On Constructing a Virtual Loom

Susan McBurney
Western Springs, IL USA
E-mail: smcburne@iit.edu

Abstract

This paper explores the simulation of the weaving process on a computer with a simple graphic language. Details of some of the commands serve to illustrate general programming techniques which could be applied elsewhere. Then specific math-related designs are explored and finally, a reference is made to the history of both weaving and computer development and a connection between the two.

The weaving of textiles is nearly as old as mankind and today is found throughout the world. The process itself, as practiced on hand looms, is very slow and labor intensive.

It is interesting to wonder if this pattern-making could be simulated on a computer. The process to do so evolved in a very modular, ordered and structured manner, and it soon became evident that this could serve also as a good example of programming techniques. This paper will present the development of the process in a manner suitable to be used for teaching in a classroom or for self-study. In particular, the following topics are introduced in a logical order that lends itself to beginning programming studies.

The outcome is not intended at all to be used on a professional level or as a general-purpose program. Rather, it is the process and programming strategies that are the most important product of the paper. Ready-to-use applications can be referenced on the web [1, 2, 3].

General Outline of Method

Roughly speaking, the topics should be introduced in the following order.

1. Analysis of the problem
2. Identification of data structures
3. Basic language commands
4. Looping and subroutine implementation. This would include passing parameters, the concept of global and local variables and nesting subroutines.
5. Modular construction of the program
6. Refinements

Basic Language Commands

The LOGO programming language is a very simple language designed to be easily learned by beginners while at the same time being powerful enough for advanced use in more complex applications. It is free for downloading from multiple sites [4, 5, 6]. Additionally, there are many other versions available. Although programming books are scarce, a helpful reference can be found on Wikipedia [7] and in the program's own help menu.

The most fundamental commands are those that move the drawing point (forward, back) and those that rotate the drawing direction of the point (right, left). Other commands used here are pen up (move without drawing), pen down, set pen color, set fill color, and fill (flood an enclosed area with specified

color). Subroutines can be defined as needed and nested which facilitates a modular design which is more easily designed, debugged, and understood. Comments begin with a semi-colon and variable names within an instruction must be preceded with a colon.

Beginning the Process

Examination of a basic under-and-over weave sample such as shown in Figure 1, suggests the fundamental unit should be a square.

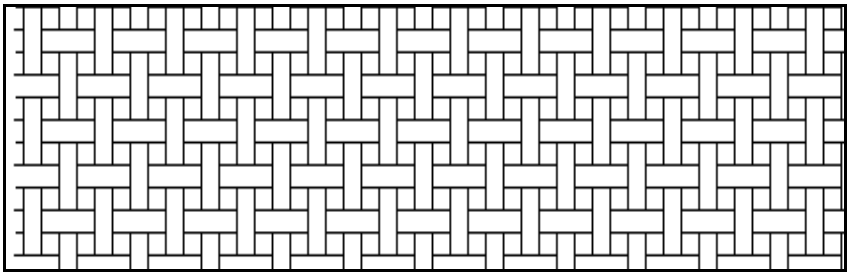

Figure 1: *Basic under-and-over weave*

Writing in the simple language called LOGO, where the basic commands are "forward", "back", "right" and "left", a subroutine named (SQ) to draw a square and fill it with color is shown below.

```
to SQ :size :color  ;Draw a Square--comments begin with semi-colons
   setpc 0          ;set pen color
   repeat 4 [forward :size   right 90]   ;repeat instructions inside [  ] Draw line, turn
   setfc :color     ;set fill color
;pu = pen up (do not draw)  pd = pen down   move inside square and fill with color
   pu   right 45  forward  2   fill   back  2  left 45   right 90   forward :size    left 90   pd
end
```

It is easy to anticipate needing to move right at each step without drawing a line, and since subroutines can be nested, it is efficient to create a **"Move Right" (MR)** and corresponding **"Move Up" (MU)** subroutine (not shown). With both of them, a negative number will indicate a move in the opposite direction.

```
to MR :amount       ;Move right without drawing
;pen up, turn right, move forward, turn left, pen down
   pu    right 90   forward :amount   left 90   pd
end
```

Finally, a pattern could be specified with two colors, one indicating the warp (vertical) thread is on top at a particular junction, and the other showing the weft (horizontal) thread is on the top. For a repeated pattern, only the fundamental portion needs to be specified. For instance, the unit in Figure 2, when repeated, results in the textile shown in Figure 3. Red indicates the *warp* thread is on top and white specifies that the *weft* thread is on top.

