Geometry Explorer:
User Guide

Michael Hvidsten
Gustavus Adolphus College

DRAFT: December 31, 2008

Contents

Introduction

> It may well be doubted whether, in all the range of science, there is any field so fascinating to the explorer-so rich in hidden treasures-so fruitful in delightful surprises-as Pure Mathematics.
> Lewis Carroll (Charles Dodgson) (1832-1898)

An *explorer* is one who seeks out new worlds and ideas. As Lewis Carroll would probably agree, exploration is not always easy—the explorer can at times find the going tough. But the treasures and surprises that active exploration of ideas brings is worth the effort.

Geometry Explorer is designed as a geometry laboratory where one can create geometric objects (like points, circles, polygons, areas, etc), carry out transformations on these objects (dilations, reflections, rotations, and translations), and measure aspects of these objects (like length, area, radius, etc). As such, it is much like doing geometry on paper (or sand) with a ruler and compass. However, on paper such constructions are static—points placed on the paper can never be moved again. In *Geometry Explorer*, all constructions are *dynamic*. One can draw a segment and then grab one of the endpoints and move it around the canvas, with the segment moving accordingly. Thus, one can create a construction and test out hypotheses about the construction with an infinite number of possible variations. *Geometry Explorer* is just what the name implies—an environment to explore geometry.

Geometry Explorer can easily be used to access Web-based information. There is an Internet browser built in to the program that allows hyperlinks to Web pages to be inserted directly into a geometry construction. The Help system consists of a series of inter-linked Web pages that are accessed via the built-in browser. (You do *not* need to be connected to the Internet to use the Help system.)

Non-Euclidean geometry can easily be explored using *Geometry Explorer*. Constructions can be carried out in Euclidean, Hyperbolic, or Elliptic environments using the same user interface. Almost all actions that apply

in the Euclidean environment can be carried out in the two non-Euclidean environments (with a few important exceptions that depend on the parallel postulate).

Fractal geometry can be explored using turtle graphics and grammatical descriptions of fractals. In turtle graphics, one controls a "turtle" on the screen by telling it to move, draw, rotate, change color, etc. Grammar-based descriptions of fractals encapsulate a fractal's structure by sentences of symbols. These sentences can then be interpreted as a series of turtle actions.

Geometry Explorer is designed to assist the classroom teacher. Text areas can be created on the screen so that additional information can be included with a construction. If a large amount of textual information must accompany a construction, this can be included in a separate Notebook with the construction. Web pages can be referenced directly from the *Geometry Explorer* window. A fully functional calculator is included for carrying out detailed calculations with measurements and other numerical values. Analytic geometry is supported in a variety of ways. Finally, there is the ability to make "recordings" of sequences of steps (macros) that can then be used in other constructions.

Additionally, any construction created in *Geometry Explorer* can be saved as a web applet that can be accessed via a web browser. With this capability one can share geometric ideas on a fully interactive web page with users from all over the world.

Audience

Geometry Explorer is designed for users having a wide variety of backgrounds in mathematics. At the simplest level, the program allows one to construct complex geometric configurations by using simple visual tools. One does not necessarily have to understand the *why* of a construction to get valuable geometric intuition and insight from playing with the construction. Such play would be quite valuable for students at even an elementary level.

At a more advanced level, *Geometry Explorer* can be used to study patterns and properties that never change under transformation. This idea of studying aspects of figures that remain invariant under transformation was the central theme of Felix Klein's Erlanger Program in the late 1800's. Klein's great insight was that a geometry is essentially defined by invariance of shape under transformation. Thus, Euclidean geometry is the geometry of figures that are invariant under transformations such as translations, ro-

tations, or reflections. For example, triangles are Euclidean figures because their shape remains essentially unchanged when translated, rotated, or reflected. By studying patterns of geometric figures, students can make their own conjectures concerning the geometry they are exploring. This can be done even without a deep understanding of the mathematical theorems and axioms underlying the geometry. Of course, ultimately the goal is to have students then find proofs of their conjectures.

One of the most significant uses of *Geometry Explorer* is to develop an *intuition* about geometry. For this reason, having three different geometries, Euclidean, Hyperbolic, and Elliptic, available to the user is crucial. By trying out constructions in these geometries, students gain an almost tactile understanding of what it would be like to live in these different worlds. Textbooks that include non-Euclidean geometry can at best give a very sterile and static explanation of the geometry. Using *Geometry Explorer* students can move around and play in a hyperbolic or elliptic worlds. Such play gives immediate feed-back and dynamic information about these geometries and also shows in very clear terms how one geometry differs from another.

Technical Requirements

Geometry Explorer will run on Macintosh, Windows, and Linux computers, and on any computer that has a Java Virtual Machine (Java 1.5 or above). The program will run best on a computer with 512 mb or more of ram. On Macintosh computers the operating system must be OS X or above. On PC's the operating system must be at least at the level of Windows 98/NT. For an acceptable level of performance *Geometry Explorer* should be installed on computers that have clock speeds of at least 500 Mhz. To install the software follow the instructions that come with the CD of the software, or follow the instructions from the *Geometry Explorer* web site http://www.gac.edu/ hvidsten/gex.

Using this Guide

Chapters 1 and 2 provide a quick introduction to the program. It is recommended that these chapters be read before reading any of the succeeding chapters.

While in no way comprehensive, the material in Chapter 1 will give a good overall introduction to using the basic user interface features of *Geometry Explorer*.

Chapter 2 consists of a series of tutorials that illustrate specific features of *Geometry Explorer*. The examples and constructions used in these tutorials are somewhat more advanced than those discussed in Chapter 1. Each tutorial guides the user step-by-step through the actions needed to produce a desired geometric figure, at the same time helping the user gain experience with using the tools of the program.

The remaining chapters of this guide consist of detailed, complete references to each of the major categories of tools available in the program. These categories correspond roughly to the graphical layout of the tools in the *Geometry Explorer* main window.

Errata

Geometry Explorer has been used in several courses at Gustavus Adolphus College and has been tested extensively by the author. However, with a program as complex as *Geometry Explorer*, there could be some bugs still out there. Please let me know if you find anything that does not seem to work quite right. Contact the author at *hvidsten@gac.edu*.

Acknowledgments

I would like to thank several people who have assisted with the development of the program. Alicia Sutphen worked on many of the algorithms used in the hyperbolic geometry section of the program during the summer of 1997 as part of a summer research program funded by the President's office at Gustavus Adolphus College. She also wrote up the initial draft of the help pages for the program. Alicia was a delight to work with and her help was very valuable.

Thanks also to the Geometry classes at Gustavus Adolphus College that have used *Geometry Explorer* over many years. The students have been very supportive of the project, even in the early stages where the program was not quite ready for prime time. Student comments and suggestions have been extremely important in achieving the goal of a useful, easy-to-understand software environment for doing geometry.

Lastly, I would like to thank my wife Rebekah for proof-reading early drafts of this guide. Her insistence on clear and concise prose and on writing for a general science audience helped keep the focus of the project where it should be—on engaging people of all ages and backgrounds in the exciting exploration of geometry.

Chapter 1

Getting Started

Euclid alone has looked on Beauty bare.
Let all who prate of Beauty hold their peace,
And lay them prone upon the earth and cease
To ponder on themselves, the while they stare
At nothing, intricately drawn nowhere
In shapes of shifting lineage; let geese
Gabble and hiss, but heroes seek release
From dusty bondage into luminous air.
O blinding hour, O holy, terrible day,
When first the shaft into his vision shone
Of light anatomized! Euclid alone
Has looked on Beauty bare. Fortunate they
Who, though once only and then but far away,
Have heard her massive sandal set on stone.

—Edna St. Vincent Millay (1892–1950)

1.1 The Main *Geometry Explorer* Window

Upon starting *Geometry Explorer* you will see the main *Geometry Explorer* Euclidean window on the screen. (Fig. 1.1)

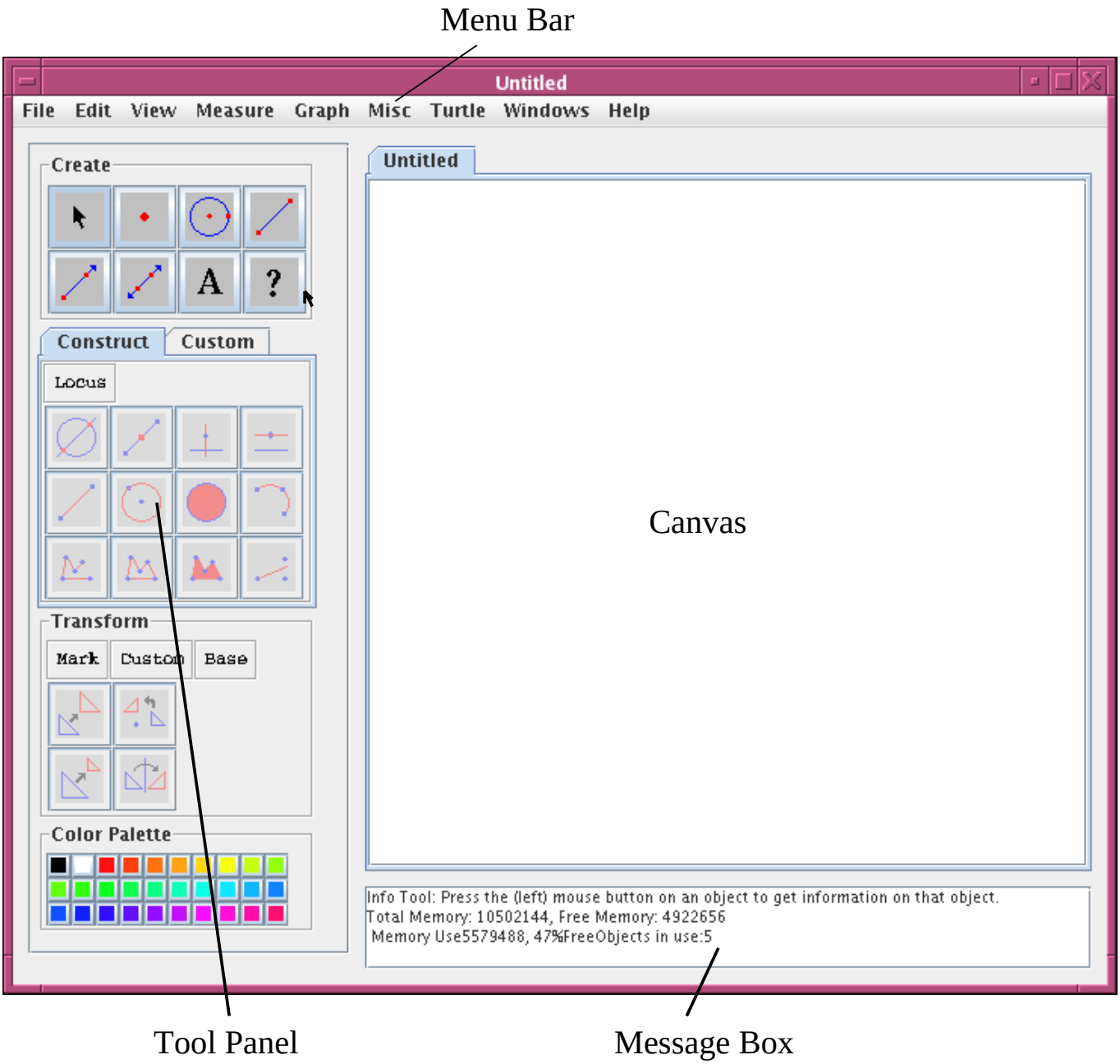


Fig. 1.1 The *Geometry Explorer* Main (Euclidean) Window

There are four important areas within this main window.

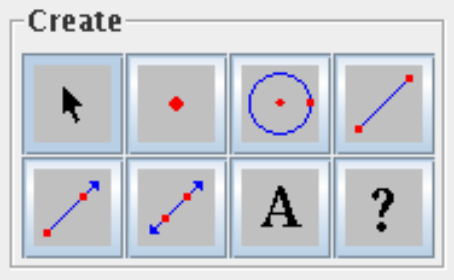
1. The *Canvas* where geometry is created and transformed. This is the large white area on the right side of the main window.

2. The *Tool Panel* where geometric tools are located. The Tool Panel is directly to the left of the Canvas. It consists of a set of iconic buttons which represent tools used to create and modify geometric figures. The icons (pictures) on the buttons depict the function that the particular button serves. Sometimes this function is quite clear, other times it is harder to figure out, but the pictures serve as reminders as to what the buttons can do. The Tool Panel is split into four sub-panels: Create, Construct, Transform, and Color Palette. Note that the cursor is over the Info tool (the one with the question mark). While not shown in the figure, a small box with the words *Get Info on Object* will appear below the Info tool when the cursor is help steady over the tool for a second or two. This box is called a *Tool Tip*. Tool tips are designed to give quick information on a tool's purpose.

3. The *Menu Bar*. There are nine menus shown in the menu bar: **File**, **Edit**, **View**, **Measure**, **Graph**, **Misc**, **Turtle**, **Windows**, and **Help**. Each of these menus will control specific actions which are spelled out in more detail in later chapters of this guide. The figure shows the set of menus available when working in Euclidean geometry. Other menus are available in Hyperbolic and Elliptic geometry.

4. The *Message Box*. This is where detailed information will be shown concerning various tools that one may wish to use. In (Fig. 1.1) the mouse cursor is over the Info tool. In the Message Box we see information concerning how this tool should be used, as well as other information provided by the tool. In the case of the Info tool, we see information regarding memory use for the program. The Message Box is located below the Canvas.

1.2 Selecting Objects



The Selection tool is perhaps the most widely used of all the *Geometry Explorer* tools. When one selects an object, that object is singled out from all of the other objects in the Canvas so that it can be uniquely identified for further use. The most important thing to remember about the selection process is that the Selection tool in the Create Panel must always be clicked in order for selection to be possible. In the figure at right the Selection tool is currently in use as indicated by its pressed-in appearance.

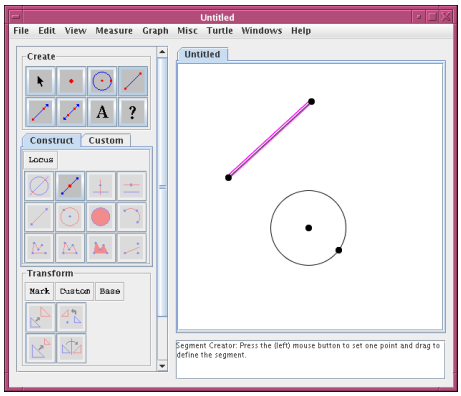
Selections are carried out using the mouse or by using the **Select All** menu option under the **Edit** menu. All mouse actions use the left mouse button, or a single button for those mice having just one button.

1.2.1 Selections Using the Mouse

The selection of objects via the mouse can be carried out in three ways.

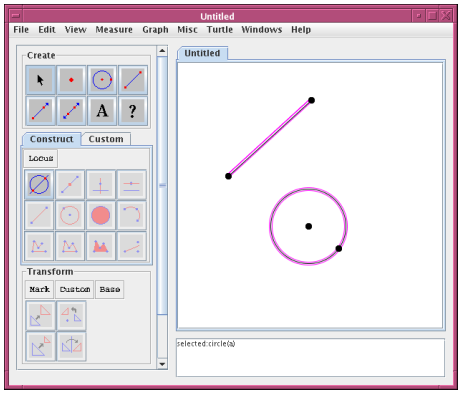
1. **Single Selection** One clicks on a single object to select it.

In the figure at the right, a circle and a segment have been created. Note that the segment was created last, as the Segment tool is shown as being clicked. Also note that the segment has an outline draw around it. This is used to visually signify that the segment is currently a selected object. When an object is created it is automatically selected. Also note that a message appears in the Message Box below the Canvas telling the user what object is being selected.



2. **Multiple Selection** One clicks on a series of objects to select them all.

Suppose we wish to select the circle as well as the segment in the figure above. We first click on the Selection tool (in the Create Panel) to make the Selection tool active. Then, we move the mouse to the Canvas and click somewhere along the circle. In the figure at right we see the circle is selected, as well as the segment. Each time we click on a new object it will be added to the current group of selections.

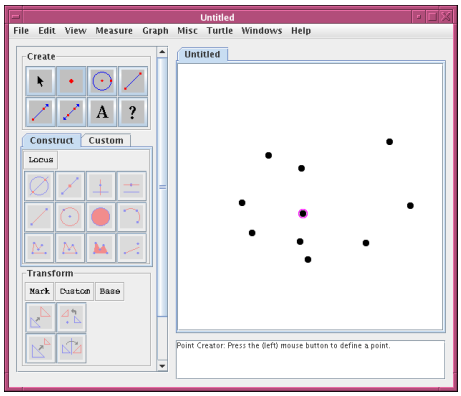


If we wanted to select just the circle, we could do so by first clicking the mouse in a white area in the canvas to clear all selections, and then click on the circle.

If we want to move a set of selected objects, we first select them all. Then, while holding the Shift key down, we click and drag to move the set of objects.

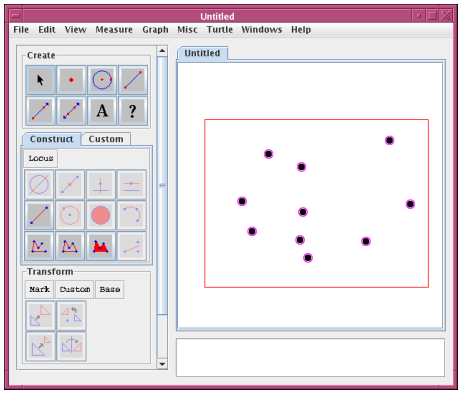
3. **Box Selection** One can draw a selection box about a set of objects to select all of the objects enclosed in the box.

In the figure at the right, a series of points have been created. Suppose that we want to draw a polygon through this set of points, not really caring which order they are connected. It would be tedious to do a multiple selection of each point.

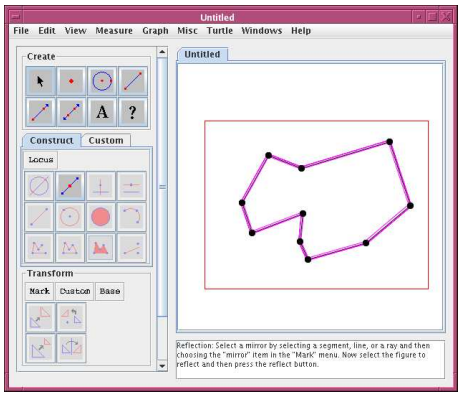


We can select all of the points at once by using the box selection feature available in *Geometry Explorer*.

To do a box selection of the set of points above, first make sure that the Selection tool is active. Then, click in the upper left-hand corner of the Canvas and drag to create a selection box surrounding all of the points. When you release the mouse button all of the objects inside the box will be selected. The selection box can be visually identified by its red appearance.



After selecting a group of objects, we can create new objects based on the selected objects. For example, after we select all of the points above, the Closed Polygon tool (second from left in bottom row of the Construct Panel) will be active. After clicking on this tool, we get a figure similar to the one on the right. Note that the selection box remains visible until we select some other object.



1.2.2 Selections Using the Select All Menu

The selection of objects via the **Select All** menu item in the **Edit** menu is designed for those situations where one wants to select all of a particular type of object. For example, one may want to select all points in a figure, or all circles. When we click on **Select All**, a sub-menu with the items **Points**, **Segments**, **Rays**, **Lines**, **Circles**, **Arcs**, and **Objects** will pop up. By dragging across to one of these options and releasing the mouse, all of the objects of that type will be selected. If one chooses **Objects** then all objects of any type on the Canvas will be selected.

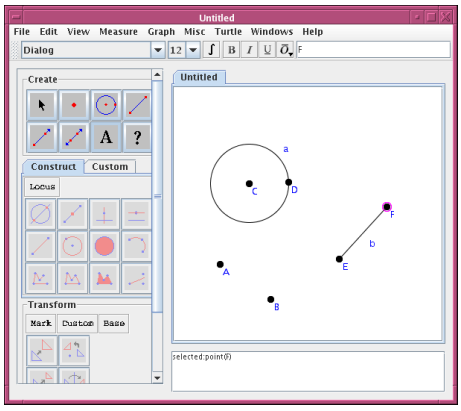
1.3 Active vs Inactive Tools

We have talked a lot so far about how to make tools *active* so that they can be utilized. Some tools are always active. Others can change from active to inactive and vice-versa depending on user actions. For example, tools in the Create Panel are always available for use, they are always in an active state. Most other tools will start out in an inactive state. An inactive tool can be visually identified by its grayed-out border or appearance. When a tool is in an inactive state, clicking on that tool will have no effect. To activate an inactive tool, one needs to select the kinds of objects that the tool needs defined in order to function. For example to activate the Midpoint tool (second tool in first row of the Construct Panel), one needs to select a segment and then the Midpoint tool will become active.

1.4 Labels

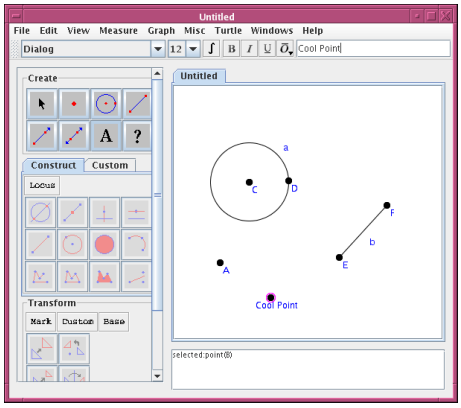
So far none of the examples discussed have had objects with visible labels. However, *all* objects that one creates are created with labels—they are just not always visible. To make a label visible we use the Text/Label tool in the Create Panel. (The one with the “A” on it).

In the figure at the right several geometric objects are shown. The labels are made visible by first clicking on the Text/Label tool in the Create Panel and then clicking on an object to make its label visible. Labels are created in alphabetical order based on the sequence of object creation. Thus, points A and B were created first, then circle a, then line b.

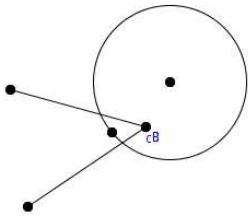


Note that points are created with capital letters whereas lines, circles, and arcs have lower-case labels. Also note that a new tool bar has appeared below the main menu bar. This tool bar can be used to modify textual attributes of a label.

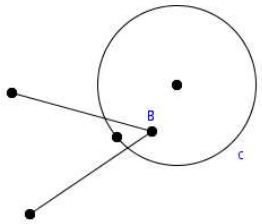
If we want to edit a label, we click on an object using the Text/Label tool and type in the new text for the label in the tool bar, as shown. Here, we have changed the label of point "B" to say "Cool Point."



Sometimes a label can get partially obscured by other objects. In the figure at the right the label "c" for the circle is partially obscured by the label for point B.



To move a label, click on the label in the Canvas and drag the label to the desired position. Note that the label cannot be placed anywhere one chooses. Labels can be only be moved within a zone around the object to which they are attached. In the figure at the right we have placed the labels in a better position.



1.5 Object Coloring

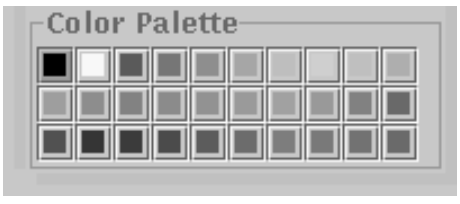
We can change the color of an object that exists in the Canvas by using the Color Palette in the Tool Panel. The Color Palette consists of a set of color squares on the bottom left of the main window. To change an object's color we first select the object and then click on a color square to immediately change that object's color. If we select a group of objects (using multiple selection) then all objects in that group will have their color changed to the desired color.

The color of the label of an object can be changed by first clicking on the object with the Text/Label tool and then clicking on a color in the Color Palette.

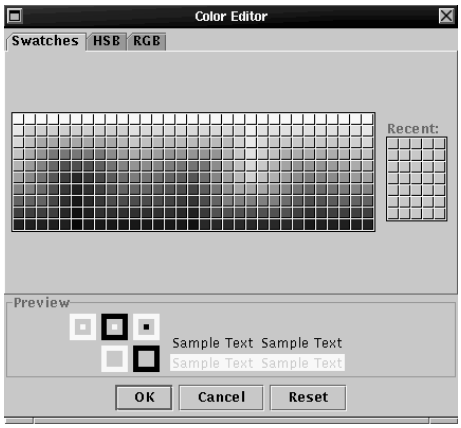
1.6 Changing Palette Colors

A color in the Color Palette can be modified by holding down the Control key while clicking on the color.

