1849

HI-RES GRAPHICS FOR THE APPLE® II / IIe / IIc

ROBERT L. TICE &
DOROTHY JO STEVENS

HI-RES GRAPHICS FOR THE APPLE® II / IIe / IIc

ROBERT L. TICE & DOROTHY JO STEVENS

TAB TAB BOOKS Inc.
Blue Ridge Summit, PA 17214

FIRST EDITION

FIRST PRINTING

Library of Congress Cataloging in Publication Data

Tice, Robert L.
Hi-res graphics for the Apple II///e///c.

Includes index.
1. Computer graphics. 2. Apple II (Computer)—Programming. 3. Apple //e (Computer)—Programming. 4. Apple //c.(Computer)—Programming. I. Stevens, Dorothy Jo. II. Title.
T385.T52 1985 001.64'43 84-23954
ISBN 0-8306-0849-4
ISBN 0-8306-1849-X (pbk).

Cover concept by Ken Jensen, Design Center, University of Nebraska-Lincoln.

Preface

This book uses a step-by-step approach to explain the fundamentals and advanced techniques of graphics. The sequencing of topics and the programs selected to demonstrate those concepts are based on experiences in commercial software design and from teaching many students ranging from junior and senior high school students to adults. Because students were extremely receptive to materials and examples used in Dr. Stevens' computing courses and respected the commercial success of Mr. Tice, we made the decision to combine these two levels of expertise in one book–this book.

BASIC has become a common language that has evolved, over a number of years, from a language intended mainly for student use to a relatively sophisticated language often used for large-scale development projects. In addition, many educational institutions and agencies, large numbers of commercial software development concerns, and personal computer owners use Apple computers. Thus we decided to focus on BASIC and the programming commands used with Apple computers. It seemed logical to concentrate on one brand of computer—and do it well—rather than weaken and interrupt the flow of content by attempting to include the graphics commands of several computers. Once the concepts are learned, however, then it is very easy to transfer those concepts to other brands of computers.

Topics are presented to teach graphics as a functional tool for software development—whether the user is a junior or senior high student, teacher, hobbyist, or commercial programmer desiring to jazz up a program with attractive displays, motion, and sound. Concepts presented are appropriate for courses, workshops, seminars, enrichment topics for programmers, supplementary material to computer courses, and computer camps, as well as for independent study. Whether there is enough content for a one- or two-semester course depends on the programming expertise of the participants and the objectives of the users.

Introduction

Although this book uses a step-by-step format, students are able to learn rapidly because the examples are specifically designed to bridge directly with the concepts being presented. Furthermore, the examples used can be related to various interests and often can be integrated, as is, into a variety of programs. This is true whether the concepts presented are simple BASIC commands or complex high-resolution graphics pictures using shape tables with motion and sound effects. We made a deliberate effort to blend examples, program descriptions, and utility programs to allow you to assimilate concepts commensurate with your level of expertise and your reasons for wanting to learn graphics. If you are just beginning to learn about graphics, you will find the detailed explanations of initial BASIC commands very helpful; if you are an experienced programmer, you may also benefit from this review of BASIC commands. Not only do you learn how to use advanced, efficient graphics techniques, but your initial knowledge of BASIC commands is also enhanced.

CONTENT ORGANIZATION

The first chapter is essentially an introduction with recommendations on how to use the book. The book is organized so that you may begin working with the computer as soon as possible. Chapter 2 contains material that involves you immediately. Low-resolution graphics are presented, sounds (such as buzzes and clicks) that can be controlled from BASIC are introduced, and all of the concepts are then incorporated into the Pink Out game at the conclusion of the chapter. Information on high-resolution graphics is presented in Chapter 3. Initial instruction begins with plotting points and moves quickly to drawing lines, figures, solid polygons, complex polar designs, and graphing equations. The highlight of this chapter is a utility mathematics program that provides all of the mathematics necessary for using some very mathematically complex programs. Consequently, little or no knowledge of mathematics is required. The chapter concludes with all of the concepts integrated into a Winter Word Game program.

Chapter 4 concentrates on the design and placement of shapes and how to save shapes on a disk. You will gain a depth of understanding in coding, translating, and saving shapes on disks.

Shape tables are presented in Chapter 5. Topics on creating motion, generating character sets, and mixing text and high-resolution graphics are explained and demonstrated. Numerous demonstrations reinforce the concepts of scaling, rotation, and color. The chapter concludes with a snappy arcade game.

