TEACHING PROGRAMMING WITH SPIDER WORLD

John Dalbey
Lawrence Hall of Science
University of California, Berkeley

Presented at the Seventh Annual Western Educational Computing Conference, San Francisco, November 18, 1983.

Abstract

This paper describes the use of "robot" languages for teaching introductory programming. It is argued that robot languages are more appropriate than high level languages which are currently being used as first languages for learning programming. The advantages of robot languages over other high level languages are presented. An exciting new robot language simulation called Spider World is introduced, and the unique features which distinguish it from other languages are described.

1. INTRODUCTION

There are two major difficulties in teaching a first course in programming which the proper choice of a programming language may be able to alleviate. The first problem is that the complexity and detail of many languages obscures the essential concepts about the design of algorithms. The immense number of superficial syntax rules makes it difficult to identify the underlying notions of procedural reasoning. The second problem is one of providing a programming environment that is stimulating enough to arouse the student's interest in programming. The kinds of problems which can be solved after an initial introduction to most languages are unattractive and of little interest to the average person. Robot languages present a possible solution to these two problems.

2. WHAT ARE ROBOT LANGUAGES?

Robot languages are simple programming languages. Characteristically they are used to control a real or imaginary robot. They are intended to be easier to learn than other high level languages like BASIC or FORTRAN, primarily because their syntax is more like natural language. Turtle graphics, a feature of the LOGO language, is perhaps the most famous example of a robot language. (LOGO itself has powers far beyond turtle graphics.) Karel the Robot (1) and Robotwars (2) are other popular examples. Less well known robot languages are Josef (3) and Pacjana (4). This paper introduces a new member to this family, Spider World.

3. WHAT ARE THE ADVANTAGES OF ROBOT LANGUAGES?

Robot languages have many characteristics which make them appropriate for a beginning course in programming. The first is their simple syntax. Most robot languages do not require explicit declaration of variables or data types. Most of the robot languages do not have restrictions on punctuation such as line numbers or statement separators. For example, the Spider instructions are very simple one word commands. All the robot languages mentioned above are graphically oriented. Thus

they are intrinsically appealing to novices because they enable one to control a visual display. Languages like BASIC or FORTRAN manipulate symbols or perform numeric computations, neither of which has the same intuitive appeal as graphics. Another advantage of a graphics orientation is that it does not stress numeric computations. Hopefully by deemphasizing computation it will be appealing even to the "mathophobic."

In BASIC it is necessary to understand the concepts of input and output (and the associated commands INPUT and PRINT) before any meaningful programs can be written. Robot languages allow the learner to create programs which draw patterns without having to understand the idea of a "data stream" or other unfamiliar input and output concepts. This allows even beginners to solve problems which are intrinsically appealing.

A disadvantage of some languages is that the source programs must be compiled before they can be run. The programmer must explicitly request the computer to perform a compilation step which translates the statements into machine language. Only then can the program be executed. On the other hand, LOGO, the Spider language and Pacjana (as well as BASIC) are interpretive languages. Students using an interpretive language have fewer things to learn before they can actually run a program. Their programs can be run immediately without an intermediate compile phase. This simplifies the process of creating, running, and modifying programs.

Another feature of many robot languages is that they are "dataless". The language carries out many data processing functions implicitly so that they are invisible to the user. The novice programmer does not need to be concerned about reserving memory locations, declaring variables, assigning values to variables, obeying data type conventions, or formatting the input and output of data. At least in the early stages of instruction there is no explicit mention of concepts such as data, counter, and variable. Rather the emphasis is on procedures and control structures for looping and branching. Variables are not introduced until after the student has mastered the use of loops and conditional expressions. In this manner the critical components of writing procedures can be explored separately from the peripheral issues of data manipulation.

4. WHAT IS SPIDER WORLD?

Spider World is an interactive simulation which is intended to be an environment for learning the fundamentals of creating computer programs. It is intended to be used by people with no previous programming experience. Like turtle graphics, Spider World provides the user with tools for creating colored patterns on the computer display screen. The robot in this simulation is a programmable entity called the Spider. Spider World is much less sophisticated than turtle graphics, in that it offers a limited set of capabilities. However this simplicity is advantageous in that novices have fewer details to learn before they encounter the central ideas of programming. The ACCCEL research project at the Lawrence Hall of Science is currently attempting to demonstrate the effectiveness of Spider World as a first programming language for junior high school students.