For example, if we want to change the black color in the Color Palette we would hold down the Control key and click on the black color square.



Once a color is selected a Color Editor dialog box will pop up as shown. This dialog box has three panels that one can use to define new colors: the Swatches, HSB, and RGB panels. The simplest way to define a new color is to just click on one of the many colors in the Swatches panel and then hit the "Okay" button. The new color will then replace the black color in the Color Palette. Any newly defined colors will automatically be saved when a *Geometry Explorer* session is saved.



The HSB and RGB panels can be used to *precisely* define new colors. The HSB panel uses the Hue-Saturation-Brilliance method of defining a color and the RGB panel uses the Red-Green-Blue 24-bit method of defining a color.

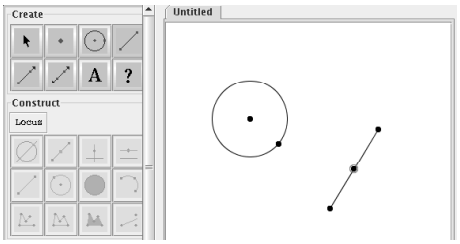
1.7 On-Line Help

There is an extensive on-line help system that can be accessed via the **Help** menu item in the Menu Bar at the top of the main *Geometry Explorer* window. Click on this menu item and then on the **Help** sub-menu to start up the help system. The help system is designed as a series of web pages that are viewed by an Internet browser that is built into *Geometry Explorer*. No additional software is needed to view these web pages. The help system is organized into categories that roughly correspond to the visual areas in the *Geometry Explorer* window—panels, menus, etc. There are many examples available in the help system from an introductory to advanced level.

1.8 Undo/Redo of Actions

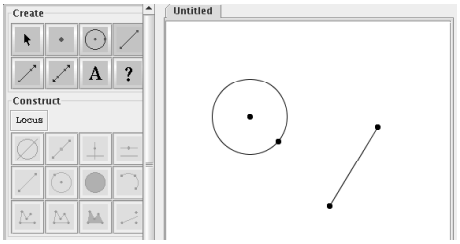
Geometry Explorer provides the user with the ability to *undo* almost any action that arises from some geometric construction.

For example, in the figure at the right we created (in order of creation) a circle, a segment, and the midpoint of the segment.



Suppose we decide that the midpoint is not actually needed. We can undo the midpoint action by going to the **Edit** menu in the Menu Bar and choosing the **Undo midpoint** sub-menu.

The midpoint construction will be undone, leaving just the circle and segment.



At this point if we decided that the segment was also a mistake we could undo again to get rid of the segment. Undoing yet another time would erase the circle and leave a totally blank Canvas.

Now, suppose we decided that we really did like the circle, segment, and midpoint that we had initially constructed. Then, we could *redo* all of the steps that we just undid. This is done by choosing **Redo** from the **Edit** menu.

Geometry Explorer provides the user with an unlimited ability to undo and redo steps. This capability is very useful for showing someone exactly

what steps were done to produce a geometric figure.

Note that objects can only be undone/redone in the order in which they were made.

1.9 Clearing, Resizing the Canvas

To clear the Canvas of all objects currently constructed we use the **Clear** menu item under the **Edit** menu. This action will clear all currently defined objects. Note that this is different than undoing the constructions. When we clear the Canvas all objects are immediately removed. However, clearing the screen is itself an action that can be undone. Thus, if we clear the screen and then change our mind we can always undo or redo this action.

On most computers a program's window can be resized by clicking somewhere on the border of the window and dragging. If the boundary window for *Geometry Explorer* is resized the Canvas will also change size, but the Tool Panel and Menu bar will not change size. As the Canvas changes size you will notice that the figures on the Canvas change so that the size of objects relative to the size of the window stays the same. For example, if we had a circle that filled half of the Canvas and then we doubled the length and width of the main window, then the circle would still fill half of the new expanded Canvas.

The reason for this is that all of the mathematical calculations for the program are done on a "virtual" Canvas that has the dimensions of a square. The virtual coordinates of this square Canvas are transformed to screen pixel coordinates and then displayed on the screen. The virtual Canvas is always fixed in size, but as the screen area changes, the transformation from the virtual Canvas to the screen Canvas preserves relative distances.

Expanding the size of the main window will have the effect of increasing the resolution of your figure. If objects are too close then expanding the window size will be like putting your figure under a microscope.

If you expand your window in such a way that the Canvas can no longer be displayed inside of a perfect square, then the square Canvas will be placed inside of a scrolling window.

Sometimes a construction will be so large that it leaves the boundaries of the Canvas. There is a way to *rescale* the figures in the Canvas so that the image will shrink or grow. Look in Chapter 13 under the section labeled "Rescaling Constructions" for information on how to rescale the image inside of the Canvas.

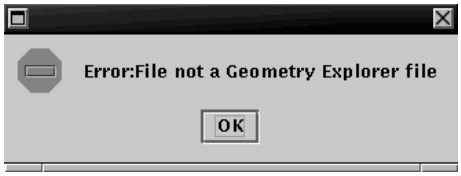
1.10 Saving, Opening, Printing of Files

File operations like saving, opening, and printing of files are operations that are very specific to a computer's operating system. For instance, printing is handled very differently in a Unix environment versus a Macintosh environment. Since *Geometry Explorer* will run on virtually any computer, it is difficult to give specific instructions on how to handle file operations. However, some universal interface elements are present in any version of *Geometry Explorer*.

When using any software environment one's motto should always be "Save Often". In *Geometry Explorer* one can save a set of geometric constructions, measurements, etc, by using the **Save** menu option under the **File menu**. After choosing **Save** a file dialog box will pop up asking you where you wish to save your work and what you wish to title the saved file. This file dialog box will look like the standard Open/Close/Save file box that is commonly used in other programs on your computer.

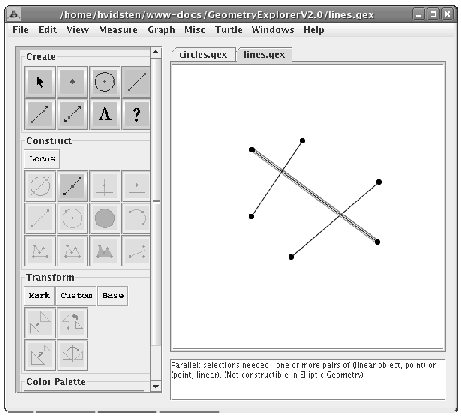
If you wish to open a previously saved *Geometry Explorer* file, choose **Open** under the **File** menu. Again, a file dialog box will open and you can choose the file you wish to open. Note that *Geometry Explorer* will open only those files that were saved from the program itself.

If you try to open a word processing document, for instance, *Geometry Explorer* will pop up an error dialog box like the one shown on the right, telling you that the file is not a valid *Geometry Explorer* file.

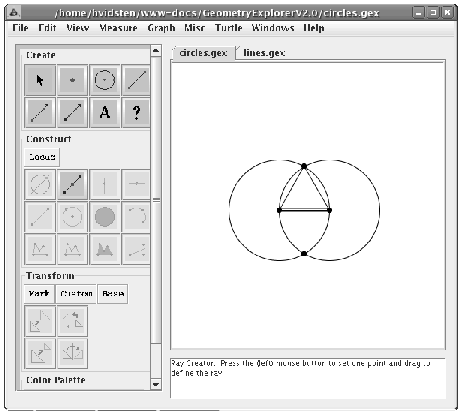


When a valid *Geometry Explorer* file is opened a new Canvas will appear inside the main *Geometry Explorer* window. This canvas will appear as a tabbed panel inside the window.

As an example, suppose that we saved two constructions as files labeled "circles" and "lines". If we opened the circles file first and then the lines file we would see the main window configured as shown.



Note that there are two tabs on the top of the Canvas. These display which files are currently loaded into *Geometry Explorer*. We can click on these tabs to move back and forth between the constructions. For example, if we click on the "circles" tab we would get the Canvas for that file, as shown.

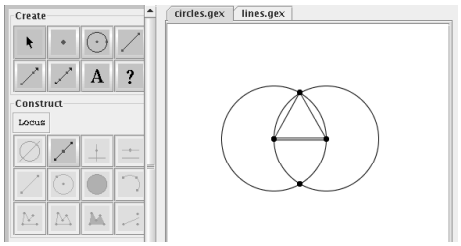


To print the contents of the Canvas, choose the **Print** menu item under the **File** menu. A print dialog box will pop up asking you to set certain printing options. This print dialog box will be similar to the print dialog boxes that appear when you print from other programs on your computer.

1.11 Saving Files as Images

It is often desirable to save the contents of the Canvas to an image file. This is useful for example if you want to add a picture of the Canvas to a web page or if you want to insert a picture of the Canvas into a word processing document.

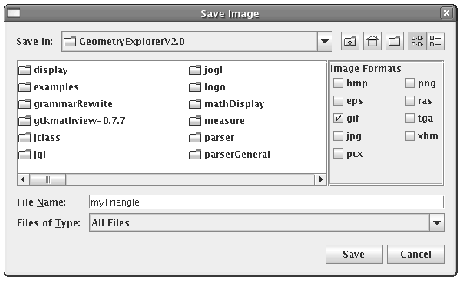
As an example suppose that we have constructed the equilateral triangle shown at the right and wish to save it as a GIF file.



To save the Canvas as an image choose **Save as Image...** under the **File** menu. A dialog box will pop up as shown at the right.

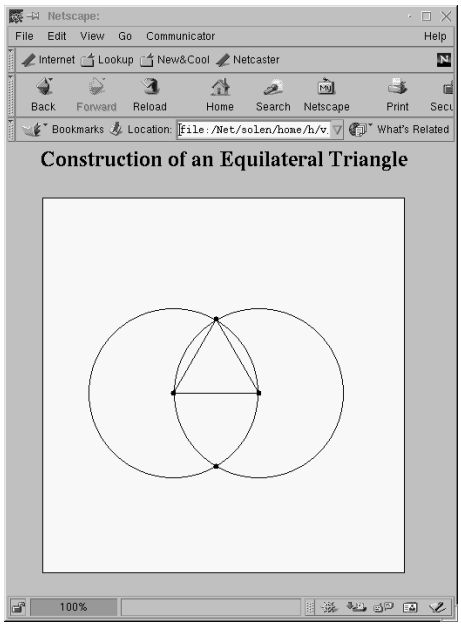
Note the rows of buttons to the right of the directory window. These allow one to specify the image format that the Canvas will be saved to. Supported image formats include most of the commonly used formats: bmp (Windows bitmap), eps (Encapsulated Postscript), gif (Graphics Interchange Format), jpg (JPEG format), pcx (PC Paintbrush), png (Portable Network Graphics), ras (Sun Raster), tga (Targa), and xbm (X Windows Bitmap). The default image format is the JPEG format.

Suppose we have created a triangle and want to save it as an image file called "myTriangle". To save it as a GIF file we click on the "gif" check box and hit the "Save" button.



At this point the cursor may switch to a wait cursor signaling that the conversion of the Canvas to a stored image i s taking some time. Actually, the image conversion computations may take a little while so be patient. Once the wait cursor switches back to a normal cursor the image will be saved as the file "myTriangle.gif". The appropriate suffix ".gif" will automatically be appended to the file name.

We can now take our image file and use it in a web page or word processor as shown.



One important note about EPS files is that a preview image is stored with an EPS file so that you can see the image when inserting it into another program such as a word processor. However, the image quality will typically be much less than the real postscript image. The image will have the original postscript quality when printed with your document on a postscript-compatible printer.

Chapter 2

Tutorials

> Ptolemy once asked Euclid whether there was any shorter way to a knowledge of geometry than by a study of the Elements, whereupon Euclid answered that there was no royal road to geometry.
>
> —Proclus Diadochus (411–485)

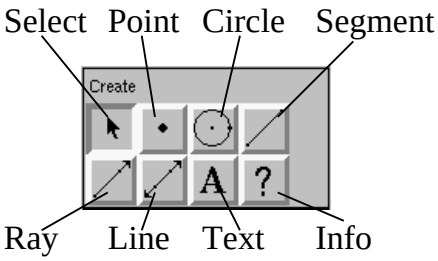
This chapter consists of a set of seven tutorials which are designed to illustrate major features of *Geometry Explorer*. The best way to carry out the tutorials is in the order that they appear in this chapter. However, each tutorial is sufficiently self-contained that one can also pick and choose what looks the most useful and interesting.

Note that one should have finished reviewing the introductory material in Chapter 1 before looking at this chapter. In particular, familiarity with the basic Tool Panel layout, with basic constructions, and with selections is essential.

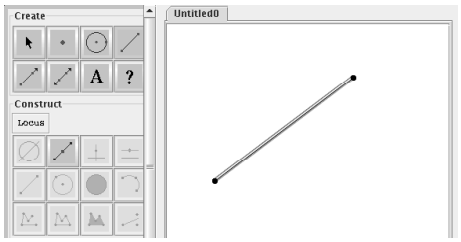
2.1 Tutorial 1 Working With Basic Geometric Figures

In this tutorial we look at how to use the tools in the Create Panel to make simple geometric figures.

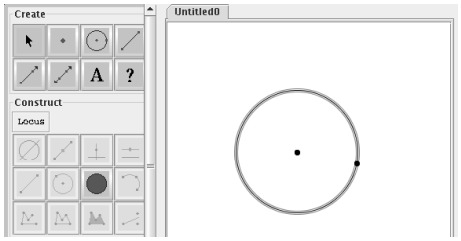
The tools in the Create Panel are used to make points, circles, segments, lines, and rays.



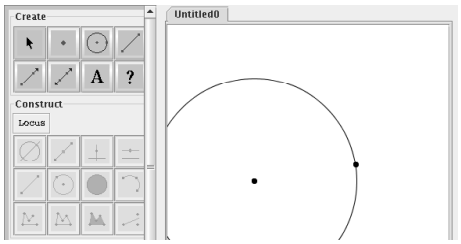
Let's start with something easy—making a segment. Click on the Segment tool in the Create Panel to make that tool active. Move to the Canvas and click and drag the mouse. A segment will be drawn and will change dynamically (like a rubber band) as you move the mouse.



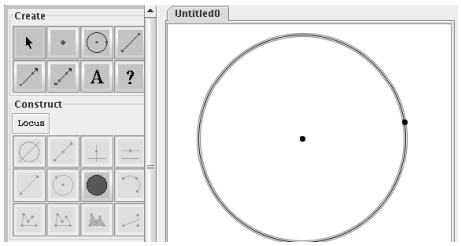
Now let's try something more complicated. First, go to the **Edit** menu and choose **Clear** to erase the segment. Then, click on the Circle tool, move to the Canvas, and click to set one point as the center of the desired circle. Drag the mouse and another point will appear under the cursor. This point acts as a radius point on the circle. Drag the radius point until the circle is the desired size and release. The circle will be drawn with a purple outline to show that the circle is currently a *selected* object.



One of *Geometry Explorer*'s major features is that geometric constructions are "live" and can be altered dynamically. Make sure that the Select tool is currently active. Click in a white section of the canvas and then on the center point of the circle. Drag the center point around the Canvas. You will see the circle change shape as you move the mouse.

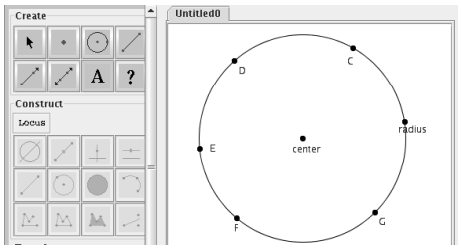


We can move the entire circle as an object also. Click on the circle somewhere other than the center or radius points. Then, drag the mouse. The entire circle will move, preserving the relationship between the center point and the radius point.



Suppose we want to inscribe a pentagonal shape inside of this circle. To do this we will place four new points on the circumference of the circle.

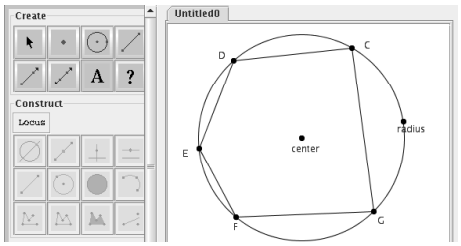
Make the point tool active and click on the circle in five places other than the radius point. Now, to double check that these new points are actually *attached* to the circle, select one of the points and move it around. It will move in a way that is always constrained to the circle.



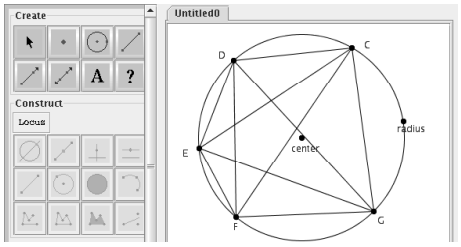
In general, to *attach* a point to another object, such as a segment or circle, we make the Point tool active and click the mouse on the object. The point created will then be forever attached to the object.

To finish the creation of our five-sided polygon, we need to join the five new points on the circle with segments. Click on the Segment tool to make it active and then click on one of the attached points on the circle.

Drag the mouse until another attached point on the circle becomes highlighted. Release the mouse and a new segment will connect the first point to the highlighted second point. Do this four more times to create the desired polygon.



Finally, let's connect all possible pairs of points among the five points on the circle by creating segments between all possible pairs of points. We end up with the star-shaped figure shown.



Experiment with moving the points on the circle, the center point, and various segments within the figure. Note how the construction changes in size, but the geometric connections within the construction stay fixed.

2.2 Tutorial 2 Constructions

The Construct Panel (Fig. 2.1) controls the construction of geometric objects which depend upon already existing objects. This panel consists of 12 tools. Note that there is a segment and circle *construction* tool in the Construct Panel as well as in the Create Panel. This is to allow for the construction of a circle or segment from already existing objects (for example, from two existing points).

There is one additional feature in the Construct Panel—the "Locus" button. This button hides a pop-down menu that will allow one to create geometric loci. To read more about this advanced feature of *Geometry Explorer* see the section on loci in Chapter 3.

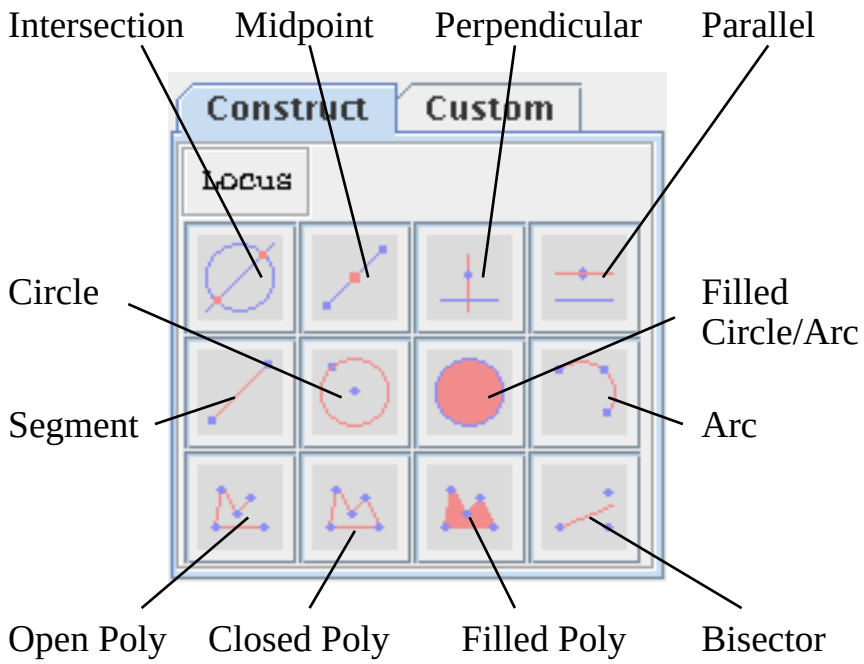


Fig. 2.1 The Construct Panel

Initially, when there are no objects defined on the Canvas, the Construct Panel tools will be inactive (i.e. grayed-out) because none of the constructions can be performed from scratch. Once the objects that are necessary for a particular construction have been built, and are selected in the correct order, that particular button will become active (i.e., darker in appearance). Clicking on the activated button will automatically perform the construction using the selected objects. To get a quick idea of what needs to be selected to activate a tool, pass the mouse cursor over that button and information will appear in the Message Box.

To illustrate how constructions work, we will look at an example from classical Greek geometry.

2.2.1 Euclid's Equilateral Triangle

This example is the first demonstration in Book I of Euclid's classic work *The Elements*. In this demonstration Euclid shows how to construct an

equilateral triangle. The construction goes as follows:

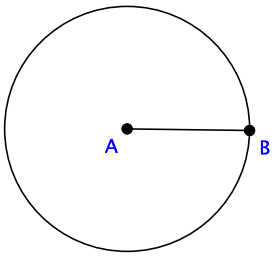
1. Construct a segment from point A to point B.
2. Construct a circle C_1 with center at A and radius of AB.
3. Construct a circle C_2 with center at B and radius of AB.
4. Let D be one of the intersection points of circles C_1 and C_2.
5. Triangle ABD will then be equilateral.

To do this in *Geometry Explorer* we would proceed as follows:

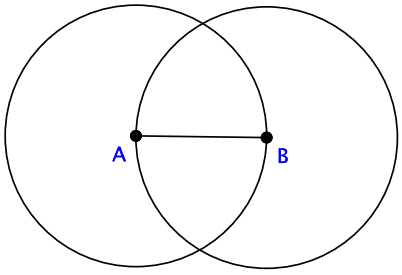
First, create a segment on the Canvas, shown here as segment $\overline{AB}$.

Initially, the labels of the endpoints of the segment will not be visible. To make them visible, go to the Create Panel and click on the Text/Label tool (The "A" button). This tool is used to edit and show/hide labels. With the Text/Label tool activated click on each endpoint and the label will appear.

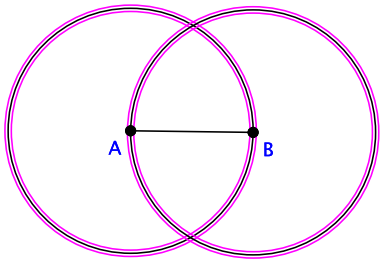
Next create a circle from point A to B by 1) activating the Circle tool in the Create Panel, 2) clicking on point A and dragging until the cursor is over point B, and 3) releasing the mouse. You will have created a circle with center A and radius B.



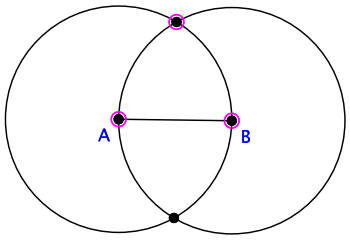
Likewise, create a second circle with the center point being point B and the radius point being point A.



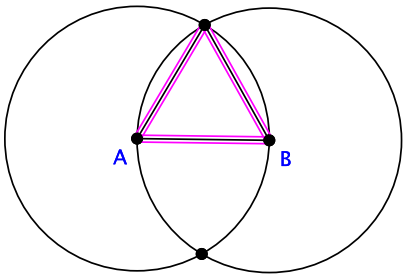
We will construct the intersection of these two circles by activating the Select tool in the Create Panel and clicking on each circle. Once both circles are selected the Intersection tool in the Construct Panel will be active.



To construct the two intersection points, click on the Intersection tool. One of these intersection points, along with the radius and center of one of the circles, will form an equilateral triangle. To construct the triangle select the top intersection point and also points A and B.



With these three points selected, many construction tools become active. In particular the Closed Polygon tool is now active. Click on this tool to construct a triangle. (For fun: why must the triangle be an equilateral triangle?) Now select either of the centers in the Canvas and drag them around.



The triangle remains equilateral. All of the constructions that were made - circles, intersections, polygons, etc, are preserved under dynamic changes

to the centers, and so the triangle will always have three sides of equal length.

