4. JavaGami

4.1 Description

JavaGami is a software environment implemented by the author to address some of the design issues discovered when children worked with HyperGami. The primary objective behind the development of JavaGami was to provide a less complicated environment in which to study children's spatial thinking as they designed and created paper sculpture. JavaGami was also intended to be more accessible for classroom teachers, parents, and children who wish to use the software on their own.

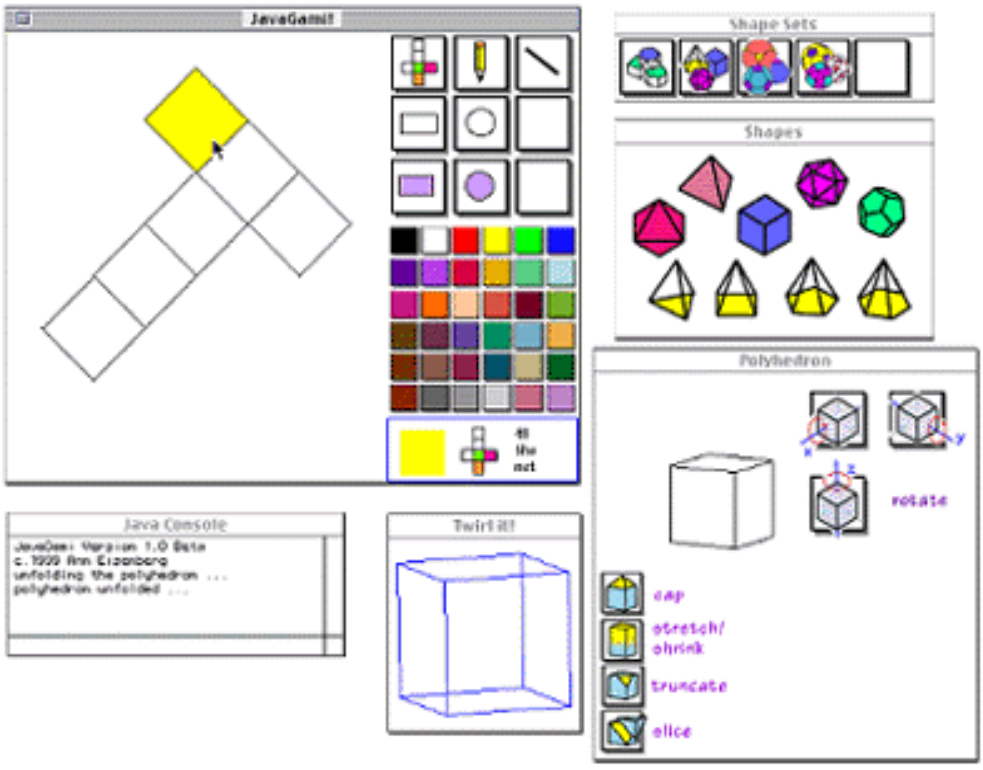

Figure 4-1. Overview of the JavaGami system. The JavaGami window at upper left displays the current folding net and tools to decorate the net. The Polyhedron window (lower right) shows the current solid object, buttons to rotate the object, and function buttons (at the lower left of the window) to cap, stretch, truncate, or slice the solid. The Shape Sets window at upper right is used to load a set of starting shapes into the Shapes window below it; the Shapes window is then used to load a single polyhedron; and the Twirl It! window contains a draggable wireframe version of the solid object. The Java Console at lower left is a read-only display for error messages.

A Sample Interaction with JavaGami

A student working with JavaGami first selects a set of polyhedra from the Shape Sets window. The polyhedra in the set will appear in the Shapes window. After clicking on one of the polyhedra, the student will see a solid rendering of the polyhedron in the Polyhedron window, a wireframe rendering of the polyhedron in the Twirl It! window, and a folding net for the polyhedron in the main JavaGami window.

The student can now engage in various activities. She may decorate the folding net with a variety of colors and designs using the decoration tools such as polygon-fills, drawing, and shape creation tools for lines, rectangles, and circles. If she is satisfied with the folding net, she may print it to a color printer and assemble it into the corresponding paper model.

Although the decoration tools built into the current version of the system are still relatively basic, students have been able to successfully employ them to paint intricate designs on folding nets. Figure 4-2 below is a net for a Martian cube designed by a 9 year-old girl.

