed Page

NPSDRAW is designed to produce figures for a standard paper size (8.5" x 11"); the size cannot be changed. NPSDRAW does allow the picture to be

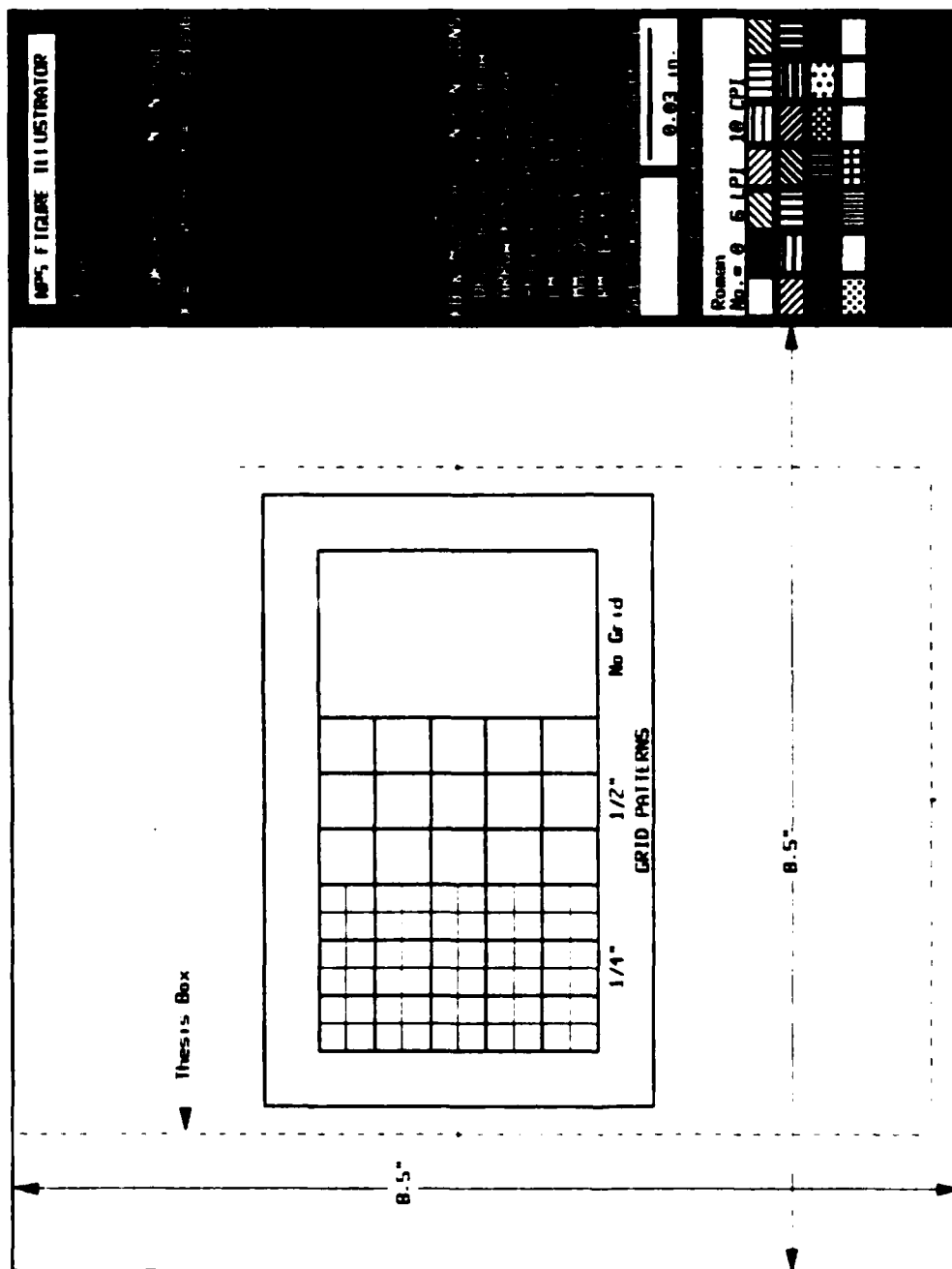


Figure 2.1 Terminal Display and Drawing Area.

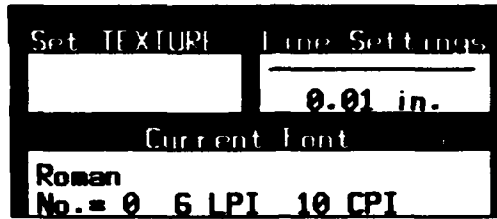


Figure 2.2 Current Attribute Settings Display

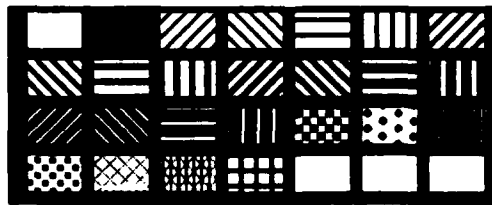


Figure 2.3 Texture (fill pattern) Palette

either vertically oriented (portrait) or horizontally oriented (landscape), i.e. either 8.5" x 11" or 11" x 8.5" respectively.

5. The IRIS Mouse and Cursor

NPSDRAW uses mainly the mouse, and to a lesser extent the keyboard, for user input. Menu choices and texture selection are done via the mouse. All text is entered via the keyboard. Full screen movement is accomplished via the cursor control keys (arrow keys).

The IRIS mouse has three buttons associated with it, located at the top end of the mouse. These buttons are referred in this manual as the Left (LM), Middle (MM) and Right (RM) mouse buttons. The locations are self-explanatory. Moving the mouse across an appropriate surface moves the position of the cursor on the workstation screen. The cursor can appear as a small red arrow (menu selections), red crosshairs (texture selection) or black pencil (drawing mode). Pressing a button will perform a unique operation for the current function*. The function of each button is explained on the screen at all times.

B. MENU ORGANIZATION

The program is menu driven with selections made through the use of the mouse. The options in a menu are highlighted through movement of the mouse up and down. When the appropriate selection is highlighted (arrow and background

* Sometimes when pressed, the mouse button gives a "double bounce", i.e. it appears to the system and its user that the key has been pressed more than once. This occurrence can be annoying as it produces results that are not expected. It is therefore recommended that the mouse buttons be pressed in a crisp, sharp manner to avoid this situation.

color change), it is selected by pressing the middle mouse button (MM). Throughout the program, the function of each mouse button is identified on the right side of the drawing area in an area titled *KB & MOUSE BUTTON FUNCTIONS*. The organization of the program is discussed in the following explanations of each available menu.

1. Main Menu

The first menu displayed after start up and selection of the screen orientation is the **MAIN MENU** (Figure 2.4). There are 16 active options in the menu. The menu should be thought of as being divided into four functional areas: **Figure Generation**, **Page Editing**, **File Storage & Recall** and **Attribute Selection**.

The first area of **Figure Generation** is comprised of the first two options. "Draw Figures" presents the user with the **FIGURE MENU** (Figure 2.5) for selection of one of the available primitives. Lines are reached through the second option "Draw Lines" which presents the **LINE MENU** (Figure 2.6). Each menu lists the various figures and line modes possible. The associated attributes of those figures can be changed from the same level.

The second functional area is comprised of options three thru eight of the main menu. These options concern editing the page and specific figures. **SINGLE EDIT** (Figure 2.7) and **BLOCK EDIT** (Figure 2.8) provide the user with means to arrange single or multiple figures on the page. "Figure Attributes" selects the **FIGURE ATTRIBUTES MENU** (Figure 2.9) display and allows

MAIN MENU
Draw Figures
Draw Lines
Write Text
Single Edit
Block Edit
Toggle Grid
Figure Attributes
Erase Page
View Page
Read File
Write File
Change LineStyle
Change LineWidth
Change Font
Change Texture
EXIT SYSTEM

Figure 2.4 MAIN MENU

FIGURE MENU	MAIN MENU
Draw Rectangle	Draw Figures
Draw Circle	Draw Lines
Draw Polygon	Write Text
Draw Ellipse	Single Edit
Draw Diamond	Block Edit
Draw Triangle	Toggle Grid
Draw Arrow	Figure Attributes
Drop a Seed	Erase Page
Change Texture	View Page
Change LineStyle	Read File
Change LineWidth	Write File
Figure Attributes	Change LineStyle
EXIT	Change LineWidth
	Change Font
	Change Texture
	EXIT SYSTEM

Figure 2.5 Figure Selection Menu

LINE MENU	MAIN MENU
Vertical Line	Draw Figures
Horizontal Line	Draw Lines
Single Line	Write Text
Connected Lines	Single Edit
Smooth Line	Block Edit
Arc	Toggle Grid
Change Line Style	Figure Attributes
Change Line Width	Erase Page
Exit	View Page
	Read File
	Write File
	Change LineStyle
	Change LineWidth
	Change Font
	Change Texture
	EXIT SYSTEM

Figure 2.6 Line Selection Menu

SINGLE EDIT	MAIN MENU
Move a Figure	Draw Figures
Remove a Figure	Draw Lines
Recover Lost Figure	Write Text
Reproduce an Object	Single Edit
Exit	Block Edit
	Toggle Grid
	Figure Attributes
	Erase Page
	View Page
	Read File
	Write File
	Change LineStyle
	Change LineWidth
	Change Font
	Change Texture
	EXIT SYSTEM

Figure 2.7 Single Edit Menu

BLOCK EDIT	MAIN MENU
Move a Block	Draw Figures
Remove a Block	Draw Lines
Move Entire Page	Write Text
Copy a Block	Single Edit
Block Append	Block Edit
Exit	Toggle Grid
	Figure Attributes
	Erase Page
	View Page
	Read File
	Write File
	Change LineStyle
	Change LineWidth
	Change Font
	Change Texture
	EXIT SYSTEM

Figure 2.8 Block Edit Menu

the changing of attributes of a selected figure. **TOGGLE GRID** varies the grid overlay between a 1/4", 1/2" and no overlay on the drawing area. This grid is not printed on the final product. The grid is displayed in green with the 1/2" divisions thicker than the 1/4" divisions. **ERASE PAGE** (Figure 2.10) clears the drawing area and **VIEW PAGE** permits the user to view his drawing in an unclipped format with reduced scale.

The **FILE RECALL MENU** (Figure 2.11) is accessed through the "Read File" option. The "Write File" option brings up the **FILE STORAGE MENU** (Figure 2.12).

The last area of the main menu includes the Attribute Change options. Each option prompts the user with the steps required to change the desired attribute setting. It is through these options that attributes are set for figures yet to be drawn. The previously discussed option of "Figure Attributes" allows changing attributes of figures already drawn. Finally the last option "Exit System", is the option used to terminate the program.

2. Figure & Line Menus

The Figure Menu is obtained through the selection of the "Draw Figures" option of the Main Menu. This menu permits the selection of the drawing routine of rectangles (squares), circles, user specified polygons, ellipses, diamonds, triangles, a fixed size arrowhead, and a seed for designating a texture for a concave area. The Line Menu provides the options of vertical, horizontal, single (any direction), and multiple attached straight lines. A smooth cursive

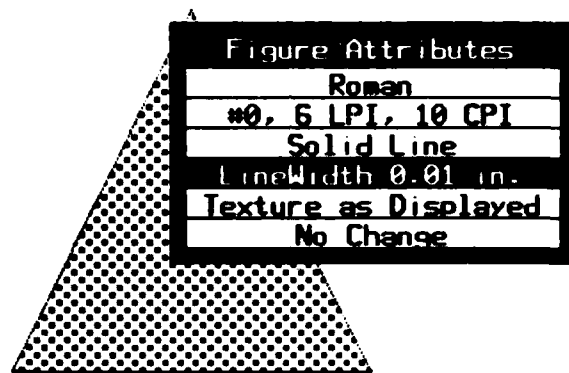


Figure 2.9 Figure Attribute Display

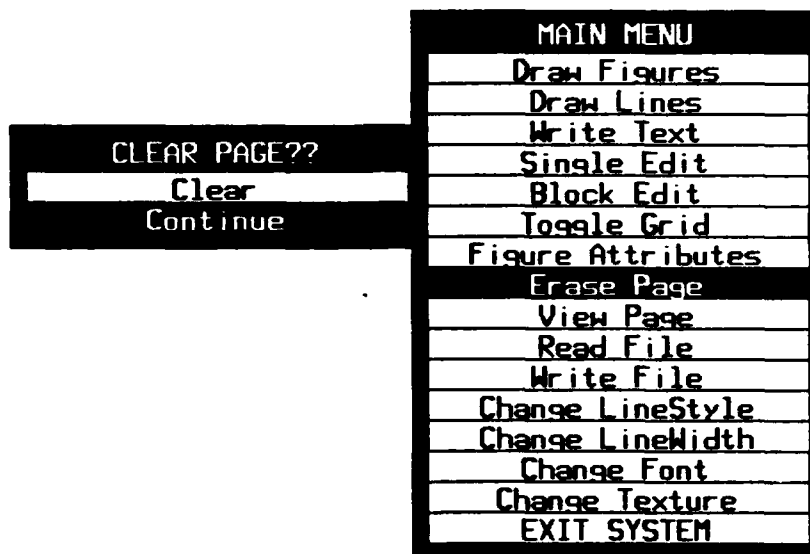


Figure 2.10 Screen Clear Menu

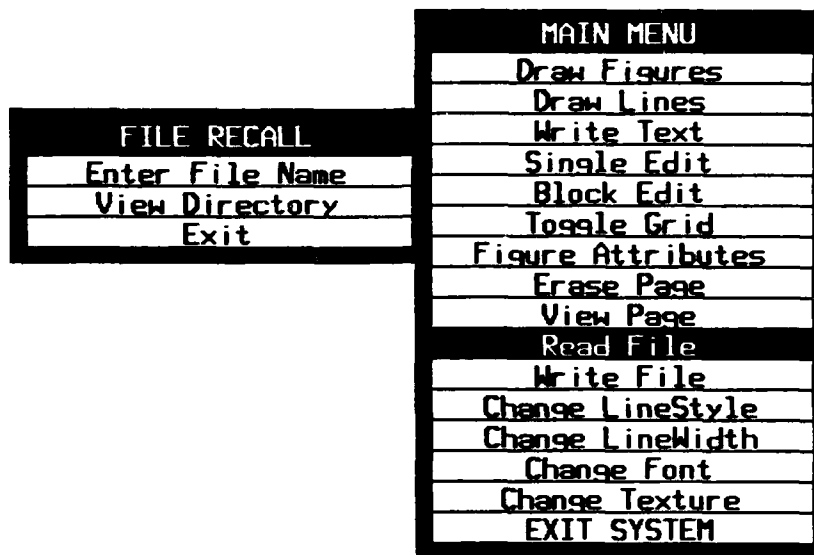


Figure 2.11 File Read Menu

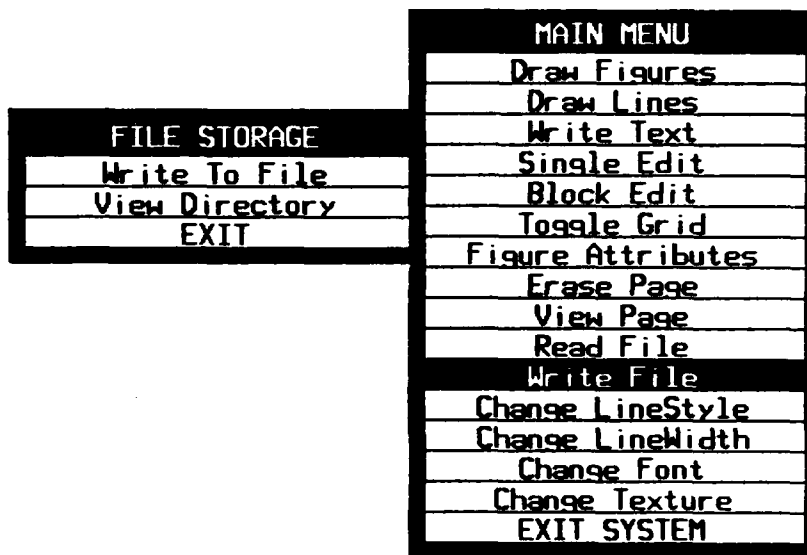


Figure 2.12 File Write Menu

style line and arcs are also available. The attributes associated with the figures can also be set from these menus. The "Exit" option returns the user to the Main Menu.

All drawing functions differ slightly in operation but are basically similar. To begin drawing, press the middle mouse button (MM). To finish drawing, press MM again. The functions of the mouse buttons are always displayed. All functions are easy to use and most users are comfortable with the drawing process after several minutes of experimentation. However, for completeness, a quick procedural sequence for each figure is given in Table 2.3.

3. Single & Block Edit Menus

The Single and Block Edit Menus are the two interactive editing facilities of NPSDRAW. Single Edit offers movement, deletion, reproduction, or recovery of a deleted figure. Selection of individual figures is accomplished by positioning the cursor over the control point of a figure and pressing the middle mouse button. Control points are corners of a polygon, centers of circles and ellipses, ends of lines and arcs, and the beginning of text strings. If an object is found, NPSDRAW will cause the figure to blink. The user confirms the blinking figure to be correct by pressing the middle mouse button, or else rejects it by pressing either outside mouse button. After a figure is rejected, another figure may blink if in the bounds of the system's pick mechanism. Once a figure is confirmed, the desired editing function can be performed. If no figure is found, than an error message is displayed.

TABLE 2.3 BRIEF PROCEDURAL DRAWING SEQUENCES	
Rectangles	Press the middle mouse button to drop the first corner of the rectangle, move the cursor to draw the rectangle and press the middle mouse button when finished.
Circles	Press the middle mouse button to choose the center of the circle, move the cursor until the circle is drawn, pressing the middle mouse button to stop drawing.
Polygon	Press the middle mouse button to drop the first point of the polygon, move the cursor and press the middle mouse button again to drop subsequent points. When the figure is complete press the left mouse button.
Ellipses, Diamonds, Triangles	Press the middle mouse button to drop the first corner of a rectangle, and move the cursor to draw the rectangle that will circumscribe the figure. Press the middle button when finished and the figure will be drawn.
Arrow heads	Designate the position of the arrowhead tip by pressing the middle mouse button and select the direction by positioning the cursor on the arrowhead axis behind the tip. A dotted line will appear indicating the axis. When in position, press the middle mouse button and the arrow head will be drawn.
Drop a seed	Select the position of the seed point by pressing the middle mouse button. The area that encloses the seed point will have the texture currently set. The seed point will not be printed.
Single Lines	Select the first point using the middle mouse button. Move the cursor to complete the line and press the middle mouse button to stop. The vertical or horizontal line option restricts line drawing to the appropriate direction.
Connected Lines	Select the first point using the middle mouse button. Move the cursor to subsequent points and drop by pressing the middle mouse button. When complete, press the left mouse button.
Smooth Line	Press the middle mouse button to set the beginning point. Move the cursor as required to complete the line. When complete, press the middle mouse button.
Arc	Press the middle mouse button to choose the center of the arc. Move the cursor to draw the circle, pressing the middle mouse button when the correct radius is selected. Next press the middle mouse button to select the start of the arc. Move the cursor counter-clockwise and press the middle mouse button to select the angle of the arc.

Block Edit offers movement, deletion, reproduction of two or more figures (blocks), movement of the complete picture about the page, and a block append mode. Selection of blocks is carried out by the user drawing a special rectangle around the desired figures. If any figure has a control point within the rectangle, it is included in the selected edit function.

Movement, deletion and reproduction is similar to the single edit function. Moving the entire page is straightforward. The page is reduced into an unclipped image so that the user can see the page as a whole. Use of the arrow keys moves the picture about the page. This option is useful when a picture is complete and requires centering on the page. The Block Append mode permits reading in a file but rather than immediately fixing the image to the drawing area, it is attached to the cursor for positioning and placement. This facility is useful for special figures not in the primitive set.

4. Figure Attributes

The Figure Attributes menu is acquired by selection of the "Figure Attributes" option presented in either the Main or Figure Menus. The user is prompted to select a single figure via control points. Once a figure is accepted, the attributes menu appears. Selection of any attribute from the menu activates the sequence to change it. Only a single attribute can be changed on a selected figure at a time. If additional attributes need to be changed, the figure must be reselected. The presentation of this menu is the method for reviewing the attributes of a figure.

5. Clear Screen Menu

There are two methods for clearing the drawing area of all images. First, the user can remove the figures through use of the single or block edit remove functions. Individual figures can be recovered at a later time in the same drawing session. If complete clearing of the screen is necessary, then selection of the "Erase Page" option of the Main Menu displays the Clear Screen Menu. The choices on that menu are to "Clear the Screen" or "Continue" drawing. If the picture is not stored before erasure, it is lost forever. If the decision to not clear is made, the user needs only to select "Continue".

6. Read & Write File Menus

To print a file or save a drawing, the figures need be stored in a file. Naturally, there is a facility to retrieve this drawing from a file. A directory listing is available, listing only those files contained in the user's current directory. The Read and Write operations are selected from the Main Menu and are similar in format. The user provides a file name entered from the keyboard.

