MATH ON A SPHERE: MAKING USE OF PUBLIC DISPLAYS IN EDUCATION

Michael Eisenberg[1], Antranig Basman[1] and Sherry Hsi[2]
[1]*Department of Computer Science, University of Colorado, Boulder CO 80309-0430 USA*
[2]*Lawrence Hall of Science, Berkeley CA 94720 USA*

ABSTRACT

Science on a Sphere (SoS) is a compelling educational display installed at numerous museums and planetariums around the world; essentially the SoS display is a large spherical surface on which multicolor high-resolution depictions of (e.g.) planetary weather maps may be depicted. Fascinating as the SoS display is, however, it is in practice restricted to the use of museum professionals; students (and for that matter, older museum visitors) are unable to create their own displays for the surface. This paper describes a working software system, *Math on a Sphere* (MoS), that democratizes the SoS display by providing a simple programming interface to the public, over the World Wide Web. Briefly, our system allows anyone to write programs for spherical graphics patterns, and then to upload those programs at a planetarium or museum site and see the result on the giant sphere. This paper describes the implementation of the MoS system; sketches a sample project; and concludes with a more wide-ranging discussion of our user testing to date, as well as strategies for empowering children and students with greater control of public displays.

KEYWORDS

Math on a sphere, spherical geometry.

1. INTRODUCTION

When educational technologists refer to "display devices", there is usually a tacit assumption that they are talking about flat screens–perhaps on a desktop or laptop computer, or mobile phone. Not all educational displays, however, fit this description. This paper focuses on one such unorthodox example–a particularly remarkable one called *Science on a Sphere* (SoS), created by the United States National Oceanic and Atmospheric Administration (NOAA), and installed in over 80 museum and planetarium settings around the world. [NOAA 2013] The SoS display, shown in Figure 1, is a large (1.73m diameter) solid-white spherical surface accompanied by four synchronized projectors; typically, the system is used to display (e.g.) planetary weather maps, animations of continental drift, the surface of the Moon, and other "spherical" graphics.

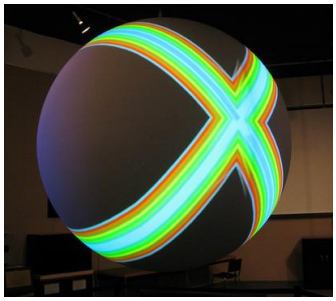

Figure 1. The Science on a Sphere (SoS) display surface at the Fiske Planetarium in Boulder, Colorado. In this photo, the sphere is displaying a geometric pattern generated by the Math on a Sphere (MoS) system described in this paper.

The graphics projected on the SoS surface are multi-color, high-resolution, and smoothly animated–in short, stunning. Still, the surface is "closed to the public" in the sense that while museum audiences can *watch* and *enjoy* graphics on the sphere, they have no accessible medium with which to *create* their own patterns for display on the sphere. In the case of the SoS display, this inaccessibility to the public is particularly regrettable, since youngsters could gain a provocative introduction to non-Euclidean geometry through interactive programming on a spherical surface. By producing their own patterns on the SoS surface, children (and interested adults as well) could encounter notions such as geodesics, intrinsic curvature, and spherical coordinates in the course of compelling, personalized projects.

This paper describes a working (and publically available) system, called *Math on a Sphere* (MoS) that allows users to create graphical patterns that may later be displayed on an actual SoS surface in a planetarium. By writing short programs in a relatively simple language (based loosely on the syntax of the early Logo language), people can create beautiful graphical patterns on their own personal computer screen; these patterns may then be retrieved and displayed on the giant public surface in a planetarium or museum. Currently, our project is conducted with the collaboration of the Fiske Planetarium in Boulder, Colorado; the NOAA offices in Boulder; and the Lawrence Hall of Science in Berkeley, California. Eventually we hope to solicit the cooperation of many more institutions so that larger numbers of planetarium visitors can be given (at least temporary) control of these remarkable spherical public display surfaces.

