GeoSVG: A WEB-BASED INTERACTIVE PLANE GEOMETRY SYSTEM FOR MATHEMATICS EDUCATION

Xun Lai and Paul Wang
Department of Computer Science
Kent State University
USA
xlai@cs.kent.edu
pwang@cs.kent.edu

ABSTRACT
Geometry is an important part of mathematics. GeoSVG is a Web-based system, written in Scalable Vector Graphics (SVG), for teaching and learning geometry and mathematics. The work is part of the WME project for enhancing mathematics education using Web technologies. GeoSVG is a totally Web-based environment for creating interactive plane geometry manipulatives to illustrate concepts, to provide hands-on experience, and to support homework and tests. GeoSVG-created manipulatives are interoperable and interact with enclosing Web pages via standard Javascript. GeoSite provides instant Web publishing of manipulatives and pages authored using GeoSVG. GeoSVG and GeoSite combine to form a complete Web-based system for creating plane geometry manipulatives and for teaching and learning with them. It offers significant advantages over traditional interactive geometry software.

KEY WORDS
Interactive Plane Geometry, Mathematics Education, Authoring Manipulatives, Learning, SVG, Web.

1. Introduction

Hands-on experimentation is one of the best ways to learn mathematics. The use of *physical manipulatives* is possible but less efficient than employing *virtual manipulatives* supplied by computer programs.

Since 1980, *Dynamic Geometry Software* (DGS) packages such as *Cabri Geometry II* [1], *Geometer's SketchPad* [2][3], *Cinderella* [4][5], *Euclides* [6], *C.a.R.*[7] have been widely used in schools and colleges all over the world. A primary function of these DGS packages is to generate *geometry manipulatives*.

DGS packages normally provide both features for authoring manipulatives (Section 3.1) and features for running them for hands-on learning.

Traditional DGS packages are stand-alone applications requiring installation and update on each computer that uses the software. Manipulatives created under such a system run well only within that particular system. These manipulatives cannot be placed in Web pages where the bulk of the teaching materials must be placed, unless they are exported as Java applets. Then they become slow to load and plug-in dependent, not to mention that they won't work nearly as well as within the DGS package. For example, Geometer SketchPad fails to include many objects into the exported applet [8]. Cinderella improves on the situation, but still misses important features such as listing the construction steps. The entire C.a.R. system can run as an applet. But digital signature is required if you want to save a construction.

Another problem with traditional systems is sharing, or reusing by others, the manipulatives created. Embedding manipulatives as applets into HTML pages does not provide much sharing because an applet cannot be easily modified by others.

A third problem is that applets do not interact well with their enclosing Web pages. They can't easily take advantage of the *Documet Object Model* (DOM) and Javascript for dynamic interactions. Nor can they provide output data for question answering or be interoperable with other manipulatives in the same page.

1.1 The GeoSVG Approach

Ideally, we want

- *A DGS runs on the Web via a browser without requiring a plug-in, or offline after installation on the local workstation if one prefers.*
- *A generated manipulative can be embeded in a Web page directly and the manipulative can include as much or as little of the DGS features as needed (the author decides).*
- *Manipulatives are in standard formats making them easily shared, modified, and reused.*

GeoSVG and *GeoSite* use a new approach to achieve these goals. GeoSVG is designed as a completely Web-based DGS. The plane geometry engine, drawing, and animation are implemented in SVG [9] and manipulatives depends only on SVG and Javascript. Mathematical formulas are represented by MathML [10].

The Graphical User Interface (GUI) provides easy access to all the features under the *authoring mode* while supplying a much simplified interface under the l*earning* *mode*. Figure 1 is a snapshot of the GeoSVG authoring environment running in a Firefox browser.

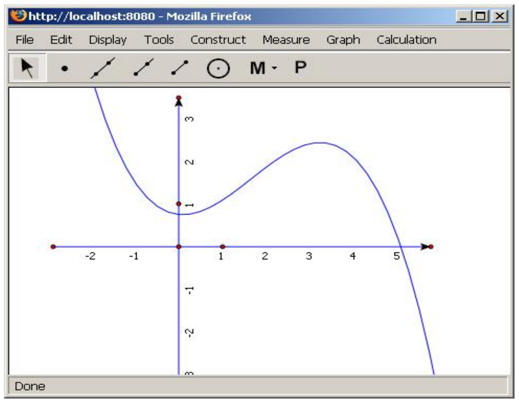

http://localhost:8080 - Mozilla Firefox
File Edit Display Tools Construct Measure Graph Calculation
M P
-2 -1 1 2 3 4 5
3 2 1 -1 -2
Done