If the user wishes to read a file, NPSDRAW checks two conditions. First it checks if the file exists. If the file does not exist, the user is informed. Second, the system checks if any figures are displayed on the screen. If there are, NPSDRAW asks the user if he wishes to "Discard" the displayed figures, or "Merge" the two files. When writing to a file, NPSDRAW checks if the file exists. If it does not, it creates the file and store the figures in it. However, if the file does exist, the user is asked if he wishes to "Append" the figures to the end of the file,

"Overwrite" the contents or "Quit". Overwriting the contents of a file destroys its contents.

7. Line Style Menu

The Line Style Menu (Figure 2.13) is presented through selection of the "Change LineStyle" option from a menu or via selection of changing the line style from the Figure Attributes Menu. The menu lists four line styles. Styles currently available are solid (continuous) line, large dashed, medium dashed and dotted lines. Upon selection of a style, the user is returned to the controlling menu. Whenever the user enters the Line Style Menu, he can exit without making any changes.

8. Line Width Menu

The Line Width Menu (Figure 2.14) is presented through selection of the "Change LineWidth" option from a menu or via selection of changing the line width from the Figure Attributes Menu. The menu lists options of line widths from 0.01" to 0.10", variable selection and no change. Variable selection provides line widths from 0.01 to 1.00" in increments of 0.01". Upon selection of a width, the user is returned to the controlling menu from which the call was made. The variable option operates and provide line widths up to 1.00" for NPSDRAW. Screen representation and storage in memory reflect the selected size, but when printed with the QMS 1200 Printer, the line width will not exceed 0.10".

LINE STYLE	MAIN MENU
Solid Line	Draw Figures
Large Dashed Line	Draw Lines
Med Dashed Line	Write Text
Dotted Line	Single Edit
No Change	Block Edit
	Toggle Grid
	Figure Attributes
	Erase Page
	View Page
	Read File
	Write File
	Change LineStyle
	Change LineWidth
	Change Font
	Change Texture
	EXIT SYSTEM

Figure 2.13 Line Style Menu

LINE WIDTH	MAIN MENU
0.01 in	Draw Figures
0.02 in	Draw Lines
0.03 in	Write Text
0.04 in	Single Edit
0.05 in	Block Edit
0.06 in	Toggle Grid
0.07 in	Figure Attributes
0.08 in	Erase Page
0.09 in	View Page
0.10 in	Read File
Variable	Write File
No Change	Change LineStyle
	Change LineWidth
	Change Font
	Change Texture
	EXIT SYSTEM

Figure 2.14 Line Width Menu Selections

9. Font Source, Font Style and Font Sizes Menus

The Font Source Menu (Figure 2.15) is presented through selection of the "Change Font" option from a menu or via selection of changing the font from the Figure Attributes Menu. This menu permits selection of the font by number, or style and size. The user can choose the number entry method of font style as a result of his experience with OZDRAW / NPSDRAW, or he can select the font by style method and make selection by the style and size that suits his needs (Figure 2.16)*.

* Not all font sizes and styles are available. See Appendix A for font availability.

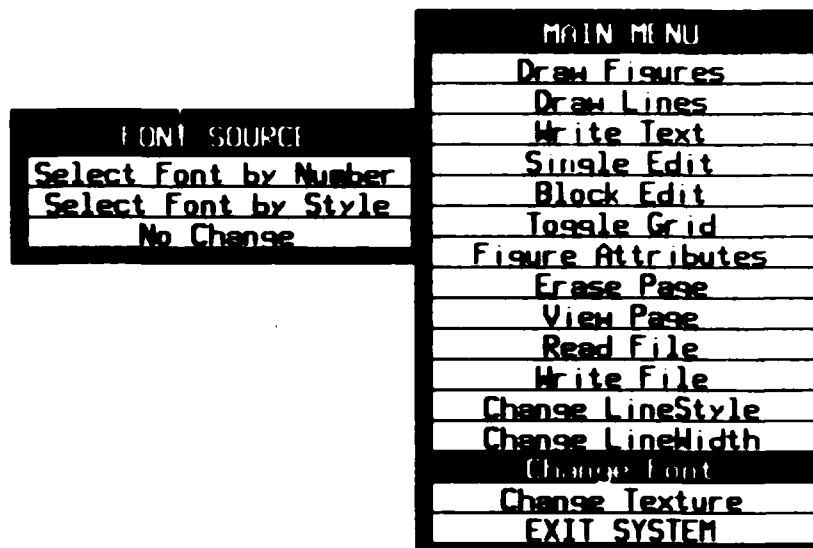


Figure 2.15 Font Selection Method

FONT SIZES	
3 LPI	5 CPI
4 LPI	7 CPI
6 LPI	10 CPI
6 LPI	12 CPI
8 LPI	10 CPI
8 LPI	12 CPI
8 LPI	14 CPI
8 LPI	15 CPI
8 LPI	25 CPI
12 LPI	25 CPI
No Change	

FONT STYLES
Roman
Roman Bold
Roman Slanted
Roman Underlined
Roman Title
Roman Bold Title
IBM Roman
Supp Roman
Supp Roman Bold
Supp Roman Slanted
San Serif
San Serif Bold
San Serif Italic
San Serif Slanted
San Serif Underlined
San Serif Title
San Serif Bold Title
IBM San Serif
Supp San Serif
Supp San Serif Bold
Supp San Serif Slant
Typewriter
Typewriter Bold
Typewriter Slanted
Typewriter Underlined
Typewriter Title
Orator
Scientific
Scientific Title
DEC Technical
DEC Technical Title
DEC Symbol
Math Symbol(mo)
Math Symbol(mt)
APL
No Change

Figure 2.16 Font Selection by Style & Size Menus

III. NPSDRAW - SOFTWARE IMPLEMENTATION DETAILS

The origin of NPSDRAW is as part of an M.S. thesis by Steve Firth [Ref. 1]. Modification of that system began with improvements to the user interface. OZDRAW is a medium size program, 10 - 20 K line of source code [Ref 2]. The difficulty with the system lies in the fact that the number of functions and their inter-relationship was not well documented. To discuss the implementation and areas of modification, one must first be aware of the basic graphics technique of double buffering and be knowledgeable of the data structures used in data manipulation. Discussion of the more than 160 functions is then clearer.

A. BASIC GRAPHICS OPERATING SYSTEM

Before any discussion on the system design as a whole can be accomplished, one must have a basic understanding of the method by which the drawn figures are presented to the user. The smooth "animated" appearance of the drawing area is accomplished through the use of a display technique called **Double Buffering**. This technique refers to the feature of dividing all available bitplanes into two groups, where only one set is viewed at a time. Each group is called a buffer with the visible buffer referred to as the **Front Buffer** and the non-visible as the **Back Buffer**. With this organization all writing and drawing is done to the back buffer, thus alleviating any apparent drawing on the front buffer. All drawings

appear "instantaneously" to the user's eye. unless the operating system as a whole is slowed down by either too large a drawing or other background processes. The back buffer is continuously drawn to and immediately "swapped" with the front buffer so that all screen activity viewed by the user is seen as "smooth & fluid".

B. DATA STRUCTURES

Within NPSDRAW, there are two primary data structures. The first, **object**, is a computer graphics system structure to store the graphics primitives that generate a figure. The second is a linked list of structures, named **drawing_struct**, that contains the object, various parameters for drawing the object. numeric information and pointers to adjacent members of the linked list.

1. Object

The **object** is the graphics system's method of storing a particular figure for later recall without the re-execution of the drawing routines. The drawing primitives are stored within the object. With a system call using the object's name, the figure definitions are retrieved and written to the back buffer. In our system, the object does not contain data such as line width, line style, texture or font selection. Each object can be redefined to reflect figure modifications. The use of objects permits easy copying for insertion into other structures.

2. Drawing Struct

The variable **drawing_struct** is a structure in the C language analogous to the record construct in PASCAL. This is the element of the linked list which

contains all data unique to each figure. As seen in Table 3.1, **drawing struct** contains the figure definition (object), the figure attributes, numeric information and additional pointers to adjacent elements of the linked list. All this data is necessary to permit the easy manipulation of single or group figures, storage in files and subsequent printing of full page pictures. It is the sequential reading of these structures in the linked list that permit the continuous drawing of the figures to the back buffer. The elements of the linked list can be "turned off" from view as required to represent removal, or can be designated a figure on which a particular operation is performed.

C. SOFTWARE IMPLEMENTATION

C is the language of implementation for NPSDRAW. All segments of the program are referred to as functions. With the large number of functions in NPSDRAW, the best approach to a description of each function is follow through the program and discuss the areas using the organization of the Main Menu as a guide.

1. Initialization and General Functions

Most functions called in the "main" function deal with the initialization of NPSDRAW for the current drawing session. The call of *main_menu()* provides the user access to drawing routines, file storage and recall, and attribute change.

TABLE 3.1 COMPONENTS OF STRUCTURE		
"DRAWING_STRUCT"		
Component	Type	Explanation
type	short	Records the TYPE of figure, i.e. CIRCLE, RECTANGLE, TEXTS etc.
exdsts	short	Is the figure in existence?, i.e. visible on the screen or deleted but recoverable.
this_font	short	Font with which this figure was drawn. Used only with TEXTS but recorded for all figures.
this_texture	short	Texture with which this figure was drawn. Recorded for all figures though all do not use it.
this_linestyle	short	Line Style with which the figure was drawn. Recorded for all figures though all do not use it.
this_linewidth	short	Line Width with which the figure was drawn. Recorded for all figures though all do not use it.
real_linewidth	float	Line Width with which the figure was drawn. Recorded for all figures though all do not use it. Line width read in can be of any value. NPSDRAW maps the value read in to one of its own for presentation, but stores the correct value back out to the file.
num_info	short	Records the (a) number of characters pointed to by the textptr for text TEXTS figures,(b) value of an attribute for attribute figure and (c) number of values pointed to by a coord ptr (e.g. DIAMOND has 4 pts, thus 8 values {x&y}).
figure	object	The object that contains the figure associated with the structure.
textbox	object	An object containing a rectangle that represents the area occupied by the TEXTS figure without text. Used in page display.
textptr	char	A pointer to a text string for a TEXTS figure, if present.
coords	float	A pointer to coordinates of a figure, if any.
fwd, bwd	ptrs	Pointers to adjacent members of the linked list.

The last function cleans the graphics workstation for a graceful exit. The functions of *main ()* are as follows:

initialise () - undertakes all required initializations including all texture definitions, line style definitions, devices for queue usage and initial settings of attributes. Texture definitions and attribute settings are accomplished via calls to *init_textures()* and *init_attributes()* respectively. The actual means of setting the attributes is discussed in section 5.

readfonts() - reads the current font table into memory. The reading of the table occurs each time the system is activated, ensuring that the most recent font definitions are available to the user for the current graphics printer. As the table is read, the font name, number, and size characteristics are placed in arrays for future reference.

initpopup() - sets NPSDRAW to accept the definitions of the various menus throughout the program.

opening_display() - welcomes the user to NPSDRAW with the logo of the Graphics and Video Laboratory.

init_guide() - the system inserts the grids and thesis box into the linked list with the appropriate alignment acquired from the user through a call of *get_user_alignment()*.

There are some general functions that get called repeatedly throughout the NPSDRAW system's operation. They are important throughout the entire program. They are:

display_position() - display the current position of the cursor.

instructions() - display up to six lines of instructions on the right hand side of the screen.

redraw_figures() - redraw the current linked list to one or both buffers

set_orthoview() - set the correct orthoview with respect to the selected page alignment.

update_bounds() - update the position of all figures, grids and the thesis box as a result of input from the arrow keys on the keyboard. Permits movement of the clipped image of the page for work on areas not visible.

set_screen() - update the right hand side of the terminal display. Each attribute setting is updated at this time.

newcursor() - select a new definition of the cursor.

get_block() - draw a block around a specified area on the drawing screen. This block is used to include figures for manipulation or represent the area in which a particular figure is to be drawn.

full_memory() - checks for available memory before reading in any figures from memory. Used in both "read file" calls.

Though not a function, most menus have an "Exit" option. This is used to exit the menu when work is complete.

2. Figure Generation

The Main Menu can be viewed as having four functional areas of activity. The first area is that of Figure Generation. That area includes two menu options that call the functions *fig_menu()* and *line_menu()*. Selection of an option from one of the menus calls either *draw_a(type figure)*, *change(an attribute)*, or *edit_obj(CHANGE_ATTRIBUTES)*. Each call to *draw_a(type)* calls the function(s) necessary for that particular figure to be drawn as found in Table 3.2. Each drawing command contains a call to *insert()* to store the figure in the linked list. If a figure has just been drawn, calls of *reproduce_last()* will copy the last drawn figure and permit placement anywhere on the screen. If the last figure drawn is not desired, then it is removed with *remove_last()*. Though the function

edit_obj(CHANGE ATTRIBUTES) is accessible from *fig_menu()* it is discussed later in section 3(d) as an editing function.

3. Editing Functions

The second functional area of the Main Menu is that of editing the screen image. This includes moving, copying, or deleting one or more figures at a time, grid changing, clearing the page or viewing the page as an unclipped image.

a. Single Edit

Single Edit functions edit only one figure at a time. A figure is selected from those visible on the drawing area by *find_figure()* and returns a pointer to the structure containing that figure. A call on *find_figure()* results in a

TABLE 3.2 FUNCTION CALLS "DRAW_A(TYPE)"	
Type Figure	Function Call(s)
RECTANGLE	draw_rectangle()
SEED	insert_seed()
CIRCLE	draw_circle()
ARROW	draw_arrow()
POLYGON	draw_polygon()
DIAMOND	get_block() insert_diamond()
TRIANGLE	get_block() insert_triangle()
HLINE, VLINE, LINE	draw_line()
TEXTS	draw_text()
MULTI_LINE	draw_multi_line()
SMOOTHLINE	draw_smooth_line()
ARC	get_circle() draw_arc()
ELLIPSE	get_block insert_ellipse

call to a *check(type)* for types of arc, circle, line or text. The pointer then is used as input to the subsequent calls that perform the actual editing. Function calls of *edit_obj(MOVE)* permit the repositioning of any object. Once the updated position is determined through movement of the object on the screen, the function *modify_object()* is called. That function calls a modify function unique for each type of figure. An edit mode of copying is accomplished through *edit_obj(COPY)* and performs similarly to MOVE but makes a duplicate of the figure for placement on the screen with *duplicate()*. Deletion and recovery of deleted objects during a drawing session is performed by *edit_obj(REMOVE)* and *edit_obj(RECOVER)*. These simply change the "exists" field of the structure **drawing_struct** as required by the function. Recovery is then performed by drawing all figures present in the list with an exists field of NO. Selection of these figures is performed similarly to selection on the current visible screen. Upon confirmation of a figure, the exists field is changed back to YES. All edit function calls immediately *set_written()* to FALSE indicating the current illustration is not saved. This flag is only set to TRUE when the illustration is saved, i.e. written to a file.

b. Block Edit

The Block Edit option selected from the Main Menu performs operations similar to those of Single Edit but on two or more figures as determined by the rectangle presented with a call to *get_block()*. Once the block is placed by the user, *find_objects()* traverses the linked list with a set of

coordinates and determines what if any figures exists within the block. Existence within the block is determined with calls to *gt1pt_inside()*, *1pt_inside ()* or *arc_inside()*, as appropriate for the specific figure type. Once the figure is determined to be within a block, a pointer to it is placed in a temporary array for further action.

Movement of a block in *blockedit()* takes the working array of structure pointers and permits movement over the drawing area. Once the block of figures are in the updated position, each object undergoes *modify_object()* for its specific type. During movement, routine *draw_page()* is called to show the changing positions of the figures. When *blockedit(COPY)* is invoked, all actions are similar to those of MOVE but additionally calls to *drawcopy()* and *duplicate()* are made.

REMOVAL in *blockedit()* simply utilizes the working array of figures found in the block and changes their "exists" field to NO. There is no recovery of figures in the *blockedit()* calls but each figure removed in this mode can be recovered through a single edit function as discussed earlier.

The Block Append option is similar to the Block Copy option discussed previously, but copies a block from a file rather than the screen. (The method of file identification is discussed later in section 3(e).) All actions and calls parallel the COPY sequence. The functions used are *bblockedit()* to coordinate the block append of figures, *bfind_objects()* creates the working array of pointers to the figures of a temporary linked list, *bmove file()* moves the new list of figures

over the drawing area as well as lets the arrow keys move the page. Additionally, it calls *bdraw_page()* which generates the image of the new figures for movement about the drawing area. Once the position of the new figures is set, *bmove_page()* links up the two separate linked lists into a single entity.

The last option to discuss from the Block Edit Menu is that of page movement. The unclipped version of the page is presented through a change in the viewport setting according to the initial page alignment. Once this image is present, the figures themselves are moved about the screen via changes in the paramters to the function *set_orthview* via inputs from the arrow keys. Once the figures are placed in the desired position, all figures in the linked list are taken through *modify_objects()* for update of their positional data.

c. Grid Change

A grid overlay that provides a guide for figure placement and drawing can be varied with spacing of 1/4", 1/2" or deselected altogether. The method of changing the grid is performed through a call to *toggle()*. When NPSDRAW is activated, the grids 1/4" and 1/2 " and the thesis box are inserted into the working linked list. As the user toggles the grid, the "exists" field of these figures are changed to represent the desired overlay. The thesis box is always visible, i.e. its structure "exists" field is always YES. The change in the grid is made through the sequential changing of the "exists" fields of the 1/2" and 1/4" figures. Though the grid in the 1/4" version appears as one, it is actually made up of two figures from the linked list, the 1/4" and 1/2" grid figures.

d. Figure Attribute Modification

The last function that deals with the actual editing of figures and structures, is the function *edit_obj(CHANGE ATTRIBUTES)*. This function is available through both the Main and Figure Menus. The function call takes a selected object from *find_figure()* and reads from **drawing_struct** the attributes associated with it via *alter_attributes()*. A menu is displayed containing those attributes. The contents of the menu items is determined by reading the value associated with an attribute for that figure and displaying it on the menu as text. The font text representation is determined through use of the font number which is used to determine an array index. This index then accesses the arrays that contain the font name, number, and size data. The line width value is acquired directly from the structure while the line style value is converted to text through a case statement. Texture is not represented by text since it is visible on the screen. All data then is placed in the menu as options for change. Selection of any menu option activates the attribute select function associated with the chosen attribute. The actual attribute selection functions are discussed in section 3(f).

Once the value for the changed attribute is selected, *alter_attributes()* selects the appropriate function of *changefont()*, *changels()*, *changelw()* and *changetexta()*. The operation of each change function is similar so a general description of the procedure follows. The call to the change functions passes a pointer to the figure and the new attribute value. The figure is first evaluated if the change is appropriate. For example, changing texture of a line

does not make sense so no change occurs and the function is terminated. Two new figures are placed in the linked list, one in front of the selected figure for the new version and an attribute figure placed behind the selected figure to reset that attribute so following figures are not effected. Upon completion of these operations, the original figure is deleted from the linked list.

4. File Manipulation

a. Read in a File

Selection of the "Read File" option calls the function *read_menu()* which presents the user with a menu having options of "Read File" or "View Directory". A choice to read a file activates *get_from_file()* which prompts the user for a file name. While the user inputs the name (visible on the screen as typed), NPSDRAW waits for the carriage return character as a signal to evaluate the given name. If the filename is valid one of two actions can occur. If no picture is presently on the screen (flag function *drawing_exists*), the file is read into the linked list via *file2list()*. This function calls individual functions for each type of figure for actual insertion into the linked list. Once the file is completely read in, *get_from_file()* is terminated. If an image does exist on the screen, the user has three options, to quit altogether, i.e. to not read the file, to merge the screen image with the file picture, or to discard the previous drawing and keep only the file picture. The user is provided a message as to the status of the file name provided, i.e. "File Read Correctly", or "File does not Exist" followed by "returning to Menu". The *get_from_file()* function also returns a value to the

calling function that indicates the status of the original read file call. It is these returned values that determines the next function call by *read_menu*. If the returned value indicated that the file existed, and was read into the linked list. *set_orthoview()*, *redraw_figures()* and *set_screen()* are called to finish the process.

Selection of the menu option "View Directory" or incorrect input of a file name calls a set of directory functions. To exit this function, the user needs only to press the middle mouse button to get back to the "Read Menu.

The method described above for reading in a file is the basis for the previously described function *block_append()*. The associated function is that of *bget_from_file()*.

b. Write File

The process of writing a file to disk is similar in format to reading a file. Upon selection of the "Write File" option from the Main Menu, the user is presented a menu with selections of "Write File" and "View Directory". The selection of "View Directory" selects the directory functions as described above under Read File.

The selection of "Write File" activates the function *go_to_file()* which then prompts the user for a file name. The file name is checked against present files in the current directory. If the name has not been used, the new file is created and the data written. If the file is found in the directory, the user is asked for one of three actions - to "O"verwrite, "A"ppend, or "Q"uit. Each action is self explanatory. The user is provided with messages from the system regarding the

status of the file such as, "unable to open", "Writing to file", "Finished Writing" and "Returning to menu". Upon completion of a file write or the selection of "quit" from the menu level, the user is returned to the Main Menu.

5. Setting Attribute Values

The setting of the various attributes is controlled through numerous short functions that utilize graphics system commands to set the screen representation of the attributes. The setting of each attribute is done through the assignment of the attribute value to a variable. This variable is read each time a new figure is drawn to the screen and inserted into the linked list. The functions called are of the form *set_current_ (attribute)* and *get_current(attribute)*.

