Tech Know Learn (2012) 17:115–122
DOI 10.1007/s10758-012-9195-4

COMPUTATIONAL DIVERSIONS

Computational Diversions: The Return of the Spherical Turtle

Michael Eisenberg

Published online: 9 December 2012

Two years ago in this column (time flies!), I discussed the idea of programming a Logo turtle to move about the surface of a sphere. This was an idea explored at length in Abelson and diSessa's (1980) book *Turtle Geometry*, and it's a fun way of introducing many of the seemingly strange ideas of non-Euclidean geometry. In that earlier column, I described a still-embryonic programming system that permitted spherical turtle programs to be drawn on the giant spherical screen, "Science on a Sphere",[1] produced by the National Oceanic and Atmospheric Administration (NOAA). The column closed with a photograph of a Logo "flower" drawn on the giant sphere at our local Fiske Planetarium here in Boulder, and I've brought back that photograph as Fig. 1 for this column: the flower was a thank-you gift of sorts for Hal Abelson and Andrea diSessa in honor of their remarkable and timeless book.

This essay returns to the topic of spherical turtle geometry, but in a more participatory spirit. As of this writing, we now have a Web-accessible programming interface for the spherical turtle, programmed by Antranig Basman and Michelle Redick at the University of Colorado; that's just a formal, academic way of saying that *you*, the reader, can now bring up a working spherical turtle interface and write programs for it. (I *could* refer to the system as a "spherical Logo interface", but that would be inaccurate—since, as you'll see if you play with the system, the language only has some superficial similarities to Logo.) The spherical programming system is available through the beautiful website created by Dr. Sherry Hsi and her colleagues at the Lawrence Hall of Science in Berkeley, California. Here's how you can access the system: first, go to the website www.mathsphere.org, and then follow the "Tools" link to get the current web address of the spherical programming interface.; currently, that interface is available at: http://mathsphere.org/mos-client/client.html.[2]

[1] NOAA website for "Science on a Sphere": http://sos.noaa.gov/index.html.

[2] The current web interface works in the latest versions of the Firefox and Google Chrome browsers.

M. Eisenberg (✉)
University of Colorado, Boulder, CO, USA
e-mail: ijcml-diversions@ccl.northwestern.edu

Fig. 1 A spherical "Logo flower" pattern drawn on the Science on a Sphere installed at the Fiske Planetarium in Boulder

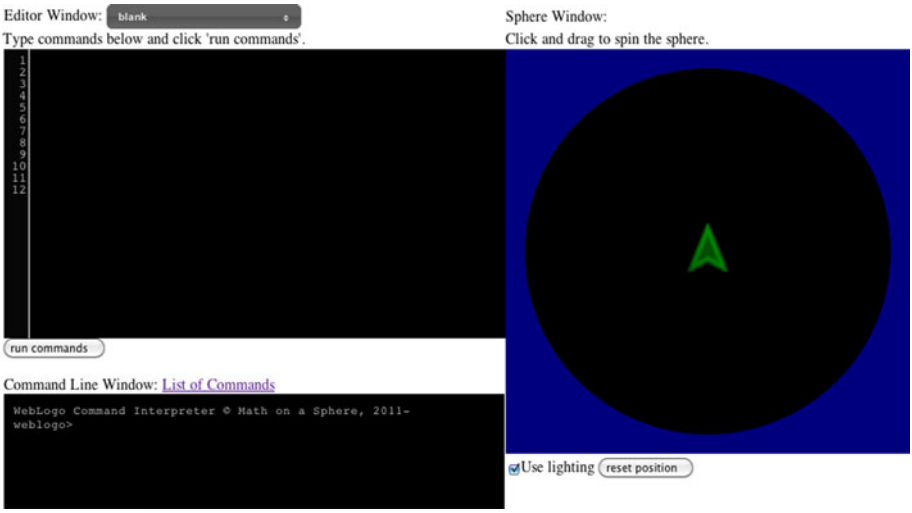

Fig. 2 The spherical programming interface. At *upper left*, an editor window in which full programs may be written and run (by selecting the "run commands" button underneath the window). At *lower left*, an interactive command window. At *right*, a rendering of the sphere, with the turtle icon initially placed at the equator, pointing due north. This sphere may be turned interactively via the mouse to see the surface from any direction

When you bring up the spherical turtle interface, you will see a window in your Web browser that looks like Fig. 2. Toward the right of the window there is a rendering of a sphere with a turtle icon on it; at the upper left, there is an editor window in which to write programs; and at the bottom left, there is an interactive window in which simple one-line turtle commands can be given. It should be noted that the system is still a work-

in-development, so if you are reading this column long after its composition, the screen may look different than it does at present; if all goes according to plan, there will be fewer bugs and more powerful language features as well. In this column, I will "talk you through" some relatively simple spherical geometry examples, just to get you started; a future column will deal with more advanced concepts.