Music and sounds are presented in Chapter 7. A sound effects utility program is very useful to programmers is presented. For example, Song Editor allows complete songs to be played; all you have to do is to type in the notes. The unique part is that Song Editor allows notes to be held longer than conventional music programs. Gaming sounds such as a laser sound are explained and demonstrated. Your imagination can really take over at this point. Concepts presented in this chapter allow for a multitude of sounds to be generated, ranging from the sound of a starting car to the sounds of an intergalactic war.

Chapter 8 combines concepts. Our intent here was not to present ideas as disjoint sets, but to encourage interesting combinations of ideas with practical or entertaining results. An excellent example is provided with the Fraction Action educational game. This program integrates high-resolution graphics, vectors, motion, music, and sounds into a clever and worthwhile, yet entertaining, educational learning experience. When this program was tested with groups of junior high students the results were amazing. In this chapter page flipping is used to create motion. You are urged to transfer these concepts to your other areas of interests. For example, the circles in this Fraction Acton program would be ideal for pie charts in a business program.

Chapter 9 is a short but interesting chapter with a different sort of ending. Programs are selected to compare the effects of using shapes from BASIC using page flipping and without page flipping to those written in machine language using page flipping and without page flipping. The primary objective is to make you aware of the advantages of learning about machine language. Many persons believed machine language to be difficult; this brief exposure may motivate you to explore and learn about raster graphics or bit mapping techniques.

While each chapter is independent of the others, you are encouraged to read the chapters sequentially. Beneficial tips and ideas are presented throughout the book. For example, paddles are presented only in Chapter 2; if that chapter is skipped then the paddle concept is missed.

Each chapter contains a description of common programming errors that may occur, as well as hints for debugging. Wherever appropriate, concepts that require conversion tables or similar types of references, appendices have been provided.

OTHER POPULAR TAB BOOKS OF INTEREST

The Computer Era—1985 Calendar Robotics and Artificial Intelligence (No. 8031—$6.95)
Programming with dBASE II® (No. 1776—$16.50 paper; $26.95 hard)
Lotus 1-2-3™ Simplified (No. 1748—$10.25 paper; $16.95 hard)
Mastering Multiplan™ (No. 1743—$11.50 paper; $16.95 hard)
Apple Logo for Kids (No. 1728—$11.50 paper; $16.95 hard)
Getting the Most from Your Pocket Computer (No. 1723—$9.95 paper; $14.95 hard)
Scuttle the Computer Pirates: Software Protection Schemes (No. 1718—$15.50 paper; $21.95 hard)
Computer Programs for the Kitchen (No. 1707—$13.50 paper; $18.95 hard)
Beginner's Guide to Microprocessors—2nd Edition (No. 1695—$9.95 paper; $14.95 hard)
Apple® Lisa™: A User-Friendly Handbook (No. 1691—[illegible]5 paper; $24.95 hard)
[illegible]reate Your Own Computer Bulletin Board (No. [illegible]—$12.95 paper; $19.95 hard)
Does Your Small Business Need a Computer? (No. 1624—$10.95 paper; $18.95 hard)
Forecasting On Your Microcomputer (No. 1607—$15.50 paper; $21.95 hard)
BASIC Computer Simulation (No. 1585—$15.50 paper; $21.95 hard)
Solving Math Problems in BASIC (No. 1564—$15.50 paper; $19.95 hard)
Learning Simulation Techniques on a Microcomputer Playing Blackjack and Other Monte Carlo Games (No 1535—$10.95 paper; $16.95 hard)
Basic BASIC-English Dictionary for the Apple™, PET®, and TRS-80™ (No. 1521—$17.95 hard)
Making Money with Your Microcomputer (No. 1506—$8.25 paper; $13.95 hard)
Investment Analysis with Your Microcomputer (No. 1479—$13.50 paper; $19.95 hard)
The Art of Computer Programming (No. 1455—$10.95 paper; $16.95 hard)
Using and Programming the Macintosh™, including 32 Ready-To-Run Programs (No. 1840—$12.50 paper; $16.95 hard)
Making CP/M-80® Work for You (No. 1764—$9.25 paper; $16.95 hard)
Going On-Line with Your Micro (No. 1746—$12.50 paper; $17.95 hard)
The Master Handbook of High-Level Microcomputer Languages (No. 1733—$15.50 paper; $21.95 hard)
How to Document Your Software (No. 1724—$13.50 paper; $19.95 hard)
101 Programming Surprises & Tricks for Your Apple II®/II®e Computer (No. 1721—$11.50 paper)
Interfacing and Digital Experiments with Your Apple® (No. 1717—$15.50 paper; $21.95 hard)
MicroProgrammer's Market 1984 (No. 1700—$13.50 paper; $18.95 hard)
PayCalc: How to Create Customized Payroll Spreadsheets (No. 1694—$15.50 paper; $19.95 hard)
The First Primer of Microcomputer Telecommunications (No. 1688—$10.25 paper; $14.95 hard)
33 Adult Computer Games in BASIC for the IBM PC®, Apple II®/II®e and TRS-80™ (No. 1627—$13.50 paper; $18.95 hard)
Microcomputers for Lawyers (No. 1614—$14.50 paper; $19.95 hard)
Computer Companion for the Apple II®/Apple II®e (No. 1603—$10.25 paper)
Database Manager in MICROSOFT® BASIC (No. 1567—$12.50 paper; $18.95 hard)
Troubleshooting and Repairing Personal Computers (No. 1539—$14.50 paper; $19.95 hard)
25 Graphics Programs in MICROSOFT® BASIC (No. 1533—$11.50 paper; $17.95 hard)
Apple II® BASIC (No. 1513—$13.50 paper; $19.95 hard)
The Handbook of Microprocessor Interfacing (No. 1501—$15.50 paper; $21.95 hard)
From BASIC to Pascal (No. 1466—$11.50 paper; $18.95 hard)
Computer Peripherals That You Can Build (No. 1449—$13.95 paper; $19.95 hard)