Figure 1: GeoSVG authoring environment

An author runs GeoSVG by accessing a URL with an appropriate Web browser such as Firefox 1.5. The constructed manipulatives are stored automatically on the GeoSite. Other users can immediately access the newly constructed manipulative to embed it in a teaching Web page, to build upon it, or to learn directly from it. The manipulative creator can decide what authoring mode features will remain in the manipulative. The author can also define input and output interfaces of a manipulative. The input interface allows other parts of a page, either an input box or the output of another manipulative, to update the current manipulative. The output interface allows other parts of a page to use results provided by the manipulative as the user interacts with it. One common use of such results is to help verify answers from the user to questions on the page. All manipulatives created by authors are saved on the GeoSite under different user folders. A search function helps people locate manipulatives for different purposes.

SVG-based manipulatives are easily included in educational pages anywhere on the Web. The WME pilot site [11][12][13] deploys many manipulatives generated by GeoSVG.

2. Architecture of GeoSVG

GeoSVG is a Web-based system. Users need just a Web browser in order to author manipulatives or employ them for teaching and learning. GeoSVG has two major components.

- The GeoSVG toolkit:
    a. An SVG-coded *Plane Geometry Engine* for authoring and viewing manipulatives (creating, moving, and animating geometric objects).
    b. GUI for the authoring environment providing authoring logic, a variety of dialogs assisting authoring, publishing, and communications with the server side.
- The GeoSite (http://wme.cs.kent.edu/geosite): a Web site that makes the GeoSVG toolkit available as well as stores manipulatives for access, searching, and sharing.

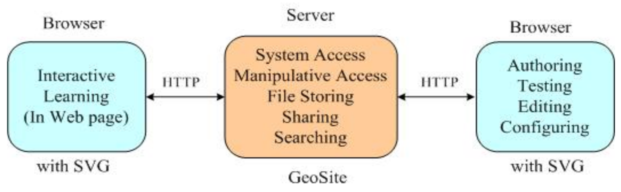
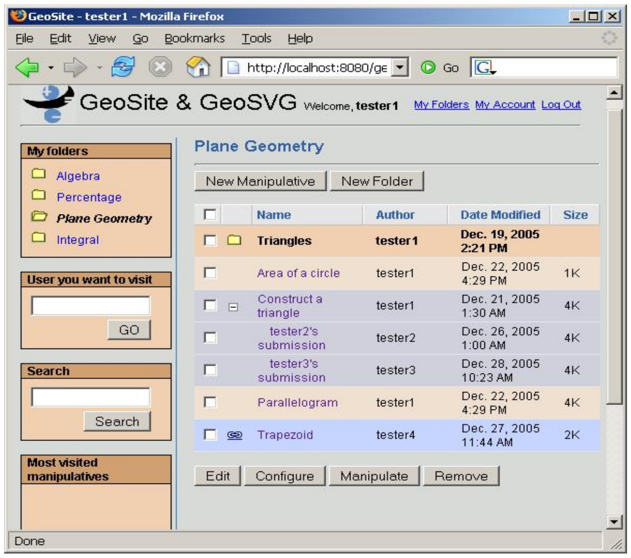


Figure 2: GeoSVG architecture and components

Figure 2 shows the relation between the client side and the server side. A GeoSite provides the functions for GeoSVG system access, manipulatives access, file storing, manipulative sharing and searching, and management of user accounts.

With a Web browser supporting both the GUI and SVG (currently Firefox 1.5 is the best choice), a user can author, test, configure, and publish manipulatives. Communication with GeoSite is via HTTP (Figure 2 right).

Users who only do interactive learning (Figure 2 left) only need a browser supporting SVG, either by the browser's native support or the Adobe SVG Viewer plugin. Users can access manipulatives by visiting GeoSite directly or through a Web page that links, via <embed>, to a manipulative stored on a GeoSite or any other website. WME lesson pages are examples that use the latter technique which makes GeoSite completely transparent to visitors.