1 Example 1: An Equilateral Right Triangle

As a starting example, we can write turtle commands that will produce an equilateral right triangle. Such things can't, of course, be found on the plane–a Euclidean equilateral triangle has three 60-degree angles. On the sphere, however, things work differently, as you can demonstrate for yourself by typing in the following sequence of six commands at the interpreter window at bottom left:

```
forward 90
right 90
forward 90
right 90
forward 90
right 90
```

It is worth examining these commands, just to get a feeling for our spherical version of the Logo turtle. First, note that a "forward" command moves the turtle along a great circle on the sphere–you can imagine the turtle moving in the direction that its nose is pointing, taking small-but-equal steps with both its left and right pairs of legs. The resulting great circle arc is thus regarded as a "straight line" on the sphere, The units of our sphere have been chosen so that a forward move of 360 steps will take the turtle all the way around the sphere in a complete great circle, returning it to its starting point; again, this is a manifest impossibility on the plane, where a Logo turtle that keeps moving forward will wander off into infinite realms beyond the computer screen and never return to its starting place.

In our initial example, the turtle begins at the equator of the sphere, pointing toward the North Pole. The first `forward 90` move, then, draws a line (i.e., a great circle arc) from the equator to the North Pole, corresponding to one quarter of the complete trip around the sphere. The turtle, now sitting at the Pole, turns 90 degrees to the right, and then makes another `forward 90` move back to the equator. At this point the turtle is sitting at a spot on the equator, a quarter of the way around the sphere from its initial equatorial position, and pointing toward the South Pole; one more `right 90` turn points the turtle along the equator, and a final `forward 90` move returns the turtle to its original starting point. The resulting graphical pattern, as it appears on the screen, is shown in Fig. 3.[3] You can see in the figure that we have created an equilateral spherical triangle: each of the three sides is one-quarter the circumference of the sphere, and all three angles are right angles.

[3] As an aside, I should mention that you can "grab" the sphere image in the right window with your computer's mouse, and turn the image about to look at it from different angles; the little "reset position" button underneath the window returns the sphere to its original viewing position.

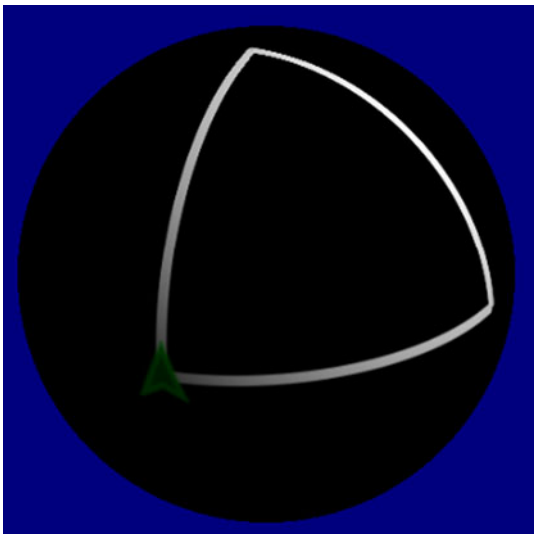

Fig. 3 An equilateral *right triangle*, drawn on the sphere

2 Example 2: Longitude Lines

For the second example, we'll make a bunch of longitude lines on the sphere; there isn't really any new spherical geometry here (beyond the previous example, that is), but we can use this project to introduce a bit more of the system's language.

We can begin, once again, by typing in some commands directly to the window at lower left:

```
clearall
pu
fd 90
```

The first `clearall` command erases any current pattern on the sphere (we start with that command in case you still have the triangle on the sphere from the earlier project), and repositions the turtle at its usual starting spot on the equator. The second command is the familiar Logo "penup" command; the third is a forward move (abbreviated in the usual fashion as "fd") to the North Pole, and since the turtle's pen is up, there is no line drawn in this step. The sphere now looks as in Fig. 4a.