2.3 Tutorial 3 Transforming Geometric Figures

The Transform Panel enables you to perform four different transformations on geometric objects. These include translations, rotations, dilations, and reflections. Transformations are carried out in a two-stage process. First, you must specify the geometric information that defines a transformation. Then, you must select the objects to be transformed and click on the appropriate transform tool. There are three pop-down menus used to define necessary geometric information for transformations. These are hidden under the "Mark", "Custom", and "Base" buttons. (Fig. 2.2)

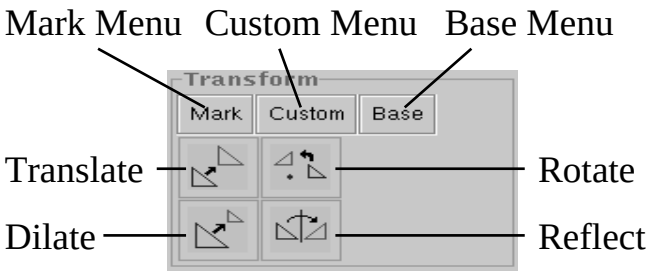


Fig. 2.2 The Transform Panel

2.3.1 Rotation of a Figure – Using a Geometric Angle

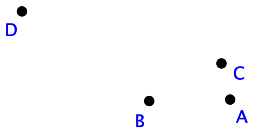
A rotation is possible once we know two pieces of information—a point about which the rotation is carried out (called the *center of rotation*), and an *angle* specifying the magnitude of the rotation about the center point.

An Angle can be defined in two ways—as a numerical value in degrees or as the angle defined by the position of three points A,B, and C. A is called the *initial* point, B is the *vertex* and C is the *terminal* point of the angle.

In *Geometry Explorer* the information needed for a transformation is defined using the **Mark** pop-up menu. If we look under that menu we will

see both a **Center** and **Angle** sub-menu. However, both will be inactive until we create and select a center and an angle.

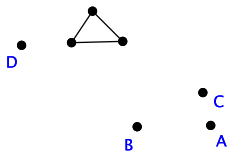
At right we have created four points A, B, C, and D. We will use points A, B, and C to define an angle and point D to define a center of rotation.



To define D as a center of rotation we select D and then choose **Center** from the **Mark** pop-up menu. Next, we define an angle by selecting points A, B, and C (in that order). Then, choose **Angle** from the **Mark** pop-up menu.

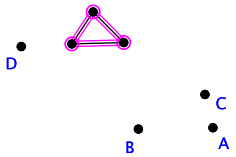
At this point we have defined all the necessary objects for a rotation. We can apply this rotation to any figure in the Canvas. Let's apply it to a triangle.

Use the Segment tool (in The Create Panel) to construct a triangle as shown in the figure at the right.

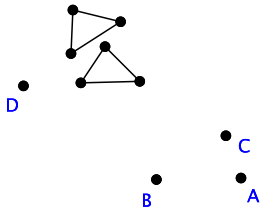


To apply our rotation to the triangle we first need to select it. One easy way to do this is to use the Select tool and do a box selection around the figure. Then, all objects entirely within the selection box will be selected.

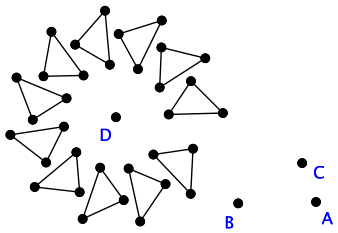
Carry out this box selection, as shown in the figure. Note that the Rotation tool in the Transform Panel has now become active.



To rotate the triangle just click on the Rotate tool. Note that the rotation will be carried out and a new, rotated, copy of the original triangle will be created and will be selected.



At this point we can carry out (iterate) the rotation on the newly created triangle, achieving a double rotation. We can iterate the rotation again and again, getting a sequence of rotated triangles all based on the original angle and center of rotation.



2.3.2 Dilation of a Figure – Using a Numerical Angle

In the above example, we defined a transformation in terms of *geometric* data, that is an angle was defined by three points and not a numerical value. Often, we need to define a transformation in terms of fixed, numerical values. For example, suppose we needed to divide a segment into exactly three equal parts. One way to do this is to scale the segment down by factors of $\frac{2}{3}$ and $\frac{1}{3}$. To scale a figure (i.e. shrink or expand) we need to use a type of transformation called a *dilation*. A dilation is a scaling down (or up) of a figure in relation to a center point. To define a dilation we need to specify a center of dilation (a point) and a scaling factor (a ratio).

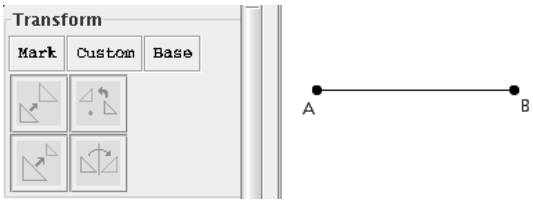
To illustrate how this works, let's return to the problem of dividing a segment into three equal parts by dilating one endpoint by scale factors of $\frac{1}{3}$ and $\frac{2}{3}$towards the other endpoint.

To begin we create a segment $\overline{AB}$ (Use the Segment tool in the Create Panel). Now, we want to define two different dilation (or scaling) transformations that will shrink point B towards point A by scale factors of $\frac{1}{3}$ and $\frac{2}{3}$. To accomplish this, we set point A as the center of dilation by selecting A and choosing **Center** from the **Mark** pop-up menu in the Construct Panel.

Next, we define a numerical scale factor for the dilation. In the *Geometry Explorer* environment transformations needing numerical values are defined as "Custom" transformations. They are specified using the **Custom** pop-up menu in the Construct Panel.

Go to the **Custom** pop-up menu and select **Dilation** from the choices. The window shown here will pop up. Type in the values of "1" and "3" for the numerator and denominator and hit "Okay" in the dialog window.

At this point the dilation transformation is fully defined – we have a center of dilation (point A) and a scale factor ($\frac{1}{3}$). To dilate point B by a factor of $\frac{1}{3}$ toward point A we first select point B, as this is the object we wish to transform. The Dilate tool will now become active, as shown.

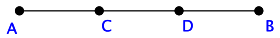


Click on the Dilate tool to construct the new point C that is $\frac{1}{3}$ of the distance from B to A.

To finish the subdivision of $\overline{AB}$ into three equal parts, we need to dilate B by a new scale factor of $\frac{2}{3}$ towards point A. *Geometry Explorer* already has point A stored as a center point, so we do not need to re-define the center. However, we need to re-define the scale factor.

To define a new scale factor, click on the **Custom** pop-up menu in the Construct Panel, choose the **Dilation** sub-menu, and put in the ratio $\frac{2}{3}$. Then click "Okay."

At this point the Dilate tool should be active. Click on B and then click the Dilate tool to create a point D that is $\frac{2}{3}$ of the distance from A to B.



When dilating objects the scale factor is always viewed as a *ratio* of the distance towards the center point of the dilation. Thus, a scale factor of $\frac{2}{3}$ means that in the example just shown the distance from dilated point D to point A divided by the distance from B to A will be equal to $\frac{2}{3}$.

2.4 Tutorial 4 Measurement

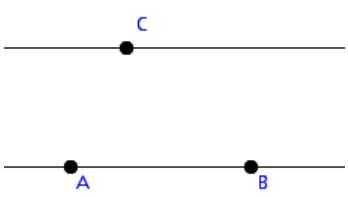
So far we have looked at how *Geometry Explorer* can be used to construct complex geometric figures and carry out transformations on those figures. Another valuable component of *Geometry Explorer* is the ability to make geometric measurements on objects. Measurements are performed by selecting an object and then choosing an appropriate measurement from the **Measure** menu.

The **Measure** menu consists of 19 different items. There are 13 measurements that can be made on geometric objects, three measures that are "special" measures, and three options that deal with the use of tables of measurements (Consult Chapter 4 for more information on special measurements and on how to create and use tables). The **Measure** menu shown here is for the Euclidean Canvas. There are slightly different menus for the Hyperbolic and Elliptic Canvases.

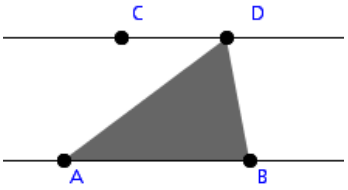
2.4.1 Triangle Area

As an example let's look at using measurements to study areas of triangles. In particular we will consider triangles constructed between two parallel lines.

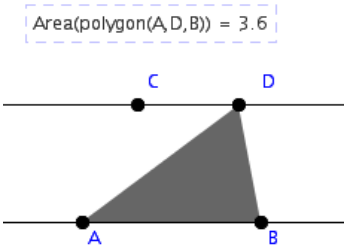
Construct a line $\overleftrightarrow{AB}$ on the Canvas and then create a point C off this line. Select the line and then the point. The Parallel tool in the Construct Panel will now be active. Click on this tool to construct the parallel to $\overleftrightarrow{AB}$ through C.



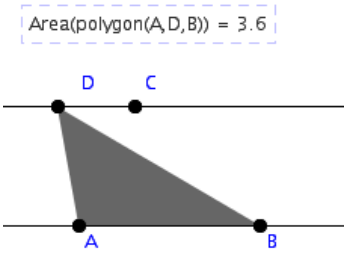
Now, create a new point D attached to the parallel through C. To do this use the Point tool and click the mouse somewhere along the line. To measure the area of triangle ABD we first need to create a filled-in polygon for this triangle. Select points A, B, and D and click on the Filled Polygon tool in the Construct Panel.



Select the triangular area by using the Selection tool and clicking somewhere in the black area. The **Area** menu item under the **Measure** menu will now be active. Choose **Area** to have *Geometry Explorer* calculate the area of the triangle. A text box should appear in the Canvas giving the area of the triangle.



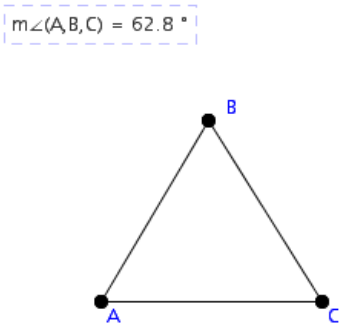
Drag point D back and forth along $\overleftrightarrow{CD}$ and notice how the area remains fixed, as it should!



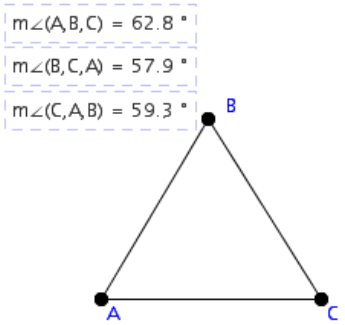
2.4.2 Triangle Angle Sum

As another example of using measurements we will analyze the angles within a triangle ABC.

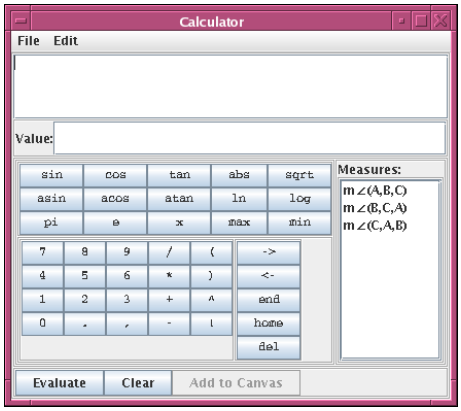
To construct ΔABC we use the Segment tool in the Create Panel to create three connecting segments. To measure an angle, we need to select three points: the initial, vertex, and terminal points of the angle. Select points A, B, and C (in that order). The **Angle** menu item in the **Measure** menu should now be active. Click on this menu item to calculate the numerical value of this angle and display it in the Canvas.



Now, measure $\angle BCA$ and $\angle CAB$. The order in which points are selected makes a difference. If we had measured $\angle CBA$ (selecting, in order, C, B, and A) rather than $\angle ABC$ we would have gotten the measure of the angle from C counter-clockwise around B to A, which would be greater than 180 degrees, as shown in the figure.



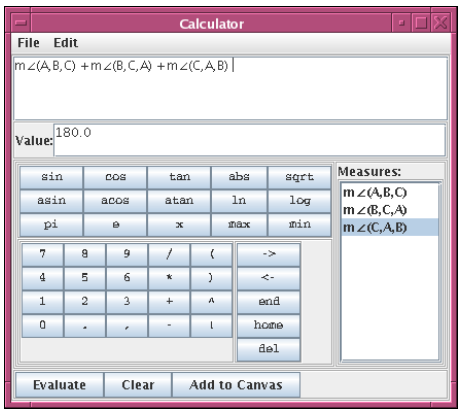
At this point we will add these three interior angles together. To do this, we use *Geometry Explorer*'s built0in Calculator. To access the Calculator, go to the **View** menu and choose **Calculator...**



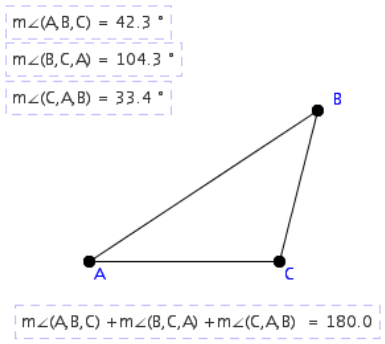
The calculator is a powerful feature of *Geometry Explorer* and will be explained fully in a later section. However, for now all we need to know is that the measures that we have just defined appear in the **Measures** list

on the right side of the Calculator window. If we double click on a measure (e.g. on m$\angle(A, B, C)$ the measure will be inserted into the main expression window at the top of the Calculator. We want to add this measure to the measure of $\angle BCA$. To do this click on the "+" key and then double click on the list entry labeled "m$\angle(B, C, A)$". Finally, add the measure for $\angle CAB$ by clicking "+" and then double clicking "m$\angle(C, A, B)$". We have now created a mathematical expression (or formula) that adds up the three angles in the triangle.

To evaluate this expression click the Evaluate button on the bottom of the Calculator. You will see the result in the "value" text area. Lo and behold, the sum of the angles is 180 degrees!



To make this new compound measurement part of the Canvas, click on the Add to Canvas button in the Calculator. *Geometry Explorer* will take this new formula for the sum of three angles and add it to the Canvas. Drag the vertexes of the triangle around and verify that the angle sum does not change, *except* when one of the vertexes is dragged so that its corresponding triangle angle reverses orientation.



To summarize, to compute a measurement it is necessary to first select the objects needed for the measurement (e.g. three points for an angle) and then choose the appropriate measurement from the **Measure** menu in the Menu Bar.

2.5 Tutorial 5 Analytic Geometry

Using *Geometry Explorer* one can graph the relationship between two measured (Euclidean) quantities. A *graph* consists of two coordinate axes (x and y) and points plotted in relation to these axes. The *coordinate system* is the system by which a point is located on the graph. For us, this will be determined by a point of origin (where the two axes intersect) and a unit point on the x-axis which fixes a distance of one unit along that axes.

In the figure below the coordinate system used in *Geometry Explorer* is shown. The origin and unit points are visible. Note that tic marks are shown on the axes to help identify coordinate values. The default coordinate system runs from -5 to 5 on both coordinate axes, as shown by the tic marks in the figure.

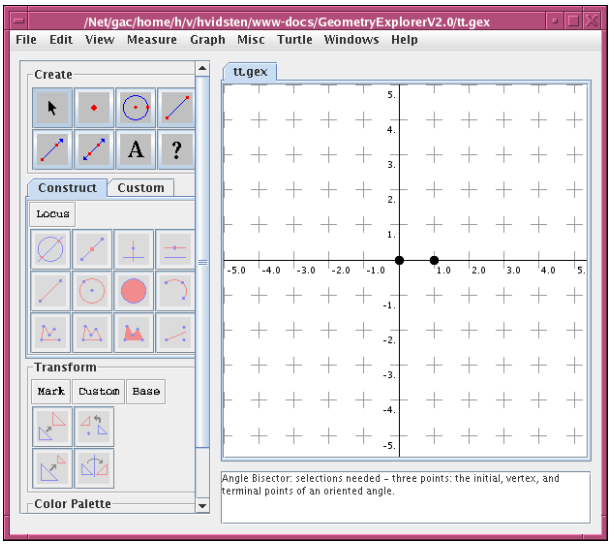


Fig. 2.3 Geometry Explorer Coordinate System

To plot a point in this graph coordinate system, we need to specify two numerical values to use as the x and y coordinate values. Since measurements are always numerical, we use two measurements to specify a coordinate pair for the graph.

The menu titled **Graph** controls the user interface to the graphing capability of *Geometry Explorer*. There are eight options under this menu which control graphing: **(Show/Hide) Axes**, **Grid (On/Off)**, **Add Function to Graph...**, **Add As (x,y) Point from Measures**, **Add Point on**

Function from x-Point, **Iterate Function from Point...**, **Derivative of Function**, **and Input Box for Function**. We will make use of three of these options in this tutorial. For more information on the analytic geometry capabilities of *Geometry Explorer* see Chapter 6.

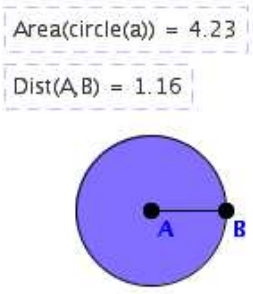
1. **(Show/Hide) Axes:** Coordinate axes are always available in a *Geometry Explorer* Euclidean session. However, the axes are hidden initially. To see the axes use this menu option. Once the axes are visible, the menu item will change to **Hide Axes**, allowing one to make the axes invisible.

2. **Add As (x,y) Point from Measures:** We can add coordinate pairs to the axes by selecting (in order) 1) the measurement that will serve as the x-coordinate and 2) the measurement that will serve as the y-coordinate. Once we click on this menu choice a point will be created at the (x, y) point corresponding to the measures selected and their relative distances on the axes in relation to the origin and unit points of the graph.

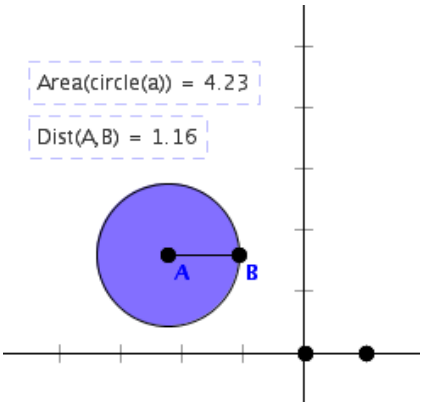
3. **Grid (On/Off):** This menu option will turn on and off a grid for the coordinate system. To enable this option, the coordinate axes must first be visible. Note: Having the grid on will slow down the dynamic behavior of *Geometry Explorer*. Use the grid only when absolutely needed.

As an example, let's explore the relationship between the radius and area of a circle.

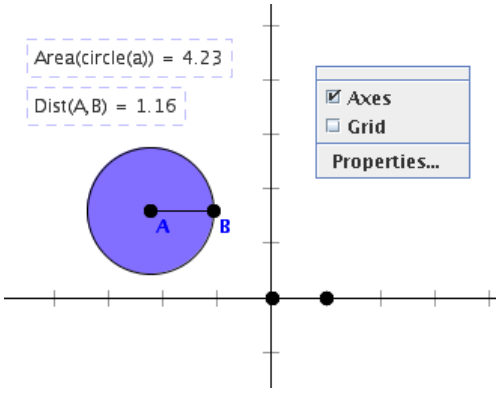
Create a circle with center A and radius point B. Select the circle and click on the Filled Circle/Arc tool of the Construct Panel (third from left in the second row). Select the circle area by clicking inside of it. Go to the **Measure** menu and select **Area**. Next, measure the distance between points A and B by selecting the two points and choosing **Distance** under the **Measure** menu.



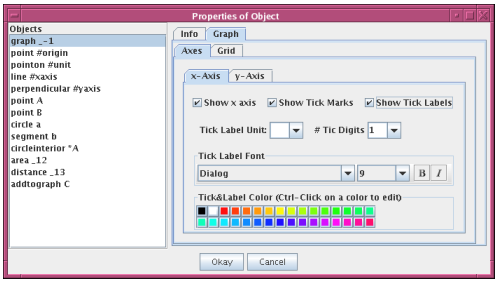
Choose **Show Axes** under the **Graph** menu to make the coordinate axes visible. We may wish to move the axes. To do so, select the origin point and drag it until the axes are in the desired position. In the figure to the right we have moved the axes down and to the right.



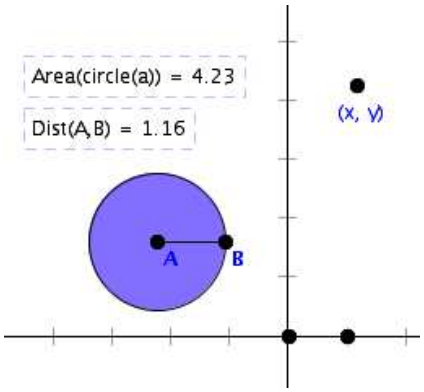
We may wish to have labels on the tick marks on the axes. To make the tick marks visible, click on a clear space of the Canvas using the right mouse button (or Apple-click on a one-button Macintosh mouse). A menu will pop up at the cursor as shown. Click on the "Properties"option.



A dialog box will pop up allowing for the setting of various properties of the grid and axes. Click on the option labeled "Show Tick Labels" in both of the tabs labeled "x-Axis" and "y-Axis" to make the labels appear. Then, close the Properties Dialog box.



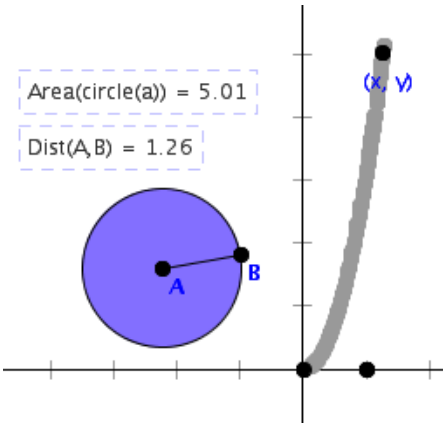
To create a point on the graph that represents the two measures we have calculated (i.e. the distance and area measurements) we select the distance measure (in the figure at right we would click on the text "Dist(A,B)=1.16"). Then we select the area measure and choose **Add as (x,y) Point from Measures** from the **Graph** menu.



The point that *Geometry Explorer* adds to the graph may lie off of the visible Canvas. To scale back the coordinate system, we can just move the unit point on the x-axis away from the origin until the (x, y) point is visible. By moving the unit point away from the origin we will scale down the radius and area measurements since the unit length on the axes will stretch out, but the circle will remain fixed.

The relationship that the point (x, y) represents is that of the radius to the area of a circle. As we move the radius point of the circle, the point (x, y) changes also. To see what path this point takes, we can use the tracing feature of *Geometry Explorer*.

Click on the (x, y) point and choose **Trace On** from the **View** menu. As we move the radius point of the circle, we will see a trace of the moving point (x, y). Note that the trace appears to be that of a quadratic relation, as it should, since the area of a circle is π times the radius squared (or πr^2).



2.6 Tutorial 6 Hyperbolic Geometry

One of the greatest mathematical discoveries of the 1800's was that of non-Euclidean geometry. At a basic level the difference between Euclidean and non-Euclidean geometry is in the area of parallels. In Euclidean geometry

given a line and a point not on the line there is only one line parallel to the given line through the point. In non-Euclidean geometry there may be many lines parallel or none parallel. *Geometry Explorer* provides a non-Euclidean Canvas with which to explore a geometry where there are many parallels, Hyperbolic Geometry. In particular, one model of Hyperbolic geometry it uses is the Poincaré model. (For more background on the Poincaré model see Chapter 7)

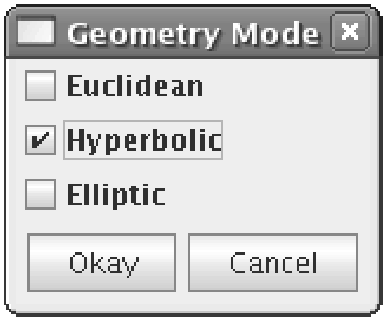
At this point we do not need to understand all of the mathematical theory behind the Poincaré model in order to experiment in this exotic geometry. It is enough to know that the "universe" of the Poincaré model is an open disk. Points on the boundary circle are not included in this universe.

In a geometry that is not Euclidean one needs to define precisely what is meant by a *point*, *line*, and *circle*. Points in the Poincaré model of Hyperbolic geometry are just regular points interior to the Poincaré disk. Lines are defined as circular arcs which meet the boundary of the disk at right angles, and Circles are Euclidean circles, but with Hyperbolic centers somewhat shifted from the usual Euclidean centers.