TAB TAB BOOKS Inc.

Blue Ridge Summit, Pa. 17214

Send for FREE TAB Catalog describing over 750 current titles in print.

Contents

1

Overview

The Apple II computer has excellent color graphics. There are three methods that can be used to create graphic displays: plot, vector, and raster (bit mapping). This book explains both the plot and vector graphics methods.

Ideas are presented in this chapter to help guide you in your quest to learn grapics programming techniques. We will discuss the following topics in Chapter 1.

1. Using this book.
2. Program presentations
3. Notation conventions.
4. Required equipment.

The format of this book is such that you can learn graphics gradually and at your own pace. Material presented becomes increasingly advanced as each chapter progresses. *Hi-Res Graphics for the Apple* II /II e /II c is designed to be useful to individuals with various levels of programming expertise. Naturally, some knowledge of BASIC is advantageous. The information is presented in such a way, however, that beginning programmers can learn to utilize most of the graphics routines demonstrated. More advanced programmers will be pleasantly surprised to find programming techniques illustrated that are coveted by most commercial programmers but are rarely shared with others.

You must take an active role in the learning process in order to master the techniques presented in this book. Computer programming cannot be learned by simply reading a book. The programs presented should be entered into the computer and executed. The monitor screen should be observed closely while the designs are displayed. If you find yourself asking, "Why did it do that?" referring to the text to seek the answers will help you gain the most from reading this book.

A conscientious effort was made to avoid the use of needless computer jargon whenever possible. However, many ideas are best conveyed with terminology uniquely associated with computers and computer programming. Brief explanations are given for words used in this book that have special meanings. Whenever detailed discussions are required to clarify these concepts, we refer you to the appropriate Appendix so that the flow of the chapter is not disrupted.

Every chapter should be read, regardless of your level of programming expertise. Skipping a chapter may cause you to miss explanations of programming techniques that are useful for all graphics modes. While critical information is repeated several times to assist you, some specific programming techniques are presented in only one chapter.

PROGRAM PRESENTATIONS

A diverse selection of programs is used to demonstrate the concepts presented in this book. The programs are designed to accommodate different interests of readers. Illustrated here are graphics programming techniques that can be used in the design of games, educational software, computer art, and many other types of programs.

Several approaches are employed to explain program statements. Line-by-line discussions of most of the programs are presented to help you understand every detail of the program. Pitfalls and errors commonly made during initial attempts to use graphics are discussed. To further meet the needs of beginning programmers, most BASIC statements are explained the first time they appear in a program. This form of explanation allows you to become comfortable with each programming skill presented before moving to the next concept.