One of the unique features of Spider World is that it has a mode of operation in

which the user can interactively guide the Spider to create designs on the screen. In "draw mode" the display shows a white square with black grid lines, which is the Spider's room (see Figure 1). The bottom four lines of the screen contain instructions. The Spider herself is displayed in the upper left corner as a solid black triangle. From this initial position the user can "pilot" the Spider, directing her to move around the room and paint squares certain colors. Each button pressed causes the Spider to respond with a specific action, and results in an immediate change in the image on the Draw screen. This beginning interactive mode of Spider World allows a novice to explore the primitive capabilities of the Spider in a very direct way. Draw mode allows the user to discover the effects of Spider commands without creating and executing a complete program.

An attractive characteristic of Spider World is that the Spider operates in a discrete world, not a continuous one like turtle graphics. The Spider functions in a plane which is divided into a small number of distinct cells which appear on the screen as a grid. The only movement available to the Spider is within this orthogonal framework. This is especially advantageous for children who don't yet understand geometry, because it is not necessary to specify a numeric value of an angle in order to turn the Spider.

One of the most powerful features of Spider World is that the Spider remembers each key that the user presses in draw mode. Then by pressing a special key ("redraw") the entire pattern can be recreated. Thus the simulation has created a stored program automatically, which the user can execute by simply pressing a key. This exemplifies the essential characteristic of programmable devices - the ability to autonomously carry out a stored sequence of instructions. In this way the fundamental power of the computer is made available to the novice almost effortlessly.

Figure 2 shows the display at a particular point during the "redraw" or "run" of a Spider program. The simulation allows the user to halt the execution, or proceed one step at a time. Included in the text at the bottom of the screen is the Spider instruction currently being executed.

The operation of the Spider in draw mode is based on a "one button per function" model which is common to most mechanical devices, such as household appliances, as well as contemporary video games. The earlier stages of interacting with the Spider are not unlike previous experiences most students have had in daily life. The novice does not need any prerequisite knowledge from mathematics or formal languages. Thus for even unsophisticated learners, Spider World serves as an accessible bridge to the "powerful ideas" contained in programming.

The Spider World simulation has a built-in editting facility. Simply pressing a key causes the commands in the Spider's memory to be printed as English text on the screen (see Figure 3). The user's program can be modified using certain keys to move the cursor on the screen and insert and delete lines of code. Additionally the simulation has the ability to load and save Spider programs the user creates on a floppy disk. Thus the user is quickly acquainted with the utilities that are commonly offered for developing and manipulating programs (see figure 4).

The Spider language has several programming constructs which are found in higher level languages. The DEFINE-END construct provides the ability to create subroutines. The REPEAT-UNTIL construct performs a looping function. The Spider also has a "counter" and a "memory register" which can be manipulated by the program. The introduction of these powerful programming structures is facilitated by the simplicity of the command language (see Figure 5).

In conclusion, robot languages are especially appropriate for teaching introductory programming concepts because they have a simple syntax, are graphic oriented, "dataless", and have no complex input/output procedures. In addition, Spider World has several characteristics which are particularly appealing. It is interpretive, contains a "draw" mode and a redraw feature, is presented in an orthogonal geometric context, has easily used utilities for program development, and incorporates several fundamental programming constructs.

5. TECHNICAL INFORMATION

Spider World runs on an Apple II-plus microcomputer with a single disk drive and 48K RAM. The simulation includes the interpreter, editor, file manager, and a demonstration mode. The source code is written in Applesoft BASIC. The Spider World program and a complete user manual are available from the author.

