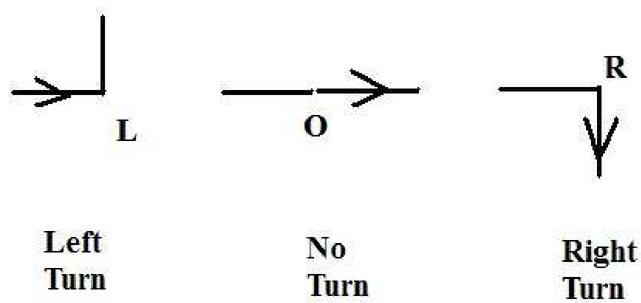
GENERATING 2D CURVES USING A GENETIC CODE

It is well known that living cells are replicated via information contained in their DNA molecules. In particular, the sequential arrangement of just four bases (Adenine, Thymine, Cytosine, and Guanine) determines all characteristics of replicated cells. It would seem reasonable that a similar blue-print like code should be able to reproduce certain mathematical curves. Our purpose here is to explore such a possibility. It should be pointed out that the idea of using a code to perform certain tasks is well known. One need only to mention the Jacquard loom of 1804 or the early 1950s punch card control for electronic computers. To keep the discussion simple we will concern ourselves here only with codes capable of producing some simpler 2D curves consisting of concatenated unit length straight lines segments. Similar to the four base in DNA molecules, the connection between line segments ends will be associated with just three possibilities , namely, R, O, and L. We characterize these math bases as follows-

THE THREE BASES FOR GENERATING 2D CURVES

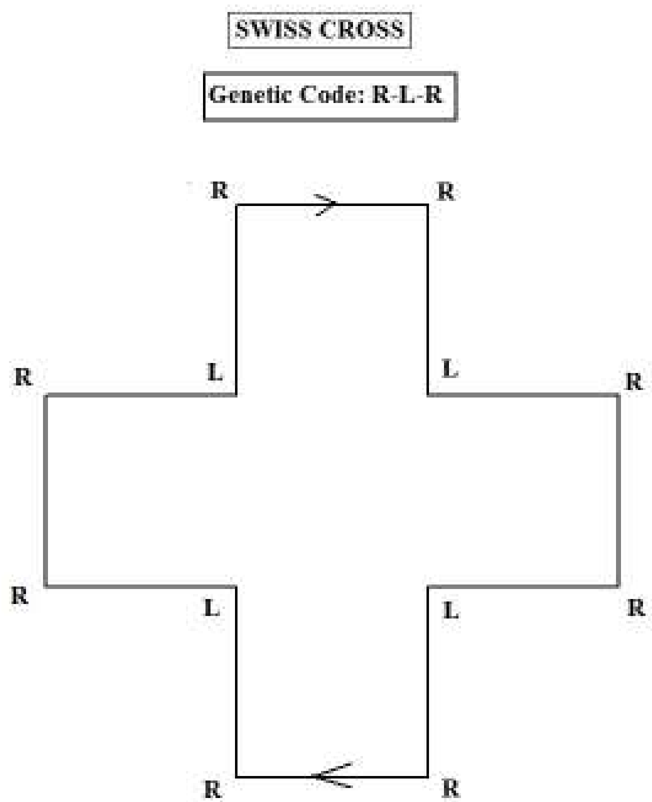


(moving from left to right around curve)

Let us look at a simple case whose genetic code reads-

$$R-L-R-R-L-R-R-L-R-R-L-R$$

This shows a repeating sequence of right turn(R) followed by a left turn (L), and followed by a right turn(R) . If one follows this three letter sequence R-L-R four times, one finds the following closed 2D figure of a Swiss Cross-