GeoSite - tester1 - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/ge
Go
GeoSite & GeoSVG
My Folders My Account Log Out
My folders
Algebra
Percentage
Plane Geometry
Integral
User you want to visit
GO
Search
Search
Most visited manipulatives
New Manipulative
New Folder
Edit Configure Manipulate Remove
Done

Welcome, **tester1**

Plane Geometry

Name	Author	Date Modified	Size
Triangles	**tester1**	**Dec. 19, 2005 2:21 PM**	
Area of a circle	tester1	Dec. 22, 2005 4:29 PM	1K
Construct a triangle	tester1	Dec. 21, 2005 1:30 AM	4K
tester2's submission	tester2	Dec. 26, 2005 1:00 AM	4K
tester3's submission	tester3	Dec. 28, 2005 10:23 AM	4K
Parallelogram	tester1	Dec. 22, 2005 4:29 PM	4K
Trapezoid	tester4	Dec. 27, 2005 11:44 AM	2K

Figure 3: GeoSite – directory for user "tester1"

Figure 3 shows the directory "Plane Geometry" under GeoSite registered user "tester1". In the directory there are a sub-directory "Triangles", and a few manipulatives with different features. The "Construct a triangle" manipulative is a submittable manipulative (section 3.3) with submissions following it. The "Trapezoid" entry holds a link to a manipulative physically located under another user "Tester4". The others are just ordinary manipulatives.

On the left side, there are a few functions provided to the users. A user can type in another user's account name to visit manipulatives under his/her account. Search function is provided (section 3.3). The GeoSite also keeps track of the most visited manipulatives and macros across all the users, and provides links to them.

3. Features of GeoSVG

Our purpose is to make GeoSVG into a modern, powerful, efficient, easy-to-use, Web-oriented, and standard-based tool for creating geometry manipulatives.

3.1 Basic Authoring Support

Here is a list of user-level features any reasonable DGS should offer.
Drawing primitives: Creating basic geometric shapes such as points, lines (segments, rays and vectors), circles (ellipses and arcs), polygons, conics, etc.
Geometric object construction: Constructing a new geometric object subject to mathematical relations with existing objects. For example, creating a line parellel to an existing line and through an existing point.
Measurement: Measuring length, slope, radius, distance, area, circumference, perimeter, angle, and coordinates.
Loci and Envelops: Constructing loci of moving points and envelops of moving lines.
Animation: Moving and changing objects to illustrate and to demonstrate.
Iteration: Repeated execution of user commands.
Calculation: Creating and evaluating mathematcial expressions based on existing measurements.
Graphing: Plotting points and function graphs in coordinate systems.
Geometric transforms: Translation, reflection, dilation, and rotation of objects.
Defining Macros: Grouping steps into one command.
Defining GUI Operations: Creating a variety of buttons, user inputs, and tables in a manipulative.

Currently, GeoSVG offers most of these features. When completed, it will support all of them. Due to its education orientation, GeoSVG won't offer advanced features such as support for non-Euclidean geometries [5] available in Cinderella.

3.2 Total Web Orientation

Table 1 compares GeoSVG with traditional systems in terms of Web orientation.

3.3 Manipulative Enhancement by the Web

Except Java applets, traditional DGS systems do not utilize the Web to any great extent. GeoSVG on the other hand supports a number of Web related features to make manipulatives more functional.

Flexible authoring support in a manipulative

	Traditional DGS System	**GeoSVG**
Software installation	Per Computer installation required	Use through browser, no installation required for authoring or learning
Manipulative sharing	Difficult because manipulatives are stored on indivi-dual computers	Easy because manipulatives are stored and searchable on the Web
Publishing manipulatives	Authors need to include Java applets in Web pages which are then deployed on servers	Saving a manipulative automatically publishes it on the Web
Download speed	Applets are binary, large and slow to download	Files are textual, smaller and can be compressed for fast download
Open Standards	Use proprietary technologies	Use W3C standard Web technologies
Interoperable with the enclosing page	No	Can be driven by data outside, and output data

Table 1: Comparing GeoSVG to Traditonal DGS

As stated, a manipulative may contain none or all of the authoring support features. Figure 4 shows the GUI for choosing additional features to include in a manipulative.