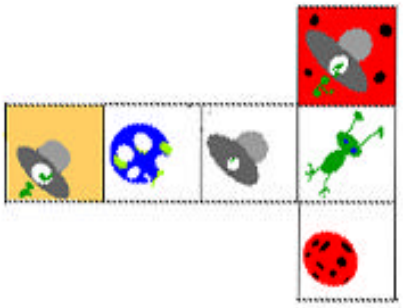

Figure 4-2. A cube net with a Martian motif designed in JavaGami by a 9 year-old girl.

Once the student has loaded a starting shape, she may apply one of four basic functions to create a new polyhedron based on the one she has selected. All solid operations take place in the Polyhedron window as shown in Figure 4-3 below. The x, y, and z-rotation buttons allow the student to turn the solid object so that all parts of the polyhedron are accessible.

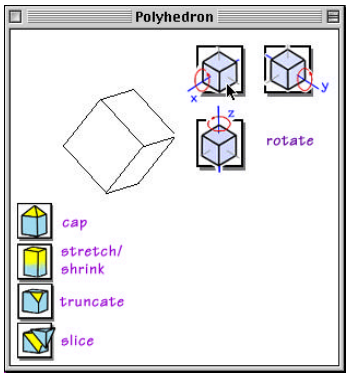

Figure 4-3. Clicking to rotate the polyhedron about the x-axis.

Clicking any of the function buttons in the lower left quadrant will produce an interactive panel in the lower right quadrant of the window. As shown in Figure 4-5 at the end of this section, the student may click on the cap button to add a pyramid to a face of a cube; she may then truncate the pyramidal cap at its tip; she may stretch the capped-cube along an axis; and then she may slice the resulting solid into two parts by selecting three vertices to determine a slicing plane. At each step of the process, she will see (and have the option to decorate) the folding net for the shapes in the JavaGami window.

The Twirl It! window shown in Figure 4-4 below allows the student to see a wireframe rendering of the current polyhedron. By dragging her mouse over the window, the student can twirl the wireframe in real-time, allowing her to easily see a transparent view of the polyhedron from all angles.

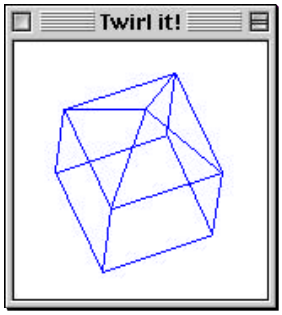

Figure 4-4. A wireframe version of a capped cube. The wireframe object can be rotated by dragging the mouse over the window.