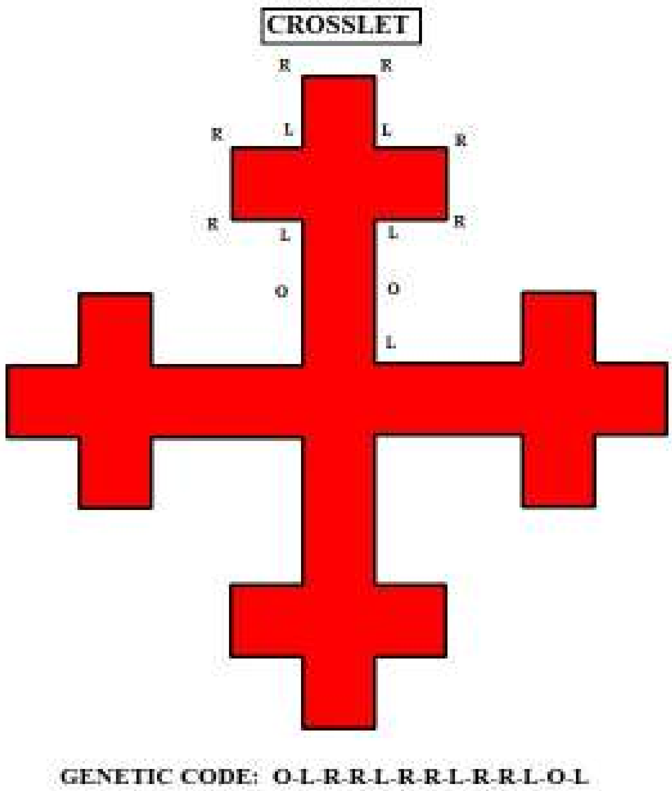


Notice this result scales to all sizes from the present set value of unity for each straight line. Also the code says nothing about the orientation of the cross. A rotation is produced by orienting the boundary R-R on top of the cross by a fixed angle relative to the horizontal.

A more complicated code is-

$$O-L-R-R-L-R-R-L-0-L$$

It produces the crosslet shown-



A slight variation of this code will produce the swastika.

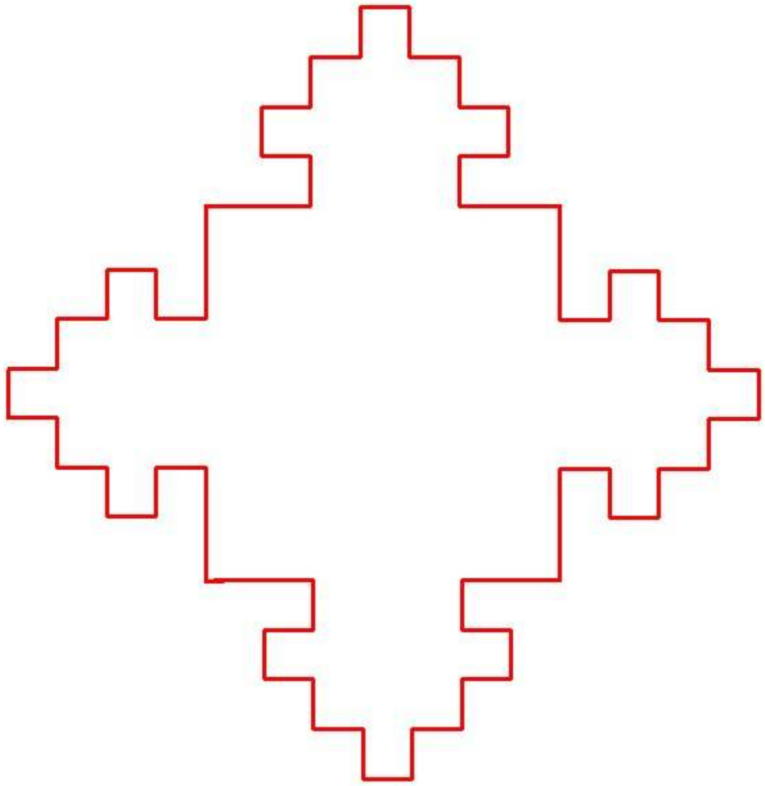
We next look at a still more complicated code whose basic 20 element sequence reads-