The remainder of this paper is structured as follows: the following (second) section will provide a structural overview of our software system and an explanation of how it can be used to communicate with an installed SoS system. The third section begins with a brief description of our language (in its current, still evolving, implementation). The section then sketches a scenario for creating a spherical design; in the course of this scenario we will touch upon some of the interesting aspects of spherical (as opposed to planar) geometry that are highlighted by designing graphics for a sphere. The third section concludes with a summary of our experiences with users. The final (fourth section) mentions key related work, and in effect is a more wide-ranging discussion of one of the key motivating ideas behind this project: namely, that large-scale, unusual, and (often) public displays can be made accessible to children for education and play. It should be noted that, we have previously discussed the MoS system in [Eisenberg 2012, Hsi and Eisenberg 2012]; those papers focused on spherical geometry and an evaluation of our first student workshop. This paper, by contrast, is the first full explanation of the system architecture; it is also the first venue for presenting recent student-created examples, and for a more wide-ranging discussion of central educational issues raised by the MoS system.

2. THE MATH ON A SPHERE (MOS) SYSTEM

It is probably most straightforward to describe the MoS system starting with what the user sees: namely, the MoS web interface. A screenshot of the web interface (in the midst of an ongoing programming project) can be seen in Figure 2. The major components of the interface are the three windows seen in the figure: at upper left, an *editor* allows the user to compose programs and define new procedures for creating graphical designs; at bottom left, a *command interpreter* allows the user to type in lines directly (e.g., procedure calls that draw patterns on the sphere); and at right, an interactive *sphere view* window allows the user to see a graphical rendering of the sphere display itself. The sphere in this window, parenthetically, may be "grabbed" and rotated by mouse button presses and movements, so that the user can view all portions of the surface.

We will briefly discuss the end-user graphics language (the one shown in the editor window) in the following section. For the present, one additional element of Figure 2 is worth noting here: the button labeled "Connect" toward the bottom of the figure. When the MoS website is loaded onto a suitably prepared machine at a local planetarium, this button can be used to activate a connection between the MoS web client and the giant spherical display at the museum. (As noted, the project currently has the cooperation of three participating sites equipped with the SoS display; and as the MoS system progresses to completion we plan to solicit collaboration from many more sites.)

The web interface of Figure 2, and the original SoS system (running the sphere in the planetarium) constitute the two major "end portions" of the system: the first, written in HTML5 and JavaScript, is our own creation, while the second is the creation of NOAA. Sitting between these two end portions is the final element of the Math on a Sphere system: the *local server*.

This element (our own creation, like the web interface) sits between the web client and the planetarium sphere, and communicates programs to the sphere. In order to do its job, the local server must be installed "on site", on the same device as the planetarium's SoS system itself.

