VISGEO

A Geometric Language
White Paper

Sumithra Gomadam
Edan Harel

An Overview of the VISGEO Language.

Introduction:

VISGEO is designed to be a proof of concept for a language designed to be simple, intuitive and user-friendly geometric oriented language for performing operations and manipulations on the basic geometrical shapes like circle, squares, triangle etc. It is intended for use in schools to complement student's mathematical education. VisGeo is an acronym for Visual Geometry, a language that enhances the understanding of geometry by giving a visual experience.

Simple:

VisGeo will be a simple, easy to learn, and intuitive language. It provides most of the intuitive and easy-to-understand functionality dealing with geometry. VisGeo also provides a number of most common control flow constructs, such as loops and conditionals. VisGeo will be easy to learn, have easy, self-explanatory syntax. It should be readable and understandable to anyone familiar with the basic knowledge of geometry and graph plotting. (ie. knowing appoint is represented by a x and y coordinate in a graph, etc).

Educational:

One of VisGeo's goals is to be educational. IT both provides a computational setting for students to create and interact with geometric shapes and objects and also gives them a simple language to learn a few of the basic concepts of Object-Oriented languages.

Architecture Neutral and Portable :

VisGeo will be an interpreted language, with its interpreter written in Java, which will make it architecture neutral. VisGeo programs can be easily seamlessly ported to any platform that has JVM running on it.

Interpreted :

VisGeo will be designed as an interpreted language. Given the basic level of computations to be carried out by the language, there is little need for optimizing. This will allow students to avoid compiling, and instead allow them to immediately go from developing a program to seeing the results. This will also allow for future versions that will allow users to interactively executing each line program using a command line interpreter, akin to a more powerful version of LOGO.

Data types :

There will be two kinds of basic data types. The first is a real number, which can be used to store and compare distances and areas, as well as being used to specify coordinates.

A second kind of data type is a geometric objects. The most basic of these are *Points*, which is comprised of two real number coordinates to specify the point's location on the X-Y axes. *Lines*, and geometric shapes such as *Circles*, *Triangles* and *Quadrilaterals* are included, each of which is constructed with one or more points (as well as a real for radius of circles).

Control Flow Structures:

VisGeo will support most common control flow structures:
Looping constructs: while loop and for loop
Conditionals: if-then-else .

Graphics:

For our prototype, graphical representations of objects can be created by generating postscript files that can be viewed or printed. Future versions of the language may introduce on-screen graphics capabilities.

Extensibility:

This prototype can also be expanded further, by encoding more shapes (eg, Pentagons, Hexagons, etc), as well as finer distinction between shapes (eg, Square vs. Rectangle vs. Quadrilateral).

Time permitting, the following extensions will be added to the language: User-defined functions: users will be able to define their own functions. This is an essential feature of a language, since it allows for extensibility, reusability, and modularity.

Example of Syntax:

Here is a sample program that draws a circle, a line, and a reflection of the circle in the line.

```
/* Create Variables */

Point p1;
Real radius;
Circle c1;
Point p2;
Point p3;
Line l1;
Point p4;
Point p5;
Real xdiff;
Real ydiff;
Real X;
Real Y;
Point p6;
Real radius2;
Circle c2;


/* Create Circle */

p1 = createPoint(5, 5);
radius = 4;
```

```
c1 = createCircle(p1, radius);

/* Create Line */
p2 = createPoint(16, 0);
p3 = createPoint(12, 7);

l1 = createLine(p2, p3);

/* Find the center of the circle and reflect it through the line */
p4 = getCenterOfCircle(c1);

p5 = getClosestPoint(p4, l1);

xdiff = xvalue(p5) - xvalue(p4);
ydiff = yvalue(p5) - yvalue(p4);

X = xvalue(p5) + xdiff;
Y = yvalue(p5) + ydiff;

p6 = createPoint(X, Y);

/* Create the reflected circle */
radius2 = getRadiusOfCircle(c1);
c2 = createCircle(p6, radius2);

/* Graph the line, the circle and it's reflection */
graphShapes(c1, l1, c2);
```

Conclusion:

VisGeo will be a simple, efficient, narrow-domain language for geometry based operations. Using intuitive, self-explanatory syntax it will allow users with little programming experience to write concise, high level programs that perform fairly complex geometrical operations.