With these definitions of basic geometric figures in Hyperbolic geometry we can explore how things work in this geometry.

To get started we open up a Hyperbolic *Geometry Explorer* window.

Under the **File** menu you will see a **New** option. Choose this option and you will see a dialog box asking you to specify a type of geometry. Choose "Hyperbolic" and click the Okay button. A new Canvas will open up like that in (Fig. 2.4).



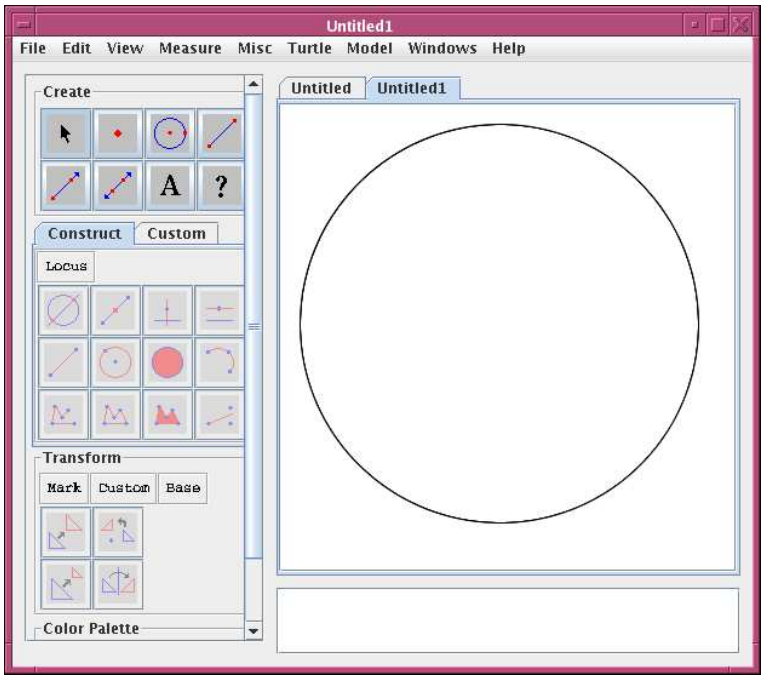


Fig. 2.4 The Hyperbolic workspace main window

Note that the Canvas looks almost identical to the Euclidean Canvas. Working in Hyperbolic geometry with *Geometry Explorer* is essentially no different than working in Euclidean geometry. Almost all of the tools work in both environments, with a few notable exceptions:

1. In the Euclidean canvas the Parallel tool in the Construct Panel is used to construct the *unique* parallel for a line and a point off the line. In Hyperbolic geometry there are no unique parallels. In the Hyperbolic environment, using the Parallel tool (with the same selection of a linear object and a point) will result in the creation of two parallels called *limiting parallels*. In (Fig. 2.5) we see the two (unique) limiting parallels to line l through point P. (The parallels are the two lines passing through P.)

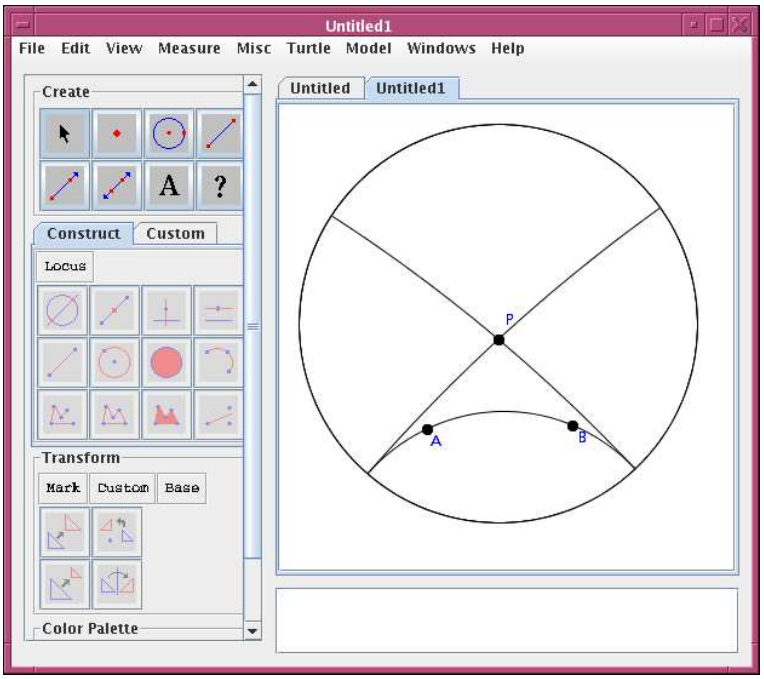


Fig. 2.5 Limiting Parallels to Line "l" through point "P"

These are parallels since they are lines through P that do not intersect line l. (They do intersect at the boundary, but they are still parallel as the boundary is not considered part of the Hyperbolic universe.)

2. In the Euclidean canvas, circles and arcs can be defined using three points. This construction depends on the Euclidean parallel postulate, (i.e. the uniqueness of parallels) and thus is not available in the Hyperbolic canvas.

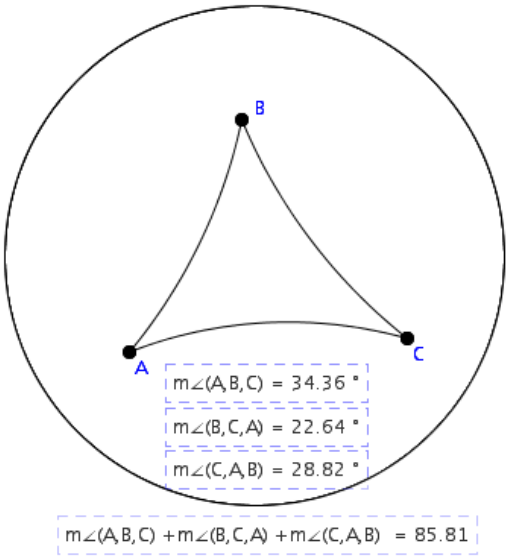
3. There is no **Graph** menu in the Hyperbolic window.

4. Some measurements are different. There is no x- or y-coordinate measure and no slope measure. These depend on a coordinate system. However, there is a new measure: the **Defect** measure. The defect is the difference between 180 degrees and the angle sum of a triangle in Hyperbolic geometry. (More on this below).

This similarity of user environments for the two geometries was deliberately designed to give the user the maximum opportunity to explore and

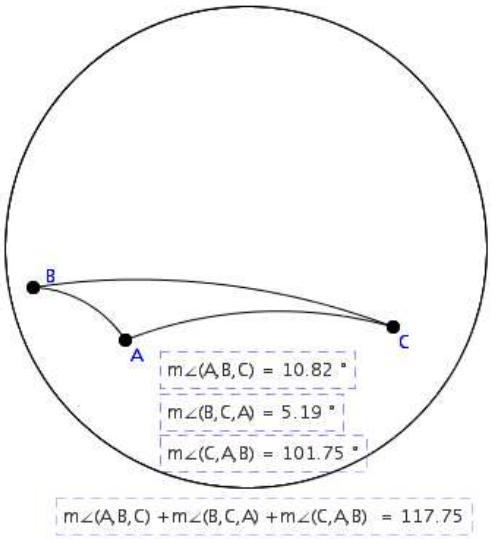
contrast these two different geometric universes using similar basic geometric ideas, such as points, lines, perpendiculars, rotations, measurements, etc. The goal in working in these geometries is to develop an intuition for how it "feels' to live in one geometry versus another.

As a first example in Hyperbolic geometry, let's create ΔABC and measure its interior angles just as we did in Tutorial 4 on measurement. Also, let's use the Calculator to find the sum of the interior angles and add this compound measure to the Hyperbolic Canvas. (Review Tutorial 4 if needed.)

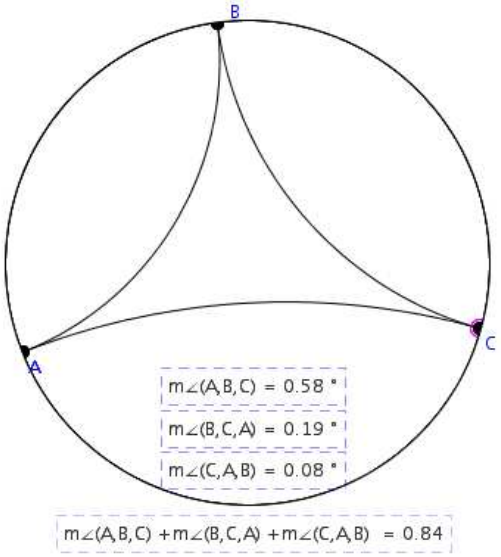


The first thing that we notice is that the sum of the angles is *not* 180 degrees, as it was in Euclidean geometry. In fact, the sum is *less* than 180 degrees. This is actually a *theorem* in Hyperbolic geometry—the sum of the interior angles in a triangle is always less than 180 degrees. A theorem is a statement which can be shown to always be true in a geometry.

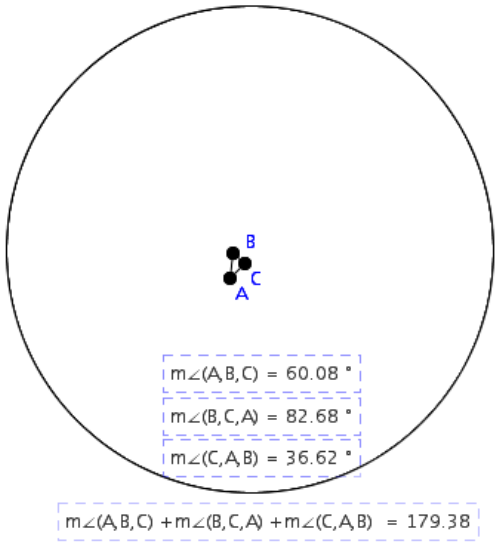
Just to verify that the angle sum is always less than 180 degrees, take any one of the triangle vertex points, say point B, and move it around. The angle sum is always less than 180 degrees. Also we notice another difference from Euclidean geometry—the interior angle sum is not constant for triangles, as it was in Euclidean geometry.



Move the triangle vertexes toward the boundary of the disk. Note how the angle sum approaches 0.



Now move the vertexes toward the center. As the vertexes get closer and closer together the angle sum appears to become 180 degrees!



In fact this is another property of Hyperbolic geometry—locally, i.e., in very small areas, Hyperbolic geometry is almost Euclidean. This fact relates to modern cosmologists' views of the universe. Some cosmologists believe that the universe is curved and possibly hyperbolic. If it was, then it would be difficult for us on Earth to experience this since we are in a very tiny area of the universe, and in tiny areas Hyperbolic geometry is essentially Euclidean!

Hyperbolic geometry is, in the words of one of its discoverers, "a strange new universe". [2, page 129] That is how Janos Bolyai described his discovery of non-Euclidean Geometry to his father in 1823. For more on this strange universe see Chapter 7.

2.7 Tutorial 7 Elliptic Geometry

In non-Euclidean geometry there may be many lines parallel, or none parallel, to a given line through a point not on that line. Elliptic geometry is the geometry where there are no parallels.

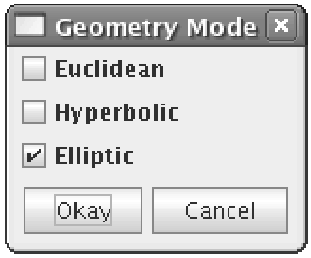
Points in Elliptic geometry are regular points interior to the unit disk. Lines are defined as circular arcs which are the images of circle arcs on the unit sphere which are mapped into the plane under stereographic projection. Circles are Euclidean circles, but with elliptic centers somewhat shifted from

the usual Euclidean centers. For more information on the precise definition of Elliptic geometry, see the text by Henle [4].

Let's explore how things work in this geometry.

To get started we open up an Elliptic *Geometry Explorer* window.

Under the **File** menu you will see a **New** option. Choose this option and you will see a dialog box asking you to specify a type of geometry. Choose "Elliptic" and click the Okay button. A new Canvas will open up like that in (Fig. 2.6).



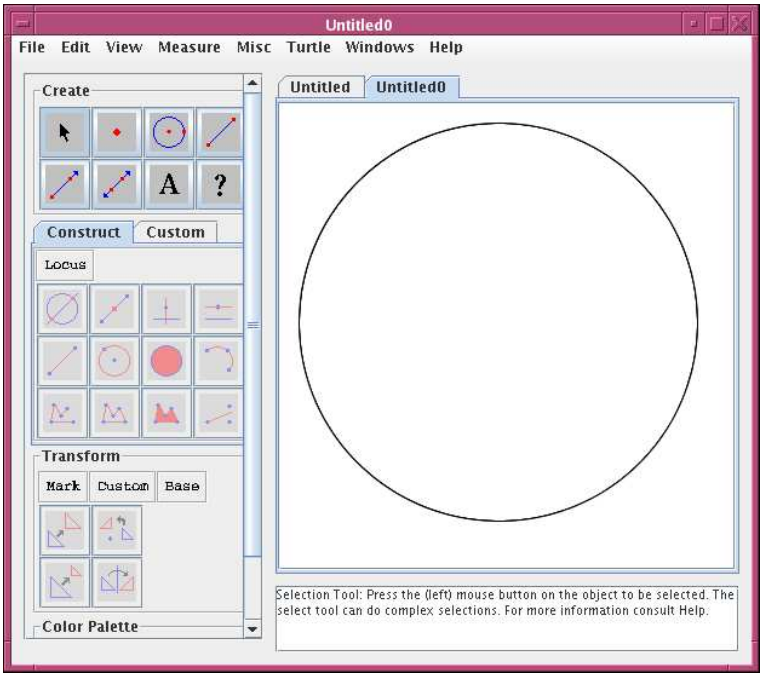


Fig. 2.6 The Elliptic workspace main window

Working in Elliptic geometry with *Geometry Explorer* is essentially no different than working in Euclidean geometry. Almost all of the tools work in both environments, with a few notable exceptions:

1. In the Euclidean canvas the Parallel tool in the Construct Panel is used to construct the *unique* parallel for a line and a point off the line.

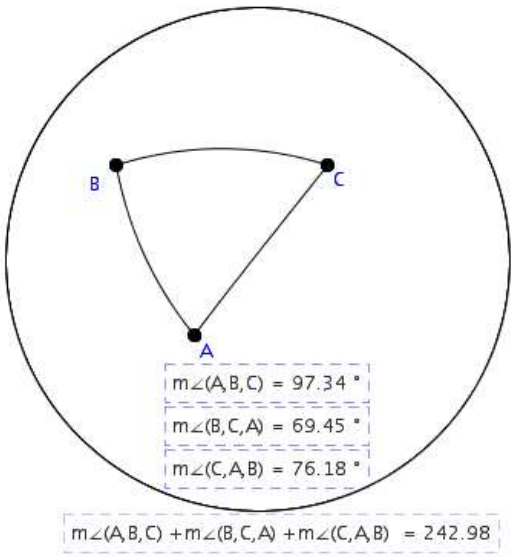
In Elliptic geometry there are no parallels, so this tool will never be enabled.

2. In the Euclidean canvas, circles and arcs can be defined using three points. This construction depends on the Euclidean parallel postulate, (i.e. the uniqueness of parallels) and thus is not available in the Elliptic canvas.

3. There is no **Graph** menu in the elliptic window.

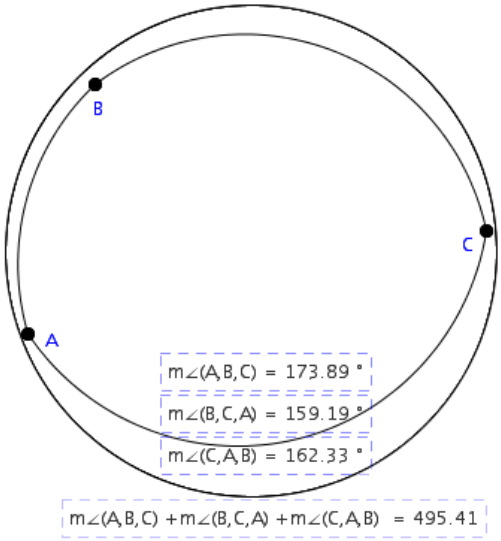
4. Some measurements are different. There is no x- or y-coordinate measure and no slope measure. These depend on a coordinate system. However, there is a new measure: the **Excess** measure. The excess is the difference between the angle sum of a triangle in Elliptic geometry and 180 degrees. (More on this below).

As a first example in Elliptic geometry, let's create ΔABC and measure its interior angles just as we did in Tutorial 7 on Hyperbolic geometry. Also, let's use the Calculator to find the sum of the interior angles and add this compound measure to the Elliptic Canvas. (Review Tutorial 4 if needed.)

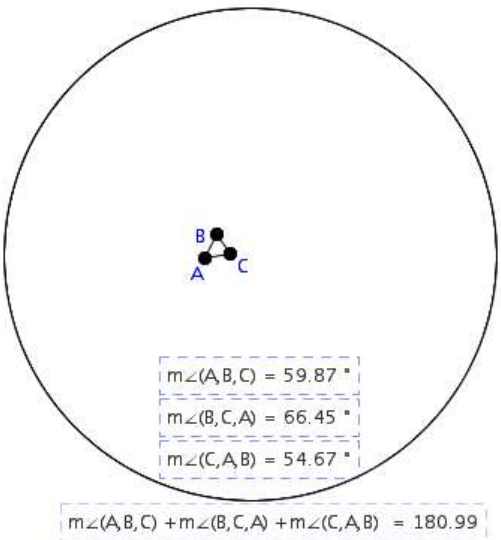


The first thing that we notice is that the sum of the angles is *not* 180 degrees, as it was in Euclidean geometry. In fact, the sum is *more* than 180 degrees. This is actually a *theorem* in Elliptic geometry—the sum of the interior angles in a triangle is always more than 180 degrees.

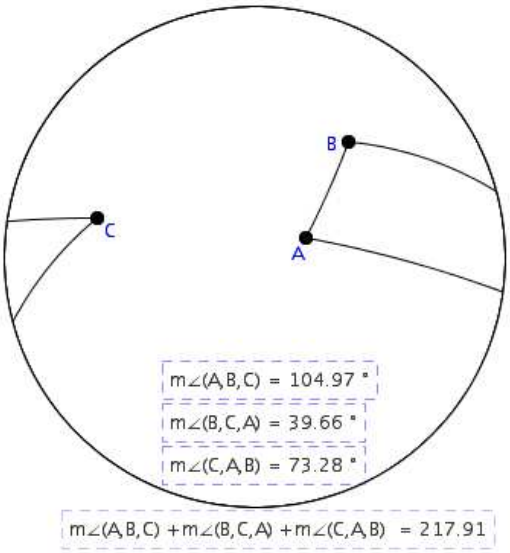
Move the triangle vertexes toward the boundary of the disk. Note how the angle sum grows larger and larger.



Now move the vertexes toward the center. As the vertexes get closer and closer together the angle sum appears to become 180 degrees! In fact this is another property of Elliptic geometry—locally, i.e., in very small areas, Elliptic geometry is almost Euclidean.



Now, expand the triangle by moving point C to the right. Then, multi-select sides AC and BC and drag the whole triangle to the right. Point C will disappear off the right of the disk area and re-appear from the left. This is due to the fact that Elliptic geometry is a *bounded* geometry. That is, the entire universe of Elliptic geometry lies in a region of bounded area.



2.8 Tutorial 8 Recording Geometric Macros

It is useful to be able to record the steps involved in a construction for playback later. This storing of steps for later use is often referred to as a "macro" in applications like spreadsheets or word processors. *Geometry Explorer* provides this capability in two ways – through the use of a Recorder window and by the creation of Custom Tools.

2.8.1 Recorder Windows

The Recorder window can be opened up by choosing **New Recording** under the **File** menu of the main *Geometry Explorer* window.

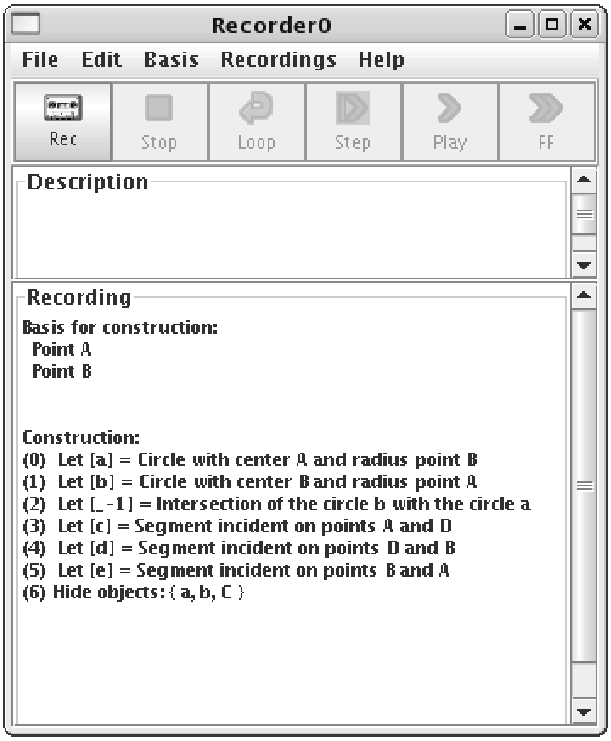


Fig. 2.7 The Recorder Window

In (Fig. 2.7) we see a Recorder window that shows a recording of the equilateral triangle construction as described in Tutorial 2. (Review Tutorial 2 if needed)

The row of buttons labeled Rec, Stop, Loop, Step, Play, and FF control how steps are recorded and played back. The text area labeled "Recording" shows the steps that were recorded. The text area labeled "Description" is used for providing an explanations of what the recording is supposed to do. For this construction there are 6 steps. The steps exactly match those described in Tutorial 2 except for the last step. In the last recorded step we hide the two circles (labeled a and b) that were used to construct the triangle and we also hide the second intersection point (D), leaving a simple equilateral triangle on the Canvas. We now look at specifically how this recording was made.

To record the equilateral triangle construction we start with an empty Canvas and an open Recorder window. To start the recording we click on the Rec button in the Recorder window. Next, we start the construction of the equilateral triangle. As we construct the triangle, the Recorder "listens" in

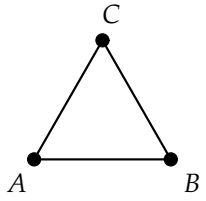
and records each step of the process. It also differentiates between geometric objects that are created and those that are constructed from other objects already existing. For example, the first step in building the equilateral triangle is to draw a circle defined by two points (in this case points A and B). These two points are recorded as the *Basis* for the construction since they are not built from already defined objects. The circle is then listed under step (0) of the recording, as it is built from points A and B which already exist. The next step in the construction is to draw another circle with center equal to the radius point of the first circle and radius point equal to the center point of the first circle. This circle is again built from already existing objects and thus is listed in the recording as step (1). We continue with the construction until the triangle is built. At that point we click the Stop button in the Recorder window to finish the recording.

To playback the recording we first select a set of basis elements which exactly match the basis that was recorded. Thus, we must first select two points that will serve as a new basis for the construction of a new equilateral triangle. Once we have selected the right number and type of objects for playback, the Step, Play, and FF buttons will become active in the Recorder window. Clicking on any of these playback buttons in the Recorder will start the playback of the recorded construction.

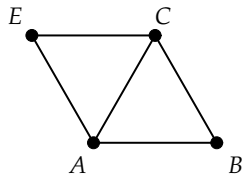
The Step button carries out one step in the construction. To execute the next step the Step button must be clicked again. The Play button carries out the steps of the construction in succession and highlights the corresponding step in the Recorder window as it continues. The FF button carries out the steps in the construction as rapidly as possible. At any point in the playback of a construction the Stop button can be clicked to stop the playback.

The Loop button is used to record constructions that are *recursive* in nature, i.e. constructions that feed back upon themselves. Look in Chapter 10 for more information on this feature.

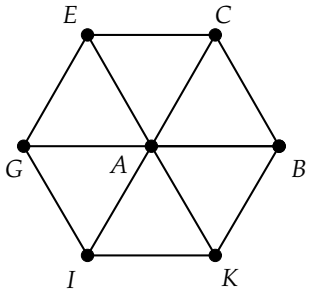
Once a recording has been made it can be used again and again. Create two points A and B on the Canvas. Since two points are all that is needed for matching the Basis of our recording above, we can select A and B and playback the recording, yielding an equilateral triangle with new point C, as shown.