R-O-L-L-R-R-L-R-L-R-R-L-R-L-R-R-L-L-O-R

Notice this sequence is in the form of a palindrome. It will produce a mirror image when split in the middle. If one runs four of these sequences in series, the following closed four-fold symmetric 2D curve is produced-

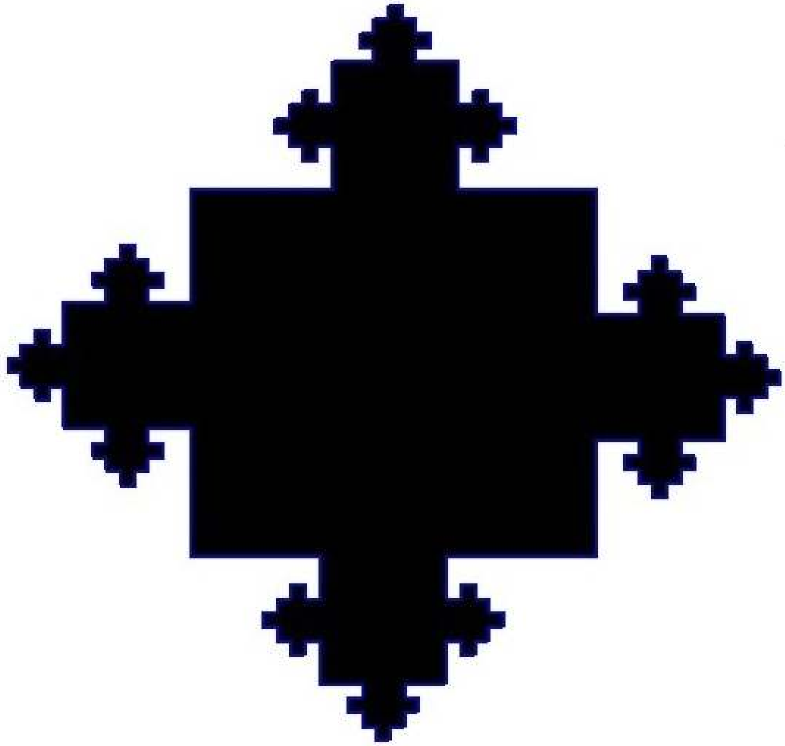
FIGURE WITH A FOUR-FOLD ROTATIONAL SYMMETRY

(graph generated by four applications of the code given below)



Genetic Code: R-O-L-L-R-R-L-R-L-R-R-L-R-L-R-R-L-L-O-R

One can think of this pattern as a square and its first two generations. An even more complex pattern, requiring a still more elaborate genetic code , is found when adding a third generation. It produces the following figure which I term the black snowflake-

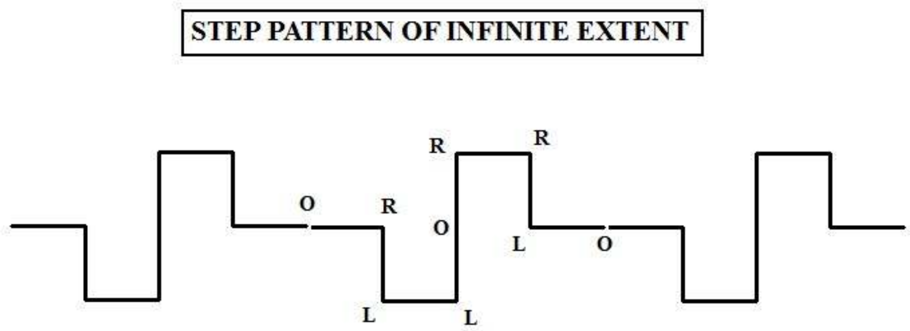


Note that each generation of squares in this pattern has a side length equal to 1/3rd of the previous generation.