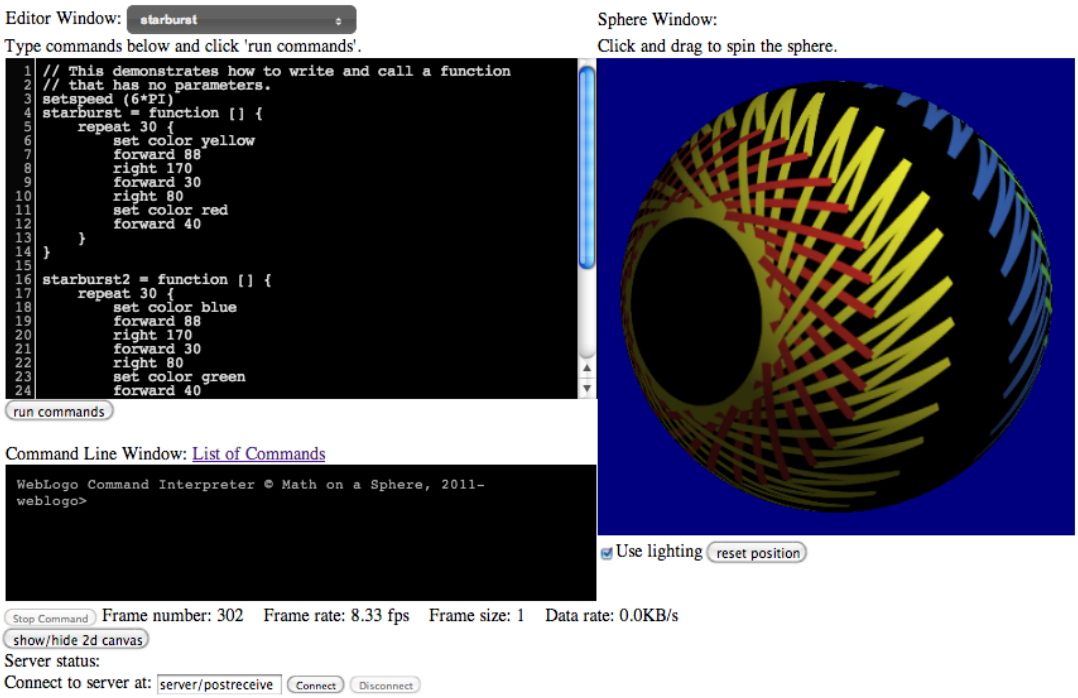

Figure 2. The web interface to the MoS system. At top left, the editor window; at bottom left, the command interpreter; at right, the interactive sphere window. The sphere rendering here is interactive; the "ball" may be turned using a mouse to view a given design from any angle. (The system can be freely accessed through the website www.mathsphere.org.)

In effect, the job of the local server is to act as the "glue" between the end-user programs written in the language interface of Figure 2, and the SoS system that displays the results of the program commands on the sphere. The local server sends to the SoS a "movie" of constant frame-rate composed of a number of individual frames in a standard image format (JPG, PNG, etc.). Each frame of this movie consists of a view of the sphere rendered into a 2:1 aspect rectangular flat image in the ECE (Equatorial Cylindrical Equidistant) projection [SoS 2013]. With the current implementation based on initial Canvas rendering, we use images of dimensions 1024x512 pixels, which (while not optimal) is visually adequate for the SoS system.

The strategy used by the local server to send this "movie" is to maintain a circular buffer of images (100-200 has proved a reasonable choice) as concrete files in a certain directory on the server machine. The SOS system treats these files as an animated movie to play even though their contents are being constantly rewritten dynamically (in circular order - the "local server" keeps in step with the SOS system and rewrites the image files on disk as they arrive from the client).

The overall architecture of the MoS system is summarized in Figure 3, which sketches (at right) the web client, (at left) the pre-existing SoS system and (toward the center) the local server and its functions.