Now, select points A and C (in that order) and playback the recording on these two points. We get a new equilateral triangle ACE.



If we continue selecting pairs of new triangle points, we can generate a regular hexagon (regular means all sides and interior angles are congruent).

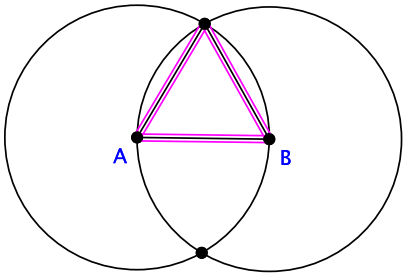


Note that if we try to move most of the points on this hexagon (e.g. if we move C,E,G,I,or K) the entire hexagon will move. This is because these points are constructed from other objects, and thus are not "free" to move as they will. However, points A and B were originally free and since they define the size of all the equilateral triangles in the hexagon, moving A or B will change the size of the hexagon.

2.8.2 Custom Tools

The second method of recording a macro is by the creation of a Custom Tool. The difference between recording a construction using a Recorder Window versus a Custom Tool is that a Recorder Window "listens in" as you carry out a construction and then stores the result. A Custom Tool is created *after* you have finished a construction.

For example, suppose we have carried out the construction of an equilateral triangle by intersecting two circles, as described in Tutorial 2. We create a Custom Tool by first selecting the objects that we want the tool to create, in this case the three segments that make up the triangle.



To create the Custom Tool we click on the tab labeled "Custom" in the Construct area of the Tool Panel. Then, we click on the button labeled "Custom Tool" to bring up a popup menu as shown in Figure 2.8)

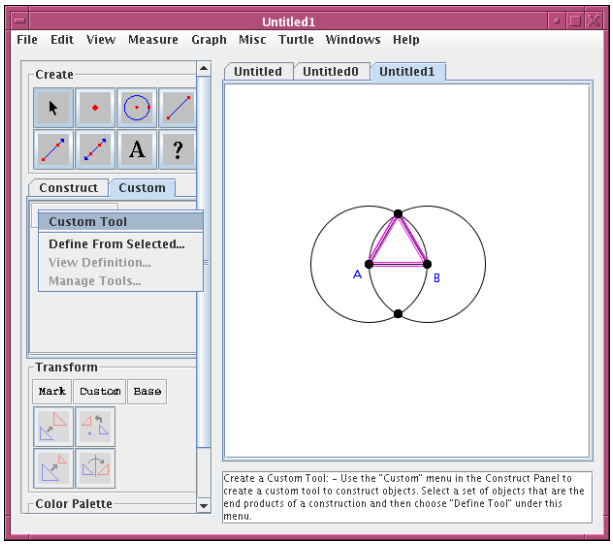
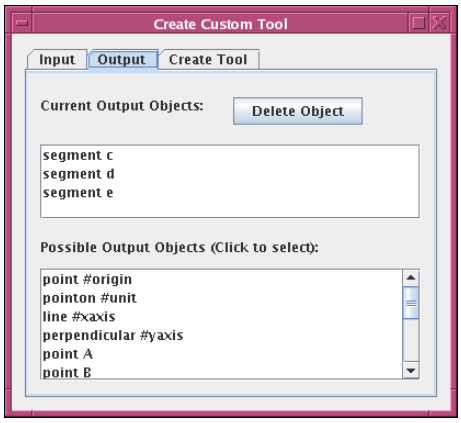
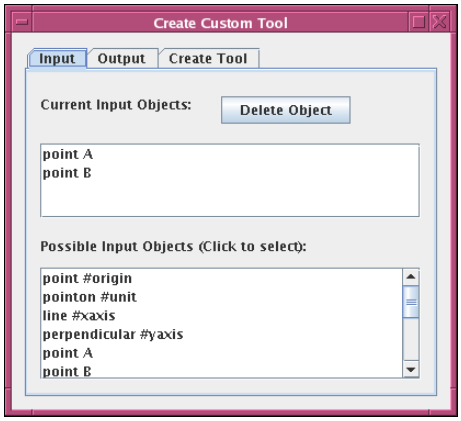


Fig. 2.8 Custom Tool Creation

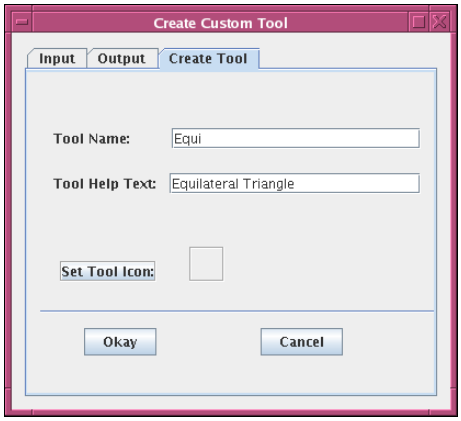
Select "Define from Selected...". A dialog box will pop up with three tabbed panels labeled "Input", "Output", and "Create Tool." The Output panel will be showing. In this window we see a list with the three objects we want the tool to create. These are the *ouput* of the tool.



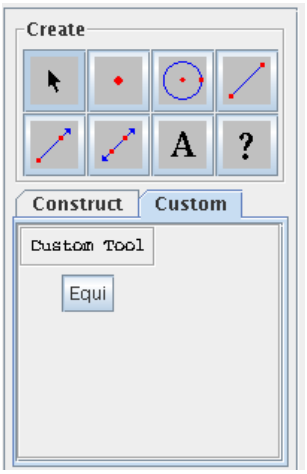
If we click on the "Input" tab we see a different list of objects. *Geometry Explorer* calculates all parent objects which the three segments depend on. In this example, there are many – points A and B, the two circles, and the points of intersection. The deepest common ancestors on which everything depends are the two points A and B. These will be the necessary *input* to the new tool.



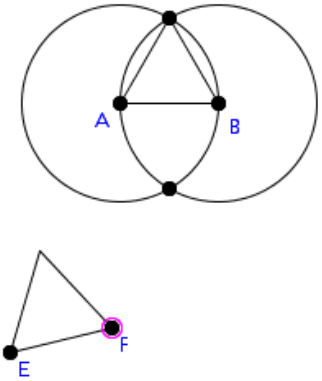
To finish the creation of the tool, click on the "Create Tool" tab. In this panel, we set the tool's name, any help text, and an icon for the new tool button that will be created. In our case, we name the tool "Equi" and have help text describing what the tool creates. For this example, we will not define an icon. Click "Okay" to define the tool.



Once the tool is defined a new button will be created in the Custom panel in the Tool Panel as is shown here (the button labeled "Equi"). Every time a new tool is defined it will be added to this subpanel of the Tool Panel.



To use the tool, first click on its button (the one labeled "Equi") and then click twice in the Canvas. Two points (E and F will be created and then all of the original construction of the equilateral triangle will be automatically carried out, beginning with points E and F. Note that all intermediate objects (such as the circles and intersection points) will be hidden.



Much more information on using Custom Tools can be found in Chapter 10

2.9 Tutorial 9 Turtle Geometry

Turtle geometry was created as part of the development of the LOGO programming language. LOGO was designed in part to give children a relatively easy way to program a computer. In turtle geometry one imagines a small turtle on the computer screen. This turtle understands commands like move forward, turn left, turn right, change color, among others.

Turtle Geometry is extremely useful in drawing simple shapes quickly, and also in creating fractals of almost any type. (For more on the use of

turtle geometry in creating fractals see Chapter 8.)

The turtles in *Geometry Explorer* can understand these basic commands:

1. **Forward** Move turtle forward one unit.
2. **Back** Move turtle backward one unit.
3. **Draw Forward** Move turtle forward one unit and draw a segment.
4. **Rotate Left** Rotate turtle counter-clockwise by a set angle.
5. **Rotate Right** Rotate turtle clockwise by a set angle.
6. **Push** Store the turtle's current heading and length.
7. **Pop** Restore the turtle's stored heading and length.

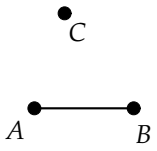
The turtle starts out with a specified heading and length. The *heading* is the direction in which the turtle will move. The *length* is how far the turtle should move when told to go forward or backward. The heading and length are given by a vector—a pair of points. The vector's length is just the distance between the points, and the vector's heading is given by an arrow from the first point towards the second.

A turtle must also know what angle to turn by. This is specified by a set of three points—the initial, vertex, and terminal points of an angle.

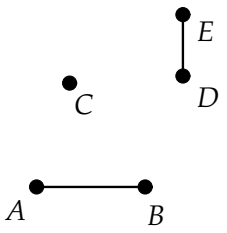
As an example, let's use a turtle to construct a regular pentagon.

First, we need to create a turtle. As described above we need to define a vector and angle. The angle that will be needed for our pentagon is one of 72 degrees.

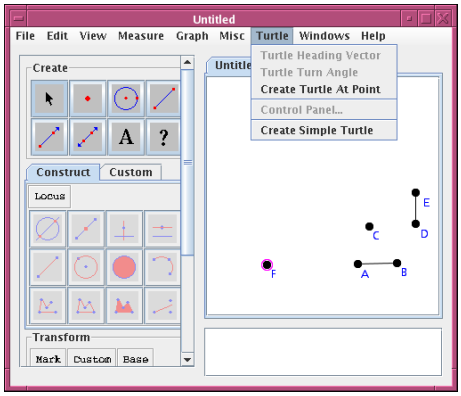
To construct a 72 degree angle first construct segment $\overline{AB}$. Then select A and set it as a center of rotation by choosing **Center** under the **Mark** menu in the Transform Panel. Click on **Rotation** under the **Custom** menu in the Transform Panel and type in 72 for the angle and hit the Okay button. Select point B and click on the Rotation tool in the Transform Panel to construct a new point C. Then $\angle BAC$ is a 72 degree angle.



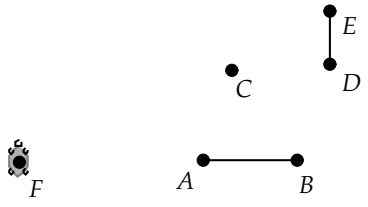
To define this angle as a turtle turning angle, we select B, A, and C (in that order) and choose **Turtle Turn Angle** under the **Turtle** menu. Next, construct a segment $\overline{DE}$ and then select D and E (in that order) and choose **Turtle Heading Vector** under the **Turtle** menu.



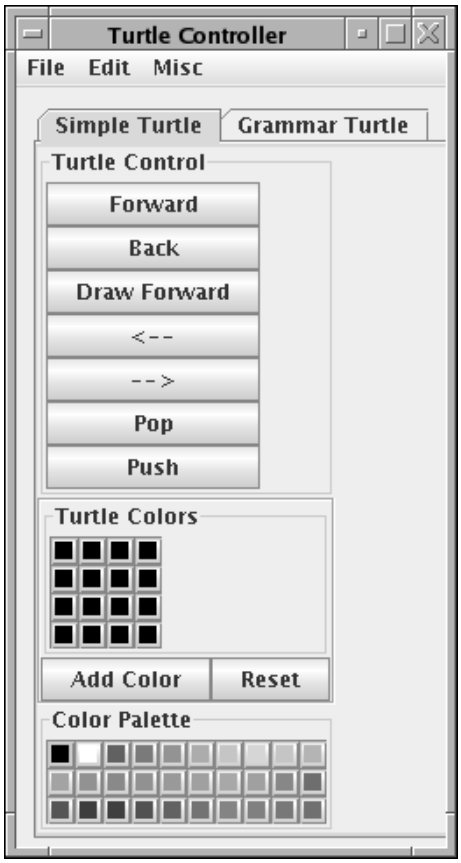
At this point we have stored an angle and vector that we can use to define a turtle. To create the turtle, however, we need to specify a point from which it will start. Create point F as shown and select it. At this point the **Create Turtle At Point** menu under the **Turtle** menu will be active.



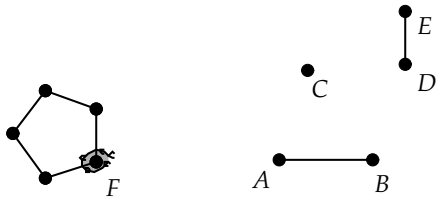
After choosing **Create Turtle At Point**, a small green turtle will be created at point F.



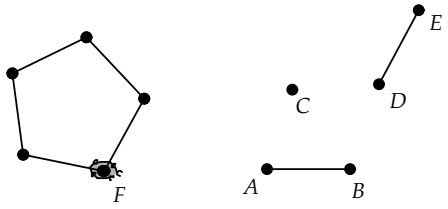
Also, a Turtle Controller window will pop up. We will use just the section of this window labeled "Simple Turtle". (For more information on turtles and the use of the Turtle Controller window see Chapter 8.)



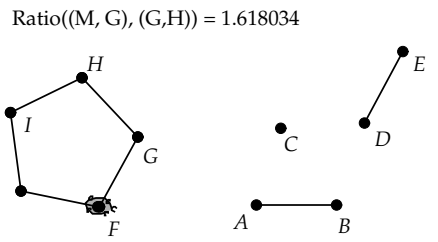
Now we will use our turtle to construct a pentagon. We will carry out a sequence of Draw Forward and Turn Left ($<--$) actions by pressing the Draw Forward button followed by the Turn Left button a total of five times, yielding the pentagon shown at the right.



Suppose we move point D. By lengthening the distance from D to E we lengthen all of the turtle movements. Also, by changing the heading of $\overline{DE}$ we change the orientation of the pentagon drawn by the turtle. However, moving point A does not change $\angle BAC$ and thus has no effect on the pentagon.



The regular pentagon has many fascinating properties. One of these has to do with the ratio of a diagonal of the pentagon to a side. Select points I, G, and H and choose **Ratio** from the **Measure** menu. The ratio of IG to GH is then calculated. This ratio is called the *Golden Ratio*. The golden ratio pops up in many surprising places in nature. A good reference for this topic is Huntley's book *The Divine Proportion* [5].



Chapter 3

Constructions

> When we cannot use the compass of mathematics or the torch of experience...it is certain that we cannot take a single step forward.
>
> —Voltaire (1694–1778)

In a geometric construction one builds a new geometric figure from *existing* geometric objects. Constructing the midpoint of a segment is different than creating a point in the plane, for example, as no pre-existing objects are needed to create the point, whereas a midpoint makes no sense unless it refers to an existing segment.

In classical Euclidean geometry all constructions are based on a straightedge and compass. In other words, all figures are composed of points, lines (or portions of lines), and circles, and on intersections of such objects. A figure that can be built in this fashion is called *constructible*. The equilateral triangle is a constructible figure, as we saw in Chapter 2. One of the most ancient of questions in geometry is to determine those figures which are constructible. For example, it is possible using a straightedge and compass to construct the bisector of an angle. That is, given an angle we can, using just a straightedge and compass, find a fourth point inside the angle so that the two new angles formed by this interior point are equal. Is it possible to trisect an angle? This question vexed geometers from the time of Euclid (about 300BC) until the 1800's, when it was finally answered in the negative. For more information on constructibility see the text by Hvidsten [6] or Smart [11].

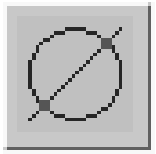
In the *Geometry Explorer* environment, the basic tools in the Create Panel provide for straightedge and compass constructions. Using just a

straightedge and compass it is possible to build complex figures such as bisectors and perpendiculars. However, figures like angle bisectors and perpendiculars are so useful and are needed so often, that these have been provided as built-in tools in the Construct Panel. These tools provide short-cuts to what one could do using *just* the creator tools (i.e. using straight edge and compass).

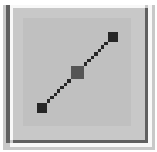
There are two tools that do not fall neatly into this category of constructibility—the two filled-area tools. These are useful in creating areas that can be measured and changed dynamically.

3.1 Tools in the Construct Panel

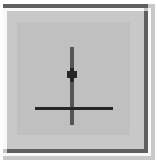
We now look at each construction tool in detail. For each tool we list the objects from which it is built.



The Intersection Constructor: To construct the intersection of two geometric objects, first select the two objects and click on the Intersection tool. Objects which can be used for intersections include lines, rays, segments, circles, and arcs.

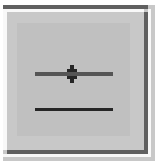


The Midpoint Constructor: To construct the midpoint of a segment(s), select any segment(s) and click on the Midpoint tool. If you select more than one segment, the midpoints of each of the selected segments will be constructed.

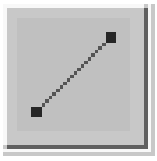


The Perpendicular Line Constructor: Two selections must be made for this particular construction to be possible: a linear object (line, segment or ray) and a point. Clicking on the Perpendicular tool will result in the perpendicular to the linear object through the selected point.

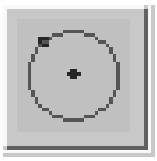
The Parallel Line Constructor: This construction is quite similar to the perpendicular line construction. A linear object and a point must be selected. Clicking on the Parallel tool will result in the parallel to the linear object through the selected point.



The Segment Constructor: Select two points and the Segment tool will become active. Clicking on this tool will result in a segment connecting the two selected points.



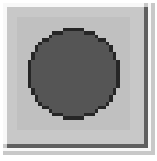
The Circle Constructor: Circles can be constructed using this tool in three ways.



1. Select two points—the first to serve as the center of the desired circle and the second to serve as a point on the desired circle.
2. Select a point and segment. The point will serve as the center of the desired circle and the segment will be the desired radius length.
3. Select three points, all of which will be located on the desired circle's circumference.

Once one of these selections has been made, the Circle tool will become active in the Construct panel and clicking on it will construct a circle. (Note: In Hyperbolic or Elliptic geometry, only the first two options apply)

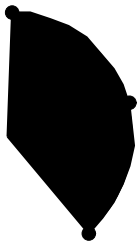
The Filled Circle/Arc Constructor: Once you have selected a circle or an arc, the Filled Circle/Arc tool will be active. Clicking on this tool will fill in the interior of the circle or arc. If you select more than one circle or arc, every circle or arc that has been selected will be filled.



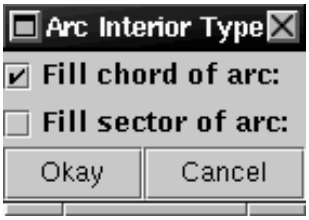
Note that arcs can be filled in two ways: we can fill the *chord* of an arc or fill in the entire *sector* of an arc.

Here we see an arc where the chord defined by the arc has been filled in.

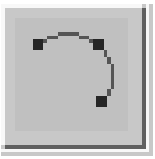
Here is the same arc where the sector defined by the arc has been filled in.



When you use the Filled Circle/Arc tool on an arc, a dialog box will pop up asking which type of filled arc is desired.



The Arc Constructor: Arcs can be constructed using this button in three ways.

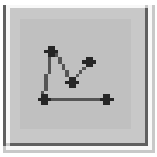


1. Select two points which define the center and radius point on which the arc will be defined.

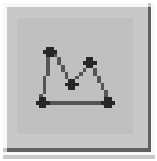
2. Select two points that are attached to an existing circle to define an arc of that circle.

3. Select three points, all of which are desired to be located on the arc's circumference. The first and last points will become endpoints of the arc.

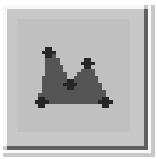
Once one of these options has been chosen and the appropriate selections made, the Arc tool will become active. Clicking on this tool will result in an arc. However, in the first option a dialog box will pop up asking for the initial and terminal angles of the arc (in degrees). (Note: In Hyperbolic or Elliptic geometry, only the first two options apply)



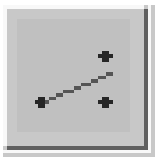
The Open Polygon Constructor: Select a series of vertex points (three or more points are necessary). Clicking on the Open Polygon tool will cause a series of segments to be drawn: from the first point selected to the second, from the second to the third, and so on. If you select the points using the box selection method, the order of connection of the selected points is the order in which the points were created.



The Closed Polygon Constructor: Select a series of vertex points (three or more points are necessary). Clicking on the Closed Polygon will result in a series of segments joining the vertices, including a final segment between the last point selected and the first point selected, thereby closing the polygon.



The Filled Polygon Constructor: Select a series of vertex points (three or more points are necessary). Clicking on the Filled Polygon tool will result in the computer filling the interior of the polygon defined by the vertices.



The Angle Bisector Constructor: Select three points to define an angle. Clicking on the Angle Bisector tool will result in a ray that bisects the angle defined by the three points. The angle bisector construction is oriented, which means that if you select the points in reverse order, a ray will be drawn in the opposite direction.

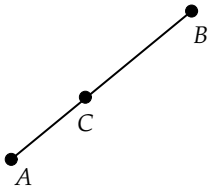
3.2 Using the Locus Tool

The word *locus* is used in geometry to refer to a set of points satisfying some defining property. For example, the locus of points satisfying the property that their distance to a fixed point is constant is a circle.

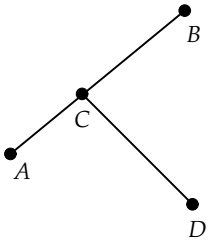
In *Geometry Explorer*, the term locus has a specific meaning. The locus is a geometric construction that is defined by two objects: a point that is *attached* to a one-dimensional object (line,ray,segment,circle, or arc), and any other geometric object (called the *locus primitive*). The locus of the primitive will be a set of copies of that primitive produced as the attached point moves along its one-dimensional path.

3.2.1 A Simple Example Using the Locus Tool

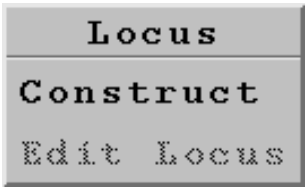
As an example, we create a segment $\overline{AB}$ and attach a point C to this segment.



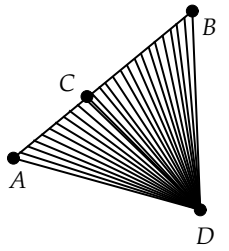
Now, create a segment from some point D (not on $\overline{AB}$) to point C.



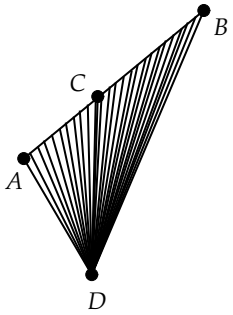
To create the locus of the segment $\overline{CD}$ as point C varies, we first select C and $\overline{CD}$. At this point the **Construct** menu item under the **Locus** pop-up menu in the Construct Panel will be active.



After choosing **Construct**, a set of equally spaced positions of the point C will be calculated along $\overline{AB}$. For each of these positions, the state of $\overline{CD}$ will be calculated and a copy of $\overline{CD}$ in that position will be displayed in a faded color.



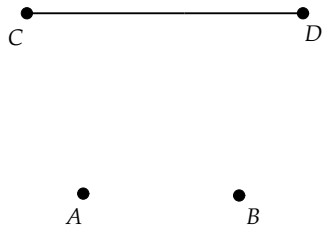
If we move any of the points in the locus construction, the locus of segments for $\overline{CD}$ will automatically be re-calculated and re-displayed.



3.2.2 The Ellipse as a Locus of a Point

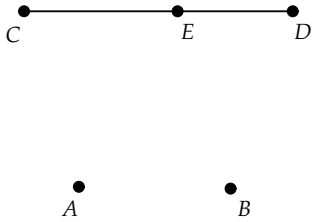
An ellipse is defined in terms of two fixed points, called the foci of the ellipse, and a numerical value d. A point E will be on the ellipse if the sum of the distances from E to the two foci equals d. Since an ellipse is defined as a set of points satisfying a certain property, it is reasonable to expect that the locus construction tool might be useful in constructing an ellipse.

To construct an ellipse we first define two focal points A and B and a geometric value representing d. We will use the length of a segment $\overline{CD}$ to represent d. Create the focal points A and B and $\overline{CD}$ as shown at the right.