Another technique used to help you understand graphics programming is to present the reasoning used in determining the functions that a module or segment of a program should perform. This presentation format allows you to examine the processes involved in planning and developing efficient programs.

Some of the programs listed in this book are designed to be used as utility programs. These programs are intended to reduce the amount of time and effort you most devote to developing interesting graphics displays. While instructions for operating and using the programs are provided, line-by-line discussions of the programs are not presented. The programs can be used effectively without investigating the programming techniques used, and those of you who decide to study the program listings will generally recognize that the techniques used in these programs are discussed in other sections of the book.

NOTATION CONVENTIONS

Certain conventions with respect to program discussions have been adopted and are used throughout the book. The names of the variables used in demonstration programs are generally based on one or two letters from the description of the variable's role. When this role is described, the letter or letters used in formulating the variable name are capitalized. For example, if the variable NP is to be used to store the number of players playing a game, then the description of the variable's role would appear as *Number of Players*.

In order to help you distinguish between variables used in a specific programming statement and those used to illustrate general forms of a BASIC command or mathematical formula, we decided to use uppercase letters for program variable names and to use lowercase letters to illustrate general-form variables.

REQUIRED EQUIPMENT

All the programs presented in this book were developed using Applesoft BASIC on a standard Apple II+ computer with 48K of RAM. Most of the applications discussed can be accomplished on a computer with a single disk drive. Either the Apple II+ or Apple *IIe* can be used for testing and studying the programs in *Hi-Res Graphics for the Apple*. A color monitor and paddles or joysticks are recommended, but are not required for most of the topics covered.

Although many printers have graphics printing capability, printer applications are not discussed in this book. Readers who want to use a printer to output graphics will find that adequate information is included in the documentation that accompanies most printers and interface cards having graphics output capabilities.

2

Low-Resolution Graphics

The graphics displays called *plot graphics* are created by instructing the computer to light or color particular points on the display screen. Apple computers allow users to create plot graphics displays with two sizes of points. The larger the points, the fewer that are available on the screen. The fewer the points available on the screen, the lower the *resolution*. The Apple's plot graphics are divided into two categories, *high-resolution graphics* and *low-resolution graphics*. Both graphics resolutions cannot be displayed at the same time on the screen, and a separate instruction set is needed for each. This chapter covers low-resolution graphics and topics that are relevant to low-resolution graphics programming:

1. Low-resolution graphics screen.
2. Low-resolution graphics instruction set.
3. Designing low-resolution graphics pictures.
4. Implementing the low-resolution graphics instruction set.
5. Random low-resolution graphics displays.
6. Integrating sound with low-resolution graphics.
7. Application of low-resolution graphics.

The Apple II computer has three screen display modes. Words and numbers are displayed in the TEXT mode. That's good, because words and numbers are the primary components of text. Two other modes are used for graphics displays, LOW-RESOLUTION mode and HIGH-RESOLUTION mode. Each display mode uses the screen in different ways. In order to understand how low-resolution graphics displays are created, it is necessary to understand the utilization of the screen in this display mode.

LOW-RESOLUTION GRAPHICS SCREEN

The low-resolution graphics display mode allows you to create colorful displays. There are 16 colors available using this type of plot graphics (see Table 2-1). Some sacrifices must be made in order to offer such an array of colors. Certain *pixels* (pictures elements) on the color monitor are capable of

Table 2-1. Low-Resolution Colors.

Code	Color	Code	Color
0	black	8	brown
1	magenta	9	orange
2	dark blue	10	grey
3	purple	11	pink
4	dark green	12	green
5	grey	13	yellow
6	medium blue	14	aqua
7	light blue	15	white

generating particular colors. In order to create many different colors, it is necessary to utilize adjacent pixels. The greater the choice of colors, the more pixels that are needed to create or generate the color, and the large number of pixels necessary to provide the low-resolution colors reduces the number of points available on the screen to 1600. Each low-resolution point is 7 pixels wide and 4 pixels high.

The low-resolution screen can display 40 points horizontally (columns) and 40 points vertically (rows). Each column of the screen is assigned a unique number from 0 through 39. The left-most column on the screen is column 0 and the right-most column is column 39. The rows are numbered from the top of the screen (row 0) to the bottom (row 39). Any point can be referenced by its column and row positions on the screen.