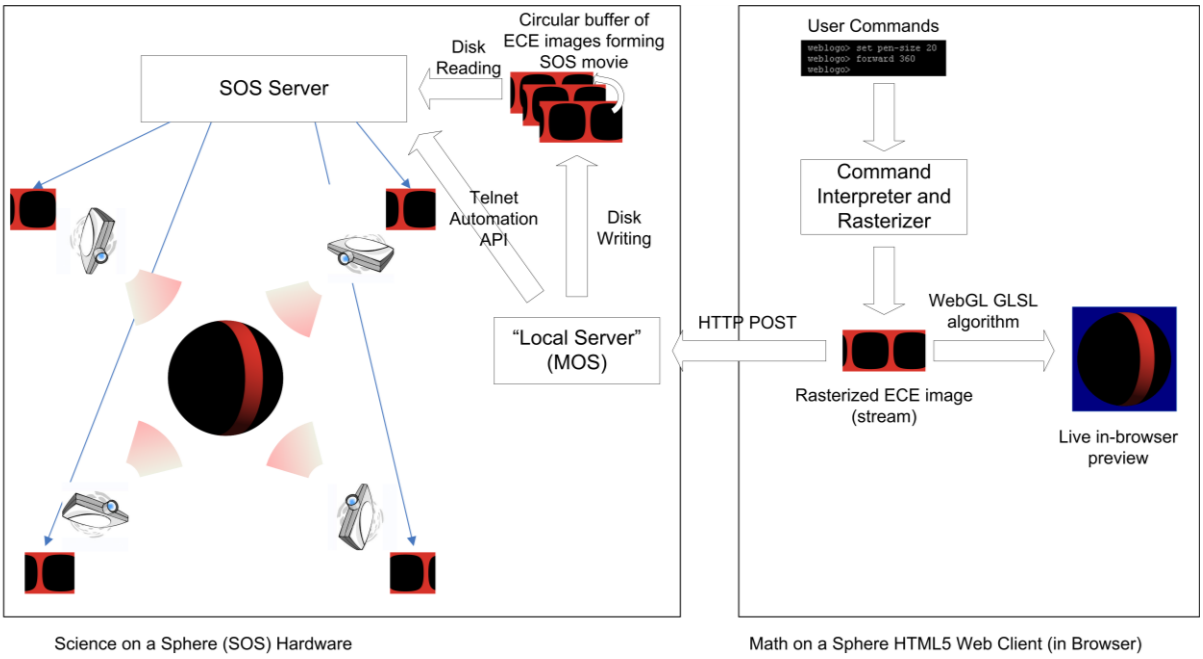

Figure 3. An architectural sketch of the MoS system. When a user runs their program from the web page (at right), the movement and image calculations are performed on this (client) side. The images are then posted by the local server (center) to the installed SoS system (at left) for display. Sequential images are posted in a circular buffer to achieve animation.

3. CREATING A SPHERICAL DESIGN USING MOS

In this section, we illustrate the use of the MoS system through a scenario in which a user creates an icosahedral pattern in the language interface, and then projects that pattern onto the planetarium sphere. In the course of working through this scenario, we will touch upon some of the interesting ways in which spherical geometry differs from the planar variety. Since earlier papers have discussed spherical programming at greater length, this section will be kept brief; the intention here is merely to provide sufficient background to motivate the discussion of the following section.

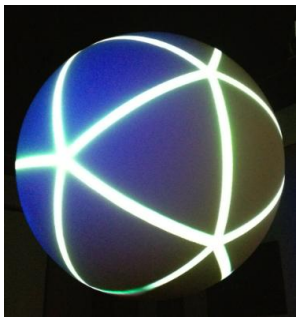

Figure 4. An icosahedral design, created by the MoS language system and displayed on the planetarium sphere.

3.1 Turtle Commands on the Sphere

The central elements of our language system are based on the "turtle commands" of the Logo language, in which a programmable pen (the "turtle") can be steered about a computer screen through commands such as **forward** and **right**.

An excellent introduction to the mathematics of turtle geometry can be found in the classic text by Abelson and diSessa [1980], while Papert [1980] gives an eloquent philosophical introduction to the Logo language design. For our purposes, we can imagine a scenario with a student who wishes to create an icosahedral pattern on her local planetarium sphere, as shown in Figure 4. (Specifically, the design in Figure 4 is the projection on the sphere of an inscribed regular icosahedron.) The student is going to create this design using **forward** and **right** commands, just like she would using standard Logo commands on the plane; except in the case of the MoS system, a **forward** command moves the turtle along a great circle path on the sphere (rather than in a straight planar line).

The Figure 4 design consists of twenty equilateral spherical triangles, arranged about the spherical surface. One immediate point to note, however, is that on a sphere–unlike the plane–triangles do not have interior angles that sum to 180 degrees. In particular, for the spherical triangles in Figure 4, the reader might note that there are five (not six) congruent equilateral triangles arranged around each vertex of the design; thus, each interior angle of each triangle is 72 degrees, and each of the equilateral triangles in the figure has interior angles that total 3*72 = 216 degrees. (For more description of the surprising differences between spherical and planar geometry, see the aforementioned text [Abelson and diSessa, 1980] and the discussions in [Eisenberg 2010, 2012].)

The upshot of these considerations is simply that we need to create an equilateral triangle on the sphere with interior angles of 72 degrees (and exterior angles of 108 degrees). Space considerations preclude a fuller discussion (and again, these issues are discussed in earlier papers), but as it happens the expression for creating an "icosahedral spherical triangle" (as seen in Figure 5a) is:

repeat 3 {forward 63.5 right 108}

The student can now extend her design by creating five icosahedral triangles arranged around a single point:

repeat 5 {repeat 3 {forward 63.5 right 108} right 72}

This will produce the set of five triangles shown in Figure 5b.

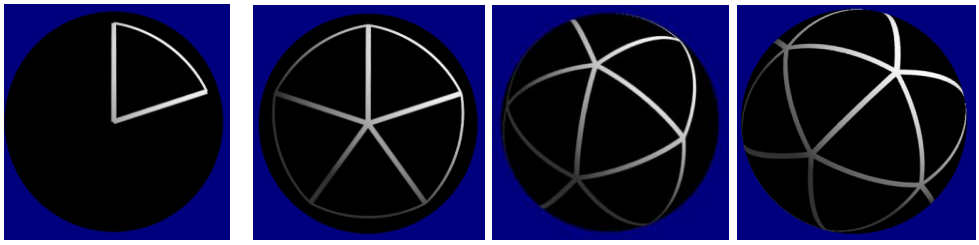

Figure 5. Creating the icosahedral display. 5a (left): a single spherical triangle. 5b (second from left): five triangles surrounding a point. 5c creating still more triangles. 5d (right): the entire icosahedron displayed in Figure 4.