Genetic math codes describing a curve need not necessarily lead to closed curves or have multiple symmetries. For instance, repeated application of the code –

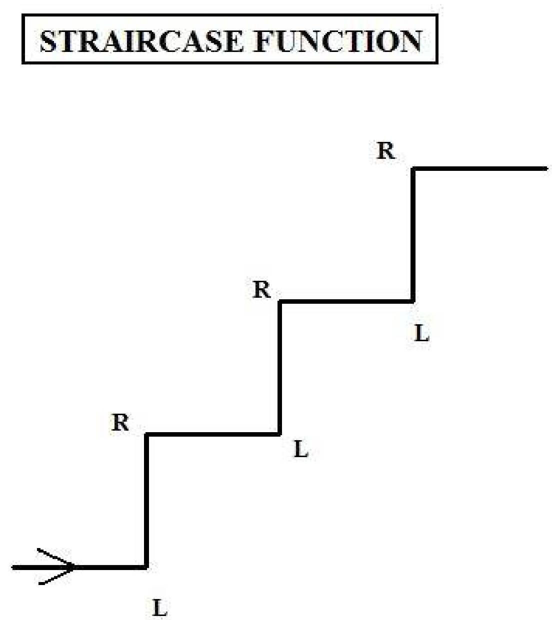
$$O-R-L-L-O-R-R-L$$

produces the following infinite long step pattern shown-

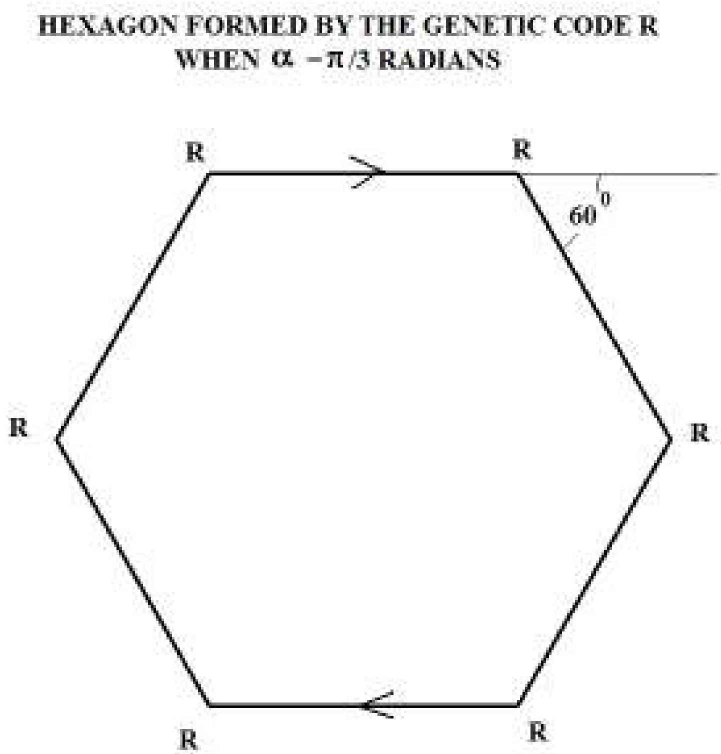


Genetic Code: O-R-L-L-O-R-R-L

A staircase function is generated by the very simple code -L-R-L-R -L-R- as shown-