At this point, we are ready to draw a set of longitude lines, as follows:

```
pd
repeat 20 {fd 360 rt 18}
```

The `pd` command is the standard Logo abbreviation for "pendown"; the `repeat` command then draws 20 complete great circles, each separated by 18 degrees. The resulting sphere appears as shown in Fig. 4b.

By playing with repeated patterns of great circles (along with the handy penup and pendown commands), and nothing else, it is already possible to make some attractive spherical designs. The pattern shown in Fig. 5, for instance, is made by positioning the turtle at several locations and then making repeated great circles; I will leave the specifics as a puzzle for the reader.

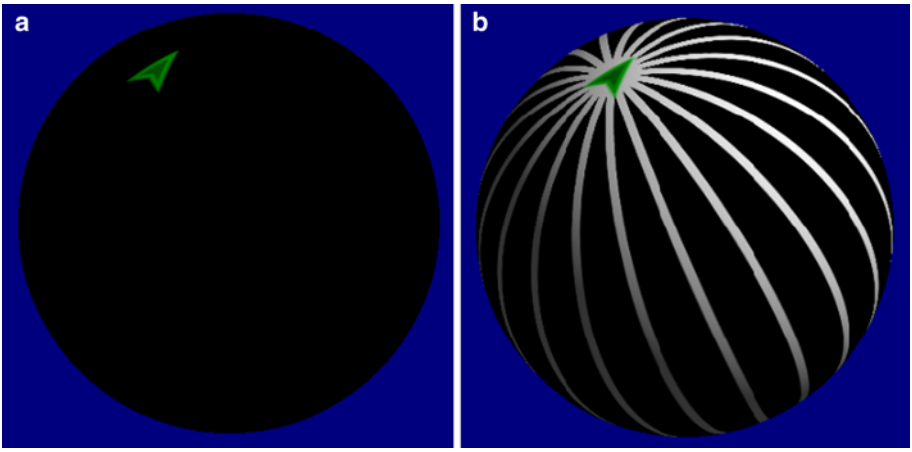

Fig. 4 **a** (*left*) The turtle is first moved to the North Pole with the pen up, so that *no lines* are drawn. **b** (*right*): The turtle now creates *20 great circles*, each separated by 18 degrees; at the end of this process the turtle is still at the North Pole, pointing along the same *longitude line* as in **a**

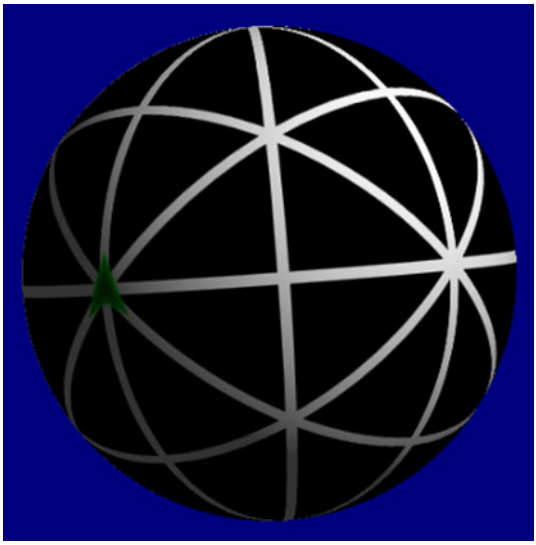

Fig. 5 A pattern of *great circles*, left as an exercise for the reader

3 Example 3: Latitude Lines

For the final example in this column, we will explore how to create latitude lines using the spherical turtle. As a first step, we note that in general, latitude lines are *not* "straight lines" on the sphere–that is, with the lone exception of the equator, latitude lines are not great circles. They are instead spherical circles centered at the poles. In order to make a spherical

circle, then, we will have to expand our repertoire beyond the mere creation of great circles.

Here is a feasible idea for creating a latitude line:

Step 1: Begin with the turtle at the North Pole.
Step 2: With the pen up, move the turtle south to the desired latitude.
Step 3: Turn the turtle left 90 degrees, so that it faces due east.