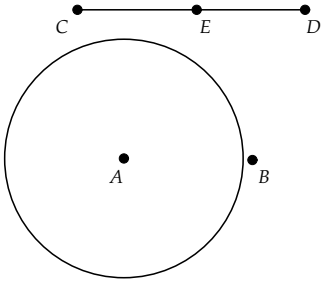
Now, we need to make a geometric construction to represent two distances that always add up to d, i.e to the length of $\overline{CD}$. We will do this by attaching a point E to $\overline{CD}$, and using CE and ED as our two distances, as these two always sum to the total segment length CD.

Using the Point tool click somewhere along $\overline{CD}$ to attach a point E to $\overline{CD}$. Hide segment $\overline{CD}$ by selecting the segment and choosing **Hide Object** from the **Edit** menu. Create $\overline{CE}$ and $\overline{CE}$.

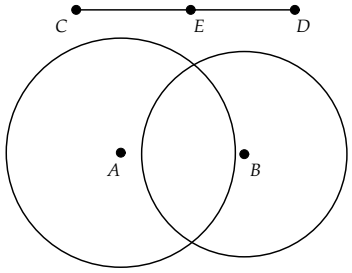


Now that we have defined the focal points A and B and have two lengths (CE and ED) which add up to the value d, we are ready to construct the points an our ellipse. We will do this by first constructing two circles centered at A and B, the one centered at A having radius equal to CE, and the one centered at B having radius equal to ED. A point F where these two circles intersect would be a point for which the sum of the distances to A and B (the sum of the radii of the circles) would be $d = CD$, as desired.

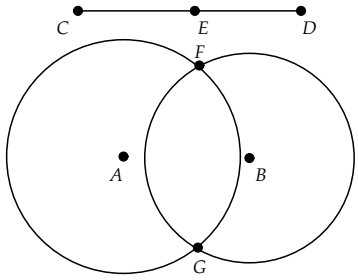
To create a circle with center A and radius CE we select A and $\overline{CE}$. Once this selection is made, the Circle tool in the Construct Panel will be active. Click on this tool to construct the desired circle.



Now, select B and $\overline{ED}$ and click on the Circle tool to construct the desired circle at B.

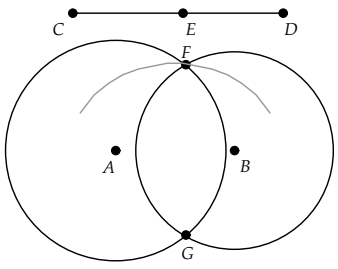


Next, select the two circles. The Intersection tool in the Construct Panel will now be active. Click on it to construct the two intersection points (F and G) for the pair of circles.

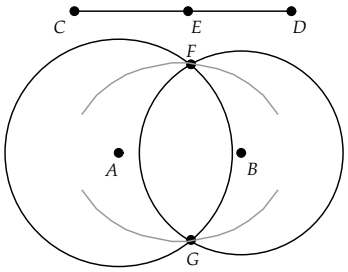


At this point if we drag point E along $\overline{CD}$ and observe the path of F and G we would see that they move along a path that appears to be an ellipse. To see this path, we construct the locus of points F and G with respect to point E.

To construct the locus of F with respect to E, select E and F and choose **Construct** from the **Locus** pop-up menu in the Construct Panel. The constructed locus will be a grayed-out half-elliptical shaped curve.



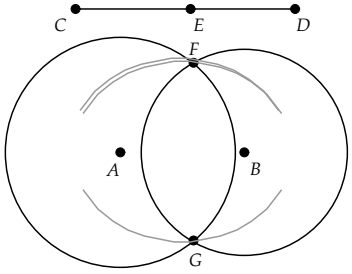
To construct the locus for the other half of the ellipse select E and G and choose **Construct** from the **Locus** pop-up menu.



We can notice a couple things about the ellipse that we have created from the locus of two points. First, our locus curves do not close up to form a complete ellipse. This is because the locus is built from only a finite number of sample points of $\overline{CD}$. By default, *Geometry Explorer* uses 25 sample points to construct a locus. The locus does not close up because we are missing points on $\overline{CD}$ that would correspond to the two boundary points of the ellipse.

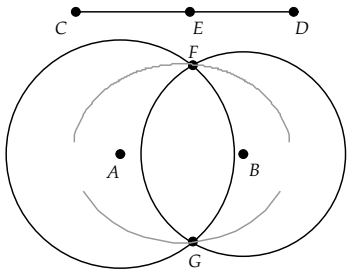
Second, we see a curve rather than a set of points. *Geometry Explorer* actually computes the set of 25 sample points for the locus, but then connects them with segments when displaying the locus. This is to help visualize what the locus looks like.

To correct the sampling problem for the locus, we can increase the number of sample points. To do this, click somewhere on one of the two loci to select it. In the figure at the right, the upper locus is selected. Selection is visually signified by a doubling of the locus set of points that are shifted a bit from the actual locus.

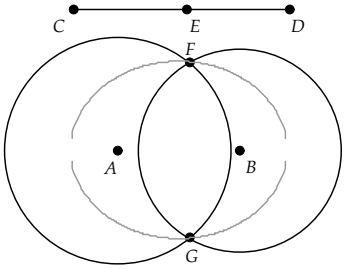


Once the locus has been selected, we can change the number of sample points by choosing **Edit Locus** from the **Locus** pop-up menu in the Construct Panel. A dialog box will pop up allowing one to change the number of sample points. We will increase the sample size to 300.

After clicking Okay in the locus dialog box, we see a more refined locus for the upper half of the ellipse.



Select the lower locus and change its sample size to 300 also. Notice that we have a much more refined set of locus points, but they still do not close up.

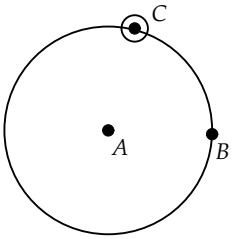


What is happening here is that the two boundary points for the ellipse (where they *should* close up) are the intersection points for the two circles when the circles are almost tangent to each other, that is touching each other at a single point. The intersection point at a tangent is quite unstable. If we move to one side we get no intersections and on the other side there are two intersections. As we sample $\overline{CD}$, we might luck out and hit one of these tangent intersections, but if we do not, then it may be difficult to hit these two tangent points exactly, even when using 300 or more sample points.

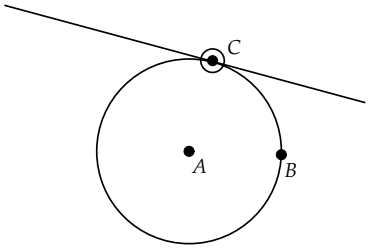
3.3 Tangent to a Circle

A *tangent* to an arc or circle at a point on the circle is a line that intersects the circle at a single point. The construction of the tangent to a circle at a point is one of the classical Euclidean constructions. Since it is such a useful construction, it has been added to the set of constructions available in *Geometry Explorer*.

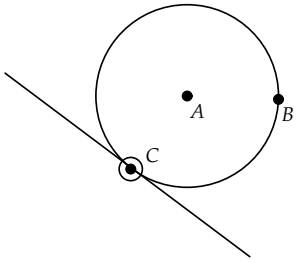
As an example, let us consider the circle with center A and radius B shown at the right. Point C has been attached to the circle. (To attach a point to a circle we use the Point tool to click somewhere on the circle)



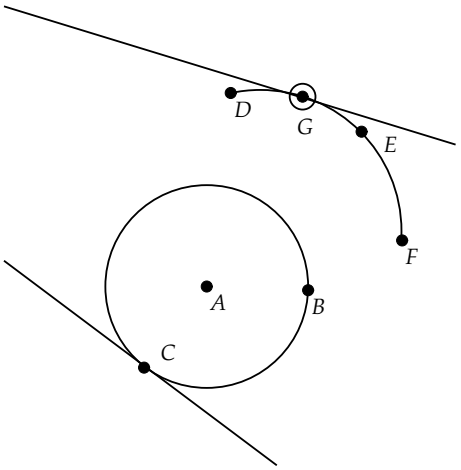
To construct the tangent to the circle at C, we select point C and then choose **Tangent to Arc/Circle/Function at Point** under the **Misc** menu. *Geometry Explorer* will calculate the tangent and construct it in the Canvas.



This tangent line will update dynamically as C is moved.



Similarly we can construct tangents to arcs at points attached to the arcs. In the figure at the right we have constructed an arc on the points D, E, and F. Point G has been attached to the arc and the tangent has been constructed at G.



Chapter 4

Measurements

> It is unworthy of excellent men to lose hours like slaves in the labor of calculation which could safely be relegated to anyone else if machines were used.
>
> —Gottfried Wilhelm von Leibniz (1646–1716)

Measurement is the process of quantifying the size of an object. One of the prime reasons for the study of geometry in ancient times was for handling the measurement of physical objects such as pyramids, roads, plots of land, or the earth itself.

In *Geometry Explorer* measurements are handled by the use of the **Measure** Menu in the Menu Bar. (Fig. 4.1) The items in this menu are split into three groups. The top group includes the basic measurements that one can perform on geometric objects. The middle group consists of a user-input measurement, a slider, and an option to duplicate an existing measurement. The bottom group controls the creation and modification of tables of measurements. Tables will be covered later in this chapter.

Fig. 4.1 The Measure Menu

To make a measurement one first must select the object(s) to be measured. For example, to measure the radius of a circle, one must first select the circle and then make the radius measurement.

We will now review the measurement types in detail. We will do this in four groups: those measurements that are applicable to either Euclidean or Hyperbolic or Elliptic geometry (so-called neutral measurements), measurements applicable only to Euclidean geometry, measurements applicable only to Hyperbolic geometry, and measurements applicable only to Elliptic geometry.

4.1 Neutral Measurements

Neutral measurements are applicable to all three geometries available in *Geometry Explorer*. There are ten neutral measurements: distance, an-

gle, length, ratio, radius, circumference, arcangle, arclength, perimeter, and area. These can be sub-classified by the types of objects they are defined on: points, segments, circles, arcs, and areas.

Note that in some cases a measurement can be defined in more than one way (for example, the distance measurement). Thus, a measurement may bridge more than one category. However, we list each measurement only once.

4.1.1 Point Measurements

1. **Distance:** Distance can be measured in two ways. In the first case we measure the distance between two points by selecting the two points and then choosing the **Distance** menu item in the **Measure** menu. In the second case we measure the distance from a point to a line by selecting the point and the line and choosing **Distance**.

2. **Angle:** An angle is defined by three points: a point on the initial ray, a point at the vertex, and a point on the terminal ray. To measure the degree value of an angle, select the three points defining the angle and choose **Angle** in the **Measure** menu. Note that angles are always measured as *oriented* angles. That is, it matters what order we specify for the initial and terminal points.

4.1.2 Segment Measurements

1. **Length:** The length of a segment is measured by selecting a segment and choosing **Length** in the **Measure** menu.

2. **Ratio:** A ratio is a proportion of two distances or lengths. For example if we had two segments, one of length 4 and one of length 3, then we could say that the two segments are in the ratio of 4 to 3. That is, one is $\frac{4}{3}$ as long as the other. Ratios can be defined in three ways. If we select two segments then the ratio measurement will calculate the ratio of the first segment's length to the second segment's length. If we select three points A, B, and C then the ratio measurement will calculate the ratio of the distance from A to B to the distance from B to C. For three points the middle point is always used twice. For example, if we select B, A, and C then the ratio will be that of BA to AC. If we select four points A, B, C, and D, then the ratio measurement will calculate the ratio of the distance from A to B to the distance from C to D.

4.1.3 Circle Measurements

1. **Radius:** The radius of a circle or an arc can be measured by selecting either the circle or arc and then choosing **Radius** in the **Measure** menu.

2. **Circumference:** The circumference of a circle can be measured by selecting a circle and then choosing **Circumference** in the **Measure** menu.

4.1.4 Arc Measurements

1. **ArcAngle:** The angle defined by an arc and the center of the circle it lies on can be measured by selecting an arc and then choosing **ArcAngle** in the **Measure** menu. The angle will be measured in degrees.

2. **ArcLength:** The length of an arc can be measured by selecting an arc and then choosing **ArcLength** in the **Measure** menu.

4.1.5 Filled Object Measurements

1. **Perimeter:** To measure the perimeter of a filled polygon area select the filled polygon and then choose **Perimeter** in the **Measure** menu.

2. **Area:** To measure the area of a filled polygon, filled circle, or filled arc select the area object and then choose **Area** in the **Measure** menu.

4.2 Euclidean-only Measurements

4.2.1 Point Measurements

1. **x-Coordinate:** To measure the x-coordinate of a point in the Canvas's underlying Euclidean coordinate system, select a point and then choose **x-Coordinate** in the **Measure** menu.

2. **y-Coordinate:** To measure the y-coordinate of a point in the Canvas's underlying Euclidean coordinate system, select a point and then choose **y-Coordinate** in the **Measure** menu.

4.2.2 Linear Object Measurements

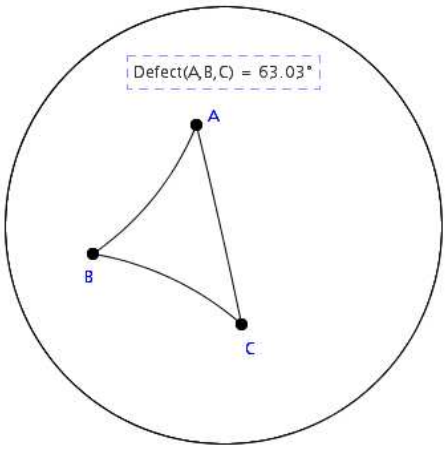
A *linear object* is a line or a part of a line, i.e., a ray or segment.

1. **Slope:** To measure the slope of a linear object select the linear object and then choose **Slope** in the **Measure** menu.

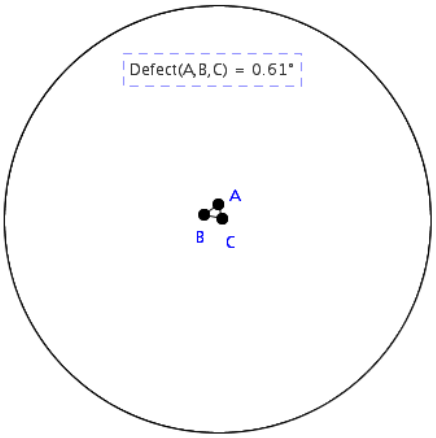
4.3 Hyperbolic-only Measurements

The only measurement that is applicable solely to Hyperbolic geometry is the **Defect** measurement. The defect measurement is defined on a set of three points in the hyperbolic plane. If one considers these three points as being the vertexes of a hyperbolic triangle, then the defect measures the difference between 180 degrees and the angle sum of a triangle in Hyperbolic geometry.

For example, in the hyperbolic triangle at the right, the defect of the triangle has been measured as 63.03 degrees. This means that the angle sum in this triangle is $180 - 63.03$ degrees.



As we noted in the tutorial on Hyperbolic geometry in Chapter 2, Hyperbolic geometry is approximately Euclidean in small areas in the hyperbolic plane. In the figure at the right we have shrunk the triangle down to a small area and notice that the defect has been reduced considerably.

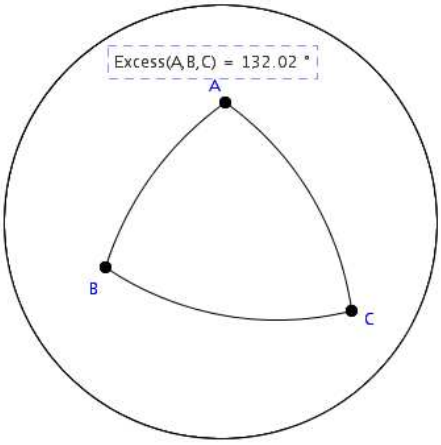


The defect is so-named because it measures how far a hyperbolic triangle is from being Euclidean.

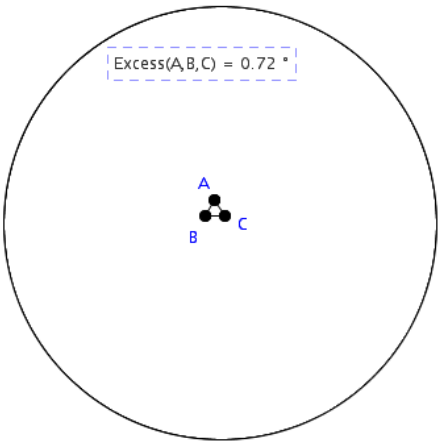
4.4 Elliptic-only Measurements

The only measurement that is applicable solely to Elliptic geometry is the **Excess** measurement. The excess measurement is defined on a set of three points in Elliptic geometry. If one considers these three points as being the vertexes of an elliptic triangle, then the excess measures the difference between the angle sum of a triangle in Elliptic geometry and 180 degrees.

For example, in the elliptic triangle at the right, the excess of the triangle has been measured as 132.02 degrees. This means that the angle sum in this triangle is $180 + 132.02$ degrees.



In the discussion above concerning the defect measurement in Hyperbolic geometry, it was noted that Hyperbolic geometry is approximately Euclidean in small areas. This is also true in Elliptic geometry. In the figure at the right we have shrunk the triangle down to a small area and notice that the excess has been reduced considerably.



The excess is so-named because it measures how far an elliptic triangle is from being Euclidean.

4.5 Precision in Measurements

In the examples shown earlier in this chapter, measurements have been displayed with a precision up to the hundredths place. Actually, all *internal* calculations done by *Geometry Explorer* are done to the maximum accuracy that the host computer is capable of. On one machine there may be 16 digits of precision internally, whereas on another there are 32 digits. For most calculations done by *Geometry Explorer* on modern computers, the internal precision carried through on measurements is sufficiently high to negate numerical errors propagating in the measurements.

The user can change the *displayed* precision of measurements by setting the precision using the Preferences Dialog Box. This can be done by choosing **Preferences** under the **Edit** menu in the menu bar.

4.6 Compound Measurements

Often we would like to compute more complicated expressions that involve measurements. For example, suppose that we wished to find the relationship between the square of the radius of a circle and the area of the circle. One way to do this would be to measure the radius and area of the circle and then create a new compound expression of the area divided by the square of the radius.

To create such mathematical expressions we use *Geometry Explorer's* Calculator.

4.7 Using the Calculator

The Calculator (Fig. 4.2) allows one to create complex mathematical expressions using measurements from the Canvas, numerical quantities, and built-in mathematical functions.

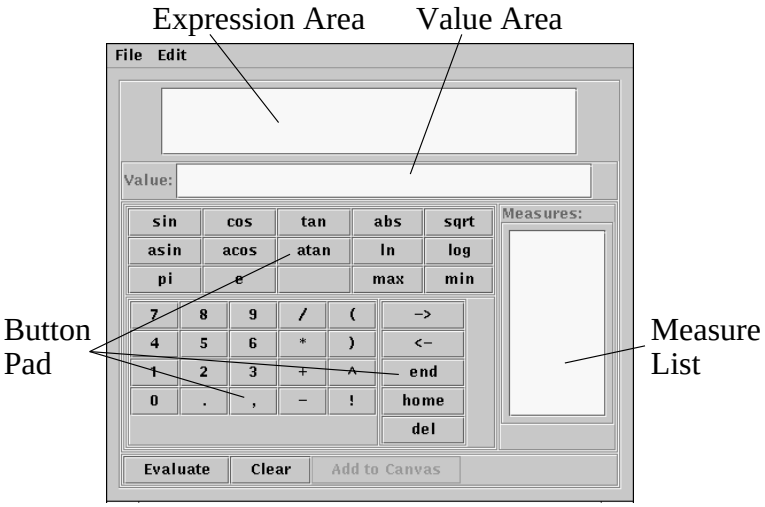


Fig. 4.2 The *Geometry Explorer* Calculator Window

The Calculator window is organized into four main sections: The Expression Area, Value Area, Button Pad, and Measure List. Additionally, there are two menus, the **File** and **Edit** menus, and three buttons on the bottom of the window labeled Evaluate, Clear, and Add to Canvas.

The Calculator interface is designed similarly to that of a modern scientific calculator. The large Expression Area at the top of the Calculator is where the expression that one builds up is visually displayed. The Button Pad consists of a series of buttons that represent numerical values, mathematical functions, mathematical operators (+,-,etc) and editing buttons.

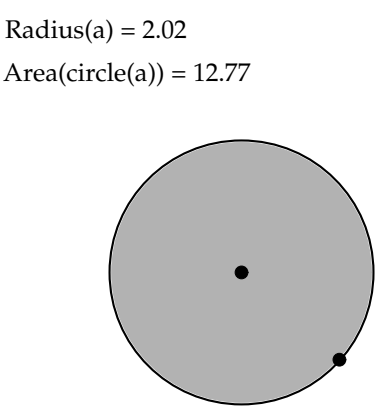
One difference between this Calculator and a hand-held calculator is the Measure List area. As measurements are made in the Canvas, they will appear in this list. One can select measurements from the list and add them to the current expression in the Expression Area. This way, compound measurements can be created.

Another difference is the Value Area. The paradigm for the Calculator is that expressions are built up in the Expression Area as *symbolic* expressions. Once the Evaluate key is pressed, the expression is numerically evaluated and the numerical result is displayed in the Value Area.

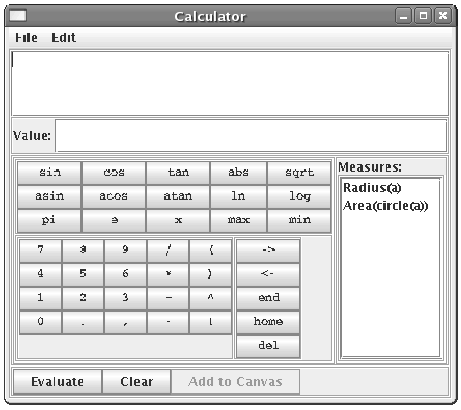
The next example shows how to use the features of the Calculator to create a compound measurement.

4.7.1 Circle Area

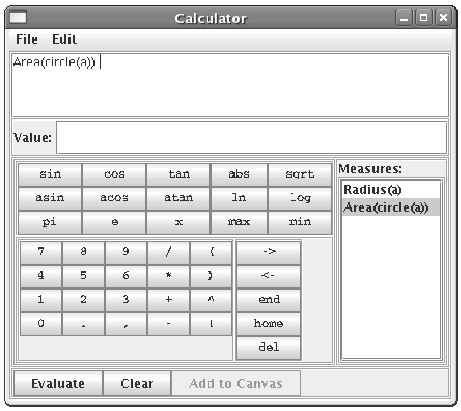
In the figure at the right we have constructed a circle and its interior area and measured the circle's radius and area.



If we now open up the Calculator window (go to the **View** menu and choose **Calculator**) we will see that these two measurements are listed in the Measure List.

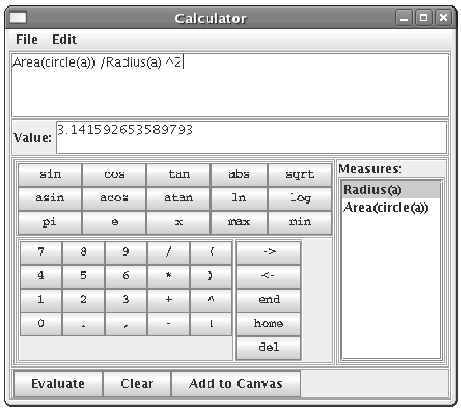


To create a new expression we can use any of the Button Pad keys and also the two measures in the Measure List. Suppose that want to create the expression for the area divided by the square of the radius. We start by double clicking on the area measurement in the Measure List to add this measurement to the Expression Area.



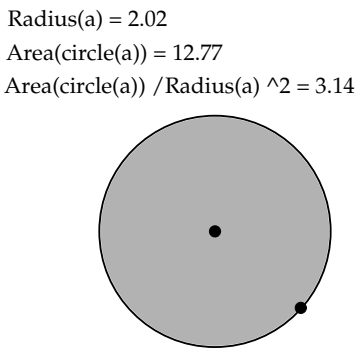
Next we click on the '/' button followed by a double click on the radius measure in the Measure List. Then we click on the '^' key (*power* key) followed by '2' getting the expression as displayed.

Once we click the Evaluate key, the expression that we have built will be evaluated and the result will be displayed in the Value Area as shown. It is no surprise that the result is 3.141592653589793 (pretty darn close to π). Here we have set the measurement precision to be very high. See section 4.5 for more info on precision in measurements.