Now, the student could, if she wished, create the full icosahedral pattern by carefully moving the turtle along already-created lines and repeating the command that generated Figure 5b. Continuing with this strategy (albeit carefully) could in fact produce the pattern in Figure 5c and, eventually, the entire icosahedral pattern shown in Figure 5d. Finally, the last step of the scenario is for the student to produce the pattern from 5d on her local planetarium display; by visiting the planetarium (which by assumption is a cooperating institution that has installed the local server to send images to its own SoS system), the student can now produce the image shown in Figure 4 earlier.

3.2 Creating a More Elaborate Design

Having created a "standard" icosahedral design on the sphere, it is now straightforward to use the features of the MoS language to extend or elaborate that pattern. Figures 6a and 6b illustrate this notion: by adding additional lines to the basic triangles of the previous example, one can create a more decorative icosahedral design, and display that design on the NOAA sphere (as in Figure 6c). This example represents, of course, a tiny initial fraction of the types of explorations that can be conducted with the MoS system. The purpose of this scenario has been merely to suggest the sorts of projects that one can realistically undertake.

The larger point–to which we return in the final section of this paper–is that we have now made a gorgeous spherical public display available for experimental use by anyone–children included–with access to a Web browser (and a cooperating local institution).

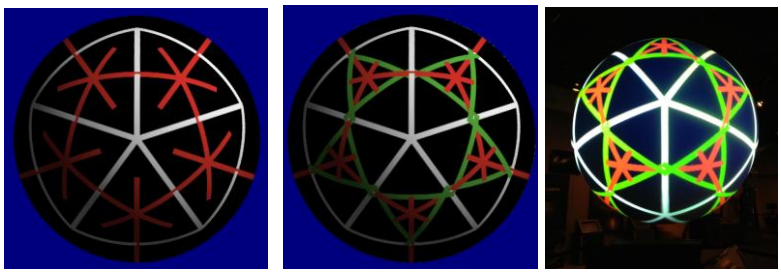

Figure 6. Left and center (6a and 6b), we take the basic triangle structure of Figure 5b and elaborate the design with additional lines and colors. Right (6c): a multi-colored icosahedral design (based on the design in 6b), rendered on the spherical planetarium display.