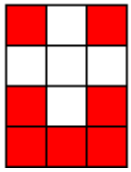

Figure 2: *Basic weave pattern*

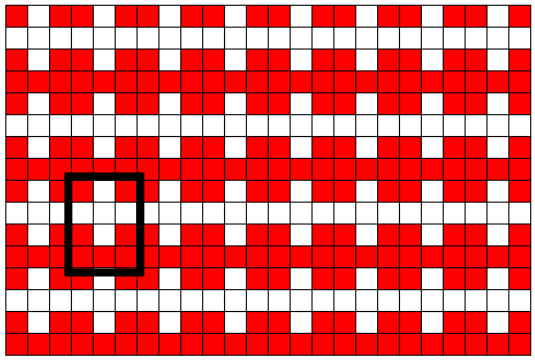

Figure 3: *Resulting textile weave*

The program that draws one row of the basic pattern is as follows:

```
to pat2 :size :number      ;size of individual squares    number of repeats of pattern
;color   4 = red   7 = white
repeat :number[ sq :size  4   sq :size  7   sq :size  4]  right 180   ; red  white  red
repeat :number[ sq :size  7   sq :size  7   sq :size  7]  right 180   ; all white
repeat :number[ sq :size  4   sq :size  7   sq :size  4]  right 180   ; red  white  red
repeat :number[ sq :size  4   sq :size  4   sq :size  4]  right 180   ; all red
end
```

The weave patterns have an attractive quality of their own, but of course in a real example the fibers are narrower and show more texture. What we need now is a variation of the SQ routine which will show both fiber width and color as in Figure 4.

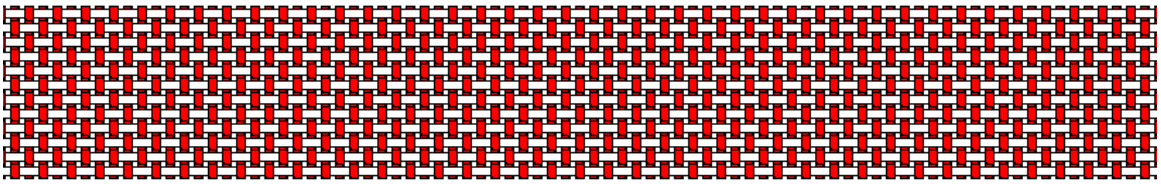

Figure 4: *Basic weave showing fiber width and color*

Again two basic routines are called repeatedly depending on whether the warp or weft thread should be on top. The procedure for the first case is to calculate the difference (d) between the width of the fiber and the size of the square, move right half that amount, draw a rectangle and move to the bottom right of the square. Then go up d/2, turn, draw the weft fiber underneath the warp and return to the bottom right of the square. See Figure 5. An analogous routine draws a square with the weft thread on top. Color fill is also added in each case (not shown). The black outline is actually not drawn.

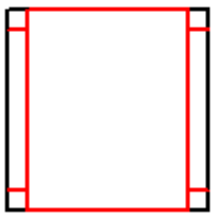

Figure 5: *Show fiber widths in square*

```
to jsq5 :a  b                          ;subroutine to draw Figure 5
;a = size of stitch square  b = width of thread
;mr = move right,  pu = pen up,  pd = pen down
;warp thread over
  mr (:a - :b)/2   forward :a   mrd :b   back :a     ; mrd moves right with the pen down
  mrd   -:a      mr :a     mr (:a - :b)/2        ; draw bottom of rectangle, move to bottom right corner
;weft thread under
  pu    forward (:a - :b)/2    left 90   pd     ; move up to begin weft and rotate 90° to left
; draw left side of weft, move under warp, continue weft
        forward (:a - :b)/2   pu     forward :b    pd    forward    (:a - :b)/2
        mrd :b   back (:a - :b)/2              ; move right with pen down and start right side of weft
  pu     back :b    pd   back (:a - :b)/2      ; go under warp, continue drawing right side of weft
  right 90    pu    back (:b + (:a - :b)/2)    ; rotate 90° to right, pen up, move to bottom right corner
end
```

Now we have the ability to "weave" patterns from a small initial diagram.