The number which corresponds to the column of the screen in which the point is located is called the *horizontal coordinate* of the point. The *vertical coordinate* of a point is the number corresponding to the screen row in which the point is located. Any point on the screen thus can be referenced by a pair of numbers. Since the numbers will always be stated in the order *horizontal coordinate, vertical coordinate,* it is called an *ordered pair.* The ordered pair which describes the location of any point on the screen are said to be the *coordinates* of that point (see Fig. 2-1).

The computer does not normally use the entire video screen for low-resolution graphics displays. Four lines at the bottom are reserved for displaying textual information. This allows programmers enough room to PRINT instructions to the user, INPUT user responses, or to describe the graphics display.

THE LOW-RESOLUTION INSTRUCTION SET

The Applesoft BASIC instructions or commands used for creating low-resolution displays are introduced and explained below.

GR. GR is the command that instructs the computer that it is to use the low-resolution graphics mode for displaying information on the screen. When the command is used, the computer erases anything that is displayed on the low-resolution graphics portion of the screen (the top 20 text lines). This command must be executed before low-resolution graphics points can be plotted.

COLOR. The color that is to be used for plotting a point on the screen must be specified prior to plotting the point the syntax for the COLOR command is

COLOR = {n}

where n is a number from 0 through 15 corresponding to the color desired. For example, COLOR=3 causes plotted points to be purple.

PLOT. Any point on the low-resolution screen can be plotted using the PLOT command. The command must be followed by the coordinates of the point to be plotted. The general form of the command is

PLOT {x,y}

where x and y are both numbers from 0 to 39. X represents the horizontal position of the point and y represents its vertical position. The statement PLOT 20,20 causes the center point of the low-resolution screen to be plotted using the most recently stated color.

HLIN. Whenever consecutive horizontal points are to be plotted, the HLIN command can be used. The syntax for using the Horizontal LINe command is