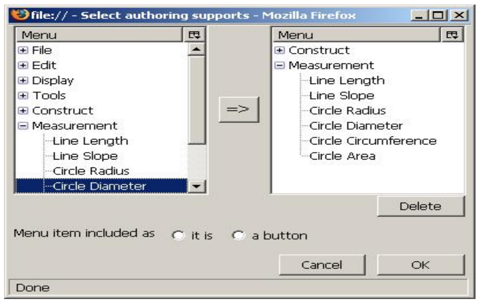


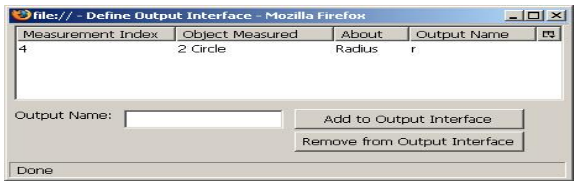
Figure 4: Add authoring supports in manipulative

Input and output interface of a manipulative

Input interface can make sources outside to change a manipulative. Sources can be input boxes or output from another manipulative.

Output interface defines what measurements in a manipu-lative can be used outside. The manipulative can actively update the outside part that uses the measurements or passively wait outside part to retrieve the measurements.
A manipulative together with its input/output interfaces forms a self-contained unit that can be embedded in any Web pages, not restricted to pages on GeoSite. WME lesson pages are such examples.

Figure 5 shows the GUI for an author to define the manipulative's output interface. Here there is only one measurement, which measures the radius of the circle. The author gives this measurement an output name that will be used later.


file:// - Define Output Interface - Mozilla Firefox
Output Name:
Add to Output Interface
Remove from Output Interface
Done

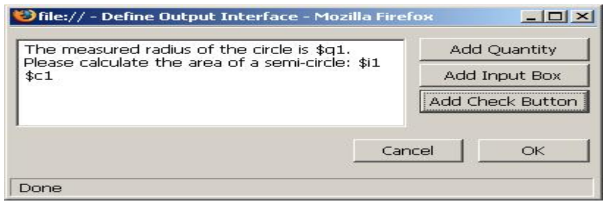
Measurement Index	Object Measured	About	Output Name
4	2 Circle	Radius	r

Figure 5: Define output interface

For programmers who want to embed manipulatives generated by GeoSVG in their own Web pages, such as the WME pages, input/output interfaces are what they can access about the manipulative. A library called GDrawing can simplify programmers' job to access the input/output interfaces.

Question composition with answer checking

As a user of the GeoSite, access to the input/output interface is much easier. GeoSite and GeoSVG provide several tools for composing questions and other tasks that need to interact with the manipulative.

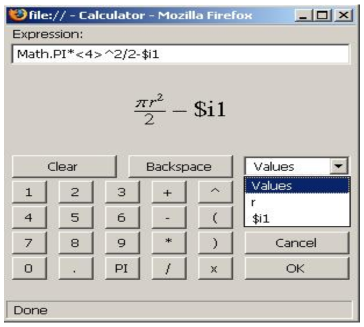


The measured radius of the circle is $q1.
Please calculate the area of a semi-circle: $i1
$c1

Figure 6: Question composition

Figure 6 shows the GUI for composing a question. Both the "Add Quantity" and "Add Check Button" will invoke the *dynamic calculator* to create expressions in terms of existing measurements from the output interface and user inputs from the question.

Syntax `$q1`, `$i1`, and `$c1` respectively refer to a quantity created by the dynamic calculator, an input box in the question, and a check button that checks if the answer in the input box is correct.



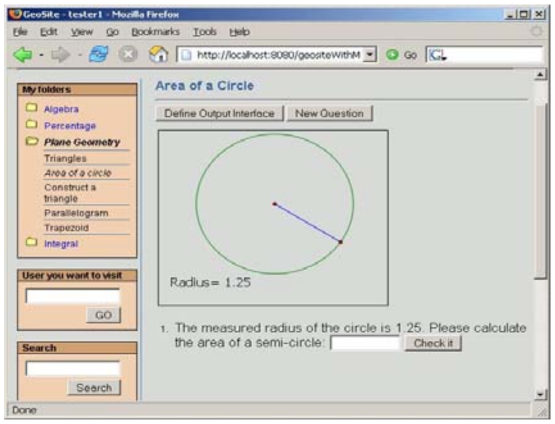
Math.PI*<4>^2/2-$i1

$$\frac{\pi r^2}{2} - \$i1$$

Figure 7: Dynamic calculator

Figure 7 shows the *dynamic calculator*. It's called dynamic because it always maintains the mathematical relations even if a user has moved around objects in the manipulative. In the pull down list of "Values", "r" is defined in the output interface of the manipulative; "$i1" refers to the input box in the question.