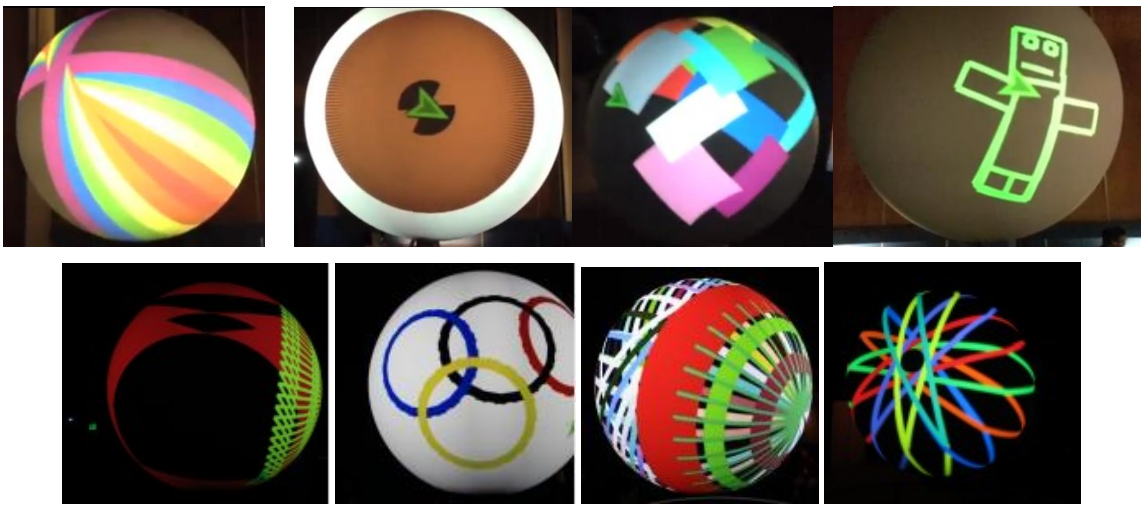

Figure 7. Student-created designs. Upper row: designs created by (in order) a 10-year-old, 12-year-old, and two 11-year-olds at the Lawrence Hall, Berkeley. Bottom row: designs created by Boulder middle school students, and displayed at NOAA Labs in Boulder.

3.3 Current State of the MoS System

The current MoS system is publically available through the website www.mathsphere.org. This version (see also [Eisenberg 2012]) represents a substantially improved and extended successor to an early prototype system (described in [Eisenberg 2010]). In particular, all of the elements shown in Figure 2–the language editor, interactive sphere view, and command interpreter–are new to this version, and the local server has been substantially redesigned. Currently, it should be noted that our MoS system is still a "work-in-progress", with many elements still under construction for future public releases. (There are also, not unexpectedly, occasional bugs in the current version still to be tracked and corrected.)

Our initial pilot tests of the system were conducted in spring 2012 at the Lawrence Hall of Science in Berkeley, California (Figure 7, top row). Two separate workshops were held for elementary- and middle-school-aged children in the San Francisco area, focusing on both spherical geometry and programming with the MoS system (see also [Hsi and Eisenberg, 2012]). Earlier this year, additional pilot tests were conducted with eight middle school students in Boulder; the students had 3 one-hour-long lessons in using the system (spaced over three consecutive weeks), and then created designs that were displayed for them and their parents at the NOAA labs (Figure 7, bottom row).

Our experience to date indicates that students are indeed able to make use of the MoS system for creating and displaying attractive (if mathematically simple) spherical programs.

Our next steps will be to work toward longer-term interactions with students and to use MoS to introduce somewhat more advanced concepts in spherical geometry (conceivably for high school or even undergraduate-level students).

4. TOWARD DEMOCRATIZED PUBLIC DISPLAYS

Our motivation for creating the MoS system described in this paper is really twofold. On the one hand, we are interested in providing an aesthetically appealing medium for introducing non-Euclidean geometry for middle- and high-school students; the sample scenario of the previous section, brief as it necessarily was, indicates the sorts of mathematical issues highlighted by the system. A second reason for creating the system, however, is less specific to mathematics per se, and more aimed at expanding the landscape of display devices (and display settings) available to students.

We can begin this reflection with the subject of "making public displays available to children". Earlier researchers such as Brignull and Rogers [2003] and Peltonen *et al.* [2008] have reported on pioneering work investigating users' activities with large-scale interactive public displays. Our SoS system has particular features that make it distinct from that earlier work: it highlights specific educational (in particular, mathematical) issues, it is replicated in many sites worldwide (suggesting that a user could in principle create a pattern to be displayed at various public sites over time, or simultaneously), and it offers a dimension of programming (as opposed to "direct interaction", as in Brignull and Rogers' "Opinionizer" or Peltonen *et al.*'s "CityWall" projects).

The MoS strategy described here–the strategy for giving students control of public display devices–is really a flexible mixture of "remote" and "present" interaction. Conceivably, students can spend a significant amount of time learning spherical programming on their own, and then (only later) display their ideas publically; or alternatively, one could imagine scenarios in which students explore ideas immediately (as in our workshops), at the museum or planetarium space itself. One might, of course, repurpose this fundamental strategy for other public displays. For example, one might create a "public opinion" space, analogous to the "Speaker's Corner" at London's Hyde Park, through which students (e.g., for the purposes of debate or political participation) can project text, graphics, and video on high-resolution screens in large public spaces; or students might be given (temporary) control over a scoreboard display at a sporting event. Naturally, there is a sociological and political dimension to such a suggestion: students might elect to project (e.g.) controversial or even offensive content through such a system. The issues raised by a public display of this sort touch upon fundamental issues of privacy and free speech. Indeed, it should be noted that even in our own MoS system, the capability exists for a user to create (e.g.) turtle-drawn text messages to display on the giant sphere; and conceivably, a diligent user could employ the system to project words on the sphere that other museum-goers might not wish to read. Once programmable public displays are made available to children (and to the public), that decision is attended by all the benefits and risks of free and open speech in other settings as well.