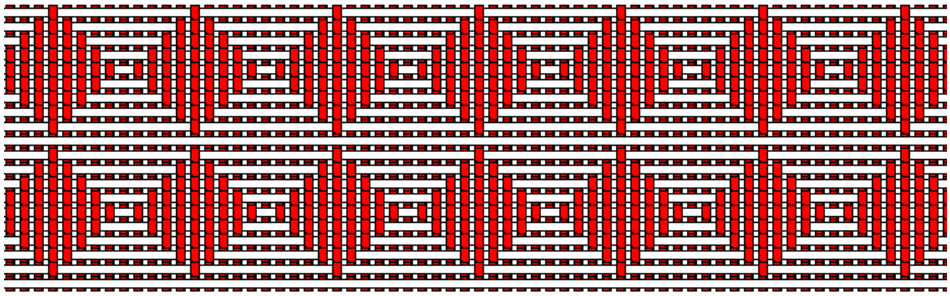

Figure 6: *Woven Diamond pattern*

We can change the orientation of the basic motif.

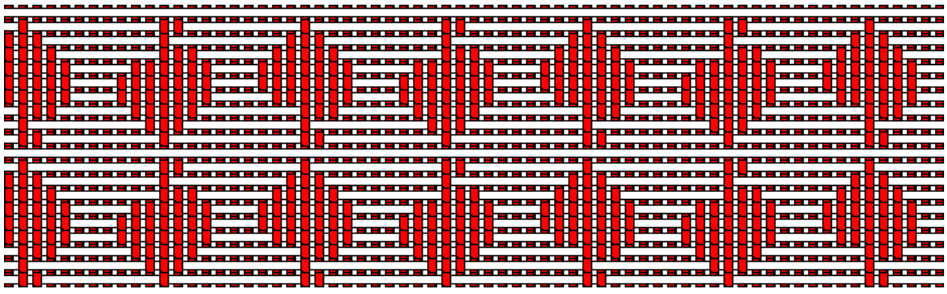

Figure 7: *Free-form object with glide reflection*

Weaving Mathematical Themes

Searching for mathematical themes to incorporate into our designs can proceed along two different tracks. First, we can look for those items which fit naturally into our binary woven structures. For instance, cellular automata and binary truth tables both have characteristics which can be easily adapted to a woven structure.

A	B	A×B
0	0	0
0	1	0
1	0	0
1	1	1

Figure 8: *Truth Table*

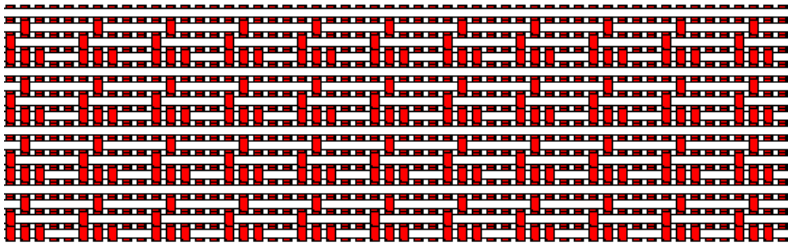

Figure 9: *Pattern woven using Figure 8*

A 9x9 cellular automata is shown in Figure 10 with two similar starting conditions, the only difference being a small hole in the center of the second one. The rule governing each state is that a square with 3, 4 or 8 red neighbors will be red in the next step. Otherwise it will be white. One of the patterns has been used in the design of Figure 11.