Notice here the mathematical expression is rendered in MathML. The calculator is not only used here, but also used in the authoring environment to create new calculation among measurements, and new functions.



1. The measured radius of the circle is 1.25. Please calculate the area of a semi-circle:

Figure 8: Manipulative with question

Figure 8 shows the question appended after the manipulative. The quantity represented by $q1 whose current value is 1.25 will change whenever a user manipulates the circle. The "Check it" button represented by $c1 uses the formula defined by the dynamic calculator to check if the input is correct. Neither of these needs any programming effort by the author of the manipulative.

Submittable manipulative

A submittable manipulative has a button for users to submit results after interacting with the manipulative. Figure 3 shows the "Plane Geometry" folder has a submittable manipulative named "Construct a triangle", in which the author has included the point and line drawing supports. Two results submitted are also shown.

Keywords and search

GeoSite provides search function for manipulatives across the site. Authors can assign keywords to a manipulative. Search is based on the keywords, and the text inside the SVG files.

4. Usage Scenarios

Authoring a manipulative

Let's consider a concrete example: teaching the relation between the area of a triangle and that of a parallelogram. Figure 9 shows a Web page containing teaching materials and the completed manipulative under item 1. The steps for creating, publishing, and using the manipulative are:

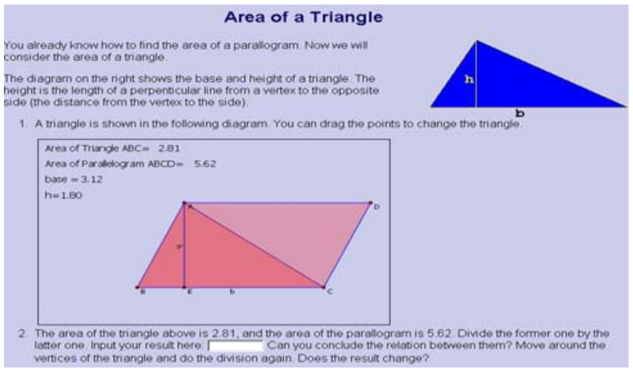
- Log on to GeoSite as, tester1, say.
- Go to the directory in which you want to create the manipulative, say Triangles. Click New Manipulative to begin authoring.
  1. Use the point and segment drawing tools to draw the vertices A, B, C, and sides of the triangle.

2. With three vertices selected, use the Construct→ Polygon menu to make a color-filled triangle.
3. With A, B, C selected in order, choose from the menu Construct→Parallelogram Point to construct the fourth point D of the parallelogram, which shares three vertices with the triangle. Then use Construct→Polygon to make a parallelogram.
4. With point A and side BC selected, do Construct→Perpendicular Point E and draw the height AE with the segment drawing tool.
5. Select the triangle and parallelogram. Use Measure→Polygon Area to display their area measurements.
6. Select BC and AE then use Measure→Line Length to display the line measurements.

- Open the object property dialog to label the geometric objects and set which are manipulable.
- Use the canvas property dialog to configure the dimensions of the manipulative canvas, 450x300 say.
- By File→Save, save the manipulative as "Triangle Area", to GeoSite. Other users can immediately access this manipulative by going to tester1's Triangles directory.
- We can make any enhancement to the manipulative as described in section 3.3. Here we put four measurements into the output interface.
- Web pages anywhere can embed this manipulative by

```
<embed width="450" height="300"
          src="http://GeoSite-server-name
              /tester1/Triangles/Triangle%20Area.svg">
</embed>
```

In the page, there is a question that uses the output data of the manipulative. Learners should be able to observe that the area of the triangle is always one half of the area of the parallelogram.



Area of a Triangle

You already know how to find the area of a parallogram. Now we will consider the area of a triangle.

The diagram on the right shows the base and height of a triangle. The height is the length of a perpenticular line from a vertex to the opposite side (the distance from the vertex to the side).

1. A triangle is shown in the following diagram. You can drag the points to change the triangle.

Area of Triangle ABC= 2.81
Area of Parallelogram ABCD= 5.62
base = 3.12
h=1.80

2. The area of the triangle above is 2.81, and the area of the parallogram is 5.62. Divide the former one by the latter one. Input your result here: ____ Can you conclude the relation between them? Move around the vertices of the triangle and do the division again. Does the result change?