The general operation of each attribute change involves two steps: (1) get the new attribute value and (2) assign that value to the appropriate variable. All values are provided by the user through (a) selection of the attribute description from a menu or (b) selection of a texture from a pallet. The values returned from the attribute select functions to the calling routine are actually character strings. These character strings are later converted to numeric values and used to update a current attribute setting or are used in the modification of a figure's attributes.

a. Font Selection

When Font change is selected from a menu, the first function activated is *fontsource()* which returns a value to signify the method of font selection. The options for font selection are via numeric value or style & sizing. If

the user chooses the numeric option, he need only enter the desired font number at the system prompt. NPSDRAW checks the value against range limits (0 to MAXFONT) and then against the "Findex" array generated during reading of the font table at system set up. If the numeric value is contained in this array, it is valid and a character string representation of the value is returned to the caller.

The second option of selection is more complex but results also in the return of a simple font number value. The user first sees a menu of all available font styles (names) . These names are not dynamically presented, i.e. they are hard coded in the menu presentation, though easily modified. Upon selection of the desired style or "No Change" (to save previous setting) the user sees a second menu of size combinations from which the user makes a selection. NPSDRAW uses both values and cross checks them with the "Fname" array for name availability and a matching size in the CPI & LPI arrays. If the user selected combination is present, the index value from the arrays is used to determine the actual font number. This number is then returned to the caller as a character string.

b. Texture Selection

Values for Texture Selection are acquired through the *sel_text()* call. The user sees a "Cross Hair Symbol" for a cursor near the lower right of the screen for movement over the various textures available on the pallet. Once the cursor is placed over the desired texture, selection is made by pressing the Middle

Mouse Button. If the mouse is within an area that defines a texture, a character string value is returned to the caller.

c. Line Style Selection

A call for a Line Style change is achieved through *sel_ls()* utilizing a menu selection of the four available line styles. Each line style is described textually for selection. The selection is returned to the caller for processing as needed.

d. Line Width Selection

Line Width Selection (*sel_lw()*) is similar to Line Style selection, but there are ten predefined line widths available for selection through a menu. There are additional choices of "variable" or "No Change". A predefined selection returns a character string value to the caller. The option of variable line width is also available. *var_lw()* gives a means for selection of line widths in 0.01" increments. The appropriate value is then returned to the caller.

6. Exiting The System

If the last option of the Main Menu is selected, the decision to exit or remain active is made as a result of the boolean response to the function *check_exit()*. This function checks if a drawing exists with *drawing_exists()* or if in fact the current drawing was written to a file by checking a static variable "drawing_written_to_file". If no drawing exists or it has been saved, then a TRUE is returned to the main menu. If the drawing was not saved, then the user

is prompted with a menu for selection of options. Upon selection of exit, the system is in *main()* and final exiting of the system occurs after a call to *cleanup()*.

IV. NPSDRAW : SOFTWARE MAINTENANCE

Maintenance is the work required to be done on a software system after it is placed in operational use. This work includes understanding the existing system, documenting the existing system, enhancing the capabilities of the system, answering questions of users, training users, rewriting and restructuring the software and adapting the system to a new environment. All maintenance work efforts can be categorized into three sub-maintenance areas: (1) CORRECTIVE MAINTENANCE - maintenance performed to identify and correct software errors and performance deficiencies, (2) ADAPTIVE MAINTENANCE - maintenance performed to adapt the system to changes in the environment and (3) PERFECTIVE MAINTENANCE - maintenance performed to enhance the capabilities and performance of the system. Regarding OZDRAW, maintenance efforts were of a perfective nature. Though corrective maintenance was performed, it is not discussed here.

No matter how well a software system meets the goals for its design and specification, it will have to undergo some change periodically during its lifetime to remain responsive to user needs. In a study of various large software systems, M. M. Lehman discussed various laws of Program Evolution that appeared to recur through the life cycles of programs [Ref. 3]. The First Law of Program

Evolution is the *Law Of Continuing Change* "which expresses a universally observed fact that large programs are never completed. They just evolve". With continuing observations and research the the First Law is now stated as:

A program that is used and that as an implementation of its specification reflects some other reality, undergoes continual change or becomes progressively less useful. [Ref 3.]

Though OZDRAW is not a large system, it still must evolve to respond to the user's needs through maintenance. The final product of this maintenance effort is NPSDRAW.

A. MAINTENANCE FACTORS

Every software product has characteristics that either decrease or increase the maintenance effort. A partial grouping of these is contain in Table 4.1. Those characteristics pertinent to the OZDRAW system are program complexity, number of user reports, poor documentation, good initial design and lack of experienced maintenance individuals.

TABLE 4.1 SYSTEM CHARACTERISTICS THAT INFLUENCE THE MAINTENANCE EFFORT	
Increase Maintenance Effort System Age System Size Program Complexity Number of User Reports Application Type Poor Documentation	Decrease Maintenance Effort Good Initial Design Modern Software Practices Automated Tools Data Base Management Techniques Good Data Administration Good Documentation Experienced Maintenance Personnel

To improve the maintenance effort of NPSDRAW in the future, there are only two characteristics that can be dealt with at the present: documentation and a log of user reports. There can be no change to program complexity or improvement on the initial design, since these factors are set with the initial version of the system. The experience of the personnel who will perform maintenance in the future is variable and most likely to be influenced by their expertise in the C programming language.

B. IMPROVEMENT OF DOCUMENTATION

Maintenance depends on information about the system, from design through actual coding. Information must be recorded that provides reasons and justifications for each design decision and modification. Information must not be lost during the life of a system. Information can only be maintained through quality documentation.

Many situations exist where the quality of documentation can be improved as maintenance is performed. Documentation can be developed and maintained with simple prologues at the beginning of a function. Data would consist of author's name, date, input, output, side effect, and exceptions if any.

The quality and careful design of code is also a part of a well documented system. A standard and straightforward coding style should be presented to maintenance personnel and enforced by management. In the academic environment, students generally code in a style that is similar to their peers, but

minor variations of style can have a disrupting and possibly confusing effect on subsequent personnel. The maintenance personnel must have documentation that can be easily read and understood sufficiently in order to maintain the system. As documentation improves, the program will be easier to maintain in the future. An additional argument for good quality documentation is that maintenance personnel prefer using embedded, "on line" documentation. As such, if the "off line" documentation is weak in any area, he will tend to disregard all further documentation. Simple and useful forms of documentation "off line" are plain language explanations of functions and call hierarchy diagrams. Cross reference listings of files and functions are also useful when tracing a program and understanding its call hierarchy. A table listing indicating the callers and the functions called by a function can be found in Appendix B.

C. PHYSICAL DIFFERENCES NPSDRAW / OZDRAW

The process of improving the user interface of the interactive graphics illustrator has brought about numerous differences between NPSDRAW and its predecessor OZDRAW. These changes are discussed in Chapter V. Some basic comparisons may be of interest at this time.

OZDRAW was a system of 139 files (165 functions). NPSDRAW has 133 files and (189 functions). The difference in file count is the result of condensing functions into common files. Though there are a number of new functions, the actual code organization has been changed slightly. The actual line count has

decreased almost 3%. This is not significant though and attributed to difference in coding style primarily for ease of changing repetitious parameters. Actual increase in compiled code was a little over 9%, which is not significant for the improved interface features.

Overall, the difference between OZDRAW and NPSDRAW source codes is primarily minor organizational changes and additional functions. Future maintenance will not be hindered by an increase in the system complexity over the original version.

V. CONCLUSIONS

A. IMPROVED USER INTERFACE

NPSDRAW follows the format of its predecessor OZDRAW. If the user is experienced with OZDRAW, he will notice the close similarity with NPSDRAW, and will notice numerous improvements to the user interface.

1. Menu Presentation & Composition

The menu system now is faster to traverse and perform related functions. Generally the user is in the second level of menus, whereas OZDRAW required the user to move between the third and first level for most related functions. The user can now access some functions from multiple locations within the menu structure rather than at a single location.

2. Attribute Display Area

The current settings for the attributes line style, line width, font and texture are visible to the user at all times. Attributes are no longer identified by an integer, but are represented numerically (line width, font), graphically (line width, line style & texture) and textually (font).

3. Font Selection

There are more than 100 different characters, and special symbol font definitions available to the user. These are selectable via their numeric designation

or through menu selection of style and size specifications. The selected font is displayed by name, number and size (CPI and LPI) in the Attribute Display Area and for the individual figure attributes when required. As new font definitions become available, simple modification to a font table provides immediate update to the system as the font table is read for each activation of NPSDRAW.

4. Texture Representation & Selection

NPSDRAW provides the user with a representation of the selected fill pattern (texture) as the individual figures are drawn (where appropriate). Additionally, the 25 different patterns are displayed in a pallet format for easy selection. The user no longer needs to remember a unique numeric value for any pattern. Once a pattern is selected, it is displayed as the current setting in the Attribute Display Area. Screen representations are unlimited, selections are limited by the printer utilized in drawing the final illustration.

5. Line Style and Line Width Representation & Selection

The current line style and line width selections are displayed attribute settings in the Attribute Display Area. The line styles remain the same (limited by the printer) but line widths are selectable from 0.01 to 0.1 in. Line width on the screen is proportional to the grid dimensions while the numeric value shown is the size of the printed line. The current line style is displayed within the Attribute Display Area. The line widths are selected from a menu containing preset values and as the capability of the printer changes, a variable scale

selection is provided for selection of widths 0.10 to 1.00 inches. Line styles are textually listed for selection.

6. Figure Attribute Change

A figure's attributes can be changed from the Main or Figure Menu. The selected figure's attributes are presented in a menu for viewing or selection of change. A particular attribute change can be selected in the same manner as current settings are selected.

7. Geometric Figures Available

Though all geometric figures were available through various drawing primitives in OZDRAW, two of the more common figures, the DIAMOND and TRIANGLE, were added to its predefined figures. Previously, the polygon function or single line functions were used to construct these. They are now available as primitives.

8. Smooth Line Generation

Within the Line Menu, a facility for drawing a free form line has been added. This facility provides the capability for enhancing drawings with smooth curved lines or cursive writing formats.

B. NPSDRAW LIMITATIONS

1. Font Representation

The IRIS has the limitation of only a single font displayable on the terminal monitor. This font has the style of San Serif and approximate sizing of 6

LPI and 10 CPI. As the user selects different styles and sizes, the characters on the screen remain the same. Even with the proportioned red box surrounding the text, this becomes annoying and somewhat messy as the complexity and amount of text in the picture increases.

A related font problem is the representation on the screen of the special symbols available with numerous font selections. The special symbols must be displayed on the picture as standard keyboard characters and symbols. Utilizing the special symbols requires the user to have a "translation table" at hand when entering text onto the picture. At present, the final picture can only be "proofed" through review of a printed image.

2. Scaling

As with any drawing process, it is easier to draw a figure in a larger format where slight errors of alignment are harder to notice. Special figures may be needed in various sizes for placement on different pictures. A facility to permit the scaling of individual or groups of figures (blocks) would provide a greater flexibility in drawing a high quality picture. Details in a large figure need to be maintained as the figure is reduced. A single drawing of a special figure could then be kept in the user's library.

3. Alignment

Consistency of figure size between similar figures is easily maintained by duplication of the first, either through the copy utility immediately after being drawn or through the utilities in single or block edit modes. The problem that

arises is the positioning of these figures with reference to a particular plane or point on the page. Each figure can be positioned if the user moves the mouse slowly. Proximity of the figure with others determines the visible accuracy of the alignment process.

A related problem is the placement of line end points and arrowhead positioning at a specified boundary. Dependent on the line widths and styles of the figures and lines, precise setting of the end points of a line, or the position of an arrowhead becomes difficult.

4. Text Placement

As the user types in a text string, it is placed on the picture in a left to right fashion. For any picture, the text can only be placed horizontally, depending on the selection of landscape or portrait page orientation. The user therefore cannot draw a picture in the landscape mode and have the page number placed in a portrait orientation.

Multiple lines of text cannot be generated automatically with the line feed and carriage return actions. Each line must be placed by the user, with the exactness of placement left up to his experience. The provision of the ability to automatically place text would provide a level of quality for more textual oriented figure displays.

C. AREAS FOR FUTURE MODIFICATION

As with all systems, there can always be improvements or enhancements to maintain their usefulness. For local usage by students, faculty and staff, NPSDRAW is more than sufficient at this time. Demand will require that it be modified further to maintain competence with commercially available systems. Below are listed various areas for future research and modification.

1. Scaling

When a user designs a custom figure for a project, he undoubtedly desires to use that figure at a later date. Just the fact that it is a special figure, means that it has more than a single primitive component in it. This gives rise to the need for a "Block Scaling" option in the system. With this feature available, the user could draw an original figure, and refine each component as the overall picture takes shape. This would make the system perform as a drawing "in pencil" available for simple modifications rather than "in ink" where each figure would need erasure before change in size.

2. Rotation

Along with scaling features, a feature to permit rotation of figures would be a nice addition. Presently, using primitives built into the system, the user can draw most figures required with little difficulty. But positioning of those figures requires a sketch of the picture beforehand to derive the general positions on the drawing page. Furthermore, the user draws two-dimensional images.

Though simple three-dimensional drawing is not altogether difficult, NPSDRAW would be enhanced and simplified even more so with a rotation capability.

3. Fixed Point Drawing

"Fixed Point Drawing" is used here to describe drawing of figures that are centered on or have a predefined common point. Additionally it may include restrictions on the drawing area defined by the user during drawing of sequential figures. It would be a multi-function facility. Functions available could be (1) the user defines a point that is common to selected figures, such as a set of concentric circles. (2) A limit could be set dynamically on the cursor preventing movement outside a specified area, implemented in similar manner as the vertical and horizontal line drawing restriction.

4. Insertion of a Bitmap Image

A bitmap image insertion capability is a necessary addition. There are images that one cannot make on NPSDRAW, i.e. figures, that can be generated and copied from a graphics terminal via a bitmap dump. The ability to include such a picture on the drawing area where the user can annotate specific points would provide a high quality product that currently can only be acquired through a strictly manual figure illustration system.

5. Point Modification

As the user develops his skill with NPSDRAW, he begins to make more complicated figures to meet his needs. In the course of drawing, some figures require the modification of a specific point of a figure, i.e., the corner of a polygon

or the end point of a line segment. The current method is to erase the current figure and redraw it as necessary. If the user could have the ability to move a single point of a figure, the modifications could be done in a relatively short time as compared to redrawing the complete figure.

D. CONCLUSION

NPSDRAW was designed to be a system accessible by unskilled users who needed a quick, easy method for generating illustrations. The system is intuitive in its operation. It still produces a high quality product that can be easily modified and reproduced. The NPSDRAW system is considered a medium sized program (at less than 15000 lines) and thus easily accepts system modification and performance enhancement. The limitations and future modifications discussed above are by no means conclusive. As NPSDRAW's users community grows, so do we expect demands for the program's expansion and modification.

LIST OF REFERENCES

1. Firth, Steven J., *OZDRAW: A Real-Time Interactive Figure Generation System*, M.S. Thesis, Naval Postgraduate School, Monterey, California, December 1985.
2. Fairley, R.E., *Software Engineering Concepts*, McGraw Hill, 1985.
3. Lehman, Meir M., *Programs, Life Cycles, and Laws of Software Evolution*, Proceedings of the IEEE, v. 68, no. 9, pp. 199-215, September 1980.

APPENDIX A

NPSDRAW WITH THE QMS 1200 LASER GRAPHIC PRINTER

This supplement to the NPSDRAW users manual is provided to illustrate the various attributes available with the QMS Laser Printer, and to identify system peculiarities.

Attributes

NPSDRAW uses the following attributes available on the QMS laser printer:

- (1) Four linestyles,
- (2) Ten linewidths,
- (3) 25 textures (fill patterns) and
- (4) 111 fonts.

An example of each attribute is shown later in this supplement. When using various combinations of attributes and primitives, unexpected results appear on the drawing area and the printed page. When drawing circles or ellipses with other than a solid line, the image will not appear on the screen until the radius and minor axis exceed 1.25" for large dashed lines, 1.0" for medium dashed lines and 0.5" for the dotted lines. If a primitive is drawn under such conditions, only through actual printing of the product can it be seen. Though the image is present on the screen, the large linewidths imposed on the broken line will render a printed image with a "scalloped" edge. Examples of this aberration are shown with the various linewidths and styles in this appendix.

OZPRINT

The utility for printing the NPSDRAW figure files is OZPRINT. This print program is a driver for the FIGURE illustration system. OZPRINT parses the parameters and spawn the background process to print the picture.

The method to print a file produced by NPSDRAW is to issue the following command:

```
ozprint -alignment filename [-alignment filename]
```

where alignment is 'h' for horizontal (landscape) and 'v' is for vertical (portrait). More than one file can be printed in this manner, as long as the correct number of alignment values are placed in the appropriate positions.

A source of error is calling OZPRINT twice, or more, in rapid succession. This error is caused by FIGURE using a scratch file to store intermediate data. With more than one process executing at a time, collisions over this scratch file may occur. To avoid this, wait between calls to print OZDRAW figures. This potential trouble area can be avoided with a compound call to the OZPRINT utility.

Calling OZPRINT with the incorrect page alignment for the current file, can produce unexpected results, especially if the picture extends beyond the page boundaries in a particular orientation. Page orientation is not encoded in the file.

The Seed Point

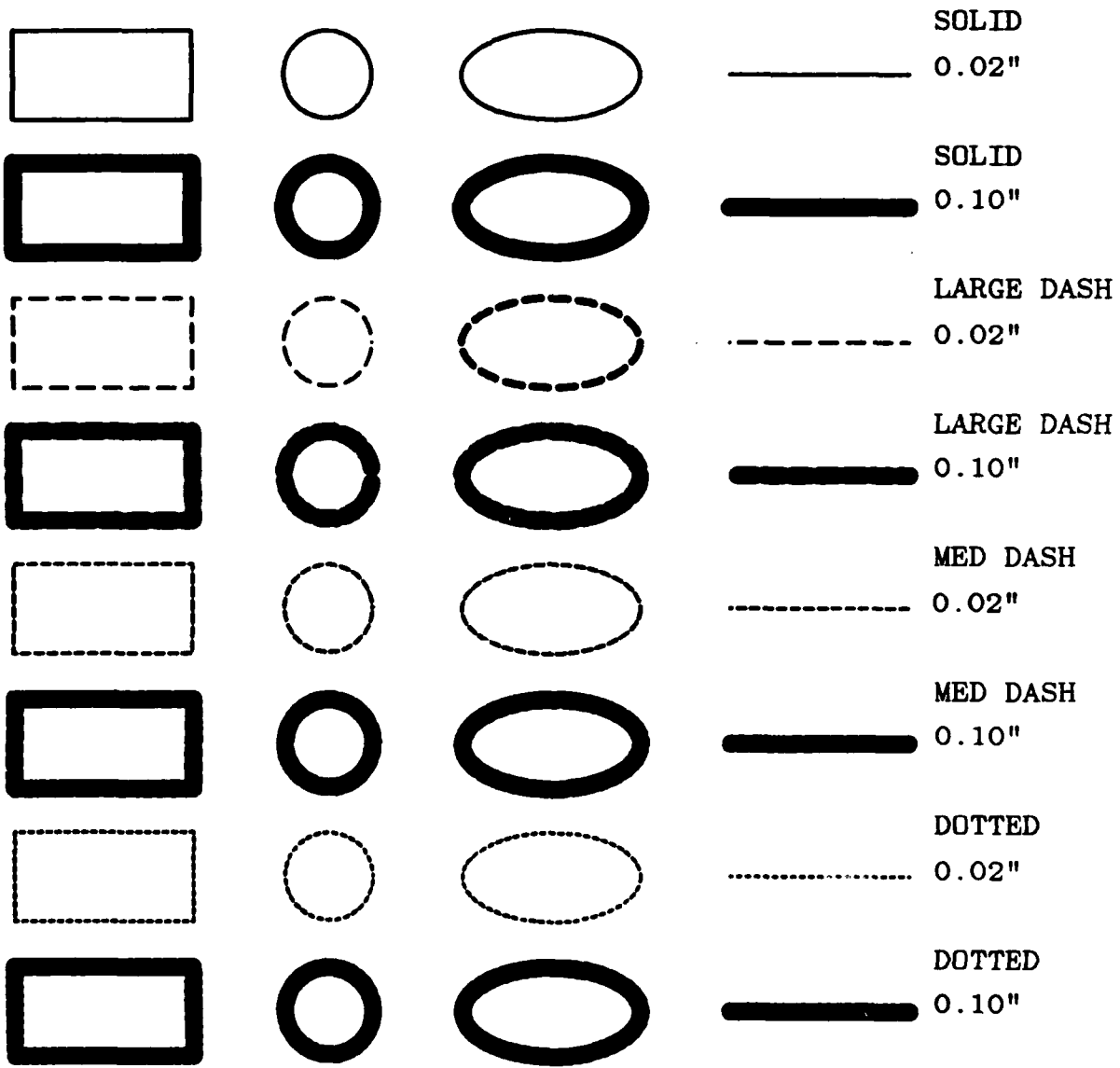
When a seed point is dropped, it will fill the enclosed area with the currently set texture. If the area is not enclosed, it will fill to the page boundary. The seed point will not be printed.