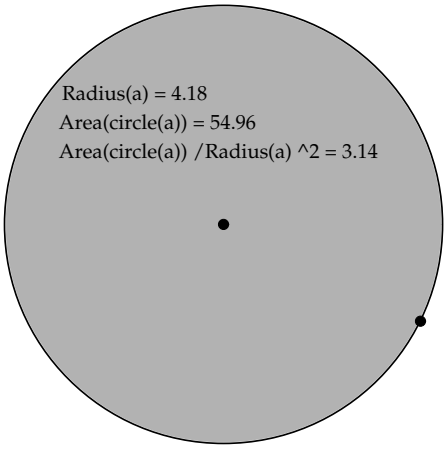


If we want to add this new compound measurement back to the Canvas we would click on the Add to Canvas button. We may wish to close the Calculator. This is done by choosing **Close** from the **File** menu in the Calculator window.

At this point, the Canvas should look like the figure at right.



If we now move the center or radius point of the circle, thus changing the size of the circle, we can see that the area-radius relationship remains as the circle changes.



We will now look at the features of the Calculator in detail.

4.7.2 The Button Pad

The buttons in the Button Pad are organized into three groups: functions and special constants, numerical values and operators, and editing tools.

Functions and Special Constants:

There are 15 keys normally visible in the top part of the Button Pad and an additional 5 keys that can be accessed by pressing the Shift key. These 20 buttons represent often-used mathematical functions and constants.

Here is how the function group normally looks.

sin	cos	tan	abs	sqrt
asin	acos	atan	ln	log
pi	e	x	max	min

The 15 basic functions include:

1. sin(x)—the trigonometric sine function.
2. cos(x)—the trigonometric cosine function.
3. tan(x)—the trigonometric tangent function.
4. asin(x)—the inverse trigonometric sine function.
5. acos(x)—the inverse trigonometric cosine function.
6. atan(x)—the inverse trigonometric tangent function.
7. ln(x)—the natural logarithm (logarithm base e).
8. log(x)—the logarithm base 10.
9. abs(x)—the absolute value function.
10. sqrt(x)—the square root function.
11. max(x, y)—finds the maximum of two variables.
12. min(x, y)—finds the minimum of two variables.

Here is how the function group of buttons looks once one hits the Shift key on the keyboard.

sinh	cosh	tanh	ival	sgnm
asin	acos	atan	ln	log
pi	e	t	max	min

The new functions include:

1. sinh(x)—the hyperbolic sine function.
2. cosh(x)—the hyperbolic cosine function.
3. tanh(x)—the hyperbolic tangent function.

4. ival(a, b, x)—the interval function (equal to 0 for $x < a$ or $x > b$ and 1 otherwise).

5. sgnm(x)—the signum function (equal to -1 if $x < 0$, 0 if $x = 0$, and 1 if $x > 0$).

When using a function as part of an expression, we put in parentheses using the '(' and ')' keys. For example, suppose we wanted to make the expression "sin(2.3)". We would do this by hitting the 'sin', '(', '2', '.', '3', and ')' keys in order. In other words, all parts of the expression "sin(2.3)" must be keyed in individually, although the sin portion is keyed in with the function key for the sine function.

There are two special constants in this section of the Button Pad—'pi' and 'e'. These represent the mathematical constants π and e. These can be used as symbolic constants in an expression. When evaluated, they are computed to a finite level accuracy.

There is one special variable key in the Button Pad—'x'. This is useful for defining and graphing functions in the coordinate system used in *Geometry Explorer*. Note that this variable button switches to 't' when one presses the Shift key on the keyboard. This allows one to create polar and parametric functions. Consult Chapter 6 for more information on how to graph functions.

Numbers and Operators: The 20 keys in this section represent the usual integers and mathematical operators that one might find on any calculator, plus a few extra keys.

The comma key is used solely for expressions involving the max or min functions. It is needed to separate the variables as in "max(2.3,3.4)".

The parentheses keys were described above. They are used in conjunction with the function keys and also to clarify expressions. For example the expression "2.0/(1+3.14)" needs parentheses to group the addition under the division.

The power key '^' is used to raise a sub-expression to a power. For example, "`2^(3*2)`" would evaluate to 64.

Finally, the '!' key is used for factorials. Factorials are defined for positive integers by multiplying together all integers less than the given integer and greater than or equal to 1. For example, $10! = 10 * 9 * 8 * 7 *$

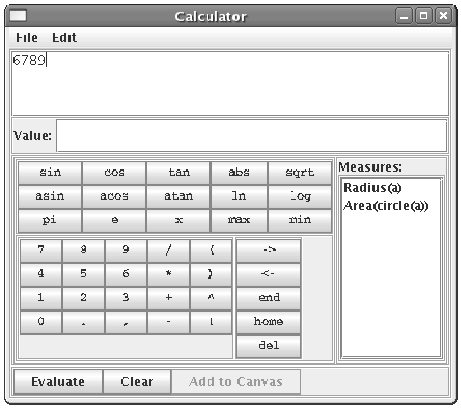
$6 * 5 * 4 * 3 * 2 * 1$. Factorials grow in size very fast. *Geometry Explorer* can handle arbitrarily large integer values and thus can handle factorials well.

For example 200! =
788657867364790503552363213932185062295135977687173263294742533244359449963403342920304284011984623904177212138919638830257642790242637105061926624952829931113462857270763317237396988943922445621451664240254033291864131227428294853277524242407573903240321257405579568660226031904170324062351700858796178922222789623703897374720000000000000000000000000000000000000000000000000.
(result taken from the Calculator Value Area)

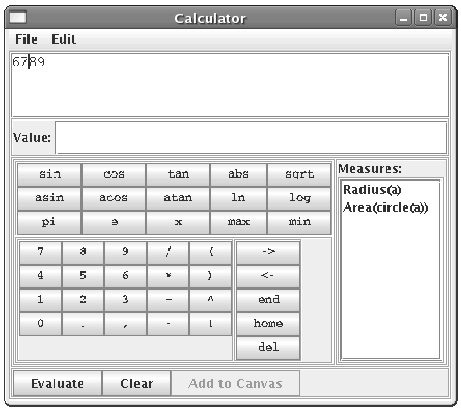
Editing Keys: The five keys in this section allow one to edit the expression currently displayed in the Expression Area.

When creating an expression in the Calculator the Expression Area keeps track of an insertion point in the expression. This insertion point is visually signified by a vertical line.

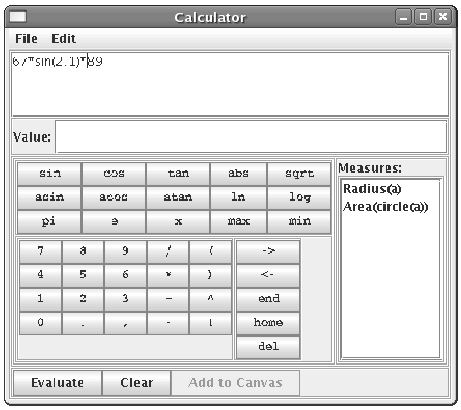
For example, if we key in the number 6789, then the insertion bar will move to the position just following the 9.



If we now click on the left arrow key ('<-') twice we will move the insertion point to just after 7.



At this point we can insert other data. For example, we could key in "*sin(2.1)*" to get the expression "67*sin(2.1)*89".



The 'end' and 'home' keys have the effect of taking you to the end of the current expression or to the beginning of the current expression.

The 'del' key is used to delete the entry before the insertion point.

Note that if one wants to *completely* clear the current expression, this can be done by clicking on the 'Clear' button on the bottom of the Calculator window.

Keyboard Equivalents

One can use keys on the computer keyboard in place of most of the keys in the Button Pad. Exceptions include the function keys and the key for π. The backspace key on the keyboard has the same effect as the 'del' key on the Button Pad.

4.7.3 Evaluation of Expressions

Once an expression has been created it can be evaluated by clicking on the Evaluate key on the bottom of the Calculator window.

If the expression has been incorrectly formed, an error dialog will pop up explaining the error.

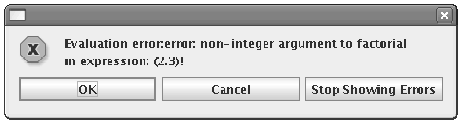
For example, if we key in "2+*3", we would get the error message shown at right.

The error trapping used in *Geometry Explorer* is designed to capture two types of errors in expressions—parsing errors and evaluation errors.

When an expression is *parsed*, or split up into small components, there is a procedure that checks to see if the expression makes mathematical sense. If it does not, then the procedure produces the error message telling the user where the error occurred. Here, the error was in column 3 as that was where the extra operator ('*') was located if we read the expression from left to right.

Checking expressions for errors in evaluation can be quite tricky. Expressions like "2.3!" are formulated correctly and thus would parse just fine. However, factorials are not defined for numbers other than integers. This is an example of an evaluation error.

If we evaluate the expression "2.3!" in the Calculator we get an evaluation error dialog. This error dialog box is informing us that there is an evaluation error in our expression and that the problem is a real number somewhere in the evaluation.



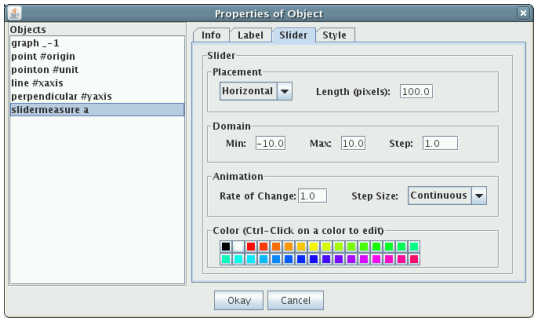
4.8 Sliders

Under the **Measure** menu you will find an option labeled **Slider**. This option can be used to create a slider with numerical values. A slider consists of a horizontal (or vertical) line segment, along with a point that moves along the segment. As the point is moved, a numerical value changes accordingly, from a minimum value at the left end of the slider to a maximum value at

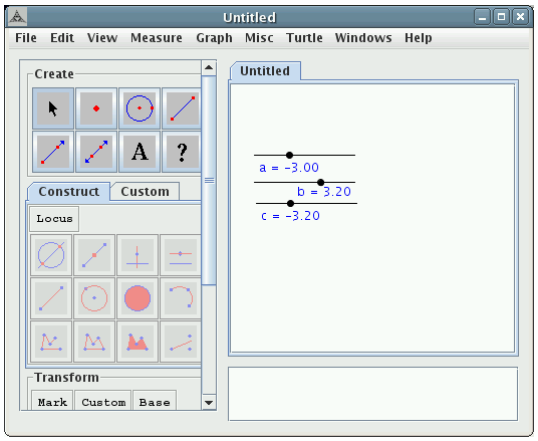
the right end. Sliders can be used like any other measurement, for example they can be used in Calculator expressions.

Suppose we want to create an illustration of how the coefficients in a general quadratic function $f(x) = ax^2 + bx + c$ affect the shape of the graph of the function.

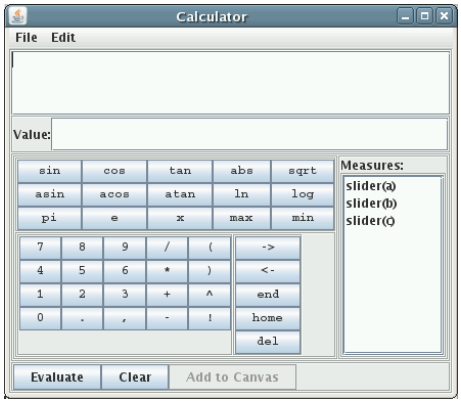
We start by creating the sliders to represent the coefficients of the quadratic. Go to the **Measure** menu and select **Slider** to create a slider. A properties dialog box will pop up asking you to define various properties for the slider. In this example, we'll just use the default values, so hit "Okay."



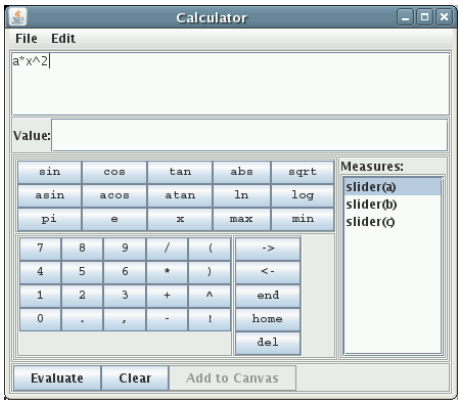
A slider will be created near the bottom of the Canvas. Create two more sliders and move them into position as shown here.



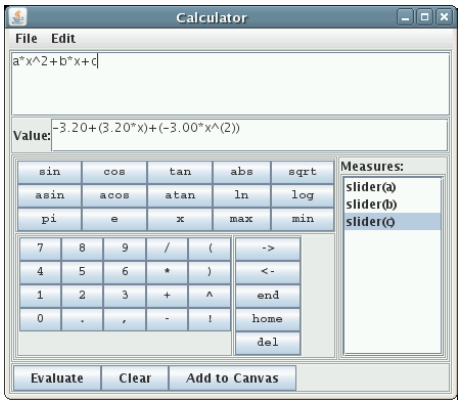
To create the quadratic function, we will use the Calculator (located under the **View** menu). The three sliders we have created will appear under the "Measures:" list.



To use a slider value in the Calculator, we double-click on it in the "Measures" list. Double-click on the item labeled "slider(a)." The symbol "a" will appear in the text area at the top of the Calculator. Next, click on the "*" button and then "x", "^", and "2" to create "a*x^2" as shown.

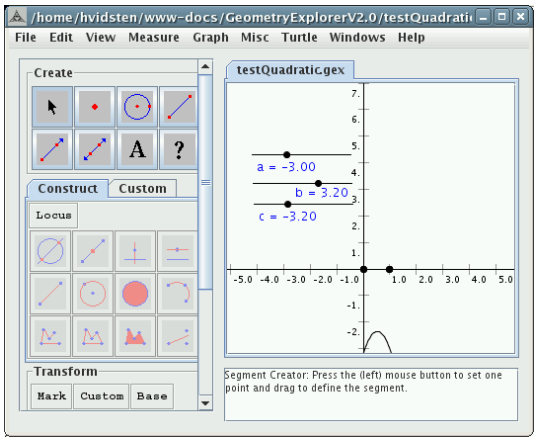


Next, use the Calculator measure list and buttons to key in the rest of the expression for "a*x^2+b*x+c" as shown. Then, hit the "Evaluate" key to evaluate the expression.

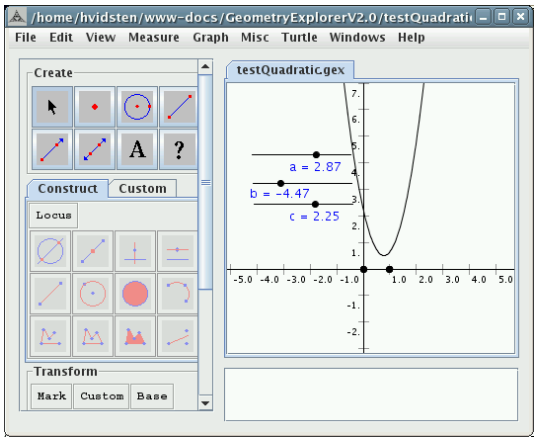


To add this function to the Canvas we click on the "Add to Canvas" button in the Calculator. A dialog box will pop up asking for the name of the function we are creating. Type in "f" and hit "Okay."

A graph of the function will appear in the Canvas. Use the **Graph** menu to show the coordinate axes and then click and drag the point at the origin of the axes down a bit to create a nice display of the axes, the function's graph, and the sliders, as shown.



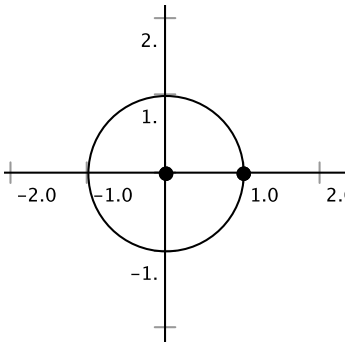
Now move the sliders back and forth to see what effect this has on the graph.



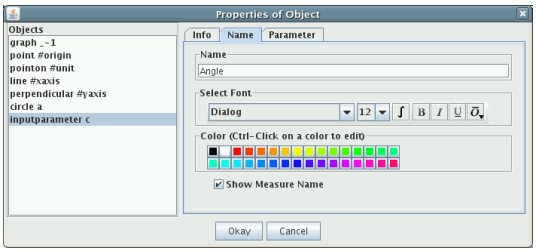
4.9 User Input Parameters

Under the **Measure** menu you will find an option labeled **Input Parameter**. This option can be used to create an input parameter. An input parameter consists of a text box for which the user can type in numerical values. These values can be used like any other measurement, for example they can be used in Calculator expressions.

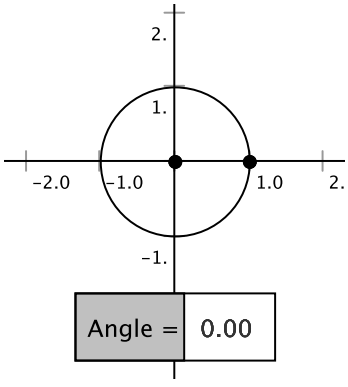
In the figure on the right we have created a circle with center at the origin and radius point at the unit point of the coordinate system used by *Geometry Explorer*.



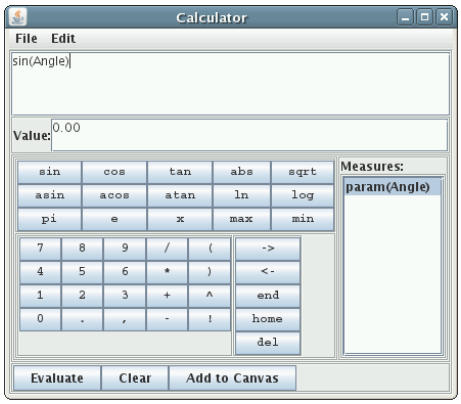
Suppose we wished to create an illustration of the relationship between the cosine and sine of an angle and a point on the unit circle. Let's start by creating an input parameter. Select **Input Parameter** from the **Measure** menu. A **Properties Dialog** box will pop up. We can use this to set the input parameter's properties. Click on the Name tab and set the name to "Angle." Then, hit "Okay."



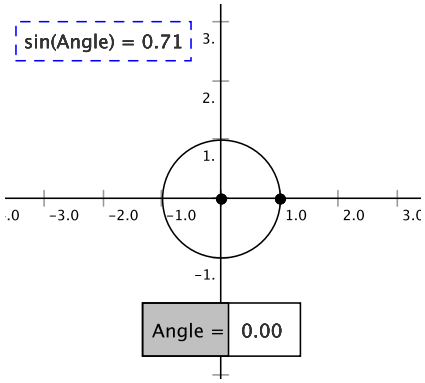
A box labeled "Angle" will be created in the Canvas.



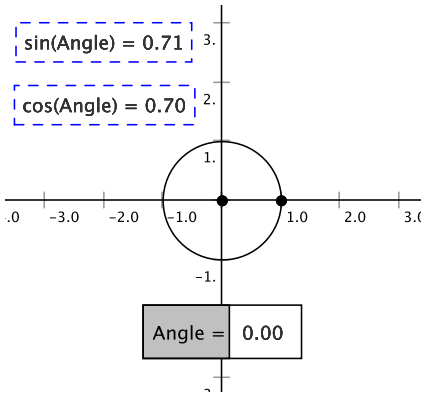
We will now use the Calculator to compute the sine and cosine of the numerical value of the angle parameter. Bring up the Calculator (under the **View** menu) . You will notice that the input parameter is listed in the Measure List in the Calculator. To compute the sine of this measure, click on the sin button, then on the left parends button, then double-click the "Angle" measure, and finally on the right parends button. Click the Evaluate key to compute the desired sine.



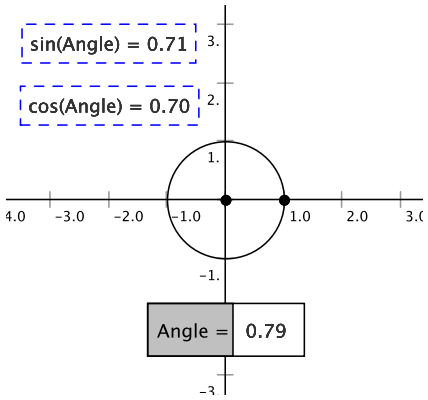
To add this new value back to the Canvas we just click on the Add to Canvas button in the Calculator.



Now, we can go back to the calculator and similarly compute the cosine of the input box value and add that to the Canvas.



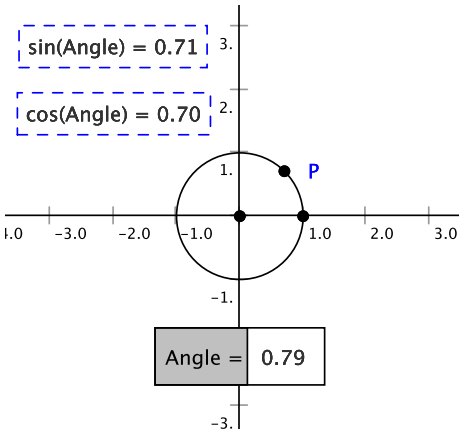
To see how everything is connected together, let's change the value in the input box and see how that changes the computed sine and cosine values. Type in the text "pi/4" in the input box and hit the return key.



Note that *Geometry Explorer* computes the numerical value of $\frac{\pi}{4}$ and replaces the text "pi/4" with this value.

To illustrate the relationship between the cosine, sine, and the unit circle in our example, let's now plot a point that has x-coordinate the cosine measure and y-coordinate the sine measure. To do this select the cosine and sine measures on the Canvas and choose **Add as (x,y) Point from Measures** under the **Graph** menu.

A new point P will be created in the Canvas with coordinates determined by these two measures. Clearly, this point lies on the unit circle and if we change the value in the input box we will see that this point always lies on the unit circle.



We make note of a special feature of *Geometry Explorer* shown in this example – the program understands the use of symbolic numerical values such as "pi" when user input is requested. The general rule of thumb is that any numerical value that can be used in the calculator can also be used as numerical input to parameters.

4.10 Using Tables

Tables of measurements are useful for analyzing relationships between measurements. For example, if we consider the interior angles of a triangle, then there is a relationship for these angles, namely that their sum is always 180 degrees.

To create tables we need to use the bottom group of three menu items located under the **Measure** menu in the main window.

1. **Create Table:** To create a table of measurements, first select all of the measurements that are to be tabulated. Then, choose **Create Table** in the **Measure** menu.

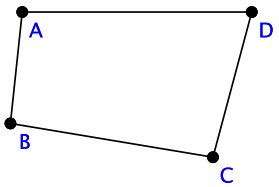
2. **Add to Table:** To add another column of data values to an existing table, first select the table and then choose **Add to Table** in the **Measure** menu.

3. **Edit Table...:** To edit an existing table, first select the table and then choose **Edit Table...** in the **Measure** menu.

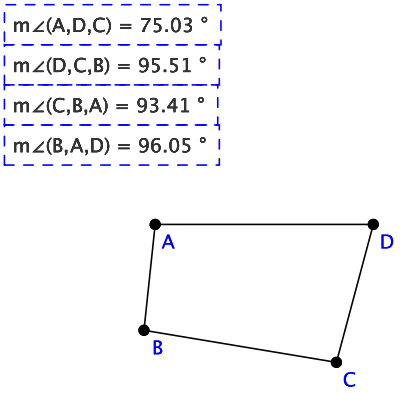
The next example illustrates how to use tables to discover a nice relationship for the interior angles of a convex quadrilateral.

4.10.1 Quad Interior Angles

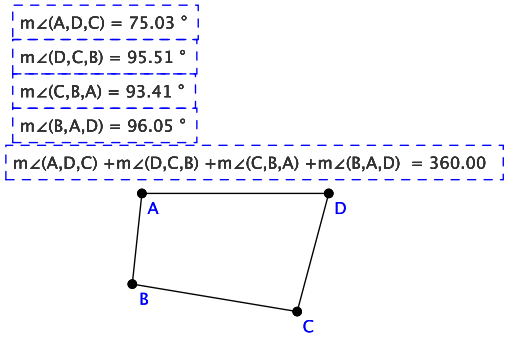
In the figure at the right we have constructed a quadrilateral $ABCD$.