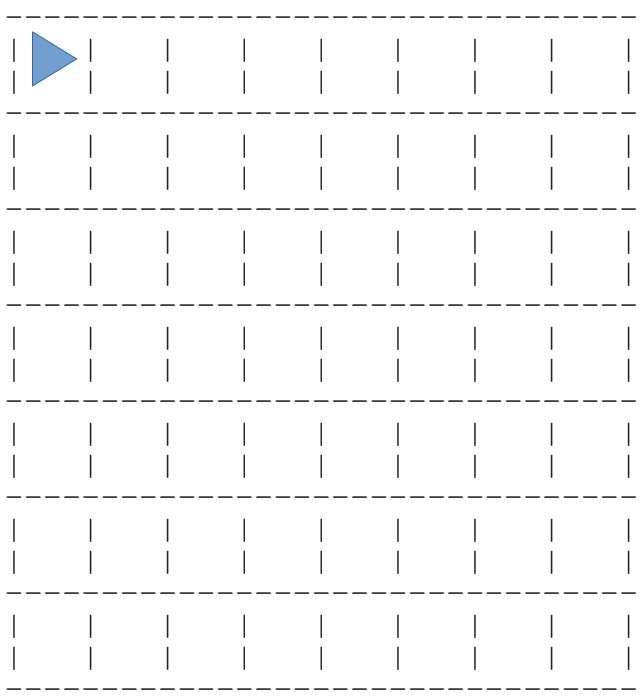
6. REFERENCES

1. Pattis, R. Karel the robot. New York: Wiley & Sons, 1981.

2. Warner, S. Robotwar. Baltimore, MD: Muse Software, 1981.

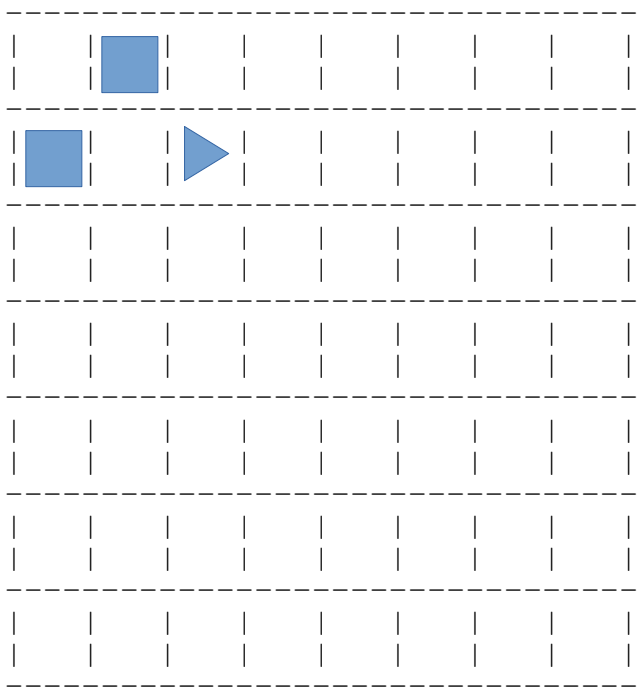
3. Tomek, I. Josef, programming for everybody. ACM SIGCSE Bulletin, 1982, 14(1), 188-192.

4. Finzer, W., and Resek, D. Computer languages and learning. Unpublished manuscript, San Francisco State University, 1983.



```
   TELL THE SPIDER WHAT TO DRAW
S=STEP, T-TURN, R=RED, B=BLUE, G=GREEN
P=PURPLE, W=WHITE, X=BLACK, !=REDRAW,
F=FORGET, <ESC>=LIST, Q=QUIT.
```

Draw Mode Screen Display
Figure 1



```
STEP
PRESS 'H' TO HALT, SPACE BAR TO RESUME,
<ESC> TO STOP THE SPIDER.
```

Run Mode Screen Display
Figure 2

```
: HERE IS A SAMPLE SPIDER PROGRAM
START
BLUE
STEP
RED
TURN
STEP
GREEN
QUIT

; = UP /=DOWN @=INSERT *=DELETE
OPTIONS:    LIST, RUN, DRAW, FORGET, SAVE, LOAD, BUILD, DEMO,
FILES,!,<ESC>
```

List Mode Screen Display
Figure 3

LIST: Displays the current spider program. You may make changes to the program by retyping lines on the screen. You may move the cursor to the desired line by pressing ";" for up and "/" for down. To insert a line press "@". To delete a line press "*".

RUN: Execute the current spider program.

DRAW: An interactive mode where you guide the Spider to create patterns. In draw mode the Spider is moved by pressing certain keys.

FORGET: Erase all the lines in the current program.

SAVE: Copy the current program to a disk file.

LOAD: Copy a file from the disk into the Spider memory. The old program is lost, and the program on the file becomes the current program.

BUILD: Append a file from the disk to the Spider's memory. The old program is not destroyed.

DEMO: Automatically run a demonstration of the spider in action. Demo executes a sequence of prewritten spider programs.

FILES: Displays the names of spider programs which have been saved on the disk.

Spider World Simulation Options
Figure 4

Action commands

START: Draw the wall and prepare to begin your journey.
STEP: Take one step in the direction you are facing.
TURN: Turn to your right.
BLUE: Paint the square you are standing on blue. (The Spider also knows RED, GREEN, PURPLE, WHITE, and BLACK.)
ADD: Add one to your counter.
SUBTRACT: Subtract one from your counter.
STORE: Place the current value of the counter in memory register.
RECALL: Replace the value of counter with the value from memory register.
QUIT: Stop, your journey is over.

Control commands

DEFINE task-name / END

Allows a sequence of statements to be referred to by a single name. When task-name is used in the program, all the statements defined in the task will be carried out. (A task must be defined before it can be invoked.)

REPEAT

Tells the spider to keep doing the instructions that are written between the "repeat" and "until" statements until the indicated condition is met.

UNTIL [WALL/ZERO/color]

WALL means until the wall is reached.
ZERO means until the counter equals zero.
color means until the Spider is standing on a square of the specified color.

Spider Language
Figure 5