Drawing a Filled Figure

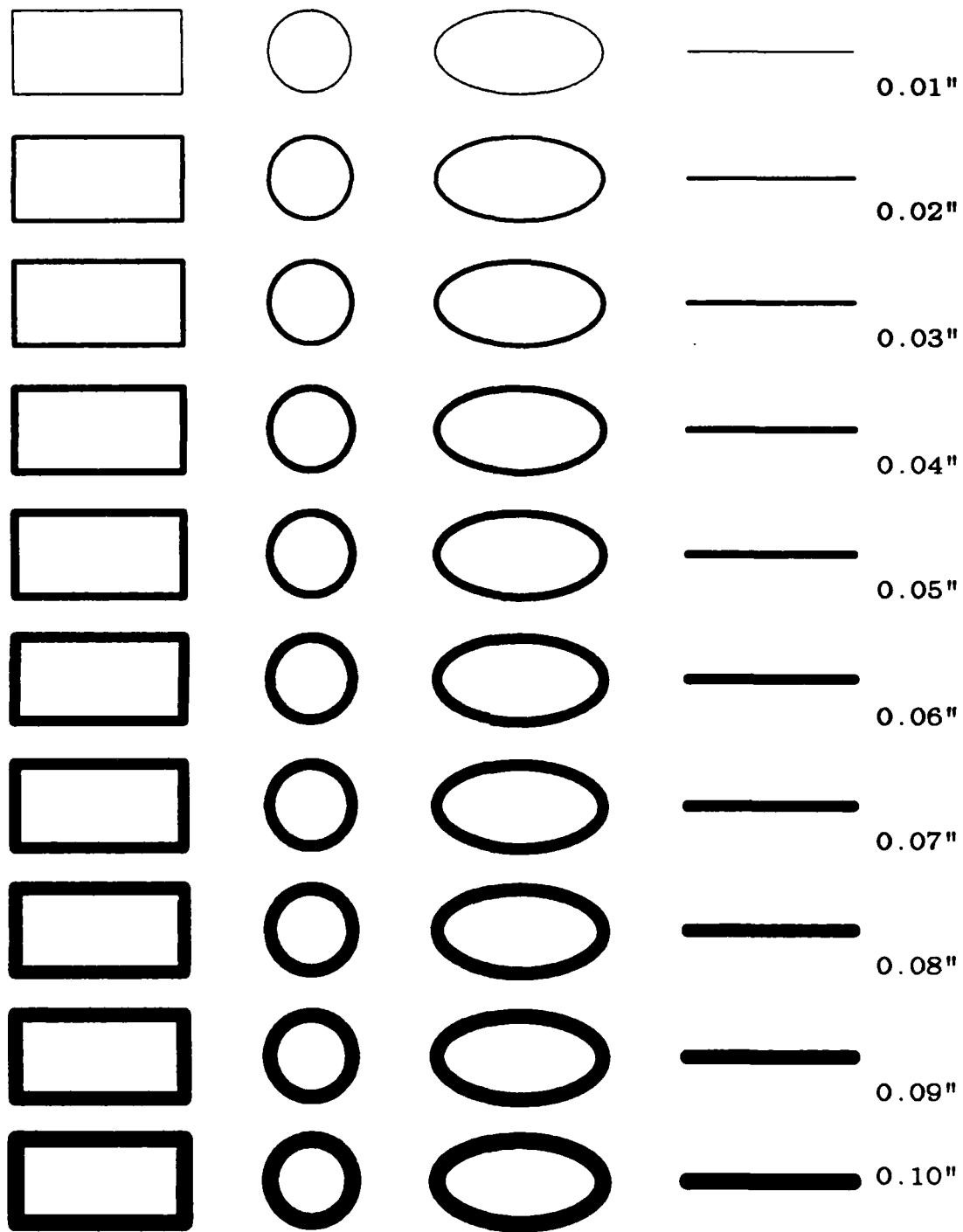
When assigning a texture to a figure there are considerations to be made: (1) Due to the polygon fill algorithm used in OZPRINT, only a concave figure can be filled. The fill pattern is unpredictable when there are overlapping figures with different textures and neither being clear. Attempts to fill a polygon with a segment of concave border can also induce undesirable results. (2) Filling a figure with other than a solid boundary, will extend the texture beyond the boundary to the next solid boundary or edge of the page. (3) Overlapping figures with other than identical textures, will only be filled with a single texture.

Picture Overrun

OZDRAW does not perform any clipping of figures that extend beyond the page boundary. This is important to consider since figures can be inadvertently moved off the page and stored in a file. Printing such images produces undesirable results. The printing system will accept the files with errors. The QMS will handle overrun to the right and top of the page, but cannot control the left or bottom margins.



NPSDRAW Available Line Styles



NPSDRAW Available Line Widths

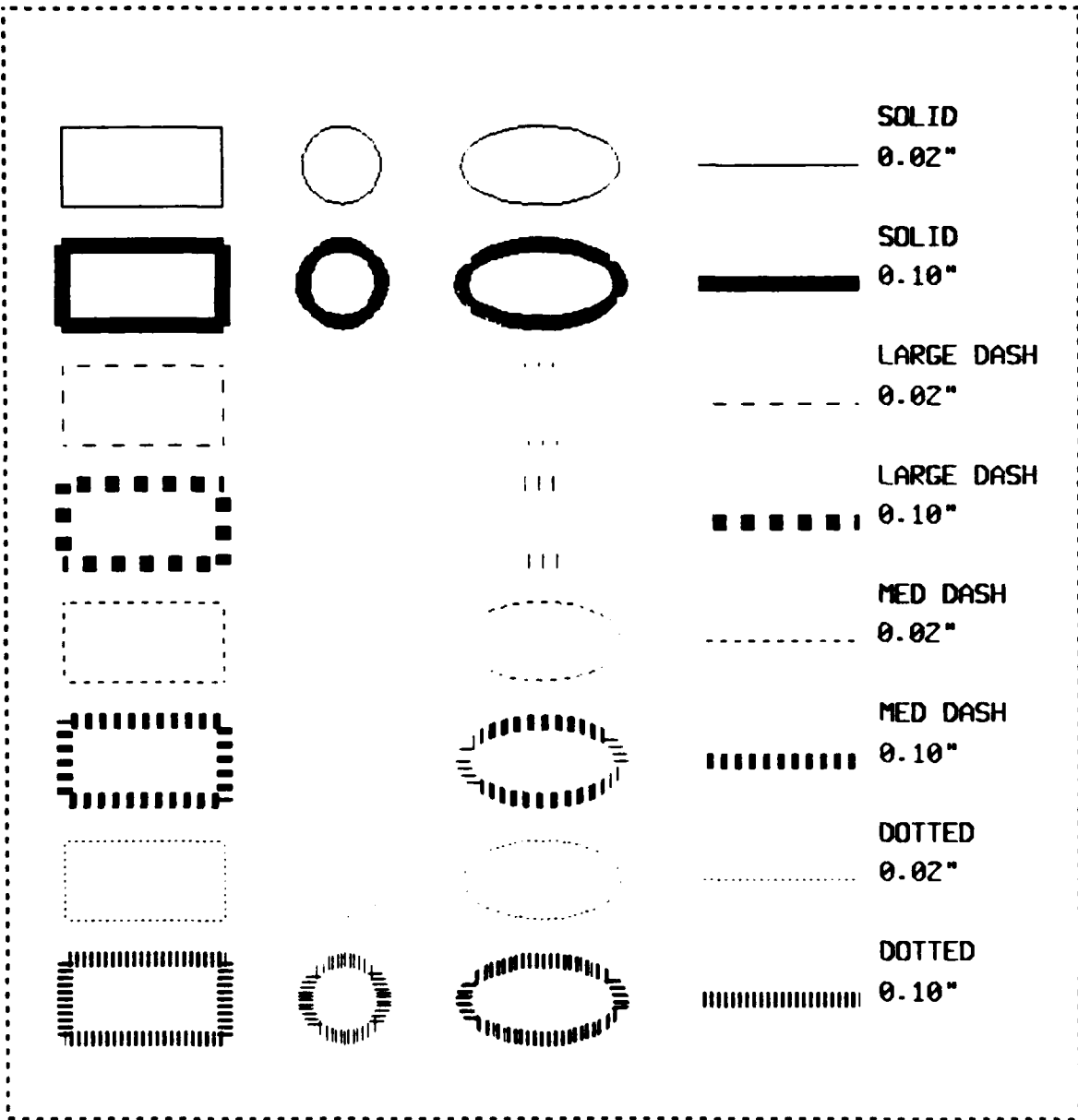
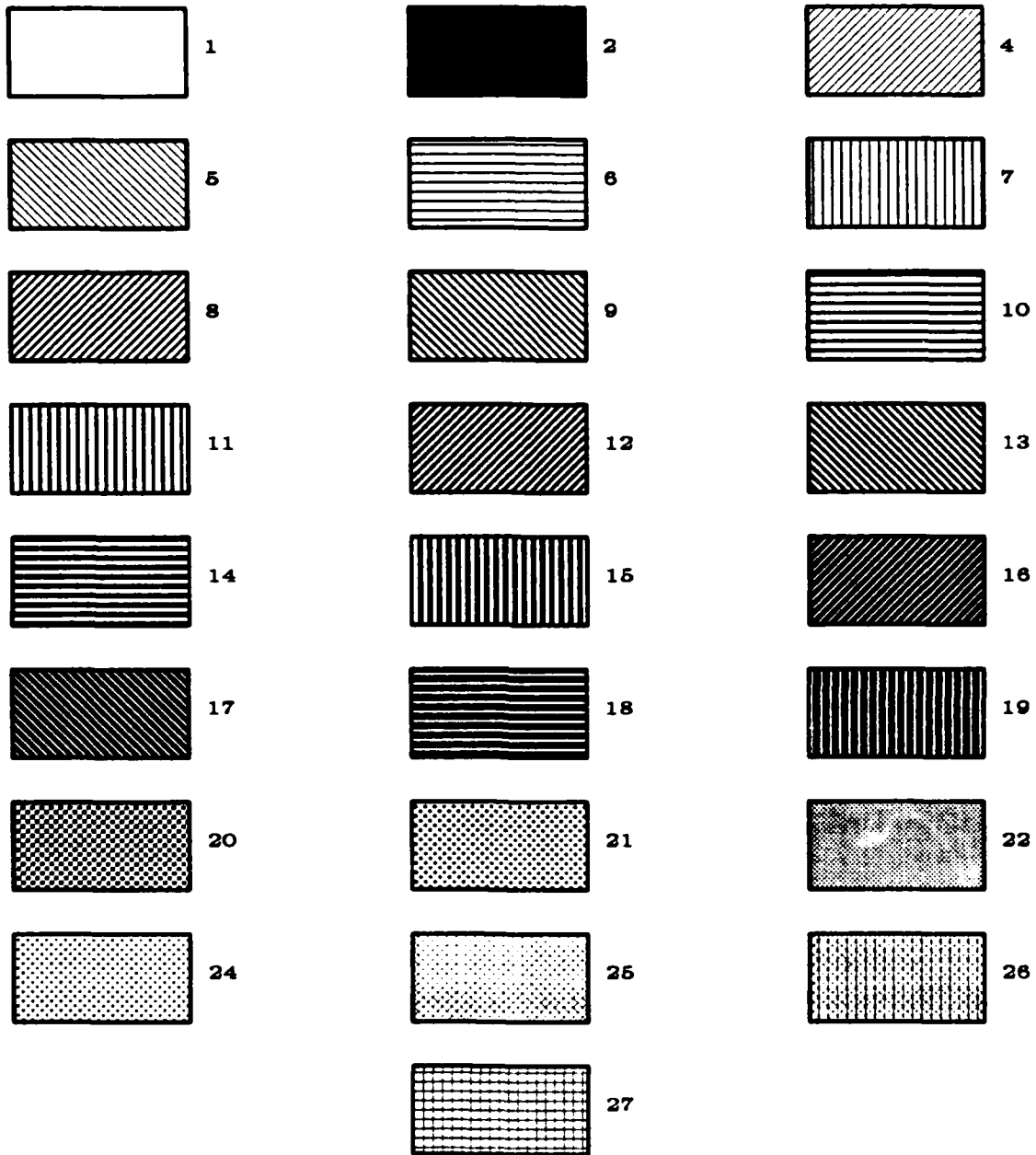


Image variations from a selection
of Line Styles & Widths



NPSDRAW Available Textures (Fill Patterns)

FONTS AVAILABLE

Arabic / Keyboard Characters

Style LPI/CPI	3/5	4/7	6/10	6/12	8/10	8/12	8/15
Roman			x	x	x	x	x
Roman Bold			x	x	x	x	
Roman Slanted			x	x	x	x	
Roman Underlined			x	x	x	x	
Roman Title	x	x					
Roman Bold Title	x	x					
San Serif*			x	x	x	x	x
San Serif Bold			x	x	x	x	x
San Serif Italic			x	x	x	x	x
San Serif Slanted			x	x	x	x	x
San Serif Underlined			x	x	x	x	x
San Serif Title	x	x					
San Serif Bold Title	x	x					
Typewriter			x	x	x	x	
Typewriter Bold			x	x	x	x	
Typewriter Slanted			x	x	x	x	
Typewriter Underlined			x	x	x	x	
Typewriter Title	x	x					
Orator			x				

* Additional San Serif of 8/14 , 8/25, 12/25.

Special Purpose Characters / Letters

IBM Roman			x	x		x	
Supplemental Roman			x	x		x	
Supplemental Roman Bold			x	x		x	
Supplemental Roman Slanted			x	x		x	
IBM San Serif			x	x		x	
Supplemental San Serif			x	x		x	
Supplemental San Serif Bold			x	x		x	
Supplemental San Serif Slanted		x	x		x		
Scientific			x				
Scientific Title	x	x					
DEC Technical Font			x	x		x	
DEC Technical Title Font	x	x					
DEC Symbol			x	x			
Math Symbol (mo)			x	x			
Math Symbol (mt)			x	x			
APL			x	x		x	

Font Styles and Sizes

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/14	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/15	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/14	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/15	a b c d e f g h i j k l m n o p q r s t u v w x y z

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/14	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/15	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 0
6/12	, . / ; ' [] < > ? : " { } 1
8/10	, . / ; ' [] < > ? : " { } 2
8/12	, . / ; ' [] < > ? : " { } 3
8/14	, . / ; ' [] < > ? : " { } 107
8/15	, . / ; ' [] < > ? : " { } 108

Fonts : ROMAN

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 4
6/12	, . / ; ' [] < > ? : " { } 5
8/10	, . / ; ' [] < > ? : " { } 6
8/12	, . / ; ' [] < > ? : " { } 7

Fonts : ROMAN BOLD

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 8
6/12	, . / ; ' [] < > ? : " { } 9
8/10	, . / ; ' [] < > ? : " { } 10
8/12	, . / ; ' [] < > ? : " { } 11

Fonts : ROMAN SLANTED

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
<u>6/10</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
<u>6/12</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
<u>8/10</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
<u>8/12</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
<u>6/10</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
<u>6/12</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
<u>8/10</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
<u>8/12</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
<u>6/10</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
<u>6/12</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
<u>8/10</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
<u>8/12</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>

REF	, . / ; ' [] < > ? : " { } Font Num
<u>6/10</u>	<u>, . / ; ' [] < > ? : " { } 12</u>
<u>6/12</u>	<u>, . / ; ' [] < > ? : " { } 13</u>
<u>8/10</u>	<u>, . / ; ' [] < > ? : " { } 14</u>
<u>8/12</u>	<u>, . / ; ' [] < > ? : " { } 15</u>

Fonts : ROMAN UNDERLINED

REF A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
3/5 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

4/7 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF a b c d e f g h i j k l m n o p q r s t u v w x y z
3/5 a b c d e f g h i j k l m n o p q r s t u v w x y z

4/7 a b c d e f g h i j k l m n o p q r s t u v w x y z

REF 1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
3/5 1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

4/7 1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

REF , . / ; ' [] < > ? : " { } Font Num

3/5 , . / ; ' [] < > ? : " { } 16

4/7 , . / ; ' [] < > ? : " { } 102

Fonts : ROMAN TITLE

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
3/5	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
4/7	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
3/5	a b c d e f g h i j k l m n o p q r s t u v w x y z
4/7	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
3/5	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
4/7	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
3/5	, . / ; ' [] < > ? : " { } 109
4/7	, . / ; ' [] < > ? : " { } 110

Fonts : ROMAN BOLD TITLE

REF	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
6/10	⊥	⊥	⊥	-	+	†	‡	§	¶	⊥	⊥	⊥	⊥	-	†	‡	§	¶	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥
6/12	⊥	⊥	⊥	-	+	†	‡	§	¶	⊥	⊥	⊥	⊥	-	†	‡	§	¶	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥
8/12	⊥	⊥	⊥	-	+	†	‡	§	¶	⊥	⊥	⊥	⊥	-	†	‡	§	¶	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥

REF	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
6/10	β	γ	π	Σ	σ	μ	τ	ϕ	θ	Ω	δ	∞	∅	∈	∩	≡	±	≥	≤	()	÷	≈	°	●	●	
6/12	β	γ	π	Σ	σ	μ	τ	ϕ	θ	Ω	δ	∞	∅	∈	∩	≡	±	≥	≤	()	÷	≈	°	●	●	
8/12	β	γ	π	Σ	σ	μ	τ	ϕ	θ	Ω	δ	∞	∅	∈	∩	≡	±	≥	≤	()	÷	≈	°	●	●	

REF	1	2	3	4	5	6	7	8	9	0	-	=	'	!	①	#	\$	%	&	*	()	_	+	~	
6/10	ü	ü		†	‡	§	¶	⊥	⊥	⊥	⊥	⊥	α	í	ℓ	ú	ñ	Ñ	a	~	¿	~	■	½	■
6/12	ü	ü		†	‡	§	¶	⊥	⊥	⊥	⊥	⊥	α	í	ℓ	ú	ñ	Ñ	a	~	¿	~	■	½	■
8/12	ü	ü		†	‡	§	¶	⊥	⊥	⊥	⊥	⊥	α	í	ℓ	ú	ñ	Ñ	a	~	¿	~	■	½	■

REF	,	.	/	;	'	[]	<	>	?	:	"	{	}	Font Num
6/10	‡	<	>	¶	∅	■	■	¶	¶	¶		ó	√	²	380
6/12	‡	<	>	¶	∅	■	■	¶	¶	¶		ó	√	²	381
8/12	‡	<	>	¶	∅	■	■	¶	¶	¶		ó	√	²	382

Fonts : IBM ROMAN

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	Â Æ Ç É Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë ì í ñ
6/12	Â Æ Ç É Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë ì í ñ
8/12	Â Æ Ç É Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë ì í ñ
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	“ < § © £ x i \ / L r T t + ’ ” > ¶ © € ÷ ¿ - / \
6/12	“ < § © £ x i \ / L r T t + ’ ” > ¶ © € ÷ ¿ - / \
8/12	“ < § © £ x i \ / L r T t + ’ ” > ¶ © € ÷ ¿ - / \
REF	1 2 3 4 5 6 7 8 9 0 - = ‘ ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ À ¾ ¼ ⅜ ¼ (+ - ü ↑ †
6/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ À ¾ ¼ ⅜ ¼ (+ - ü ↑ †
8/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ À ¾ ¼ ⅜ ¼ (+ - ü ↑ †
REF	, . / ; ’ [] < > ? : " { } Font Num
6/10	+ ± Œ ↓ ⅞ ô ù → μ œ) ¼ † ⊥ 310
6/12	+ ± Œ ↓ ⅞ ô ù → μ œ) ¼ † ⊥ 311
8/12	+ ± Œ ↓ ⅞ ô ù → μ œ) ¼ † ⊥ 322

Fonts : SUPP(lemental) ROMAN

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à á â ç é è ê ë ì ï ñ
6/12	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à á â ç é è ê ë ì ï ñ
8/12	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à á â ç é è ê ë ì ï ñ

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	“ < § © £ x i \ / L r T † ‡ ’ ” > ¶ • € † ‡ - / \
6/12	“ < § © £ x i \ / L r T † ‡ ’ ” > ¶ • € † ‡ - / \
8/12	“ < § © £ x i \ / L r T † ‡ ’ ” > ¶ • € † ‡ - / \

REF	1 2 3 4 5 6 7 8 9 0 - = ‘ ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ Å ¾ ⅞ ⅝ ⅜ (+ - ü † ‡
6/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ Å ¾ ⅞ ⅝ ⅜ (+ - ü † ‡
8/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ Å ¾ ⅞ ⅝ ⅜ (+ - ü † ‡

REF	, . / ; ’ [] < > ? : " { } Font Num
6/10	+ ± ¶ ↓ ⅔ ô ù → μ α) ¼ † ‡ 320
6/12	+ ± ¶ ↓ ⅔ ô ù → μ α) ¼ † ‡ 321
8/12	+ ± ¶ ↓ ⅔ ô ù → μ α) ¼ † ‡ 322

Fonts : SUPP(lemental) ROMAN BOLD

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë î ï ñ
6/12	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë î ï ñ
8/12	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë î ï ñ