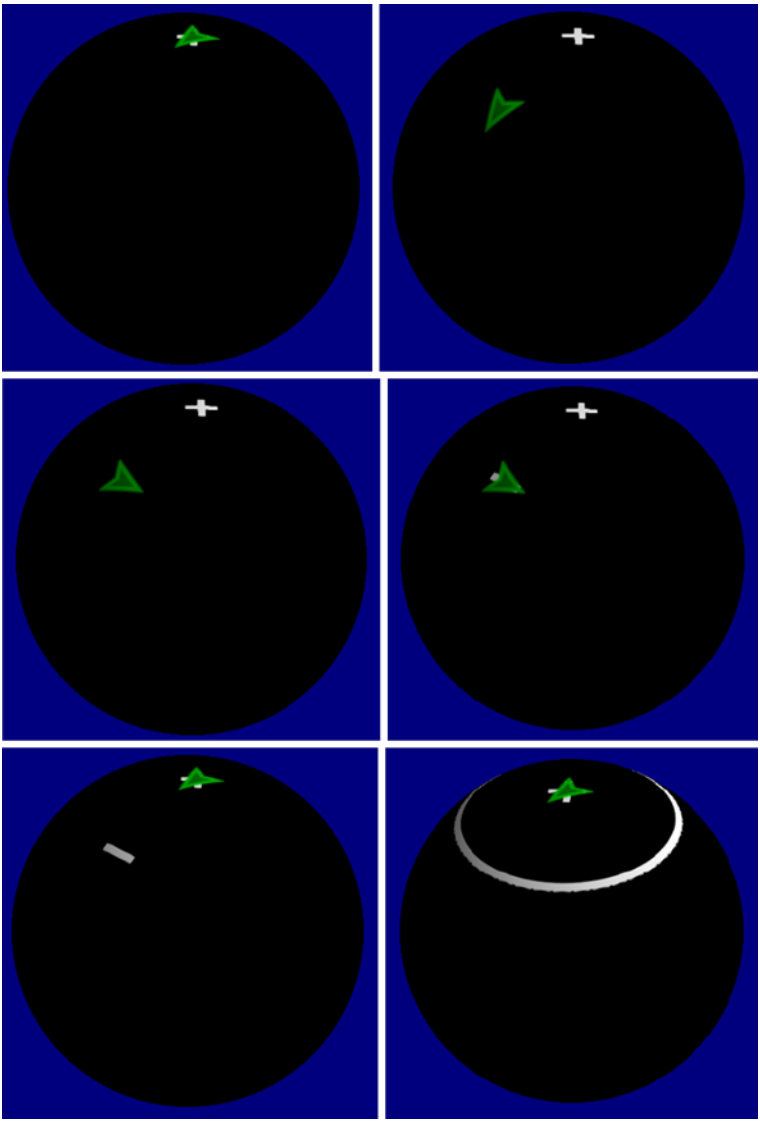

Fig. 6 A sequence of diagrams showing the drawing of a *latitude line*. *Top row* the turtle is positioned at the North Pole (marked with an "*X*"), and moves down 40 steps. *Middle row* the turtle draws a *short line*, representing an approximate chunk of the eventual *spherical circle*. *Bottom row* the turtle returns to the North Pole, turns a bit, and then repeatedly draws more chunks of the *circle* until the *latitude line* is complete

Step 4: Put the turtle's pen down, then move it back and forth a short distance; in effect we are making a small chunk of the latitude circle.
Step 5: Pick up the pen, turn the turtle so that it is facing south once more, and then move the turtle back to the North Pole.
Step 6: With the turtle back at the pole, turn it a bit. Now go back to step 2 and repeat creating short chunks of latitude line until the entire spherical circle is complete.

This idea is illustrated in Fig. 6; in the figure, we use this basic template to create a latitude line 40 degrees from the North Pole.

To implement this idea, we can create a function called latitudeline in the program window at the upper left of the screen. In showing the text of the function below, we can use this example to illustrate the syntax of our language, and parenthetically to introduce a new command to set the turtle's pen color:

```
latitudeline = function [dist, bar, col] {
 set color col
 repeat 36 {
    fd dist
    lt 90
    pd
    fd bar
    bk (2 * bar)
    fd bar
    rt 90
    pu
    bk dist
    rt 10
}
}

pu
fd 90
latitudeline [40, 6, white]
```

In prose, we have created a function `latitudeline` that takes three arguments: a distance (corresponding to the latitude circle that we wish to draw), a bar-length (corresponding to how big each drawn segment should be), and a pen color. The text of the program reflects the idea that we summarized above: here, we assume that the turtle has initially been positioned at the center of the desired spherical circle. The `repeat` form creates 36 short bars, each of twice the value of the bar-length parameter, to create a full spherical circle. The final three lines in the text above show how we can create a white latitude line at 40 degrees from the North Pole. The reader can copy this entire textual sequence into the program window and then press the button labeled "run commands" to see the latitude line drawn.