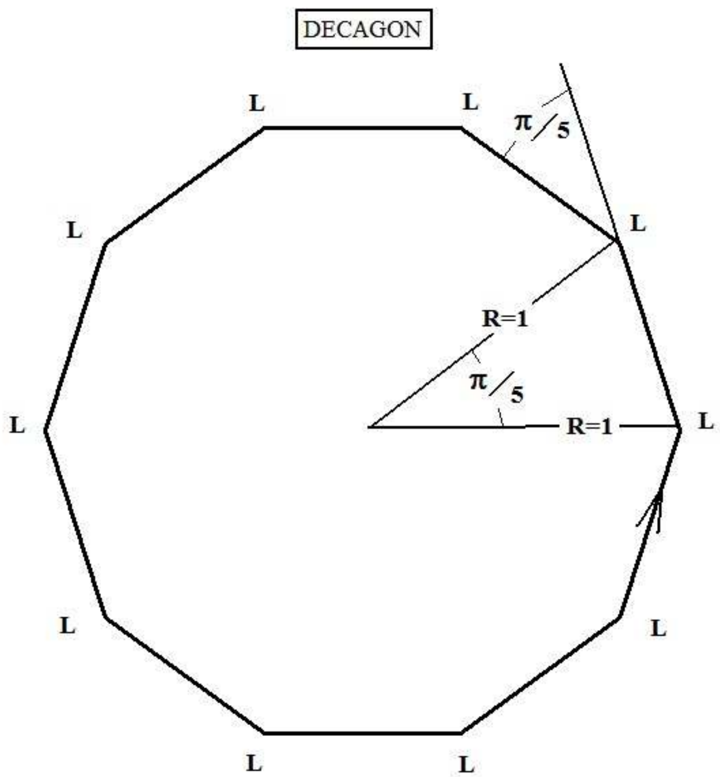
The next question which arises is how does one handle curves which are still composed of unit length lines but the bend angle between neighboring increments is no longer just ±π/2 rad or 0. In this case one can still use the three bases genetic code with the understanding that R and L mean bending by angle ±α. Take the case of a hexagon where we have the one letter code R meaning a bend by 60 deg to the right. A concatenation of six of these operations will produce the hexagon shown-



Any regular polygon can be generated by the simple code L-L-L-L-… or by R-R-R-R… with α =2π/k, where k represents the number of sides to the polygon. A simple MAPLE program which will do this is-

```
with(plots):
listplot([seq([1,2*Pi*n/k],n=1..k+1)],coords=polar,color=black,axes=
none,thickness=2,numpoints=2000,scaling=constrained);
```

For regular polygons the problem is simplified by use of polar coordinates as indicated and letting the radial distance from the center of a polygon to each of its vertexes be equal to one. Here is an example for a decagon (k=10) -



Other figures may be constructed using alphas which vary for different line increments of the figure. In that case one can just write down the bend angles of neighboring lines calling an angle positive if it involves a bend to the left and negative if the bend is to the right when transversing the figure in a clockwise manner. Look at the following genetic code-

$$(\pi/4),(-3\pi/4),(0),(3\pi/4),(-\pi/4)$$

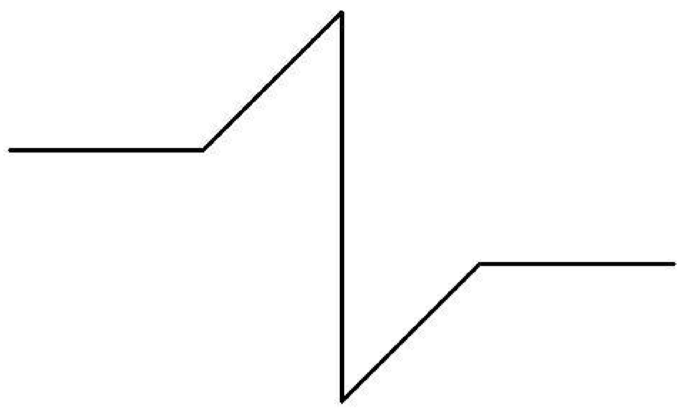
which can be more conveniently written as-

$$\frac{\pi}{4}[1,-3,0,3,-1]$$

What does it represent? It starts with a horizontal unit length line which connects to a second unit length line bent to the left at 45 deg. This is followed by a third line bent to the right at 135 deg and which in turn is connected to a fourth unit length line segment going in the same direction, This is followed by a fifth line bent to the left at 135 deg.

Finally a sixth line segment is bent to the right by 45 deg. Here is the graph-

ZORRO FUNCTION



Genetic Code: $(\pi/4)[1,-3,0,3,-1]$

I call this the Zorro Function because of the appearance of the slanted Z.

Finally, one can envision other genetic math codes including ones where the restriction on a unit length line for all segments is relaxed. Also 3D curves and surfaces should be definable by genetic math codes. I leave these considerations for a future note.

February 2012