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	“ < § © £ X i \ / L r T † † ’ ” » ¶ ® € ÷ ÷ - / \
6/12	“ < § © £ X i \ / L r T † † ’ ” » ¶ ® € ÷ ÷ - / \
8/12	“ < § © £ X i \ / L r T † † ’ ” » ¶ ® € ÷ ÷ - / \

REF	1 2 3 4 5 6 7 8 9 0 - = ‘ ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ¼ ½ ¾ ¼ (+ - ü ↑ †
6/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ¼ ½ ¾ ¼ (+ - ü ↑ †
8/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ¼ ½ ¾ ¼ (+ - ü ↑ †

REF	, . / ; ’ [] < > ? : " { } Font Num
6/10	← ± ™ ↓ ⅞ ô ù → μ α) ¼ † ⊥ 323
6/12	← ± ™ ↓ ⅞ ô ù → μ α) ¼ † ⊥ 324
8/12	← ± ™ ↓ ⅞ ô ù → μ α) ¼ † ⊥ 325

Fonts : SUPP(lemental) ROMAN SLANTED

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 17
6/12	, . / ; ' [] < > ? : " { } 18
8/10	, . / ; ' [] < > ? : " { } 19
8/12	, . / ; ' [] < > ? : " { } 20

Fonts : SAN SERIF

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/14	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/15	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/25	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
12/25	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/14	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/15	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/25	a b c d e f g h i j k l m n o p q r s t u v w x y z
12/25	a b c d e f g h i j k l m n o p q r s t u v w x y z

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/14	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/15	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/25	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
12/25	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

REF	, . / ; ' [] < > ? : " { } Font Num
8/14	, . / ; ' [] < > ? : " { } 207
8/15	, . / ; ' [] < > ? : " { } 208
8/25	, . / ; ' [] < > ? : " { } 212
12/25	, . / ; ' [] < > ? : " { } 200

Fonts : SAN SERIF

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/15	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/15	a b c d e f g h i j k l m n o p q r s t u v w x y z

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/15	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 21
6/12	, . / ; ' [] < > ? : " { } 22
8/10	, . / ; ' [] < > ? : " { } 23
8/12	, . / ; ' [] < > ? : " { } 24
8/15	, . / ; ' [] < > ? : " { } 215

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/15	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/15	a b c d e f g h i j k l m n o p q r s t u v w x y z

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/15	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 25
6/12	, . / ; ' [] < > ? : " { } 26
8/10	, . / ; ' [] < > ? : " { } 27
8/12	, . / ; ' [] < > ? : " { } 28
8/15	, . / ; ' [] < > ? : " { } 219

Fonts : SAN SERIF ITALIC

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/15	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/15	a b c d e f g h i j k l m n o p q r s t u v w x y z

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/15	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 29
6/12	, . / ; ' [] < > ? : " { } 30
8/10	, . / ; ' [] < > ? : " { } 31
8/12	, . / ; ' [] < > ? : " { } 32
8/15	, . / ; ' [] < > ? : " { } 220

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
<u>6/10</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
<u>6/12</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
<u>8/10</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
<u>8/12</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
<u>8/15</u>	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
<u>6/10</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
<u>6/12</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
<u>8/10</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
<u>8/12</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
<u>8/15</u>	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
<u>6/10</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
<u>6/12</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
<u>8/10</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
<u>8/12</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
<u>8/15</u>	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>

REF	, . / ; ' [] < > ? : " { } Font Num
<u>6/10</u>	<u>, . / ; ' [] < > ? : " { }</u> 33
<u>6/12</u>	<u>, . / ; ' [] < > ? : " { }</u> 34
<u>8/10</u>	<u>, . / ; ' [] < > ? : " { }</u> 35
<u>8/12</u>	<u>, . / ; ' [] < > ? : " { }</u> 36
<u>8/15</u>	<u>, . / ; ' [] < > ? : " { }</u> 224

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
3/5	ABCDEF GHI JKLMNOP QRSTUVWXYZ
4/7	ABCDEF GHI JKLMNOP QRSTUVWXYZ
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
3/5	abcdefghijklmnopqrstu vwxyz
4/7	abcdefghijklmnopqrstu vwxyz
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
3/5	1234567890-='!@#\$%&*()_+~
4/7	1234567890-='!@#\$%&*()_+~
REF	, . / ; ' [] < > ? : " { } Font Num
3/5	, . / ; ' [] < > ? : " { } 201
4/7	, . / ; ' [] < > ? : " { } 202

Fonts : SAN SERIF TITLE

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
3/5	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
4/7	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
3/5	a b c d e f g h i j k l m n o p q r s t u v w x y z
4/7	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
3/5	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
4/7	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
3/5	, . / ; ' [] < > ? : " { } 213
4/7	, . / ; ' [] < > ? : " { } 214

Fonts : SAN SERIF BOLD TITLE

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	Â Æ Ç É Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë ì í ñ
6/12	Â Æ Ç É Ê Ë Ì Í Ñ Ò Ó Ù Ú Û à â æ ç é è ê ë ì í ñ
8/12	Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ñ Ò Ó Ô Õ Ö Ø Ù Ú

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	“ < § © £ X i \ / L r T + ’ ” > ¶ © € ÷ ¿ - / \
6/12	“ < § © £ X i \ / L r T + ’ ” > ¶ © € ÷ ¿ - / \
8/12	á â ã ä å æ ç è é ê ë ì í î ï ñ ò ó ô õ ö ø ù ú

REF	1 2 3 4 5 6 7 8 9 0 - = ‘ ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ Å ¾ ⅞ ¼ (+ - ü ↑
6/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ Å ¾ ⅞ ¼ (+ - ü ↑
8/12	± ² ³ µ ¶ · ¸ ¹ º ¼ à á â ß € ¥ ¤ © ® «

REF	, . / ; ’ [] < > ? : " { } Font Num
6/10	← ± (E ↓ ⅞ ô ù → µ œ) ¼ ⊥ 330
6/12	← ± (E ↓ ⅞ ô ù → µ œ) ¼ ⊥ 331
8/12	> § Ò Ÿ † ¿ œ € 0 y 332

Fonts : SUPP(lemental) SAN SERIF

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ò Ü à á â ç é è ê ë ì í ñ
6/12	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ò Ü à á â ç é è ê ë ì í ñ
8/12	Â Æ Ç É È Ê Ë Ì Í Ñ Ò Ó Ù Ò Ü à á â ç é è ê ë ì í ñ

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	“ ‹ § ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿
6/12	“ ‹ § ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿
8/12	“ ‹ § ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿

REF	1 2 3 4 5 6 7 8 9 0 - = ‘ ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ ª ¼ ½ ¾ ¿ (+ - ü ↑
6/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ ª ¼ ½ ¾ ¿ (+ - ü ↑
8/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ½ ª ¼ ½ ¾ ¿ (+ - ü ↑

REF	, . / ; ’ [] < > ? : " { } Font Num
6/10	← ± ⊕ ↓ ⅓ ⅔ ÷ → μ ∞) ¼] ⊥ 340
6/12	← ± ⊕ ↓ ⅓ ⅔ ÷ → μ ∞) ¼] ⊥ 341
8/12	← ± ⊕ ↓ ⅓ ⅔ ÷ → μ ∞) ¼] ⊥ 342

Fonts : SUPP(lemental) SAN SERIF BOLD

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	Â Æ Ç É È Ê Ë Î Ï Ñ Ò Ó Ù Ú Û à á â ç é è ê ë ì í ñ
6/12	Â Æ Ç É È Ê Ë Î Ï Ñ Ò Ó Ù Ú Û à á â ç é è ê ë ì í ñ
8/12	Â Æ Ç É È Ê Ë Î Ï Ñ Ò Ó Ù Ú Û à á â ç é è ê ë ì í ñ
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	“ ‹ § © £ ¤ \ / , L r T † ‡ ’ ” » ¶ © ¢ ÷ ¿ - / \
6/12	“ ‹ § © £ ¤ \ / , L r T † ‡ ’ ” » ¶ © ¢ ÷ ¿ - / \
8/12	“ ‹ § © £ ¤ \ / , L r T † ‡ ’ ” » ¶ © ¢ ÷ ¿ - / \
REF	1 2 3 4 5 6 7 8 9 0 - = ‘ ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ¼ ½ ¾ ⅓ ⅔ (+ - ù † ‡
6/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ¼ ½ ¾ ⅓ ⅔ (+ - ù † ‡
8/12	1 2 3 4 5 6 7 8 9 0 • ■ ‘ ¼ ½ ¾ ⅓ ⅔ (+ - ù † ‡
REF	, . / ; ’ [] < > ? : " { } Font Num
6/10	← ± Œ ↓ ⅓ ô ù → μ œ) ¼ † ‡ 350
6/12	← ± Œ ↓ ⅓ ô ù → μ œ) ¼ † ‡ 351
8/12	← ± Œ ↓ ⅓ ô ù → μ œ) ¼ † ‡ 352

Fonts : SUPP(lemental) SAN SERIF SLANTED

AD-A103 199

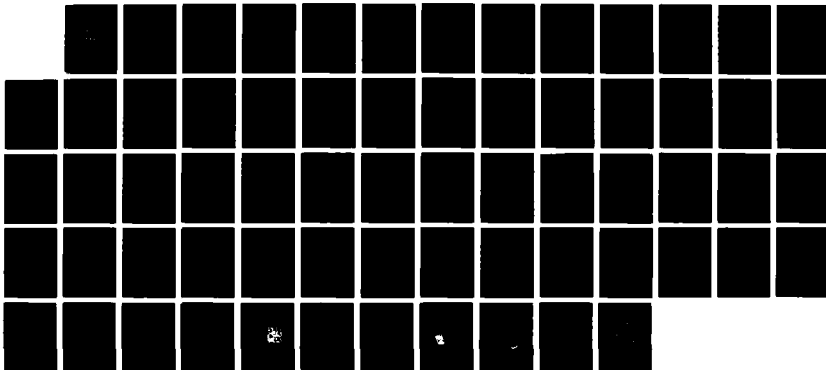
AN IMPROVED USER INTERFACE FOR AN INTERACTIVE GRAPHICS
FIGURE ILLUSTRATOR(U) NAVAL POSTGRADUATE SCHOOL
MONTEREY CA T J BEDA JUN 87

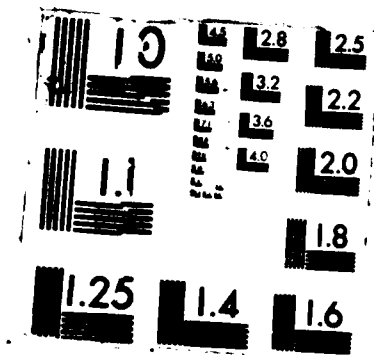
2/2

UNCLASSIFIED

F/G 12/5

ML





REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 38
6/12	, . / ; ' [] < > ? : " { } 39
8/10	, . / ; ' [] < > ? : " { } 40
8/12	, . / ; ' [] < > ? : " { } 41

Fonts : TYPEWRITER

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 42
6/12	, . / ; ' [] < > ? : " { } 43
8/10	, . / ; ' [] < > ? : " { } 44
8/12	, . / ; ' [] < > ? : " { } 45

Fonts : TYPEWRITER BOLD

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/10	a b c d e f g h i j k l m n o p q r s t u v w x y z
8/12	a b c d e f g h i j k l m n o p q r s t u v w x y z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! 0 # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 - = ' ! 0 # \$ % & * () _ + ~
6/12	1 2 3 4 5 6 7 8 9 0 - = ' ! 0 # \$ % & * () _ + ~
8/10	1 2 3 4 5 6 7 8 9 0 - = ' ! 0 # \$ % & * () _ + ~
8/12	1 2 3 4 5 6 7 8 9 0 - = ' ! 0 # \$ % & * () _ + ~
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / ; ' [] < > ? : " { } 46
6/12	, . / ; ' [] < > ? : " { } 47
8/10	, . / ; ' [] < > ? : " { } 48
8/12	, . / ; ' [] < > ? : " { } 49

Fonts : TYPEWRITER SLANTED

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
6/12	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
8/10	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
8/12	<u>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</u>
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
6/12	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
8/10	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
8/12	<u>a b c d e f g h i j k l m n o p q r s t u v w x y z</u>
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
6/12	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
8/10	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
8/12	<u>1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~</u>
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	<u>, . / ; ' [] < > ? : " { }</u> 50
6/12	<u>, . / ; ' [] < > ? : " { }</u> 51
8/10	<u>, . / ; ' [] < > ? : " { }</u> 52
8/12	<u>, . / ; ' [] < > ? : " { }</u> 53

Fonts : TYPEWRITER UNDERLINED

REF A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

3/5 ABCDEF GHIJKLMNOPQRSTUVWXYZ

REF a b c d e f g h i j k l m n o p q r s t u v w x y z

3/5 abcdefghijklmnopqrstuvwxyz

REF 1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

3/5 1234567890-='!@#\$%&*()_+~

REF , . / ; ' [] < > ? : " { } Font Num

3/5 , . / ; ' [] < > ? : " { } 54

Fonts : TYPEWRITER TITLE

REF A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

6/10 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF a b c d e f g h i j k l m n o p q r s t u v w x y z

6/10 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

REF 1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

6/10 1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~

REF , . / ; ' [] < > ? : " { } Font Num

6/10 , . / ; ' [] < > ? : " { } 470

Fonts : ORATOR

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	α ∞ + Δ ∇ φ Γ ~ ≈ θ × Λ # ⇒ ≡ Π ψ ° Σ § ¶ √ Ω Ξ Τ Ϛ
6/12	α ∞ + Δ ∇ φ Γ ~ ≈ θ × Λ # ⇒ ≡ Π ψ ° Σ § ¶ √ Ω Ξ Τ Ϛ
8/12	α ∞ + Δ ∇ φ Γ ~ ≈ θ × Λ # ⇒ ≡ Π ψ ° Σ § ¶ √ Ω Ξ Τ Ϛ

REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	α β χ δ ε φ γ η λ θ κ λ μ ν θ π φ ρ σ τ f ω ξ υ Ϛ
6/12	α β χ δ ε φ γ η λ θ κ λ μ ν θ π φ ρ σ τ f ω ξ υ Ϛ
8/12	α β χ δ ε φ γ η λ θ κ λ μ ν θ π φ ρ σ τ f ω ξ υ Ϛ

REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	< < \ / - - > [• } } ≠ ~ ↓ ∴ - { }] L l v r ↓
6/12	< < \ / - - > [• } } ≠ ~ ↓ ∴ - { }] L l v r ↓
8/12	< < \ / - - > [• } } ≠ ~ ↓ ∴ - { }] L l v r ↓

REF	, . / ; ' [] < > ? : " { } Font Num
6/10	{ } { ± [] n ≤ ≥ f] r + + 362
6/12	{ } { ± [] n ≤ ≥ f] r + + 363
8/12	{ } { ± [] n ≤ ≥ f] r + + 364

Fonts : DEC Technical

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
3/5	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
4/7	α ∞ ÷ Δ ∇ Φ Γ ~ ≈ Θ × Λ Η ⇒ ≡ Π Ψ ° Σ § ¶ √ Ω Ξ Υ ζ
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
3/5	a b c d e f g h i j k l m n o p q r s t u v w x y z
4/7	α β χ δ ε φ γ η ι θ κ λ μ ν ϑ π ψ ρ σ τ ϖ ω ξ υ ζ
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
3/5	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
4/7	< ≃ ↔ → [• }) ≠ ↯ ↓ ∴ − ()] L] V (↓
REF	, . / ; ' [] < > ? : " { } Font Num
3/5	, . / ; ' [] < > ? : " { } 360
4/7	() { ± [∩ ∪ ≤ ≥ ∫] ↯ ↰ ↱ 361

Fonts : DEC TECHNICAL TITLE

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	Α ≡ □ Δ † θ Γ Θ ∇ < > Λ { } α Π Φ ↓ Σ † Γ Ψ Ω Ξ †
6/12	Α ≡ □ Δ † θ Γ Θ ∇ < > Λ { } α Π Φ ↓ Σ † Γ Ψ Ω Ξ †
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	α β χ δ ε η γ θ λ κ λ μ ν ο π φ ρ σ τ υ ψ ω ξ x ς
6/12	α β χ δ ε η γ θ λ κ λ μ ν ο π φ ρ σ τ υ ψ ω ξ x ς
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 ÷) L ↓ * - - ± - ∞ ∫ ∫ Γ
6/12	1 2 3 4 5 6 7 8 9 0 +) L ↓ * - - ± - ∞ ∫ ∫ Γ
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	[] ~ " ' () [] ° · () 55
6/12	[] ~ " ' () [] ° · () 56

Fonts : DEC Symbol Font

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z	
6/10mo	$\Lambda \equiv \square \Delta \uparrow \theta \Gamma \theta \nabla < > \Lambda \vee \exists \alpha \Pi \phi \downarrow \Sigma + \tau \psi \Omega \Xi \rightarrow \dagger$	
6/12mo	$\Lambda \equiv \square \Delta \uparrow \theta \Gamma \theta \nabla < > \Lambda \vee \exists \alpha \Pi \phi \downarrow \Sigma + \tau \psi \Omega \Xi \rightarrow \dagger$	
6/10mt	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z	
6/12mt	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z	
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z	
6/10mo	$\alpha \beta \chi \delta \epsilon \eta \gamma \theta \iota \rho \kappa \lambda \mu \nu \circ \pi \phi \rho \sigma \tau \upsilon \psi \omega \xi \times \varsigma$	
6/12mo	$\alpha \beta \chi \delta \epsilon \eta \gamma \theta \iota \rho \kappa \lambda \mu \nu \circ \pi \phi \rho \sigma \tau \upsilon \psi \omega \xi \times \varsigma$	
6/10mt	$\Pi \Xi \phi \diamond \heartsuit \spadesuit / \setminus \backslash /$	
6/12mt	$\Pi \Xi \phi \diamond \heartsuit \spadesuit / \setminus \backslash / \text{T T}$	
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~	
6/10mo	1 2 3 4 5 6 7 8 9 0 $\div \diamond \approx \wedge \bullet \cup \otimes \neq \sqrt{\infty} \in \int \emptyset \neq \emptyset$	
6/12mo	1 2 3 4 5 6 7 8 9 0 $\div \diamond \approx \wedge \bullet \cup \otimes \neq \sqrt{\infty} \in \int \emptyset \neq \emptyset$	
6/10mt	* $\uparrow \uparrow \rightarrow \leftarrow \rightleftarrows \times \times \times \downarrow \leftarrow \rightarrow \cup \perp \lrcorner \supset \leq \geq \approx \lt \gt \Gamma \gg$	
6/12mt	* $\uparrow \uparrow \rightarrow \leftarrow \rightleftarrows \times \times \times \downarrow \leftarrow \rightarrow \cup \perp \lrcorner \supset \leq \geq \approx \lt \gt \Gamma \gg \text{T}$	
REF	, . / ; ' [] < > ? : " { } Font Num	
6/10mo	$\neg \leq \geq \cdot \circ \neq \emptyset \pm \oplus \ominus \subset \vee \angle \pounds$ 57	
6/12mo	$\neg \leq \geq \cdot \circ \neq \emptyset \pm \oplus \ominus \subset \vee \angle \pounds$ 58	
6/10mt	$\approx \Rightarrow \Uparrow \Upsilon \div \Pi \text{I} \cup \neg \text{L} \text{S} \text{C}$ 59	
6/12mt	$\approx \Rightarrow \Uparrow \Upsilon = \text{I} \text{I} \cup \neg \text{L} \text{S} \text{C} \text{T T}$ 60	

Fonts : MATH mo & mt

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ σ τ υ φ χ ψ ω ρ •
6/12	α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ σ τ υ φ χ ψ ω ρ •
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	1 2 3 4 5 6 7 8 9 ± ⁻ ≤ ≥ ~ ≈ 0 1 2 3 4 5 6 7 8 9 +
6/12	1 2 3 4 5 6 7 8 9 ± ⁻ ≤ ≥ ~ ≈ 0 1 2 3 4 5 6 7 8 9 +
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	Ρ Σ Τ Υ Φ Χ Ψ Ω [Π Ν [° Α ε Γ Δ Ε Ζ Κ Θ Ι / Λ ∞
6/12	Ρ Σ Τ Υ Φ Χ Ψ Ω [Π Ν [° Α ε Γ Δ Ε Ζ Κ Θ Ι / Λ ∞
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	Μ Ξ Ο (Η √ { }] ∫ ∇ Β - ÷ 372
6/12	Μ Ξ Ο (Η √ { }] ∫ ∇ Β - ÷ 373

Fonts : SCIENTIFIC

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
3/5	α β γ δ ε ζ η θ λ κ μ ν ξ ο π ρ σ τ υ φ χ ψ ω ρ •
4/7	α ∞ ÷ Δ ∇ Φ Γ ~ ≈ Θ × Λ Η ⇒ ≡ Π Ψ ° Σ § ¶ √ Ω Ξ Υ C
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
3/5	1 2 3 4 5 6 7 8 9 ± ⁻ ≤ ≥ ~ ≈ 0 1 2 3 4 5 6 7 8 9 +
4/7	α β χ δ ε φ γ η ι θ κ λ μ ν θ π ψ ρ σ τ f ω ξ υ ζ
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
3/5	Ρ Σ Τ Υ Φ Χ Ψ Ω [Π Ν] ° Α ε Γ Δ Ε Ζ Κ Θ Ι / Λ ∞
4/7	< ^ √ - - > [• } } ≠ ¬ ↓ ∴ - ()] L] √ (↓
REF	, . / ; ' [] < > ? : " { } Font Num
3/5	Μ Ε Ο (Η √ {)] ∫ ∇ Β — ÷ 370
4/7	() { ± [∩ ∪ ≤ ≥ ∫] ← → 371

Fonts : SCIENTIFIC TITLE

REF	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/10	α τ η λ ε _ ∇ Δ ι ° ' □ ⊥ ο * ? ρ Γ ~ ↓ U ω ∩ ↑ C
6/12	α τ η λ ε _ ∇ Δ ι ° ' □ ⊥ ο * ? ρ Γ ~ ↓ U ω ∩ ↑ C
8/12	α τ η λ ε _ ∇ Δ ι ° ' □ ⊥ ο * ? ρ Γ ~ ↓ U ω ∩ ↑ C
REF	a b c d e f g h i j k l m n o p q r s t u v w x y z
6/10	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
6/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8/12	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REF	1 2 3 4 5 6 7 8 9 0 - = ' ! @ # \$ % & * () _ + ~
6/10	1 2 3 4 5 6 7 8 9 0 + x ◊ " ~ < ≤ = > ≠ √ ∆ - ÷ \$
6/12	1 2 3 4 5 6 7 8 9 0 + x ◊ " ~ < ≤ = > ≠ √ ∆ - ÷ \$
8/12	1 2 3 4 5 6 7 8 9 0 + x ◊ " ~ < ≤ = > ≠ √ ∆ - ÷ \$
REF	, . / ; ' [] < > ? : " { } Font Num
6/10	, . / [] + + ; : \ () { } 300
6/12	, . / [] + + ; : \ () { } 301
8/12	, . / [] + + ; : \ () { } 302

Fonts : APL

APPENDIX B

FUNCTION & FILE REFERENCE

The following pages are provided for easy reference of each file, its functions and general operation as well as the location of each function. The functions listed are only those written for NPSDRAW, i.e. does not include any system functions.