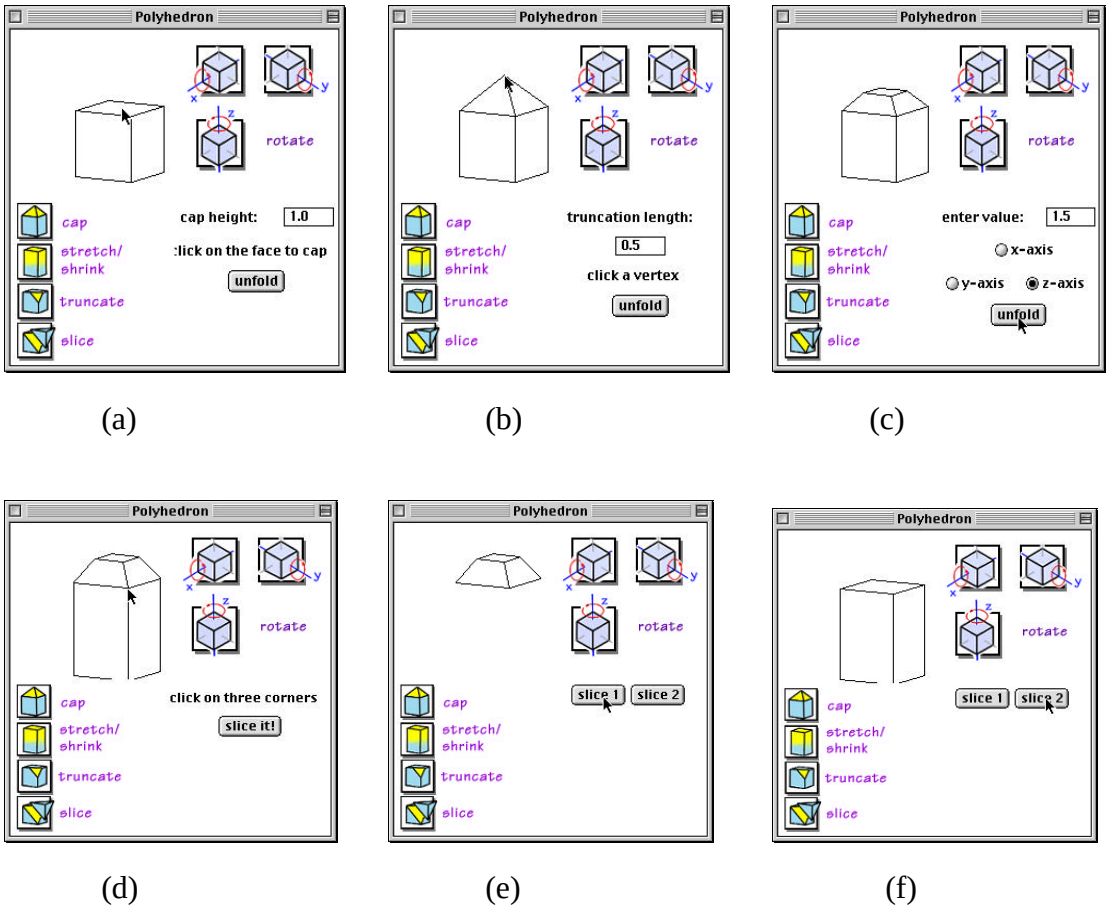

Figure 4-5. Applying successive functions to a cube in JavaGami. The cube in (a) is capped; the capped cube (b) is truncated at a vertex; the resulting solid (c) is then stretched along the z-axis; and the resulting shape (d) is sliced into two parts: (e) and (f).

4.2 Implementation of JavaGami

JavaGami is written in Java, using the Sun JDK 1.2 compiler within Symantec's Visual Cafe 2.0 development environment (Symantec). The software runs on Macintosh Power PCs with OS 8.0 or higher, using Apple Computer's Macintosh Runtime Environment for Java, version 2.0 or higher. The classes of JavaGami are deployed as a JAR object, and JavaGami itself runs as an application on the user's local machine. All of the classes -- geometric computation, graphics engines, and user interface elements -- are designed and written by the author, with the exception of the

wireframe renderer which is based on public domain (Sun, 1997) source code. Algorithms are based on those in Bowyer and Woodwark (1983), Foley et al. (1990), and Chan et al. (1997). For a description of early prototype work in JavaGami, see Nishioka and Eisenberg (1997).

The primary computational work in JavaGami takes place in the *unfolding algorithm*. Solid polyhedra are represented as vectors of vertex coordinates and face indices. Given a polyhedron object, the unfolding algorithm will attempt to calculate the corresponding folding net by depth-first search.

At the outset of the unfolding algorithm, an initial face of the polyhedron is selected, a transformation map is applied, and the first face is placed on the xy-plane with the center of the polygon located at the origin. The next face to place is selected from among the *eligible neighbors* of the first face -- i.e., those faces which (a) share an edge with the first face, but (b) have not already been placed. The selected neighbor face is placed onto the plane, aligned with the appropriate edge of the first polygon. If there is no overlap with existing polygons on the plane, the face is set; its neighbors join the list of eligible neighbors; and the algorithm attempts to attach an eligible neighbor of this polygon to the net. If a conflict occurs, the algorithm tries a different eligible neighbor of the original face to be expanded. This process continues until the shape is unfolded or until all eligible neighbors are exhausted. The folding net for the polyhedron is then drawn in the JavaGami window.

The unfolding algorithm may be implemented using a variety of search strategies including breadth-first search, a combination of depth-first and breadth-first searches, or with other user-defined search strategies. Each variation of search strategy produces different characteristics in the configuration of the folding net polygons. For a more detailed discussion of the unfolding algorithm, refer to Eisenberg (1996).

In order to create a more pictorial interface than is usually supported by Java through the standard interface toolkit, button and picker components were designed

graphically and then were rendered as gifs by superclasses of awt.Canvas objects. Mouse handlers were written for the layered gif objects so that they could act upon mouse clicks. The folding net canvas is implemented as a paintable canvas object with polygonal fills, scribbles, and geometric shape drawing.

Capping, truncation, slicing, and stretching are implemented by applying standard computational geometry calculations to the solid object and then calling the unfolding algorithm to generate the new folding net for the resulting shape.

4.3 Addressing Design Issues in HyperGami

JavaGami was built to address specific usability issues in HyperGami including those discussed in Section 3.3 of the previous chapter. JavaGami was designed with the primary goal of making it more accessible to a wider audience of children, teachers, and polyhedron model enthusiasts. In some cases -- such as decreasing the number of windows in the application -- the design decisions were straightforward. But in other interface decisions -- especially in the elimination of end-user programmability -- there were design tradeoffs made which will be discussed below.