There are many more examples and programming ideas that we could present here, but this should be enough to get the interested reader started; as mentioned above, I will pursue more advanced program ideas in a subsequent column. It should be noted in any event that with the ideas already presented here, there are many wonderful patterns to be created. Figure 7 shows a sampler of graphical ideas that make use exclusively of great circle arcs

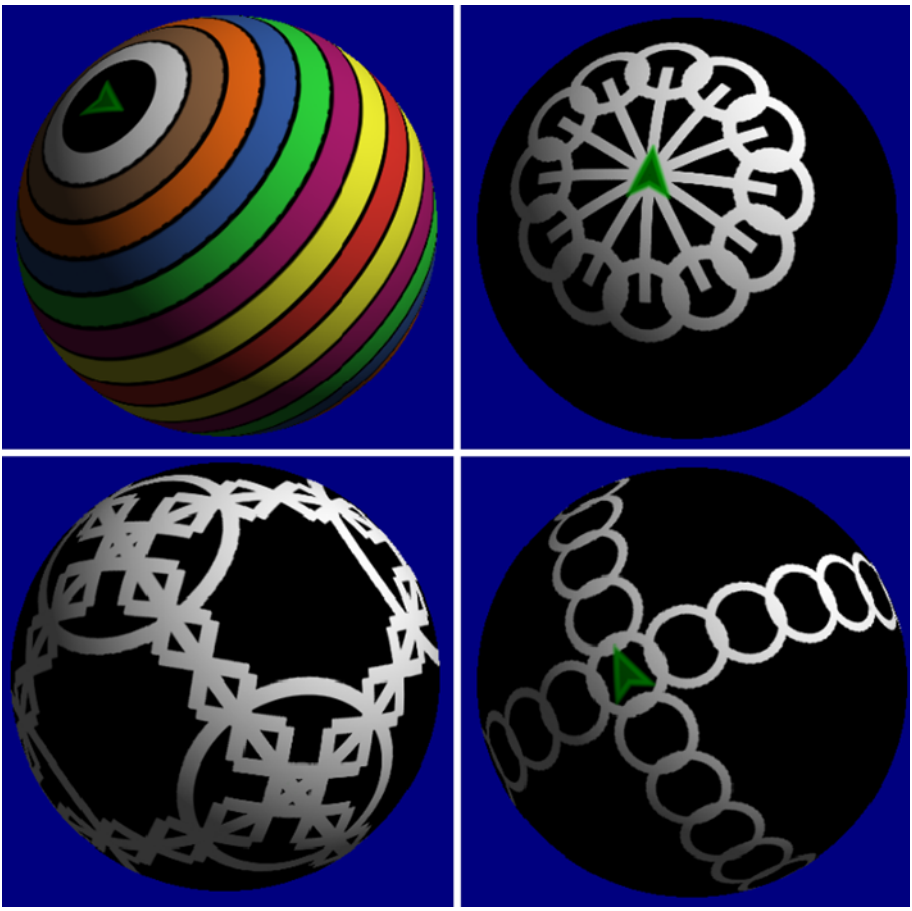

Fig. 7 A sampler of spherical turtle-drawn designs, all created using only *spherical circles* and *great circle arcs*. The *top left* example employs a variety of pen *colors*; all the examples make use of commands to specify the pen size for the turtle (for instance, the *bottom right* example uses the command `set pensize 3` to create a slightly *thicker line*). Except for this one novel technique, all the examples shown use only language elements presented in the text of this column. (Color figure online)

(i.e., "straight lines") and spherical circles. The reader is of course encouraged to explore his or her own novel patterns, and to send their favorite examples to the email address for this column at: ijcml-diversions@ccl.northwestern.edu.

Acknowledgments Antranig Basman and Michelle Redick programmed the web interface described in this column; Sherry Hsi of the Lawrence Hall of Science has collaborated on this project and directed both student workshops and the construction of the Math on a Sphere website. Thanks to Francisco (Tito) Salas of the Fiske Planetarium in Boulder, and to Mike MacFerrin, whose initial mathematical programming was instrumental in pursuing this project. This work has been partially funded by the National Science Foundation under grant DRL1114388.

References

Abelson, H., & diSessa, A. (1980). *Turtle geometry*. Cambridge, MA: MIT Press.