1. FUNCTION TO FILE CROSS REFERENCE

Files - Functions		
File Name	Function Name	Operation
1pt_inside.c	one_pt_inside()	To view a certain type of figure that is uniquely defined by one point, and to see if it lies inside a given rectangle.
align.c	get_user_alignment()	To prompt the user for page alignment.
alter.c	alter_attributes()	To get the users values to change and call the appropriate functions.
angle.c	get_angle()	To calculate the angle that the second point makes with the first point and the horizontal. Used in drawing arcs.
arc.c	draw_arc()	To draw an arc and insert it into the list.
arc2fnl.c	arc_to_file() arc2list() barc2list()	(1) To write the arc definition to a file, (2) to read an arc definition from a file into the working linked list and (3) to read an arc definition from a file into a temporary linked list for block append.
arc_inside.c	arc_inside()	To determine if a given arc is inside a given rectangle.
arrow.c	draw_arrow()	To draw an arrow
bad.c	bad_data()	To inform the user that there is bad data in the file being read in.
bfile2list.c	bfile2list()	To read a file and insert the drawing in the temporary block linked list used in block append.
binsert.c	binsert_into_list()	To insert the figure, with its associated data into the block linked list.
blockedit.c	blockedit() bblockedit() move_picked_objs() bmove_file() draw_page() bdraw_page() find_objects() bfind_objects() copy-block() draw copy()	Functions used in Move a Block, Copy a block and Block Append.
border.c	get_bottom() set_bottom() get_left() set left() update_bounds()	To manage the values that allow the page to be moved up and down, or across. These values represent the bottom or left values of the ortho call.
bread.c	bget_from_file()	The driver function for block append - read a drawing in from file and attach to cursor for positioning.

File Name	Function Name	Operation
change.c	change()	To input new values for the current set attributes
changefont.c	changefont()	To alter the font of a chosen text string.
changel.s.c	changel.s()	To alter the linestyle of a chosen figure.
changelw.c	changelw()	To alter the linewidth of a chosen figure.
changetexta.c	changetexta()	To alter the texture of a chosen figure.
check_figure.c	check_arc() check_circle() check_line() check_text()	Determines whether a chosen point is near the control point of a figure.
circle.c	draw_circle()	To draw a circle given a start point and insert it into the linked list.
circle2fnl.c	circle_to_file() circle2list() bcircle2list()	(1) Writes a circle definition to a file (2) writes a circle definition into the working linked list and (3) writes a circle definition into a temporary linked list for block append.
clean_list.c	clean_list() clean_linestyle() clean_font() clean_texture() check 1st not default()	Removes attributes from the list that are unused.
cleanup.c	cleanup()	To exit the system and reset all values.
clearpage.c	clearpage()	Reset the list so no user drawings exist.
copy_last.c	reproduce last()	Copy the last figure in the linked list
copy_obj.c	copy_obj()	To move the picked figure to the required position to copy it.
correct.c	check_correct()	To check that the chosen figure is in fact the figure that the user intended to pick.
current.c	init_attributes() set_real_linewidth() get_real_linewidth() set_current_linewidth() get_current_linewidth() set_current_texture() get_current_texture() set_current_font() get_current_font() set_current_linestyle() get_current_linestyle()	These functions manage the variables that are the current state of the attributes.
cursor.c	newcursor()	Changes the cursor definition and turns on the new cursor.

File Name	Function Name	Operation
dirdraw.c	dirdraw() dir-scroll()	This subroutine queries the user to access an external file and load it into the Init global structure. Additional scrolling menu added to the Mike Gaddis Directory package
display_pos.c	display_position()	To display the position of the cursor in page coords ie 8.5x11"
dist.c	distance_between()	To calculate the distance between two points.
draw.c	draw_a()	The driver for drawing all the primitives available.
draw_erased.c	draw_erased()	To draw the page displaying only those figures that have been erased.
duplicate.c	duplicate()	To duplicate the required figure
edit_menu.c	edit_menu block edit()	To display the edit or block edit menu and get the users selection
edit_obj.c	edit_opj()	To be the "driver" for the single edit functions.
ellipse2fnl.c	ellipse to file() ellipse2list() bellipse2list()	(1) To write an ellipse definition to file, (2) To read an ellipse definition from a file and insert into the working linked list and (3) To read an ellipse definition from a file and insert into a temporary linked list.
exit.c	check_exit()	Check if the user wishes to exit without saving the picture.
fig_menu.c	fig_menu()	To display the figure menu and get the user's selection.
file2list.c	file2list() get_next()	To read a file and insert the drawing in the working linked list.
find.c	find_figure()	To find if a figure exists.
font2fnl.c	font to file() font2list() bfont2list()	(1) Write the font value to a file (2) read the font value from a file to the working list and (3) read the font value from a file into the working temporary linked list
font_size.c	font_size()	To calculate the size of the red box to be placed around text to indicate the different size of the font.
fontsource.c	fontsource()	To get the font selection source from the user i.e. by numeric value or by selection of style and size
full.c	full_memory()	To flag the user that he has filled up all available memory.

File Name	Function Name	Operation
get_dir.c	get_directory()	To call M GADDIS's directory functions.
getblock.c	get_block()	To get the coords for a rectangle that inscribes an area for figure selection.
getcicle.c	get_circle()	To get from the user a circular section of the screen. The center point has already been determined, now all that is needed is the radius.
getlist.c	getlist()	Get the listing for the directory
getq.c	get_queue()	Steve Firths idea of a queue's operation
graphlab.c	graphlab()	It is the logo for the NPS Graphics and Video Laboratory
gt1_inside.c	gt1ptinside()	To see if a certain class of figures that has many points, lies within a region.
init.c	initialise()	Undertake all required initialization of variables and settings for NPSDRAW.
init_guides.c	init_guides()	Inserts four structures into the working linked list that represent the grids & thesis box - guides on the drawing area
init_textures.c	init_textures()	Initialization of all possible textures. All texture definitions are kept here.
insert.c	insert_into_list()	The driver that inserts all figures with data into the working linked list.
insert_arc.c	insert_arc() binsert_arc()	Takes all arc relevant information and inserts it into either the working linked list or the temporary linked list for block append.
insert_ell.c	insert_ellipse() binsert_ellipse()	Takes all ellipse data and inserts it into the working linked list or the temporary linked list for block append.
insert_poly.c	insert_diamond() insert_triangle()	Takes diamond or triangle data and inserts it into the working linked list from a drawing sequence. Not used for file to list or block append since in files they are referred to as polygons only.
inside.c	inside()	To see if a particular given point lies within a given rectangle. Used for selecting "rectangle" prompts on the screen.
instruct.c	instructions()	To display the function of the mouse buttons and keys.
invalid.c	invalid_entry()	To flag to the user that the entry he has entered to change an attribute is invalid.

File Name	Function Name	Operation
line.c	draw_line()	To draw a single line given the start point.
line2fnl.c	line_to_file() line2list() bline2list()	(1) To write a line to a file (2) To read a line definition from a file into the working linked list and (3) to read a line definition from a file into the temporary linked list for block append.
line_menu.c	line_menu()	To prompt the user for the various lines that can be drawn via a menu selection
line_size.c	line_size() lw2pix()	To map the set linewidth attribute to the real linewidth and map the chosen linewidth to a number of pixels on the screen.
list2file.c	list_to_file()	The driver that controls the storage of all existing figures to the file.
ls2fnl.c	linestyle_to_file() linestyle2list() blinestyle2list()	(1) to write a line style to a file, (2) to read a line style from a file into the working linked list and (3) read a line style from a file and into the temporary linked list for block append.
lw2fnl.c	linewidth_to_file() linewidth2list() blinewidth2list()	(1) to write a line width to a file, (2) to read a line width from a file into the working linked list and (3) read a line width from a file and into the temporary linked list for block append.
main_menu.c	main_menu()	To display the main menu and get the users selection.
map.c	map() map to list()	To map from page to screen coords and vice versa.
mod_arc.c	modify_arc()	To change the position of a given arc by the given amount without changing its attributes.
mod_circle.c	modify_circle()	To move the chosen circle by the given amount.
mod_ellipse.c	modify_ellipse()	To change the position of a given ellipse by a given amount without changing its attributes
mod_line.c	modify_line()	To change the position of a given line by a given amount without changing its attributes.
mod_obj.c	modify_object()	To select the given function to move the chosen figure.
mod_poly.c	modify_polygon()	To change the position of a given polygon a given amount without changing its attributes.

File Name	Function Name	Operation
mod_seed.c	modify_seed()	To change the position of a seed
mod_text.c	modify_text()	To change the position of a given text string a given amount without changing any of its attributes.
move_obj.c	move_obj()	To move an object on the page.
move_page.c	move_page()	To reduce the page so it can be seen in its entirety and allow the user to move all the figures in one motion.
multi_line.c	draw_multi_line()	Draw continuous lines , i.e. adjoined non parallel segments.
npsdraw.c	main()	The driver for the illustrator system.
not_found.c	not_found()	To inform the user that when he chose an object for editing the cursor was not close enough.
poly2fnl.c	poly_to_file() polygon2list() bpolygon2list() do_rectangle() bdo_rectangle() do_polygon() bdo_polygon()	(1) To write a polygon definition to a file , (2) to read a polygon definition from a file into the working linked list and (3) to read the polygon definition from a file into the temporary linked list for block append.
polygon.c	draw_polygon()	To draw a polygon given the start point.
poputil.c	initpopup() pop- up() showpopup()	(1) Initialize the system for use of the popup menu system, (2) generate the menus and get the users selection, and (3) show the menus.
printable.c	printable()	To calculate if a value is a printable ASCII character.
read.c	get_from_file()	The driver function to read a drawing in from file.
read_menu.c	read_menu()	Displays the read menu and get the users selection.
readfonts.c	readfonttable() index2ftable()	(1) Read the font table and fill arrays with the data, (2) Take a font number and find the index to its data in the array.
rectangle.c	draw_rectangle()	Draws a rectangle given the start point.
redraw.c	redraw_figures()	Redraw the complete picture from linked list to one or both buffers.
remove.c	remove_last()	To remove the last chosen drawn figure. e.g. remove the last drawn circle when circles are selected as the primitive to be drawn.

File Name	Function Name	Operation
seed.c	insert_seed() binsert_seed()	(1) Insert a seed into the working link list and (2) insert a seed into the temporary linked list for block append.
seed2fnl.c	seed_to_file() seed2list() bseed2list()	(1) To write the seed definition from the list to a file, (2) to write the seedpoint definition from a file into the working linked list and (3) to write the seedpoint definition from a file into the temporary linked list for block append.
sel_font.c	select_font()	To get the user font selection.
sel_ls.c	select_ls()	To get the user line style selection.
sel_lw.c	select_lw() var_lw()	To get the user line width selection.
sel_text.c	select-text()	To get the user texture selection.
sensible.c	not_sensible()	To flag to the user that he has tried to alter an attribute of a figure that does not make sense. For example changing a texture of a text string would be useless.
set.c	set_orthoview()	To set the ortho and viewport for the correct alignment so the page can be drawn.
set_fname.c	set_filename()	To display the filename of the file read in on the top right corner of the screen.
set_mouse.c	set_mouse() restrict_mouse() reset_mouse()	Various functions to manipulate the mouse. The last two functions are used by the line functions to force the mouse to be either horizontal or vertical.
setnewscrn.c	set_screen()	To set up the right hand side of the screen.
smoothline.c	draw_smooth_line()	To draw a smooth continuous, cursive style line.
store_alignment.c	set_alignment() get_alignment()	To store and manage the value of page alignment.
syntax.c	syntax_error()	To flag to the user that one of the op-codes read in from the file was not recognized.
text.c	draw_text() insert_hdgl() replace_hdgl w hdg2()	Draws text given the starting point. The headings are instruction boxes that appear on the screen.

File Name	Function Name	Operation
text2fnl.c	text_to_file() text2list() btext2list()	(1) To write a text string from the linked list into a file (2) to write a text string from a file to the working linked list and (3) to write a text string from a file to the temporary linked list for block append.
texture2fnl.c	texture_to_file() texture2list() btexture2list()	(1) To write a texture value from the linked list into a file (2) to write a texture value from a file to the working linked list and (3) to write a text value from a file to the temporary linked list for block append.
toggle.c	toggle_grid()	To toggle the grid overlay that is displayed on the screen.
view_page.c	view_page()	To allow the user to view the entire page unclipped at a reduced scale.
welcome.c	opening_display()	To display the opening banner page with logo.
width2lwnum.c	width2lwnum()	Return the attribute value of the passed real linewidth.
write.c	go_to_file()	The driver function to write the drawing to a file.
write_menu.c	write_menu()	To display the write menu and to get the users selection.

2. FILE TO FUNCTION CROSS REFERENCE

Functions - Files	
Functions	Files
alter_attributes()	alter.c
arc2list()	arc2fnl.c
arc_inside()	arc_inside.c
arc_to_file()	arc2fnl.c
bad_data()	bad.c
barc2list()	arc2fnl.c
blockedit()	blockedit.c
bcircle2list()	circle2fnl.c
bdo_polygon()	poly2fnl.c
bdo_rectangle()	poly2fnl.c
bdraw_page()	blockedit.c
bellipse2list()	ellipse2fnl.c
bfile2list()	bfile2list.c
bfind_objects()	blockedit.c
bfont2list()	font2fnl.c
bget_from_file()	bread.c
binsert_arc()	insert.c
binsert_ellipse()	insert_ell.c
binsert_into_list()	binsert.c
binsert_seed()	seed.c
bline2list()	line2fnl.c
blinestyle2list()	ls2fnl.c
blinewidth2list()	lw2fnl.c
block_edit()	edit_menu.c
blockedit()	blockedit.c
bmove_file()	blockedit.c
bpolygon2list()	poly2fnl.c
bseed2list()	seed2fnl.c
btext2list()	text2fnl.c
btexture2list()	texture2fnl.c
change()	change.c
changefont()	changefont.c
changels()	changels.c
changelw()	changelw.c
changetexta()	changetexta.c
check_1st_lw()	clean_list.c
check_1st_not_default()	clean_list.c
check_arc()	check_figure.c
check_circle()	check_figure.c
check_correct()	correct.c
check_exit()	exit.c
check_line()	check_figure.c
check_text()	check_figure.c
circle2list()	circle2fnl.c
circle_to_file()	circle2fnl.c

Functions	Files
clean_font()	clean_list.c
clean_linestyle()	clean_list.c
clean_list()	clean_list.c
clean_texture()	clean_list.c
cleanup()	cleanup.c
clearpage()	clearpage.c
copy_obj()	copy_obj.c
dirdraw()	dirdraw.c
dirscroll()	dirdraw.c
display_position()	display_pos.c
distance_between()	dist.c
do_polygon()	poly2fnl.c
do_rectangle()	poly2fnl.c
draw_a()	draw.c
draw_arc()	arc.c
draw_arrow()	arrow.c
draw_circle()	circle.c
draw_erased()	draw_erased.c
draw_line()	line.c
draw_multi_line()	multi_line.c
draw_page()	blockedit.c
draw_polygon()	polygon.c
draw_rectangle()	rectangle.c
draw_smooth_line()	smoothline.c()
draw_text()	text.c
duplicate()	duplicate.c
edit_menu()	edit_menu.c
edit_obj()	edit_obj.c
ellipse2list()	ellipse2fnl.c
ellipse_to_file()	ellipse2fnl.c
fig_menu()	fig_menu.c
file2list()	file2list.c
find_figure()	find.c
find_objects()	blockedit.c
font2list()	font2fnl.c
font_size()	font_size.c
font_to_file()	font2fnl.c
fontsource()	fontsource.c
full_memory()	full.c
get_alignment()	store_alignment.c
get_angle()	angle.c
get_block()	getblock.c
get_bottom()	border.c
get_circle()	getcircle.c

Functions	Files
get_current_font()	current.c
get_current_linestyle()	current.c
get_current_linewidth()	current.c
get_current_texture()	current.c
get_directory()	get_dir.c
get_from_file()	read.c
get_left()	border.c
get_next()	file2list.c
get_queue()	getq.c
get_real_linewidth()	current.c
get_user_alignment()	alignment.c
getlist()	getlist.c
go_to_file()	write.c
graphlab()	graphlab.c
gt1pt_inside()	gt1ptinside.c
index2ftable()	readfonts.c
init_attributes()	current.c
init_guides()	init_guides.c
init_textures()	init_textures.c
initialise()	init.c
initpopup()	poputil.c
insert_arc()	insert_arc.c
insert_diamond()	insert_poly.c
insert_ellipse()	insert_ell.c
insert_hdgl()	text.c
insert_into_list()	insert.c
insert_seed()	seed.c
insert_triangle()	insert_poly.c
inside()	inside.c
instructions()	instruct.c
invalid_entry()	invalid.c
line2list()	line2fnl.c
line_menu()	line_menu.c
line_size()	line_size.c
line_to_file()	line2fnl.c
linestyle2list()	ls2fnl.c
linestyle to file()	ls2fnl.c
linewidth2list()	lw2fnl.c
linewidth to file()	lw2fnl.c()
list to file()	list2file.c
lw2pix()	line_size.c
main()	npsdraw.c
main menu()	main_menu.c
map()	map.c
map to list()	map.c

Functions	Files
modify_arc()	mod_arc.c
modify_circle()	mod_circle.c
modify_ellipse()	mod_ellipse.c
modify_line()	mod_line.c
modify_object()	mod_obj.c
modify_polygon()	mod_poly.c
modify_seed()	mod_seed.c
modify_text()	mod_text.c
move_obj()	move_obj.c
move_page()	move_page.c
move_picked_objs()	blockedit.c
newcursor()	cursor.c
not_found()	not_found.c
not_sensible()	sensible.c
one_pt_inside()	1pt_inside.c
opening_display()	welcome.c
poly_to_file()	poly2fnl.c
polygon2list()	poly2fnl.c
popup()	poputil.c
printable()	printable.c
read_menu()	read_menu.c
readfonttable()	readfonts.c
redraw_figures()	redraw.c
remove_last()	remove.c
replace_hdgl_w_hdg2()	text.c
reproduce_last()	copy_last.c
reset_mouse()	set_mouse.c
restrict_mouse()	set_mouse.c
seed to file()	seed2fnl.c
seed2list()	seed2fnl.c
select-text()	sel_text.c
select_font()	sel_font.c
select_ls()	sel_ls.c
select_lw()	sel_lw.c
set_alignment()	store_alignment.c
set_bottom()	border.c
set_current_font()	current.c
set_current_linestyle()	current.c
set_current_linewidth()	current.c
set_current_texture()	current.c
set_filename()	set_fname.c
set_mouse()	set_mouse.c
set_orthoview()	set.c
set_real_linewidth()	current.c
set_screen()	setnewscrn.c

Functions	Files
showpopup()	poputil.c
syntax_error()	syntax.c
text2list()	text2fnl.c
text to file()	text2fnl.c
texture2list()	texture2fnl.c
texture to file()	texture2fnl.c
toggle()	toggle_grid.c
update bounds()	border.c
var_lw()	sel_lw.c
view_page()	view_page.c
width2lwnum()	width2lwnum.c
write menu()	write_menu.c