HLIN{x1,x2} AT{y}

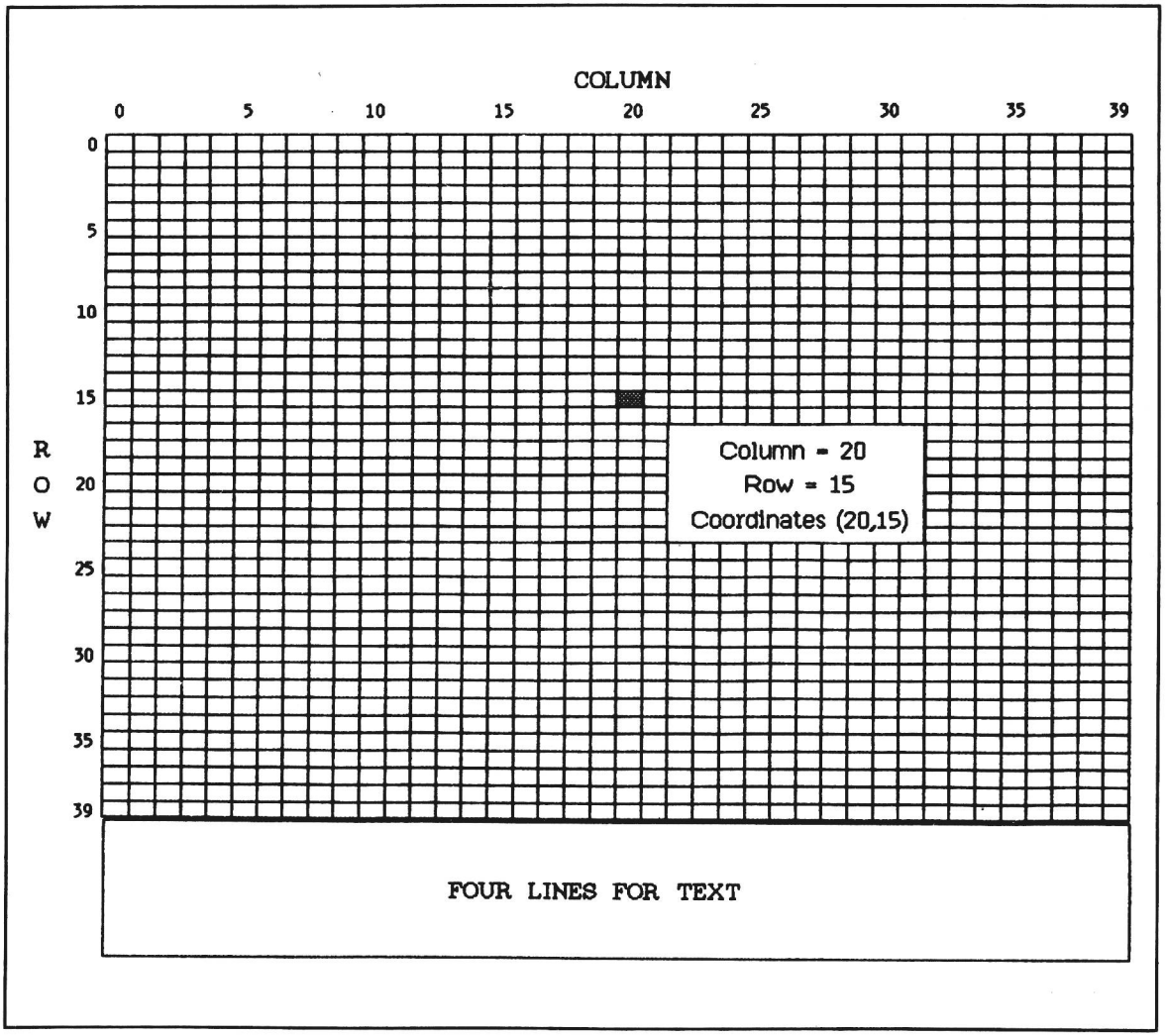

Fig. 2-1. The positions of points on the low-resolution graphics screen are identified by their column and row coordinates.

the term x1 specifies one horizontal endpoint of the line, x2 specifies the other endpoint, and y specifies the vertical coordinate of each point on the line. Legal values for x1, x2, and y are 0 to 39. In order to plot a horizontal line across the top of the screen, use the command HLIN 0,39 AT 0. A horizontal line across the left half of the screen on the bottom row is plotted using the command HLIN 0,20 AT 39.

VLIN. This command is similar to HLIN, except it is used to plot Vertical LINes. The general form for the VLIN command is

VLIN{y1,y2} AT{x}

where y1 and y2 are the endpoints of the vertical line and x is the horizontal coordinate for each point on the line. A vertical line extending from the middle of the screen to the top of the screen at the 10th horizontal coordinate is plotted when the following statement is executed: VLIN 20,0 AT 10.

TEXT. While TEXT is not a graphics command, it is included because it is needed to return to the text display mode after graphics have been used.

When TEXT is executed to terminate use of the low-resolution graphics screen, the computer places inverse video @ characters and other equally distracting images on the top 20 rows of the screen. The TEXT command should be followed immediately by the HOME command, which clears the screen.

There is not a great number of low-resolution commands, but interesting displays can be achieved when the above commands are used in conjunction with other BASIC statements.

DESIGNING LOW-RESOLUTION PICTURES

As is the case with many Applesoft BASIC commands, the low-resolution graphics commands can be used in *immediate mode.* This simply means that the computer can execute the commands immediately if they are entered without line numbers. The statements used in the previous section are presented here to demonstrate the use of graphics commands in immediate mode. After the computer is operating, enter the following statements one at a time, just as they appear below, and observe the display before entering the subsequent statement.

```
GR
COLOR = 3
PLOT 20,20
HLIN 0,39 AT 0
COLOR = 5
HLIN 0,20 AT 39
COLOR = 1
VLIN 20,0 AT 10
TEXT
HOME
```

These commands are generally used from within an Applesoft program. When line numbers are assigned to statements, they are not executed until the computer receives the RUN command. Since the computer defers execution of the statements until the RUN command is received, this is called *deferred mode.* The statements listed above can be executed in deferred mode by simply typing a line number prior to entering each command. The list of a program using the preceding commands appear in Listing 2-1.