4.3.1 No programming language

JavaGami is not a programmable application. All polyhedron functions take place by directly clicking buttons and entering numerical values for heights or sizes. Rather than working back and forth between a transcript window and the polyhedron, as in HyperGami, the student works with a window which prompts him to enter values or to click on vertices or faces of the polyhedron. This design decision limits the kind of shapes that the student can create with JavaGami--he is limited to a combination of capping, stretching, slicing, and truncation--but it greatly simplifies the set of steps that students must carry out to customize their shapes.

The issues around programmability in applications are not new ones (cf. Nardi, 1993; DiGiano, 1996; Eisenberg 1991) -- end-user programmability can be an indispensable tool for customizing applications by users and (perhaps more interestingly) a means of exploring new notations and formalisms for particular domains. Nonetheless, the guiding design principle of JavaGami was to have an easily accessible interface for a wide range of ages and ability levels, and as a result the language interface was eliminated.

4.3.2 Pre-scaling of folding nets

In response to the difficulty in setting coordinate values to display folding nets that students experienced in HyperGami (refer to section 3.3.2), folding nets in JavaGami are internally calculated to always appear completely contained within the folding net window. This eliminates the need for the student to set a view coordinate system.

This design decision, like the previous one, is not without tradeoffs: the student using HyperGami has more control over the scale at which to display his folding net. By setting his own viewing coordinates, he may zoom into or out of various portions of the net. However, the frustration exhibited by the children using HyperGami's coordinate view system led to this decision. Subsequent versions of JavaGami may incorporate some user-defined coordinates, with the default set to scale the net to the viewing window, along with zoom-in/zoom-out functions which display the view window at different scales.

4.3.3 Fewer windows/No menu commands

JavaGami interface uses fewer windows than the HyperGami system. This design decision attempts to eliminate some of the confusion shown by students working among HyperGami's various windows, as described in Section 3.3.3.

Operations on the solid object all take place within one window, operations on the net take place in a second; the remaining three windows are for loading shape sets, selecting shapes, and for display of the wireframe image.

An early version of JavaGami made use of menu toggles to access the different sets of shapes. Testing with elementary and middle school students showed that menu toggles were confusing to the students: they did not remember what to do (or where to go in the interface) to make the sets of shapes appear and disappear. As a result, the current version of the software uses a window showing the available shape sets, with no menu bar commands.

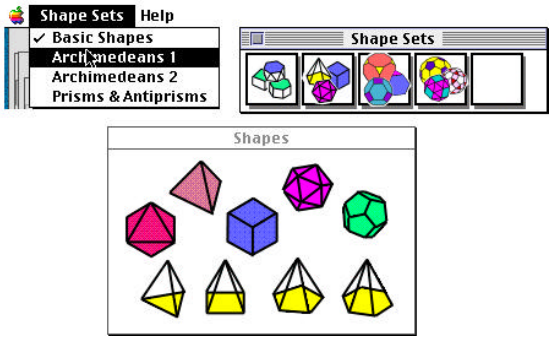

Figure 4-6. Instead of menu toggles (left) to access sets of shapes, the current version of JavaGami has a Shape Sets picker bar (upper right). The set of shapes is loaded into the Shapes window at the bottom.

4.3.4 Faster net generation time

Net generation in JavaGami is considerably faster than net generation in HyperGami. This is primarily due to the change in implementation from Scheme to Java, as well as running JavaGami on higher-end Power PC platforms. JavaGami is limited to higher end Power PCs because these are the only platforms which support the required operating system and Java run-time environment on the Macintosh for this implementation; while the MacScheme system in which HyperGami is implemented has never been written to compile down to native Power PC machines, and thus must be interpreted from 68030 instructions. Finally, the HyperGami unfolding algorithm is

somewhat more complex (and featureful) than the JavaGami version, allowing (e.g.) for user-programmable settings for the underlying search algorithm, and these features presumably make the HyperGami unfolding process slower (though the quantitative effects of these features have not been explored).

4.3.5 Less clicking for paint tools

JavaGami allows students to access paint tools with a single mouse click. Early versions of the software used a highlight indicator to show the active tool, but this often caused confusion because the highlights tended to lag behind children's mouse actions. The current version of JavaGami uses an indicator panel which shows the active tool and color, eliminating the need for object highlighting.

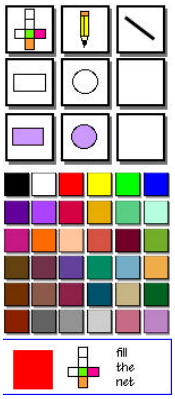

Figure 4-7. The indicator panel at the bottom displays the current color and tool selected.