3. FUNCTIONS: CALLED FROM & CALLS TO

FUNCTIONS		
Function	Called From	Calls To
alter_attributes()	edit_obj()	changefont() changels() changelw() changetex- ta() instructions() select_font() select_ls() select_lw() select_text()
arc2list()	file2list()	insert_arc() map_to_list()
arc_inside()	find_objects()	inside()
arc_to_file()	list2file()	map()
bad_data()	bfile2list() file2list()	get_queue() redraw_figures() set_orthoview()
barc2list()	bfile2list()	map_to_list() binsert_arc()
bblockedit()	block_edit()	bfind_objects() bmove_file() newcursor
bcircle2list()	bfile2list()	binsert_into_list() map_to_list()
bdo_polygon()	bpolygon2list()	binsert_into_list() map_to_list()
bdraw_page()	bmove_file()	get_bottom() get_left() set_orthoview()
bdo_rectangle()	bpolygon2list()	binsert_into_list()
bellipse2list()	bfile2list()	binsert_into_list() map_to_list()
bfile2list()	bget_from_file()	bfile2list() instructions() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture() set_real_linewidth() set_written()
bfind_objects	bblockedit()	[none]

Function	Called From	Calls To
bfont2list()	bfile2list()	binsert_into_list() set_current_font()
bget_from_file()	block_edit()	bfile2list() instructions() get_queue() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture() set_real_linewidth() set_written()
binsert_arc()	barc2list()	binsert_into_list()
binsert_ellipse()	bellipse2list()	binsert_into_list()
binsert_into_list()	barc2list() bcircle2list() bellipse2list() bfile2list() bfont2list() binsert_seed() bline2list() bpoly2list() btext2list() btexture2list()	font_size() full_memory() get_current_font() get_current_texture() get_current_linestyle() get_current_linewidth() get_real_linewidth() set_written()
binsert_seed()	bseed2list()	binsert_into_list()
bline2list()	bfile2list()	binsert2list() full_memory() map_to_list()
blinestyle2list()	bfile2list()	binsert_to_list() set_current_linestyle()
blinewidth2list()	bfile2list()	binsert_to_list() lw2pix() set_current_line() widthdth2lwnum()

Function	Called From	Calls To
blockedit()	block_edit()	copyblock() find_objects() get_block() get_bottom() get_left() get_queue() move_picked_objs() newcursor() redraw_figures() set_orhtoview() update_bounds()
bmove_file()	bblockedit()	bdraw_page() display_position() get_bottom() get_left() get_queue() instructions() modify_object() update_bounds()
bpolygon2list()	bfile2list()	bdo_rectangle() bdo_polygon() full_memory()
bseed2list()	bfile2list()	binsert_seed() map_to_list()
bttext2list()	bfile2list()	binsert_into_list() get_current_font() get_current_linestyle() get_current_linewidth() map_to_list()
bttexture2list()	bfile2list()	binsert_into_list() set_current_texture()
change()	fig_menu() line_menu() main_menu()	insert_into_list() select_font() select_ls() select_lw() select_text() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture()

Function	Called From	Calls To
changefont()	alter_attributes()	insert_into_list() invalid_entry() not_sensible()
changels()	alter_attributes()	insert_into_list() invalid_entry() not_sensible()
changelw()	alter_attributes()	insert_into_list() invalid_entry() not_sensible()
changetexta()	alter_attributes	insert_into_list() invalid_entry() not_sensible()
check_1st_not_default()	clean_list()	[none]
check_arc()	find_figure()	[none]
check_circle()	find_figure()	[none]
check_correct()	find_figure()	draw_erased() get_queue() instructions() redraw_figures() set_orthoview()
check_exit()	main_menu()()	instructions() newcursor() popup() showpopup()
check_line()	find_figure()	[none]
check_text()	find_figure()	[none]
circle2list()	file2list()	insert_into_list() map_to_list()
circle_to_file()	list2file()	map()
clean_font()	clean_list()	[none]
clean_linestyle()	clean_list()	[none]
clean_linestyle()	clean_list()	[none]
clean_list()	list2file()	check_1st_lw() check_font() check_1st_not_default() check_linestyle() check_linewidth() check_texture()
clean_font()	clean_list()	[none]
clean_texture()	clean_list()	[none]

Function	Called From	Calls To
cleanup()	getlist() list2file() main()	{none}
clearpage()	main_menu()	instructions() newcursor() redraw_figures() popup() showpopup() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture() set_filename() set_orthoview() set_real_linewidth() set_screen()
copyblock()	blockedit()	display_position() draw_copy() duplicate() get_bottom() get_left() get_queue() instructions() set_orthoview() set_written() update_bounds()
copy_obj()	copy_last() edit_obj()	get_alignment() get_bottom() get_left() get_queue() instructions() set_orthoview() update_bounds
dirdraw()	dirscroll() get_directory()	none
dirscroll()	get_directory()	{none}

Function	Called From	Calls To
display_position()	block_edit() block- edit() bmove_file() copy_obj() copy- block() draw_a() draw_arc() draw_arrow.c() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_smooth_line() draw_text() edit_menu() edit_obj() fig_menu() get_block() get_circle() line_menu() main_menu() move_picked_objs() not_found() replace_hdgl_w_hdg2() set_screen()	[none]
distance_between()	draw_circle() get_circle()	[none]
do_polygon()	polygon2list()	insert_into_list() map_to_list()
do_rectangle()	polygon2list()	insert_into_list() map_to_list()

Function	Called From	Calls To
draw_a()	fig_menu() line_menu() main_menu()	display_position() draw_arc() draw_arrow() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_smooth_line() draw_text() get_block() get_bottom() get_circle() get_left() get_queue() insert_diamond() insert_ellipse() insert_seed() insert_triangle() instruc- tions() newcursor() remove_last() redraw_figures() reproduce_last() set_orthoview() update_bounds()
draw_arc()	draw_a()	display_position() get_left() get_bottom() get_queue() insert_arc() instructions() redraw_figures() set_orthoview()

Function	Called From	Calls To
draw_arrow()	draw_a()	display_position() get_left() get_bottom() get_queue() insert_into_list() instructions() redraw_figures() set_current_linestyle() set_current_linewidth() set_current_texture() set_orthoview() set_real_linewidth() update_bounds()
draw_circle()	draw_a()	get_bottom() get_left() instructions() display_position() get_queue() set_orthoview() redraw_figures() update_bounds() insert_into_list()
draw_copy()	blockedit()	get_bottom() get_left() set_orthoview()
draw_erased()	check_correct() edit_obj()	[NONE]
draw_line()	draw_a()	get_bottom() get_left() get_queue() instructions() restrict_mouse() set_orthoview() redraw_figures() update_bounds()
draw_multi_line()	draw_a()	get_bottom() get_left() get_queue() instructions() redraw_figures() set_orthoview() update_bounds()

Function	Called From	Calls To
draw_page()	move_picked_objs()	get_bottom() get_left() set_orthoview()
draw_polygon()	draw_a()	get_bottom() get_left() get_queue() update_bounds() insert_into_list() instruc- tions() redraw_figures() set_orthoview()
draw_rectangle()	draw_a()	display_position() get_bottom() get_left() get_queue() instruc- tions() insert_into_list() redraw_figures() set_orthoview() update_bounds()
draw_smooth_line()	draw_a()	display_position() get_bottom() get_left() insert_into_list() instruc- tions() redraw_figures() set_orthoview() update_bounds()
draw_text()	draw_a()	display_position() get_bottom() get_current_font() get_current_linewidth() get_current_linestyle() get_left() get_queue() insert_hdgl() insert_into_list() instruc- tions() lw2pix() print- able() redraw_figures() remove_last() replace_hdgl_w_hdgl2() set_ortoview() update_bounds()

Function	Called From	Calls To
duplicate()	copyblock() copy_object()	full_memory() get_current_font() get_current_linestyle() get_current_linewidth() get_real_linewidth() get_current_texture() insert_into_list() modify_object() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture() set_real_linewidth()
edit_menu()	main_menu()	display_position() in- structions() popup() redraw_figures() showpopup() set_screen()
edit_obj()	edit_menu() fig_menu() main_menu()	alter_attributes() copy_obj() display_position() find_figure() get_bottom() get_current_linestyle() get_current_linewidth() get_current_texture() get_left() get_queue() in- structions() newcursor() redraw_figures() set_current_texture() set_current_linestyle() set_current_linewidth() set_orthoview() set_written() update_bounds()
ellipse2list()	file2list()	insert_ellipse() map_to_list()
ellipse_to_file()	list2file()	map()

Function	Called From	Calls To
fig_menu()	main_menu()	change() display_position() draw_a() edit_obj() in- structions() popup() redraw_figures() showpopup() set_screen()
file2list()	get_from_file()	arc2list() bad_data() circle2list() ellipse2list() font2list() get_current_font() get_current_linestyle() get_current_linewidth() get_current_texture() get_next() insert into list() line2list() linestyle2list() linewidth2list() polygon2list() seed2list() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture() syntax_error() text2list() texture2list()
find_figure()	edit_obj()	check_arc() check_circle() check_correct() check_line() check_text() instructions()
font2list()	file2list()	insert_into_list() set_current_font()
font_size()	binsert into_list() btext2list() changefont() draw_text() insert_into_list() modify_text() text2list()	index2ftable()
font_to_file()	list2file()	{none}

Function	Called From	Calls To
fontsource()	select_font()	instructions() newcursor() popup() redraw_figures() set_screer.() showpopup() up()
full_memory()	binsert_into_list() bline2list() bpoly2list() dupli- cate() insert_into_list() line2list() poly2list()	get_queue() redraw_figures() set_orthoview()
get_alignment()	bmove_file() copy- block() copy_obj() init_guides() move_obj() move_page() move_picked_objs() restrict_mouse() set_orthoview() set_mouse() set_screen() update_bounds() view_page()	get_queue()
get_angle()	draw_arc() draw_arrow()	[none]
get_block()	blockedit() draw_a()	display_position() get_bottom() get_current_linewidth() get_left() get_queue() in- structions() lw2pix() redraw_figures() set_orthoview() update_bounds()

Function	Called From	Calls To
get_bottom()	bdraw_page() block- edit() bmove_file() copy_obj() copy- block() draw_a() draw_arc() draw_arrow() draw_circle() draw_line() draw_page() draw_copy() draw_multi_line() draw_polygon() draw_rectangle() draw_smooth_line() draw_text() edit_obj() get_block() getcircle() move_picked_objs() move_obj() not_found() restrict_mouse() set_orthoview()	[none]
get_circle()	draw_a()	display_position() distance_between() get_bottom() get_left() get_queue() instruc- tions() redraw_figures() set_orthoview() update_bounds()
get_current_font()	bfile2list() binsert_into_list() btext2list() draw_text() dupli- cate() file2list() insert_into_list() select_font() set_screen() text2list()	[none]

Function	Called From	Calls To
get_current_linestyle()	bfile2list() binsert_into list() binsert_seed() btext2list() draw_arrow() draw_text() duplicate() file2list() insert_into list() insert_seed() modify_seed() set_screen() text2list()	[none]
get_current_linewidth()	bfile2list() binsert_into list() binsert_seed() btext2list() draw_arrow() draw_text() duplicate() edit_obj() file2list() getblock() insert_into list() insert_seed() set_screen() text2list()	[none]
get_current_texture()	bfile2list() binsert_into list() draw_arrow() duplicate() edit_obj() file2list() insert_into list() insert_seed() binsert_seed() set_screen()	[none]
get_directory()	read_menu() write_menu()	dirdraw() dirscroll() getlist() get_queue() instructions()

Function	Called From	Calls To
get_from_file()	read_menu()	drawing_exists() get_queue() file2list() instructions() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture() set_real_linewidth() set_filename() set_written()
get_left()	bdraw_page() block-edit() bmove_file() copy_obj() copy-block() draw_a() draw_arc() draw_arrow() draw_circle() draw_copy() draw_multi_line() draw_page() draw_polygon() draw_rectangle() draw_smooth_line() edit_obj() getblock() getcirlce() draw_line() move_obj() move_picked_objs() not_found() restrict_mouse() replace_hdg1_w_dhg2() set_orthoview()	[none]
get_next()	bfile2list() file2list()	[none]

Function	Called From	Calls To
get_queue()	bad_data() bget_from_file() block- edit() bmovefile() check_correct() copy_obj() copy- block() draw_a() draw_arc() draw_arrow() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_text() edit_obj() full_memory() get_user_alignment() get_directory() get_circle() get_from_file() get- block() go_to_file() in- valid() move_obj() move_page() move_picked_objs() not_found() select_font() select_lw() sensible() syntax_error() view_page()	[none]

Function	Called From	Calls To
get_real_linewidth()	binsert_into_list() draw_arrow() insert_into_list()	[none]
get_user_alignment()	main()	get_queue() inside() set_alignment()
go_to_file()	wite_menu()	get_queue() instruc- tions() list_to_file() set_written()
gt1pt_inside()	find_objects()	inside()
index2ftable()	alter_attributes() font_size() select_font() set_screen()	changefont() changetex- ta() changels() changelw() index2ftable() instruc- tions() popup() select_font() select_texture() select_ls() select_lw() set_screen() showpop- up()
init_attributes()	initialise()	[none]
init_guides()	main()	get_alignment() insert_into_list()
initpopup()	main()	[none]
insert_arc()	arc2list() barc2list() insert_arc()	insert_into_list()
insert_diamond()	draw_a()	insert_into_list()
insert_ellipse()	draw_a()	insert_into_list()
insert_hdgl()	draw_text()	get_bottom() get_left() insert_into_list()

Function	Called From	Calls To
insert_into_list()	change() changefont() changels() changelw() changetexta() draw_a() draw_arrow() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_smoothline() draw_text() dupli- cate() file2list() font_to_file() init_guides() insert_arc() insert_diamond() insert_ellipse() insert_seed() insert_triangle() line_to_file() ls_to_file() lw_to_file() poly_to_file() text_to_file() texture_to_file()	full_memory() font_size() get_current_font() get_current_linestyle() get_current_linewidth() get_real_linewidth() get_current_texture() set_written()
insert_seed()	seed2file()	get_current_linestyle() get_current_linewidth() get_current_texture() insert_into_list()
insert_triangle()	draw_a()	insert_into_list()
inside()	get_user_alignment() gtlpt_inside() one_pt_inside() select_text()	[none]

Function	Called From	Calls To
instructions()	alter_attributes() bget_from_file() block_edit() block- edit() bmove_file() check_exit() clear- page() copyblock() copy_obj() check_correct() draw_a() draw_arc() draw_arrow() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_smooth_line() draw_text() edit_menu() edit_obj() fig_menu() find_figure() fontsource() get_from_file() go_to_file() get_directory() get- block() get_circle() in- valid() line_menu() main_menu() move_obj() move_page() move_picked_objects() not_found() read_menu() select_font() select_ls() select_lw() select_text() sensible() set_screen() view_page() write_menu()	[none]
invalid_entry()	changels() changelw() changetexta()	get_gueue() instruc- tions() redraw_figures() set_orthoview()

Function	Called From	Calls To
line2list()	file2list()	full_memory() insert_into_list() map_to_list()
line_menu()	main_menu()	change() display_position() draw_a() instructions() popup() redraw_figures() showpopup() set_screen()
line_size()	change() duplicate()	[none]
line_to_file()	list2file()	map()
linestyle2list()	file2list()	insert_into_list() set_current_linestyle()
linestyle_to_file()	list_to_file()	[none]
linewidth2list()	file2list()	insert_into_list() lw2pix() set_current_linewidth() width2lwnum()
linewidth_to_file()	list_to_file()	[none]
list_to_file()	go_to_file()	arc_to_file() circle_to_file() cleanup() clean_list() ellipse_to_file() font_to_file() line_to_file() linestyle_to_file() linewidth_to_file() poly_to_file() seed_to_file() text_to_file() texture_to_file()

Function	Called From	Calls To
lw2pix()	binsert_into_list() binsert_seed() blinewidth2list() btext2list() change() changefont() changelw() draw_arrow() draw_text() dupli- cate() getblock() insert_into_list() insert_seed() linewidth2list() modify_seed() modify_text() move_obj() text2list()	[none]
main()	Activated from the illustrator system call NPSDRAW	cleanup() get_user_alignment() init_guides() initialise() initpopup main_menu() opening_display() read- fonttable()
main_menu()	main()	block_edit() change() check_exit() clearpage() display_position() draw_a() edit_obj() edit_menu() fig_menu() instructions() line_menu() popup() read_menu() redraw_figures() set_screen() set_orthoview() showpopup() toggle_grid() view_page() write_menu()

Function	Called From	Calls To
map()	arc_to_file() circle_to_file() ellipse_to_file() line_to_file() poly_to_file() seed2file() text_to_file()	
map_to_list()	arc2list() barc2list() bcircle2list() bellipse2list() bline2list() bpolygon2list() bseed2list() btext2list() circle2list() ellipse2list() line2list() polygon2list() seed2list() text2list()	[none]
modify_arc()	modify_object()	[none]
modify_circle()	modify_object()	[none]
modify_ellipse()	modify_object()	[none]
modify_line()	modify_object()	[none]
modify_object()	bmove_file() dupli- cate() move_page() move_picked_objs() move_obj()	modify_arc() modify_circle() modify_ellipse() modify_line() modify_seed() modify_text() modify_polygon()
modify_polygon()	modify_object()	none
modify_seed()	modify_object()	get_current_linestyle()
modify_text()	modify_object()	[none]
move_obj()	edit_obj()	get_left() get_bottom() get_queue() instruc- tions() modify_object() set_orthoview() update_bounds()
move_page()	edit_menu()	get_alignment() get_queue() instruc- tions() modify_object() set_written()

Function	Called From	Calls To
move_picked_objs()	blockedit()	display_position() draw_page() get_alignment() get_bottom() get_left() get_queue() instructions() modify_object() set_orthoview() set_written() update_bounds()
newcursor()	bblockedit() blockedit() check_exit() clearpage() draw_a() edit_obj() fontsource() opening_display() popup() read_menu() select_font() select_ls() select_lw() select_text() showpopup() write_menu()	[none]
not_found()	blockedit() find_figure()	display_position() get_bottom() get_left() get_queue() instructions() set_orthoview()
not_sensible()	changefont() changels() changelw()	get_queue() instructions() redraw_figures() set_orthoview()
one_pt_inside()	find_objects()	inside()
opening_display()	main()	newcursor()
poly_to_file()	list_to_file()	map()

Function	Called From	Calls To
popup()	alter_attributes() check_exit() clear- page() edit_menu() fig_menu() fontsource() line_menu() main_menu() read_menu() select_font() select_ls() select_lw() write_menu()	[none]
printable()	draw_text()	[none]
read_menu()	main_menu()	get_directory() get_from_file() instruc- tions() newcursor() pop- up() redraw_figures() set_orthoview() set_screen() showpop- up()
readfontable()	main()	[none]

Function	Called From	Calls To
redraw_figures()	bad_data() blockedit() check_correct() clear- page() draw_a() draw_arc() draw_arrow() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_smooth_line() draw_text() edit_menu() edit_obj() fig_menu() fontsource() full_memory() get- block() get_circle() in- valid() line_menu() main_menu() read_menu() select_font() sensible() set_screen() view_age() write_menu()	[none]
remove_last()	draw_a() draw_text()	[none]
replace_hdg1_whdg2()	draw_text()	get_bottom() get_left() insert_into_list() remove_last()
reproduce_last()	draw_a()	[none]
reset_mouse()	draw_line() select_lw()	get_alignment()
restrict_mouse()	draw_line()	get_alignment() get_left() get_bottom()
seed_to_file()	list_to_file()	map()
seed2list()	file2list()	insert_seed() map_to_list()
select_text()	alter_attributes() change()	inside() instructions() newcursor()

Function	Called From	Calls To
select_font()	alter_attributes() change()	fontsource() get_current_font() get_queue() index2ftable() instructions() newcursor() popup() redraw_figures() set_screen() showpopup()
select_ls()	alter_attributes() change()	instructions() newcursor() popup() showpopup()
select_lw()	alter_attributes() change()	get_queue() instructions() newcursor() popup() showpopup() varlw()
set_alignment()	get_user_alignment()	[none]
set_bottom()	set_screen()	[none]
set_current_font()	bfile2list() bfont2list() change() clearpage() duplicate() file2list() font2list() get_from_file()	set_screen()
set_current_linestyle()	bfile2list() bget_from_file() blinestyle2list() change() clearpage() draw_arrow() duplicate() edit_obj() file2list() get_from_file() linestyle2list()	set_screen()
set_current_linewidth()	draw_arrow() bfile2list() bget_from_file() blinewidth2list() change() clearpage() duplicate() edit_obj() file2list() get_from_file() linewidth2list()	set_screen()

Function	Called From	Calls To
set_current_texture()	bfile2list() bget_from_file() change() clearpage() draw_arrow() dupli- cate() edit_obj() file2list() get_from_file() texture2list() texture2list()	set_screen()
set_filename()	clearpage() get_from_file()	[none]
set_left()	set_screen()	[none]
set_orthoview()	bad_data() bdraw_page() block- edit() bmove_file() clearpage() check_correct() copy_obj() copy- block() draw_arc() draw_arrow() draw_copy() draw_page() draw_a() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_smooth_line() draw_text() edit_obj() full_memory() get_bottom() get_circle() get_left() getblock() insert_diamond() in- valid() main_menu() move_obj() move_picked_objs() not_found() read_menu() sensible() set_screen() syntax_error()	get_alignment() get_bottom() get_left() get_queue()