Listing 2-1. Low-Res Commands.

```
5  REM   LOW RES COMMANDS
10  GR
20  COLOR= 3
30  PLOT 20,20
40  HLIN 0,39 AT 0
50  COLOR= 5
60  HLIN 0,20 AT 39
70  COLOR= 1
80  VLIN 20,0 AT 10
90  TEXT
100  HOME
```

The above program should be entered and run. Notice that the graphics are displayed in a flash, then removed from the screen. In order to leave the graphics displayed for a set amount of time, a time delay loop could be placed in the program. The Apple II can count to approximately 800 in a single second. Placing the following line in the program causes the graphics to be displayed for approximately three seconds.

```
85  FOR TIME = 1 TO 2400: NEXT TIME
```

Perhaps a better way of regulating the amount of time the graphics are left on the screen is by using the GET statement. GET causes the execution of the program to halt until a key is pressed. The command must be followed by a variable to which the character corresponding to the pressed key is assigned. A string variable should always be used with the GET command. If a real or integer variable is used instead, and the user presses a key other than a number key, a syntax error occurs. Insert the following line in the above program to allow the user to regulate the amount of time the graphic is displayed on the screen.

```
85  GET GO$
```

Careful planning plays a significant role in creating successful graphics displays. A very helpful planning aid is a piece of graph paper. Shading or coloring grids on paper is useful in determining which of the low-resolution graphics commands should be used to generate the picture. Figure 2-2 shows a picture of a boat shaded on graph paper. In order to simplify the planning process, the boat will be a solid color.

Once the picture has been drawn, the coordinates of the points must be determined. It is not necessary to find the coordinates of every point. Try to look for lines, either vertical or horizontal, in the figure. If a vertical or horizontal sequence of