Figure 9: A Sample WME Page

Learning from a manipulative

A student uses a Web browser to access the instructional materials online and interacts with the manipulative for hands-on experimentation. Questions connected to values from the manipulative tests learner comprehension. For a submittable manipulative, the manipulation result can be collected for teacher review.

5. Implementation Overview

As stated in section 2, the two major parts of the GeoSVG toolkit are the *Geometry Engine* and the GUI. The Geometry Engine is an SVG and Javascript based library, and occupies a rectangular canvas area on the screen. A manipulative in either authoring mode or learning mode needs the engine for rendering and interaction. The GUI consists of the main menu and toolbar, and a variety of dialogs. Both the engine and the GUI are easily extensible to add new features and functionalities.

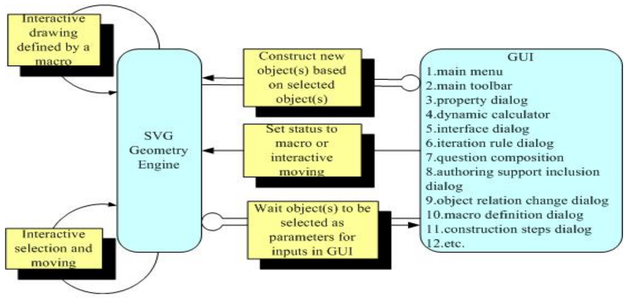

Interactive drawing defined by a macro
SVG Geometry Engine
Construct new object(s) based on selected object(s)
Set status to macro or interactive moving
Wait object(s) to be selected as parameters for inputs in GUI
Interactive selection and moving
GUI
1.main menu
2.main toolbar
3.property dialog
4.dynamic calculator
5.interface dialog
6.iteration rule dialog
7.question composition
8.authoring support inclusion dialog
9.object relation change dialog
10.macro definition dialog
11.construction steps dialog
12.etc.

Figure 10: Geometry Engine and GUI

Figure 10 shows the interaction between the Geometry Engine and the GUI.

- Drawing and interactive selection and moving take place within the engine, without involving the GUI.
- Construction of a new geometric object is done by choosing menu items or toolbar buttons, possibly composing existing objects.
- For some operations a user needs to pick an existing object to obtain input values. For instance, with the dynamic calculator opened, a user can pick a measurement object on the canvas to be inserted as a parameter of an expression to be computed.

5.1 Geometry Engine Implementation

The geometry engine is the core of GeoSVG. It is implemented as an SVG file around 80KB gziped for Web delivery containing SVG and Javascript code. When displayed, it appears as the drawing canvas to the user.

The logic control part of the engine is written in Javascript and the drawing part, including all geometric objects, is in SVG. Animation uses a combination of SVG and Javascript. Object-oriented design and implementation techniques help make the programs well-organized and easily extensible.

SVG is critical and it brings these important advantages:

- An open W3C standard and an XML application with wide acceptance and readily available (free) browser support
- Compact vector notation and easy scalability
- Fast loading and easy textual code generation
- Sophisticated coordinate transformation and animation support

Displaying SVG requires a browser that handles SVG either natively, such as Firefox 1.5, or via a plug-in such as the Adobe SVG Viewer. The Firefox native SVG support provides much better communication between SVG objects and the enclosing Web page, an ability we require for the authoring environment of GeoSVG.

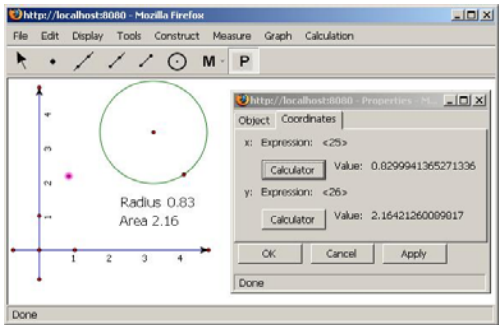
5.2 GUI Implementation

While the engine provides the core functionalities for GeoSVG, the GUI makes user control simple and intuitive, an important requirement especially for teaching and learning.

Since GeoSVG runs in a browser, its GUI depends on the widgets made available by the browser to a Web page.

For GeoSVG, buttons, selection lists and other GUI features, known as *widgets*, available from XHTML are not enough. With Firefox, we can also utilize widgets made available in XUL (XML-based User Interface Language) [14] including different menus, listboxes, trees, and tabboxes. Further, our GUI implementation uses XBL (Extensible Binding Language) [15] for creating user-defined widgets.