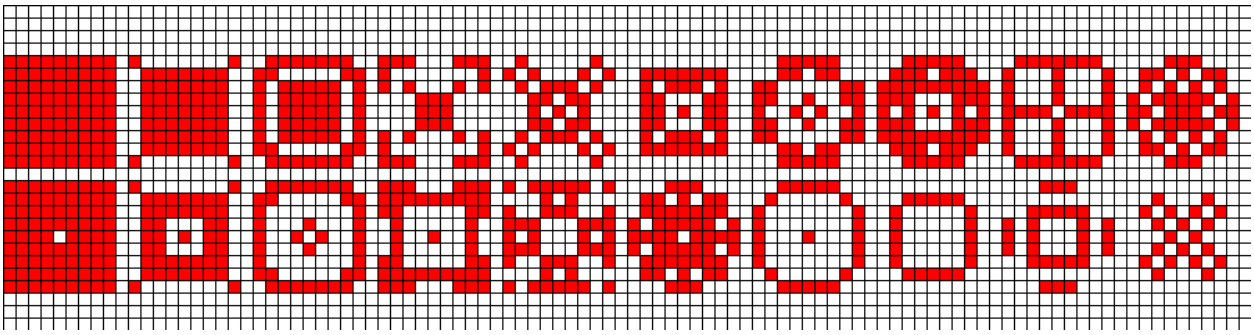

Figure 10: *A cellular Automata with two similar starting states*

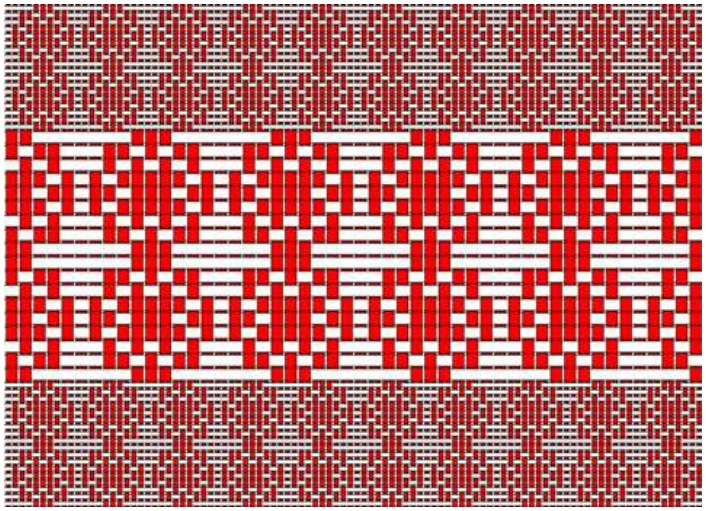

Figure 11: *Design from last state of top row in Figure 10*

Secondly, we can look at less obvious mathematical items and try to find characteristics which can be expressed on the "loom". For example, even multiples of seven appear in this pattern in Pascal's triangle [7].

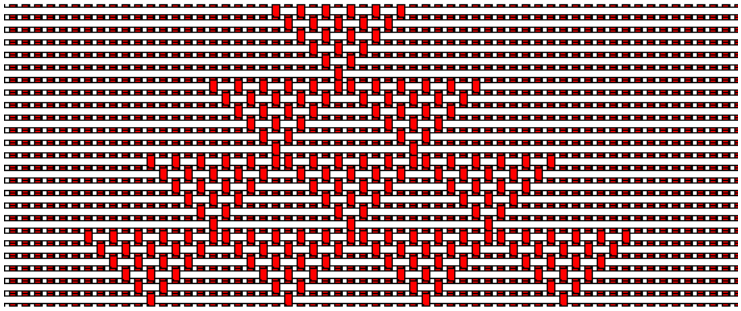

Figure 12: *Even Multiples of seven in Pascal's Triangle*

7		21		35		35		21		7
	28		56		70		56		28	
		84		126		126		84		
			210		252		210			
				462		462				
					924					

Figure 13: *Rows 8 - 13 of Pascal's Triangle (partial)*

Refinements

Additional programs come readily to mind for the development of designs. Specifically, we could use subroutines which rotate a design by 180°, allow a design to be copied and saved, allow motifs to be varied within a row, add a border, and so on. The direction of enhancement can be tailored to the specific interests and abilities of the students as well.

Historical Note

Finally, it would be remiss to not comment on the connection this theme has to computing history [8]. The mechanical Jacquard Loom (patented in 1804) revolutionized the weaving industry and the key idea behind it was the use of punched cards to control the lifting of particular warp threads at each passage of the weft. The use of punched cards was carried forward in other applications (Babbage's Analytic Engine, Hollerith's census-counting machines) and eventually appeared as input devices to early computers. Now we have used the modern computer to simulate weaving on a loom, bringing the process full circle. It seems especially appropriate to celebrate the progress made over the years by recreating the output of one of the early starting points in the computer's developmental history.

References

[1] Fiberworks PCW, www.fiberworks-pcw.com
[2] Winweave 1.1, Free downloadable software, www.softronix.com
[3] Interweave Press, Article on weaving software, www.interweave.com/weave
[4] MSW Logo (Free download), http://www.softronix.com/logo.html
[5] UCB Logo, http://www.eecs.berkeley.edu/~bh/
[6] Logo instructions, http://en.wikipedia.org/wiki/Logo_(programming_language)
[7] David Wells, *Dictionary of Curious and Interesting Geometry*, Penguin Books, 1991
[8] James Essinger, *Jacquard's Web*, Oxford University Press, 2004