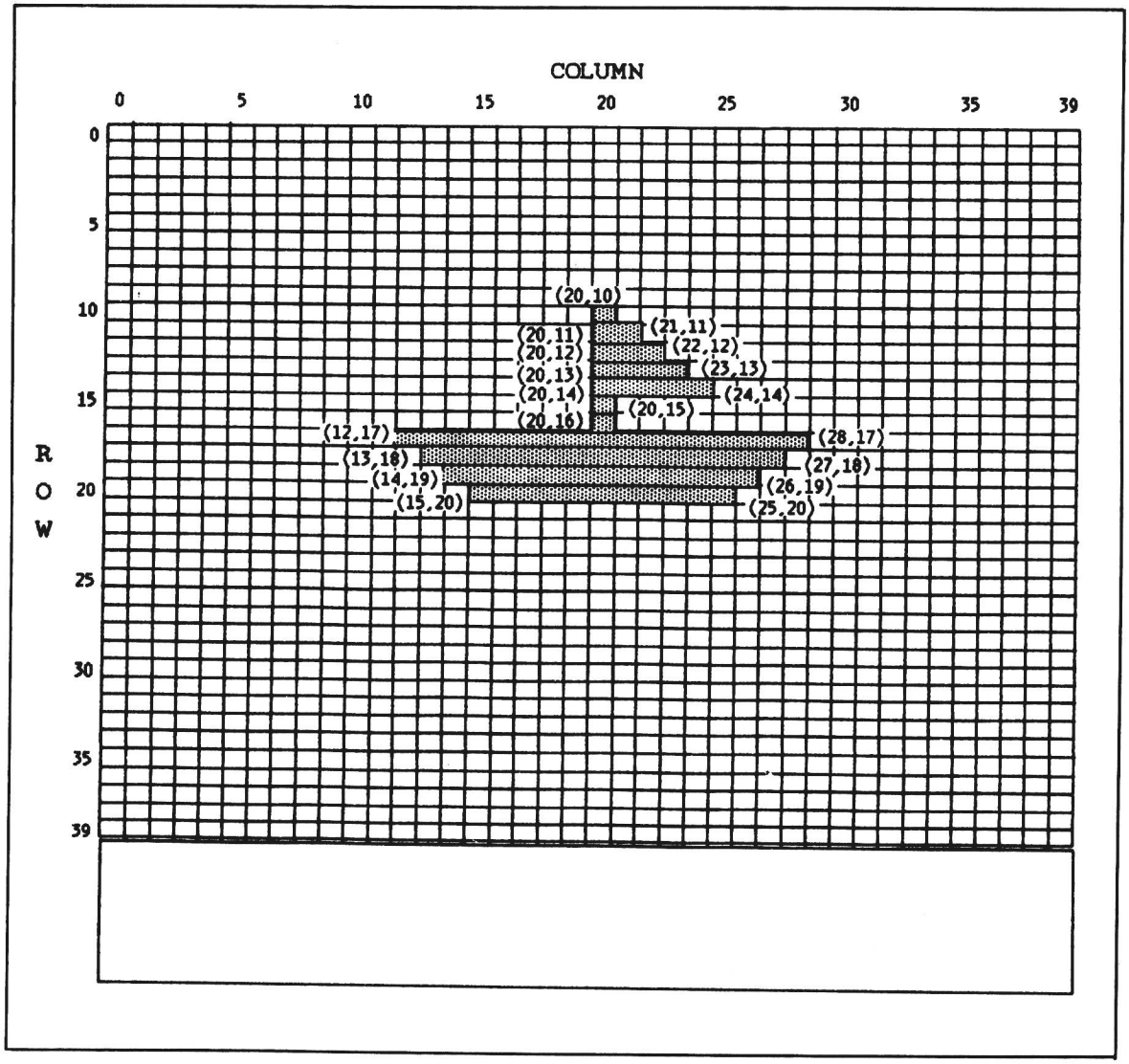

Fig. 2-2. The figure gives the coordinates of the endpoints of the horizontal lines and single points that were used to display the boat in the Purple Boat program.

Listing 2-2. Purple Boat.

```
5  REM     PURPLE BOAT
10  GR
20  COLOR= 3
500  HLIN 15,25 AT 20
510  HLIN 14,26 AT 19
520  HLIN 13,27 AT 18
530  HLIN 12,28 AT 17
540  PLOT 20,16
550  PLOT 20,15
560  HLIN 20,24 AT 14
570  HLIN 20,23 AT 13
580  HLIN 20,22 AT 12
590  HLIN 20,21 AT 11
600  PLOT 20,10
```

grids all require the same color, then only the coordinates of the endpoints need to be determined. Remember that lines can be plotted using the HLIN and VLIN commands. The coordinates of the points used to construct the boat are also presented in Fig. 2-2. Listing 2-2 is a program that displays the purple boat.

IMPLEMENTING THE INSTRUCTION SET

The above approach for programming a low-resolution graphics scene is straightforward and useful. However, it is often necessary to display a picture of an object in several areas of the screen. If a unique program segment were needed each time the figure was to appear in a different location on the screen, the work involved would discourage most programmers. In addition, programs using graphics would be very long. A more efficient programming technique is to use variables to describe the relative positions of the points needed to create the design.

This approach requires selecting one coordinate pair to be used to mark the figure's position on the screen. The point the coordinates describe is called the *reference point.* The reference point should be chosen carefully in order to minimize programming effort. Any point within the picture can serve as the reference point. The vertical coordinate of the lowest point on the boat (20) is used for the vertical coordinate in this example. While the horizontal coordinate most frequently used (also 20) is the horizontal coordinate of the reference point. Each of the selected coordinates should have a variable assigned to it. The variable X is used to represent the horizontal reference coordinate, and Y is used to represent the vertical reference coordinate. The ordered pair of coordinates for the reference point, (20,20), becomes (X,Y).

The next task is to describe the position of the remaining points in relation to the reference point. By inspecting the preceding program and Fig. 2-2, the coordinates of the points can be written in terms of X and Y. All the horizontal coordinates must be expressed in terms of X, and all the vertical coordinates must be expressed in terms of Y. For example, the statement HLIN 15,25 AT 20 must be changed to HLIN X−5,X+5 AT Y. Since X=20, X−5 is equivalent to 15 and X+5 equals 25. Y has a value of 20, so no addend is necessary.

A program that displays the boat using the reference point concept is listed as Listing 2-3.

Listing 2-3. Reference Point Boat.

```
5  REM   REFERENCE POINT BOAT
10  GR
20 X = 20
30 Y = 20
40  COLOR= 3
500  HLIN X - 5,X + 5 AT Y
510  HLIN X - 6,X + 6 AT Y - 1
520  HLIN X - 7,X + 7 AT Y - 2
530  HLIN X - 8,X + 8 AT Y - 3
540  PLOT X,Y - 4
550  PLOT X,Y - 5
560  HLIN X,X + 4 AT Y - 6
570  HLIN X,X + 3 AT Y - 7
580  HLIN X,X + 2 AT Y - 8
590  HLIN X,X + 1 AT Y - 9
600  PLOT X,Y - 10
```

Using this programming technique simplifies the programming task, particularly when more than one boat is to be displayed on the screen at a time. This is also true if one boat is to be positioned at different