Function	Called From	Calls To
set_real_linewidth()	bfile2list() bget_from_file() change() clearpage() draw_arrow() dupli- cate() file2list() get_from_file()	[none]
set_screen()	alter_attributes() clearpage() copy_last() edit_menu() fig_menu() fontsource() line_menu() main_menu() read_menu() set_current_font() set_current_linestyle() set_current_linewidth() set_current_texture() select_font() view_page() write_menu()	display_position() get_alignment() get_current_texture() get_current_linestyle() get_current_linewidth() get_current_font() index2ftable() instruc- tions() redraw_figures() set_bottom() set_left() set_orthoview()
showpopup()	alter_attributes() clearpage() check_exit() edit_menu() fig_menu() fontsource() line_menu() main_menu() read_menu() select_font() select_ls() select_lw() write_menu()	newcursor()
syntax_error()	bfile2list() file2list()	get_queue() redraw_figures() set_orthoview()
text2list()	file2list()	font_size() get_current_font() get_current_linestyle() get_currenet_linewidth() insert_into_list() map_to_list()

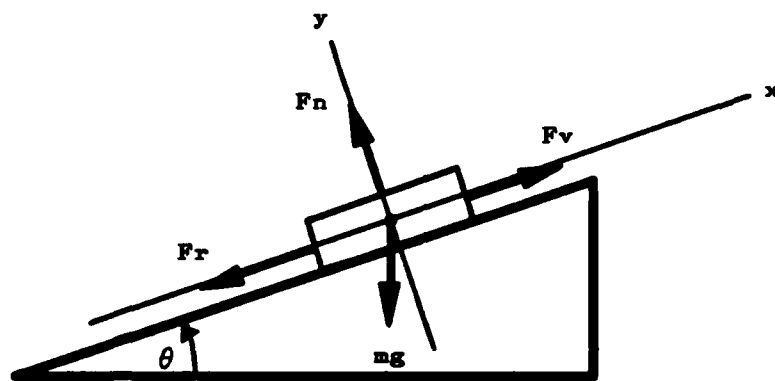
Function	Called From	Calls To
text_to_file()	list2file()	map()
texture2list()	list2file()	insert_into_list() set_current_texture()
texture_to_file()	list2file()	[none]
toggle_grid()	main_menu()	[none]
update_bounds()	blockedit() bmove_file() copy- block() copy_obj() draw_a() draw_arrow() draw_circle() draw_line() draw_multi_line() draw_polygon() draw_rectangle() draw_smooth_line() draw_text() edit_obj() get_block() get_circle() move_obj() move_picked_objs()	get_alignment()
view_page()	main_menu()	get_alignment() get_queue() instruc- tions() redraw_figures() set_screen()
width2lwnum()	blinewidth2list() linewidth2list()	[none]
write_menu()	main_menu()	get_directory() go_to file() instruc- tions() newcursor() pop- up() redraw_figures() showpopup() set_screen()

APPENDIX C

PRODUCTS OF NPSDRAW

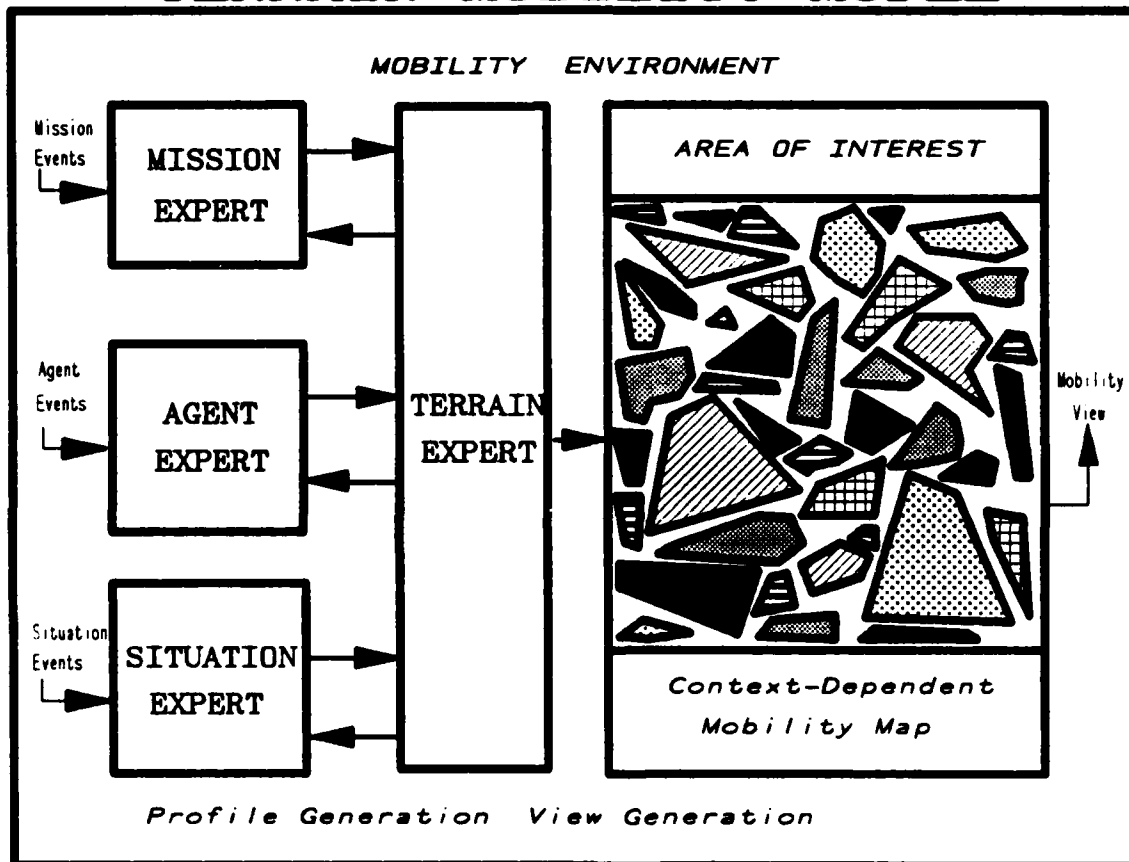
The following pages are examples of pictures created on NPSDRAW.

FREE BODY DIAGRAM
(Agent on Surface)

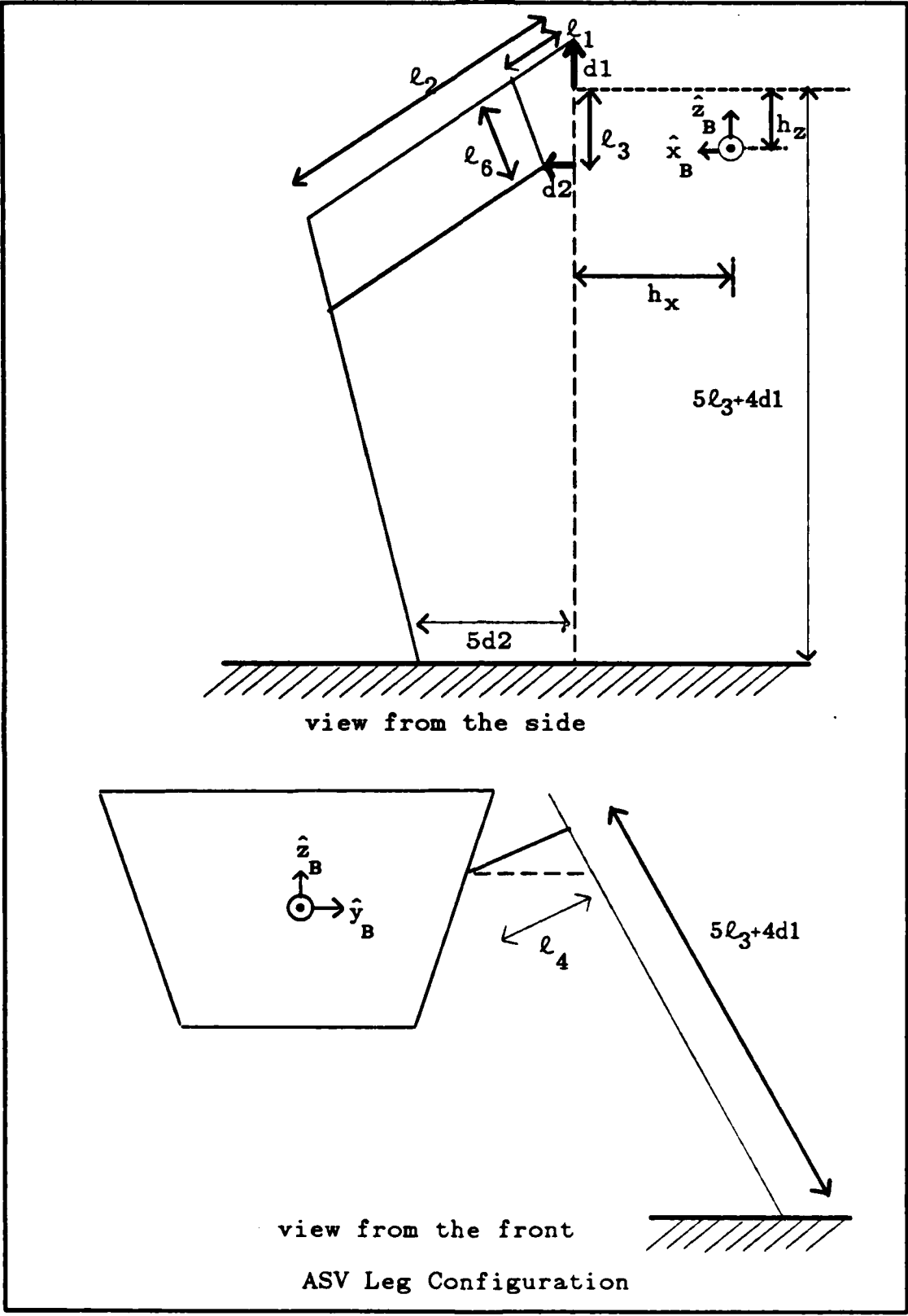


Example 1

TERRAIN MOBILITY MODEL

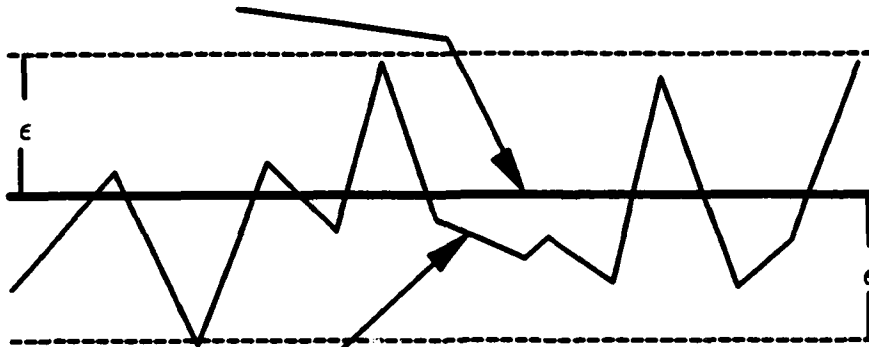


Example 2



Example 3

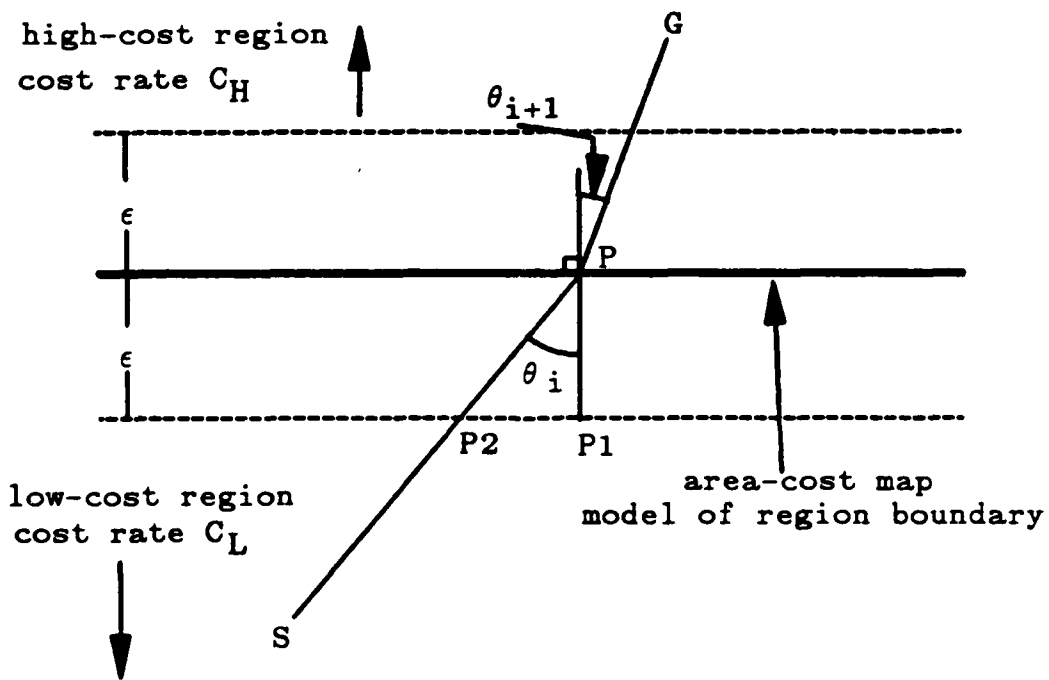
area-cost map model of region boundary



Real-world region boundary

(a). Real-World and Model Region Boundaries

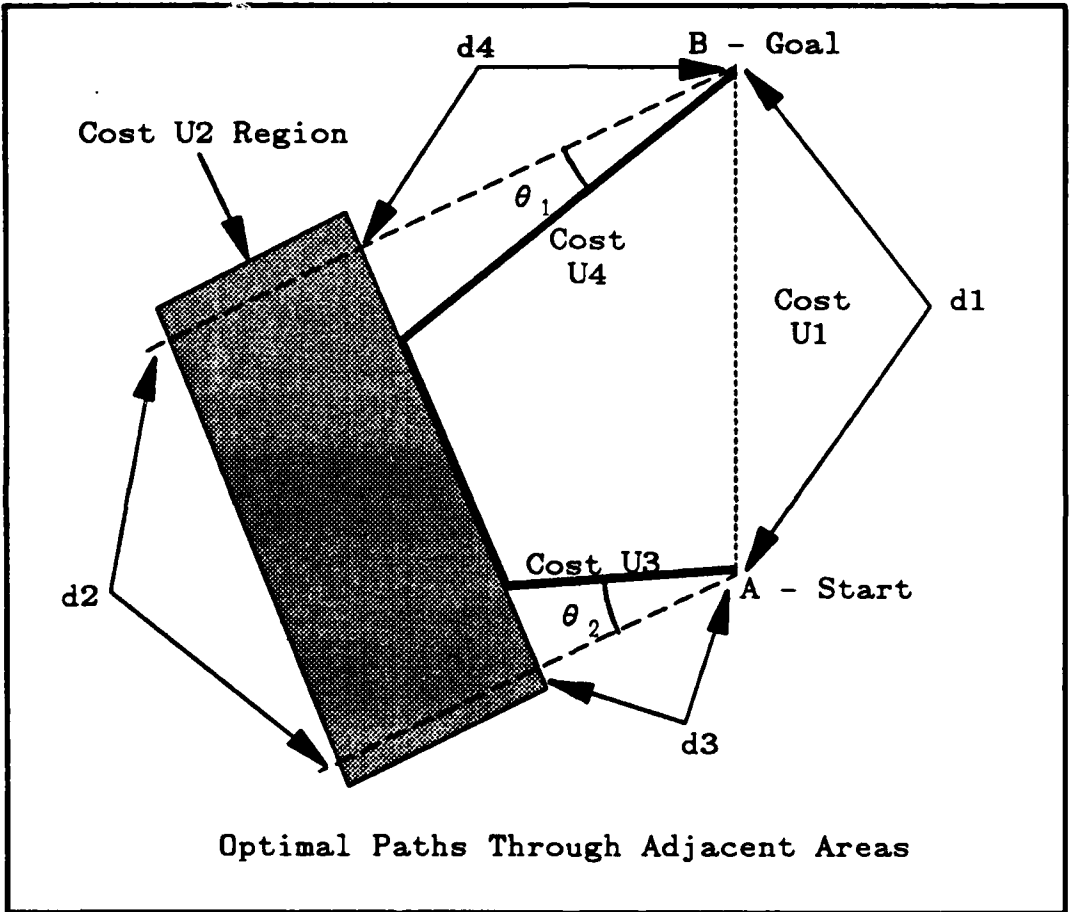
high-cost region
cost rate C_H



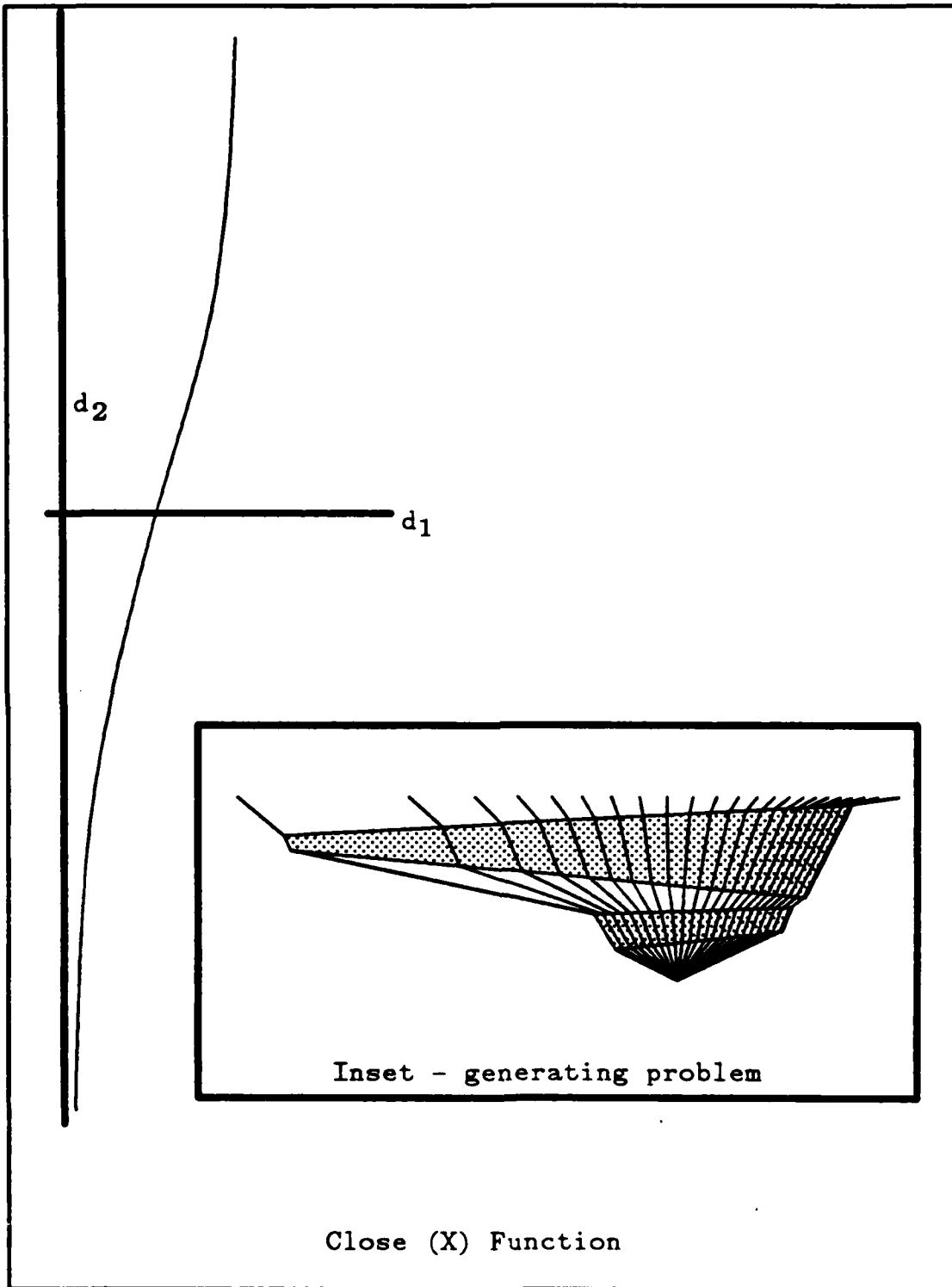
(b). Error Analysis Illustration

Path-Cost Error Due To Misplaced Boundaries

Example 4



Example 5



Example 6

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2. Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3. Dr. Michael J. Zyda, Code 52Zk Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	2
4. Chief of Naval Operations Director, Information Systems (OP-945) Navy Department Washington, DC 20350-2000	1
5. Lieutenant Commander Thomas J. Beda Tactical Electronic Warfare Squadron Thirty Three NAS Key West, Florida 33040	1
6. Chairman (Code 52) Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	1
7. Computer Technology Curricular Officer (Code 37) Naval Postgraduate School Monterey, California 93943-5000	1

END

9-87

Dtic