4.3.6 No solid coloring

Students can only color the folding net object in JavaGami -- colors cannot be placed by the students on the three-dimensional drawings. This was done to eliminate the confusion that students sometimes exhibited by decorating the solid object instead of the folding net (see 3.3.6).

JavaGami does not support mapping of painted decorations between the folding net and the solid object. This feature is available to a limited extent in HyperGami: students may choose to have the system map their folding net decorations onto the solid object so that they may have a preview of what the folded shape might look like. The reverse -- mapping decorations from the solid object to the folding net -- is not implemented in either system.

4.3.7 Addition of a wireframe window

JavaGami supports a wireframe shape browser as discussed in Section 4.1 above. Although HyperGami allows students to rotate shapes by either clicking on buttons or by entering rotation values in a dialog box, it does not support figures which can be dragged by mouse alone. The wireframe browser is included in JavaGami because it provides an elegant way for children to view polyhedra from different angles; the transparent wireframe view also provides children with the opportunity to observe the overall geometry of the shape. For example in Figure 4-8 below, it is easy to note that the top and bottom pentagons in a dodecahedron are rotated relative to one another, something that would not be as readily apparent in a solid rendering. These kinds of explorations may serve as a springboard for activities in orientation and symmetry of shapes.

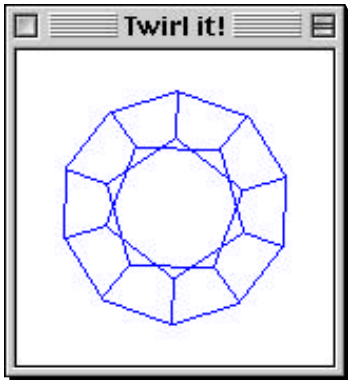

Figure 4-8. A dodecahedron in the wireframe window illustrating the orientation of opposite pentagons relative to one another.

4.4 Use of JavaGami

Figure 4-9. A modular sculpture composed of cuboctahedra and antiprisms designed in JavaGami by a 5th-grade boy.

4.4.1 Students

Seven students (3 girls and 4 boys) in grades 5 to 8 took part in a small scale pilot test of JavaGami in Fall 1998. The students were recruited by community announcements and were self-selected for interest in working with software and paper sculpture.

After an initial pre-test session of spatial thinking activities (which is discussed in detail in the next chapter), students spent an average of 5 one-hour sessions working with the software and then participated in a post-test session. Five of the students worked individually; one pair of sisters attended sessions together with the exception of the pre- and post-tests. In addition to design lessons learned in HyperGami, JavaGami also evolved in response to the feedback from these pilot students. A detailed case study of one of these students is described in Chapter 6.

(a) (b)

Figure 4-10. (a) A giant-sized pencil designed by a 10 year-old boy; (b) An ice cream cone designed by a 9 year-old girl.

The JavaGami sessions were conducted in much the same fashion as the HyperGami pilot. The sessions were student-directed, with students choosing the format of their projects. One of the participants chose to spend all of his sessions working on a single project. The result is the multi-part cuboctahedron sculpture shown in Figure 4-9 above. The other children worked on shorter-term projects which took one or two sessions to complete. Examples of two of these projects are shown in Figure 4-10 above.

4.4.2 General Distribution

Since its initial release date of February 1, 1998, JavaGami has been distributed to over 50 users over the World Wide Web. The user community includes student teachers, classroom mathematics teachers in Canada, Australia, and the U.K., as well as professional artists and scientists. JavaGami will also be published on CD-ROM as part of a text edited by Roberts, et al. (1999).

The software has been used in human-computer interaction courses as well as teacher-education classes. Feedback from users has been generally positive: users have been able to install and run JavaGami with relatively little trouble, and have produced sculptures with it. One student teacher downloaded the software and created

paper creations to share with her teacher-education class. The excerpt below is part of a message from her instructor:

> One of my students, Christy ... came to see me all excited about JavaGami. This summer I will have about 40 elementary and middle school teachers on campus looking at topics such as ... probability, statistics, and geometry. I am constantly looking for activities and this looks like it will fit perfectly with the geometry that I want the students to do. I hope that you will allow me ... to have it loaded onto our campus server so that I can take my teachers to our computer lab and let them make some of the neat objects that Christy showed me. This will fit well with other investigations that I have them do on computers such as geogolfer, tesselmania, and cabrii.

We have also received reports that students have been successful in running the software without parental guidance.