Figure 11 shows the frequently used *Property Dialog* for configuring an object. The object being configured is a *Plot Point* object. A tabbox widget from XUL is used here. Different properties are grouped into tabs. The Object tab configures some general properties. The Coordinates tab configures the x-y coordinates of the point. This tab displays a user-defined widget that is composed of text labels and a button. The Calculator button opens a dynamic calculator allowing the author to define the coordinates in terms of existing measurements


http://localhost:8080 - Mozilla Firefox
File Edit Display Tools Construct Measure Graph Calculation
M P
Radius 0.83
Area 2.16
1 2 3 4
1 2 3 4
http://localhost:8080 - Properties - M...
Object Coordinates
Calculator
Calculator
OK Cancel Apply
Done
Done

x: Expression: <25>
Value: 0.8299941365271336
y: Expression: <26>
Value: 2.16421260089817

Figure 11: Property dialog using XUL and XBL

5.3 Stand-alone GeoSVG

To make use of GeoSVG more flexible, we also have the GeoSVG authoring system installable as an extension of the Firefox browser so that it can be used offline and constructions can be saved directly to the local hard disk.

6. Conclusions and Future Work

The Web-based interactive plane geometry system, GeoSVG, takes full advantage of the Web to enhance manipulatives for mathematics education. GeoSVG uses the XML-based, W3C standard SVG for the implementation of the geometry engine. SVG is still at its early stage and browser support is still improving. Performance of complicated SVG is still not ideal. Users sometimes suffer from interaction delay when a manipulative contains too many objects. The development of GeoSVG will continue by adding desired features and improving speed. People can log on to the GeoSite as guest and try it online as we continue to complete the work.

Acknowledgements

Work reported herein has been supported in part by the National Science Foundation under Grant CCR-0201772. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any funding agency.

References

[1] Cabri Geometry II [Computer software], http://www-cabri.imag.fr/cabri2/accueil-e.php
[2] The Geometer's Sketchpad, [Computer software], http://www.keypress.com/sketchpad/
[3] Daniel Scher, Lifting the Curtain: The Evolution of The Geometer's Sketchpad, *Mathematics Educator*, Vol. 10, Num 2, pp. 42-48
[4] Cinderella, http://www.cinderella.de/tiki-index.php
[5] Ulrich Kortenkamp, Foundations of Dynamic Geometry, Dissertation, ETH Zurich, October 1999
[6] Euclides, http://www.euklides.hu/eng/euklides.htm
[7] C.a.R. (compass and ruler) [Computer software], http://mathsrv.ku-eichstaett.de/MGF/homes/ grothmann/java/zirkel/index.html
[8] Major Differences between *JavaSketchpad* and Desktop *Sketchpad,* http://www.keypress.com/ sketchpad/ javasketchpad/differences.php
[9] *Scalable Vector Graphics (SVG) 1.1 Specification*, www.w3.org/TR/SVG/ W3C Recommendation 14 January 2003
[10] *Mathmetical Markup Language (MathML) 1.01 Specification*, www.w3.org/TR/REC-MathML/ W3C Recommendation 7 January 1999
[11] P. S. Wang, M. Mikusa, S. Al-shomrani, D. Chiu, X. Lai, and X. Zou. Features and Advantages of WME: a Web-based Mathematics Education System, *Proceedings of the IEEE Southeast Conference*, IEEE, Ft. Lauderdale, FL, April 8-10 2005, pp. 621-629.
[12] David Chiu. Customization and Interoperability in WME, *Proceedings of the IEEE Southeast Conference*, IEEE, Ft. Lauderdale, FL, April 8-10 2005, pp. 636-640.
[13] M. Mikusa, P. S. Wang, D. Chiu, X. Lai, X. Zou. Web-based Mathematics Education Pilot Project, *Conference on Information Technology in Education*, Elizabethtown College Elizabethtown, PA, September 18, 2004.
[14] *XML User Language (XUL) 1.0 Specification*, http://www.mozilla.org/projects/xul/xul.html, 2001 Mozilla.org
[15] *XBL-Extensible Binding Language 1.0 Specification*, http://www.mozilla.org/projects/xbl/, Mozilla.org
[16] X. Lai, P. S. Wang. An SVG Based Tool for Plane Geometry and Mathematics Education, *Online Proceedings of the Internet Accessible Mathematical Computation (IAMC) 2005 Workshop*, Beijing, China, 24 July 2005