Of course, there are many more examples of this sort of idea that might be tried, not all of them as politically sensitive. For instance, one might return to a planetarium setting, and create an interface through which visiting school groups can project patterns on the interior of a planetarium dome. Art museums might include specialized surfaces of various sorts, perhaps using geometric forms other than the SoS sphere; for instance, one might create an "SoS-like" cube, cylinder, or cone-shaped display surface (just to name a few examples), and allow visitors to create designs for those surfaces through their own "MoS-like" web interface. Such scenarios highlight once more the specifically mathematical aspects of the MoS project.

There are many other technological challenges that could emerge from experiments along these lines. For example, the MoS system is based on a default scenario in which a single programmer creates a design to send to a particular sphere. We might imagine an alternative in which several programmers, working from several distinct web interfaces, each contributes a portion of a spherical design; for instance, one programmer might create a "background" design for the sphere, while others create programs for animated "foreground" figures that move over the background graphics.

Or we might try implementing an MoS system in which users could write programs at the web interface and then broadcast those to one or more distant spherical surfaces; in this variant, it would no longer be necessary to be physically present at the local planetarium in order to run one's program on the giant sphere.

Conceivably, the user in this case might be able to watch the results of his running program via a webcam or similar remote viewing device.[1] Yet another possibility would be to employ lightweight handheld projectors (held by students themselves) that can be overlapped or superimposed to create large-scale displays or animations under distributed control from multiple users. Or–to imagine still another alternative–one might create an MoS-like interface to display graphics on a moving or dynamic surface (such as a hanging mobile).

The intent of these examples is simply to illustrate the ways in which a strategy like that employed in the MoS system can be adapted to all sorts of other public and pervasive displays. The control of such displays can now be distributed; even a complex, large, or expensive display associated with a particular physical location can now be made available for web-based graphics and programming. We look forward to the day when museums and planetariums (among many other institutions) routinely install beautiful, specialized displays that have relatively "open access" for their visitors.

ACKNOWLEDGEMENT

The MoS system was implemented in our lab with the invaluable contributions of Michelle B. Redick and Michael MacFerrin. Many thanks to Michael Biere, Hilary Peddicord and their colleagues at NOAA; thanks also to Hilarie Nickerson for helpful conversations. The work described in this paper was supported in part by a grant from the National Science Foundation, DRL1114388.

REFERENCES

Abelson, H. and diSessa, A. [1980] *Turtle Geometry*. Cambridge, MA: MIT Press.

Brignull, H. and Rogers, Y. [2003] "Enticing People to Interact with Large Public Displays in Public Spaces." In *Proceedings INTERACT 2003*, pp. 17-24.

Eisenberg, M. [2010] "Computational Diversions: Turtle Really and Truly Escapes the Plane". *International Journal of Computers in Mathematical Learning*, 15: 73-79.

Eisenberg, M. [2012] "Computational Diversions: The Return of the Spherical Turtle." In *Technology, Knowledge, and Learning*, 17:3, pp. 115-122.

Hsi, S. and Eisenberg, M. [2012] "Math on a Sphere: Using Public Displays to Support Children's Creativity and Computational Thinking on 3D Surfaces." In *Proceedings of Interaction Design and Children* (IDC 2012), 248-251.

NOAA website for SoS: http://sos.noaa.gov/index.html

Papert, S. [1980] *Mindstorms*. NY: Basic Books.

Peltonen, P. *et al.* [2008] "'It's Mine, Don't Touch!' Interactions at a Large Multi-Touch Display in a City Centre." In *CHI 2008 Proceedings*, pp. 1285-94.

Science on a Sphere projection requirements: http://sos.noaa.gov/docs/faq.html

[1] In fact, we recently implemented a "pilot-testing showcase event" (see www.mathsphere.org) to illustrate how users can send in MoS designs to be displayed at the Lawrence Hall sphere; a sent-in design could then be viewed remotely, live, over the Web. Future implementations of events of this sort are in the planning stage.