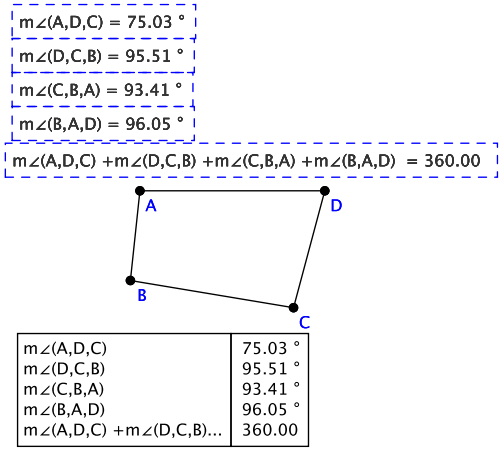
In order to discover a relationship between the interior angles of this figure we need to measure the four interior angles. Select the points of the quad in groups of three and select **Angle** under the **Measure** menu for each group of three. (be careful that you select them in the right order for *interior* angles)



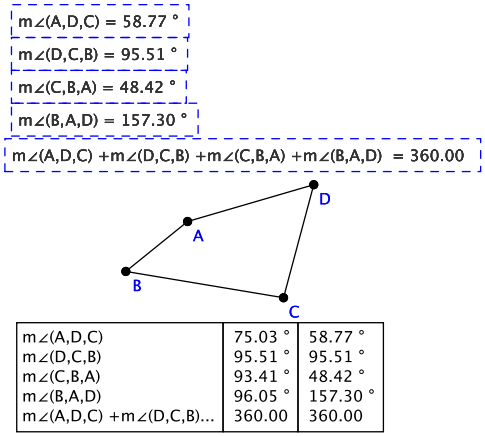
Let's create a new compound measurement that reflects the sum of the four interior angles. Using the Calculator sum up the four angles and add that new compound measurement to the canvas. (Review the previous section if you need help on using the Calculator to create compound measurements)



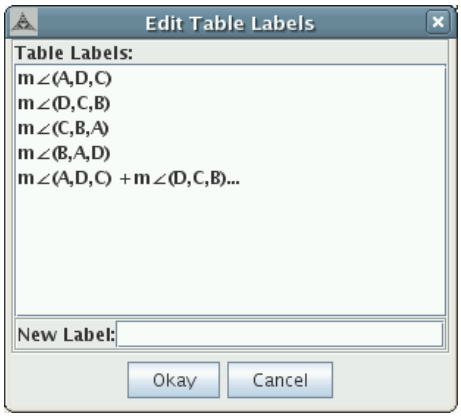
Interesting! The sum of the angles is 360 degrees. Let's create a table that will hold the four angle measurements plus the new sum of angles compound measurement. To do this, select the four angle measurements and the compound measurement and then choose **Create Table** under the **Measure** menu.



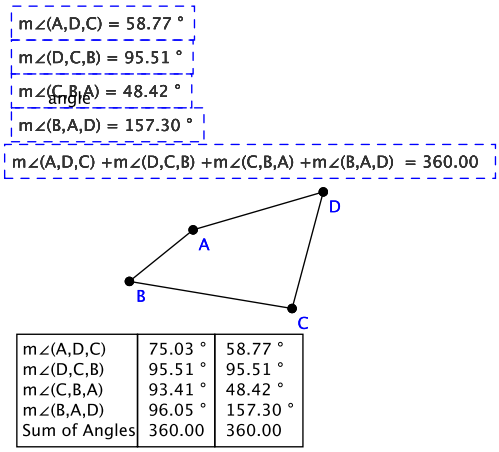
Note that the long expression for the sum of the four angles has been truncated in the table. If expressions are longer than twenty characters they will be truncated in this fashion. Let's create another table entry. Move some of the points of $ABCD$ to alter the interior angles. Then, select the table and choose **Add to Table** under the **Measure** menu. A new column of data values will be added to the table.



The sum of the four angles is still 360. This somewhat meager evidence should suggest a relationship between the four angles of a convex quadrilateral $ABCD$. (A figure is *convex* if any segment connecting two points in the figure is entirely contained in that figure). Suppose that we did not like that last truncated label in the table. To edit the table we select the table and choose **Edit Table...** under the **Measure** menu. The dialog box at right will pop up.



To change the last label click on the list entry titled "Angle(A,D,C)+Angle(..." and then type in a new label in the text field labeled "New Label:". In our case we will type in "Sum of Angles". Hit the Okay button to make the change. The new label will now appear in the table.



m∠(A,D,C)	75.03 °	58.77 °
m∠(D,C,B)	95.51 °	95.51 °
m∠(C,B,A)	93.41 °	48.42 °
m∠(B,A,D)	96.05 °	157.30 °
Sum of Angles	360.00	360.00

Chapter 5

Transformations

> Geometry is the study of those properties of a set which are preserved under a group of transformations on that set.
>
> —Felix Klein (1849–1925)

Transformations are basic to both a practical and theoretical understanding of geometry. Object permanence, the idea that we can move an object to a different position but the object itself remains the same, is one of the first ideas that we learn as infants. From the earliest stage of development we learn how objects are affected by transformations—that is by actions that we carry out that affect the configuration of an object.

Transformations that leave an object unchanged (or *invariant*) include translations (i.e movement), rotations, and reflections. A reflection leaves an object invariant in the sense that the reflected image of an object is not changed in size or shape, although the orientation of the object (left to right) might change.

Felix Klein, one of the great geometers of the late 19th century, gave an address at Erlanger in 1872 in which he proposed that geometry should be *defined* as the study of transformations and of the objects which such transformations leave invariant.

In Klein's view Euclidean geometry should be defined as the study of figures which remain invariant under Euclidean mappings such as translations, rotations, and reflections. Such figures would include triangles, lines, circles, and other figures from classical geometry.

In *Geometry Explorer* translations, rotations, and reflections are provided as built-in tools. These tools are also available for use in Hyperbolic and Elliptic geometry. A fourth transformation, dilation, is also provided,

although it is different than the other three in that it does *not* preserve Euclidean figures. A dilation of a figure in Euclidean Geometry is essentially a shrinking down or scaling of that figure. Dilations in Hyperbolic and Elliptic geometry are not as well-defined as dilations in Euclidean geometry. Dilations alter the scale of a Euclidean object, but do not change the generic nature of that object (for example Euclidean triangles remain triangles).

Transformations are carried out in a three-stage process in *Geometry Explorer*. First, geometric data must be specified that defines a transformation. Then, one must specify the object(s) to be transformed. Finally, the transformation is carried out by clicking on one of the transform buttons in the Transform Panel.

One non-obvious feature about the Transform Panel is the fact that the **Mark**, **Custom**, and **Base** buttons are actually pull-down menus. These are used to define either data needed by transformations or particular types of transformations.

5.1 Quick Overview of Transformations

1. **Translation:** To translate (move) an object in the Canvas one must define a *vector* as described in the next section, or define a *custom* translation using the **Custom** pull-down menu. Once the definition is complete, select the object(s) to translate and click on the Translate tool in the Transform Panel.

2. **Rotation:** To rotate an object in the Canvas you must define an angle and a center of rotation as described in the next section, or define a *custom* rotation using the **Custom** pull-down menu. Once the definition is complete, select the object(s) to rotate and click on the Rotate tool in the Transform Panel.

3. **Dilation:** To dilate an object in the Canvas you must define a ratio and a center of dilation as described in the next section, or define a *custom* dilation using the **Custom** pull-down menu. Once the definition is complete, select the object(s) to dilate and click on the Dilate tool in the Transform Panel.

4. **Reflection:** To reflect an object in the Canvas about a linear object (line, ray, or segment) you must define the linear object to be a " mirror," as described in the next section. Once the definition is complete, select the object(s) to reflect and click on the Reflect tool in the Transform Panel. One cannot define a custom reflection.

5.2 Defining Transformations

Transformations need to be defined in terms of certain geometric information. For example, a rotation must be defined in terms of a center of rotation and an angle.

In *Geometry Explorer* all definitions are carried out by the use of two pull-down menus in the Construct Panel. These are the **Mark** and **Custom** pull-down menus.

5.2.1 Setting Geometric Transformation Data

The **Mark** pull-down menu is used to define *geometric* data that are needed to specify a transformation.

Each item in this menu is defined by selecting a set of geometric objects in the Canvas as follows:

1. **Center:** To define either a center of rotation or a center of dilation, select a point and then choose **Center** under the **Mark** menu.

2. **Mirror:** To define a mirror of reflection, select a linear object (line, ray, or segment) and then choose **Mirror** under the **Mark** menu.

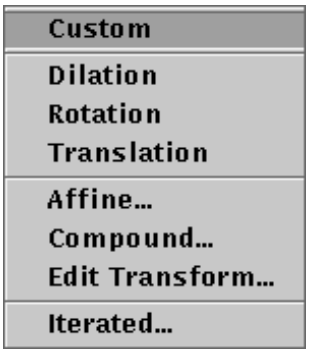
3. **Vector:** To define a vector of translation select two points and then choose **Vector** under the **Mark** menu. A dialog box will pop up asking whether to interpret this vector as a simple vector in rectangular coordinates, or if it should be interpreted as a magnitude to be used in polar coordinates. If one chooses the polar coordinates option, then an angle must also be defined.

4. **Angle:** To define an angle, select three points (the initial, vertex, and terminal points of the angle). Then, choose **Angle** under the **Mark** menu.

5. **Ratio:** To define a ratio, select two segments. The length of the first will be the numerator in the ratio and the length of the second will be the denominator. Then, choose **Ratio** under the **Mark** menu.

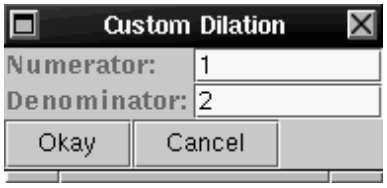
5.2.2 Defining Custom Transformations

The **Custom** pull-down menu is used to define special types of transformations.

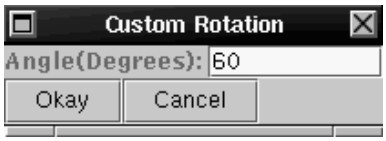


There are seven possible menu options under the **Custom** menu. When choosing any of the first three options, a dialog box will pop-up asking for the appropriate numerical data needed to define that transformation. The fourth option is used to define a general type of transformation known as an *affine* transformation (*Möbius* transformation for hyperbolic geometry). More on that later. The fifth option is used for defining transformations built from existing transformations. This **Compound Transform** capability will be described later in this chapter. The sixth option is used to edit previously defined transformations. Finally, the seventh menu option is used to carry out multiple iterations of a transformation.

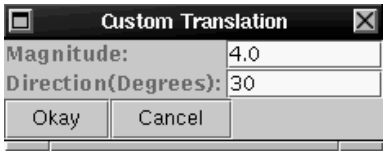
As an example suppose we wanted to define a dilation that scaled objects by a numerical ratio of $\frac{1}{2}$. We would first define a center of dilation using the **Mark** menu. Then, we choose **Dilation** from the **Custom** menu. A dialog box like that the one at right will pop up. Here we have defined a dilation with a scale ratio of $\frac{1}{2}$.



For a custom rotation we need to specify an angle of rotation (in degrees). In the figure at the right we have defined a rotation of 60 degrees. Note that a center of rotation must still be defined before the rotation can be applied. Do this by using the **Mark** menu.



For a custom translation we need to specify a vector by defining the magnitude and direction (in degrees) of the vector. Here we have defined a translation that will translate objects 4.0 units in the direction that is 30 degrees up from the horizontal.



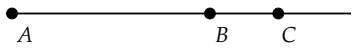
5.3 Example: The Hyperbola

As an example showcasing the use of transformations let's look at constructing a hyperbola. A hyperbola is similar to an ellipse in that it is defined in terms of a set of two fixed points and a numerical value. The hyperbola is the set of points such that the difference of the distances to the two fixed points is equal to the numerical value.

As we did with the ellipse example in Chapter 3, we can translate the distance requirement in the definition for hyperbolas to a geometric relationship between segments. However, this will not be as easy as it was in the ellipse case, since now we need to ensure that the *difference* in two segment lengths is constant, rather than the sum.

We can accomplish this difference condition by using a translation as follows:

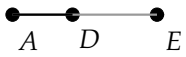
First we create a ray $\overrightarrow{AB}$ to serve as a guide for our construction of segments. On this ray attach a point C to the right of B. We will use the distance BC as the constant difference value in the definition of a hyperbola.



Now, attach a point D to the ray somewhere between A and B, and hide the original ray as we no longer need it.

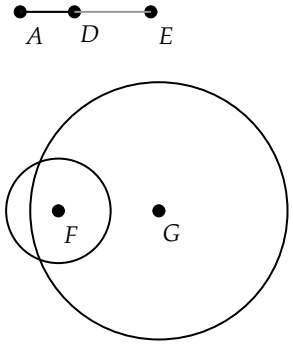
Next we translate (move) D the distance determined by the vector from B to C. To define this translation, select B and C and choose **Vector** from the **Mark** menu. When the translate dialog box pops up choose Rectangular and hit Okay. Select point D and click the Translate tool in the Transform Panel. This will produce a new point E that is a distance of BC from D.

Now, connect A and E by a segment and A and D by another segment. To keep these two overlapping segments straight, change the color of $\overline{AE}$. Hide B and C as they are no longer needed.

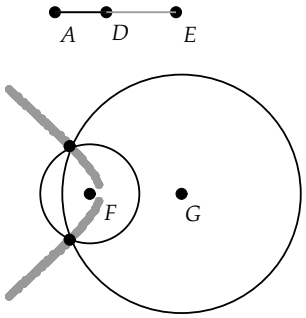


At this point $\overline{AE}$ and $\overline{AD}$ will have the property that their length difference will always be that of $\overline{DE}$. Since E is actually D translated by a fixed length of BC, then the difference between the lengths of $\overline{AE}$ and $\overline{AD}$ will be constant and equal to the length of $\overline{BC}$, even as we vary point D. Experiment with this concept by moving points D, E, and A.

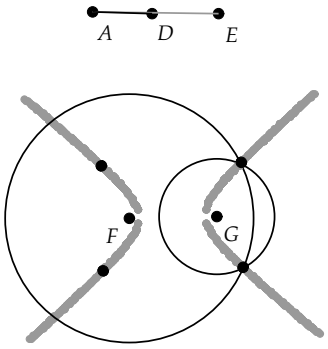
All that is needed to construct the desired hyperbola is to create two focal points F and G and to construct two circles at F and G of lengths AD and AE. We construct the circle at F by selecting F and $\overline{AD}$ and clicking the Circle tool in the Construct Panel. Similarly we construct the circle at G of radius equal to AE. (Be careful when selecting overlapping segments that you select the one you really want)



To construct points of the hyperbola, we need to find the intersection of these two circles. Select the two circles and click on the Intersect tool in the Construct Panel. As we vary D, the two intersection points should trace out one branch of the hyperbola. Select the two intersection points and choose **Trace On** from the **View** menu. Then, move D back and forth along $\overline{AE}$ to trace a branch of the hyperbola.

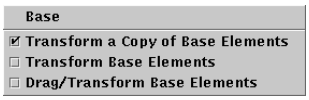


How could one construct the other branch? Hint: look at this figure.



5.4 Copying Figures in Transformations

In the examples discussed above, a transformation, when applied to a selected group of objects, always duplicates, or copies, those objects and then transforms the copied objects to their new location. This is the default mode of operation when transforming objects in *Geometry Explorer*.



However, sometimes one may wish to directly transform the original or *base* objects without copying them. To do this, we use the pop-up menu labeled **Base** which is in the Transform Panel.

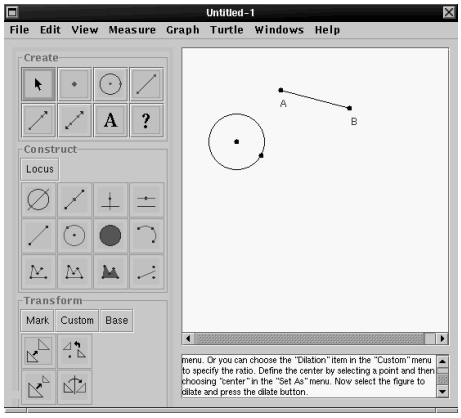
Each item in this menu controls the action of transformations on the base geometric objects that are to be transformed. There are three possible actions:

1. **Transform a Copy of Base Elements:** This is the default behavior for a given transformation. All base elements are copied and the copy is then transformed.

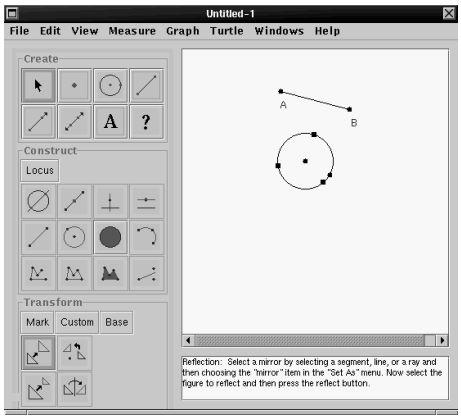
2. **Transform Base Elements:** In this mode a transformation will directly affect base objects. Base objects are not copied.

3. **Drag/Transform Base Elements:** In this mode a transformation will be controlled by dragging the mouse. Base elements will be transformed directly, as in the second case above, without being copied.

As an example of the second type of transformation, we have a circle and a segment $\overline{AB}$ on the Canvas in the figure at the right. We wish to translate the circle by the vector from A to B.

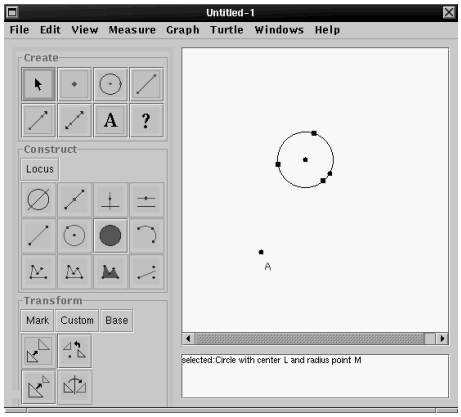


To translate the circle we need to select **Transform Base Elements** from the **Base** menu, as we want to directly translate the circle and not a copy of the circle. Next, we select A and B and choose **Vector** under the **Mark** menu in the Transform Panel. Then, we choose Rectangular in the vector dialog box that pops up. Finally, we select the circle and click on the Translate tool in the Transform Panel. The original circle will then be translated, but not copied.



For the third type of transformation listed under the **Base** menu, the effect of a transformation is under the control of the *mouse*.

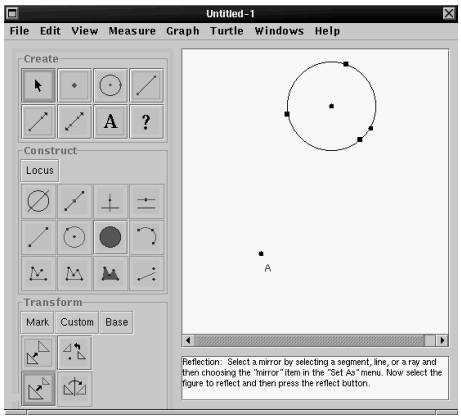
As an example, in the figure at the right we have a circle and a point A. Suppose we wish to dilate the circle towards A, but we want to be able to vary the scale of the dilation. This can be done by using the third option under the **Base** menu. Choose this menu option and then set point A as a center of rotation/dilation using the **Mark** menu. If we select the circle we will see that three transform tools are active — the Translate, Rotate, and Dilate tools.



The Translate tool is always active, no matter what mode we are in, as we can always select an object and move it. The Rotate and Dilate tools are the only tools that are truly active when in the **Drag/Transform Base Elements** mode.

At this point we can carry out either a dilation or rotation by clicking on the Dilate or Rotate tool in the Transform Panel and then clicking and dragging somewhere in the Canvas. A side-to-side motion will increase and decrease the transformation parameter.

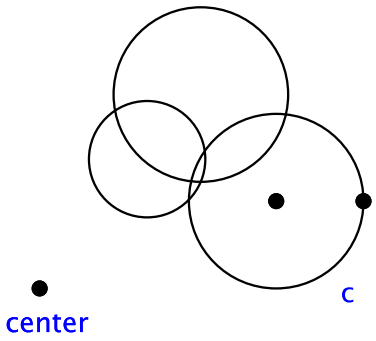
Try this out for the circle. Click on the Dilate tool and then click and drag from left to right in the white space on the Canvas. Note that the circle expands as you drag to the right. Dragging the mouse to the left will shrink the circle.



5.5 Compound Transformations - Fixed

Suppose that we want to study a spiral pattern where an object is alternately rotated and dilated about a central point. We could do this by creating the appropriate rotation and dilation transformations and repeatedly applying these to a figure.

In the figure at the right we have defined a center point of rotation (and dilation) and have carried out a rotation followed by a dilation of a circle c. Note that each time we carry out a transformation we get a copy of the original object. It would be convenient to be able to define a *compound* transformation which carries out both the rotation and dilation on the original figure.

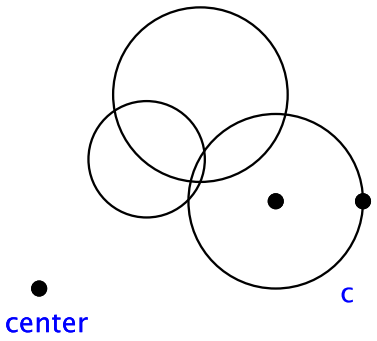


The definition of a compound transformation is a two-stage process. First, we must carry out the individual component transformations on geometric objects in the Canvas. Then, we use the Compound Transformation Dialog Box to define a compound transformation.

To see how this works, lets define a compound transformation that will carry out the rotation-dilation described above. First, create a point somewhere on the Canvas to act as a center of rotation (and dilation). Next, select this point and choose **Center** from the **Mark** menu in the Transform Panel.

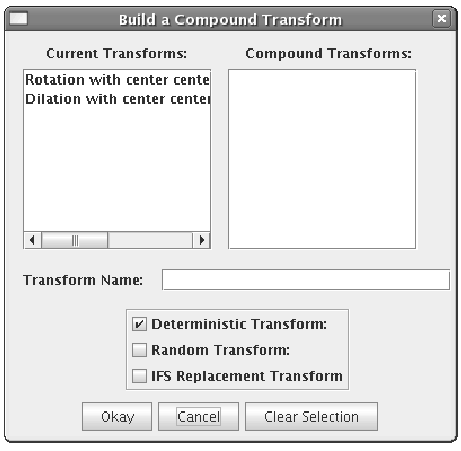
Now, we will define a rotation of 30 degrees and a dilation of $\frac{2}{3}$ about this center point. Choose **Rotation** from the **Custom** menu in the Transform Panel. Type in "30" and hit Okay. Then, choose **Dilation** under the **Custom** menu and type in "2" for the numerator and "3" for the denominator. We now have defined the desired rotation and dilation. In order for *Geometry Explorer* to store these transformations for later use, we need to use them somewhere in the Canvas.

Create a circle somewhere in the Canvas. Select the circle. The Rotate and Dilate tools in the Transform Panel should now be active. Click on the Rotate and Dilate tool (order does not matter). You should now see something like the figure on the right.

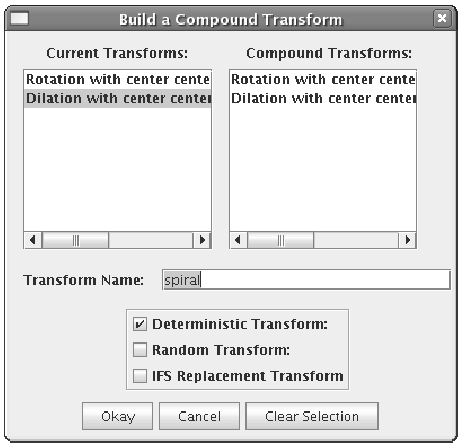


Hide the two circles that were created by the Rotate and Dilate tools. (We only need them to define the two transformations.) To define a *compound* transformation, we choose **Compound...** from the **Custom** menu in the Transform Panel. A dialog box titled "Build a Compound Transform" will pop up. In the upper half of this dialog box there are two lists.

The list on the left, labeled "Current Transforms:" contains all of the transformations that have been carried out so far, in our case the rotation and dilation. The list on the right, labeled "Compound Transforms:" will contain the transforms that we wish to sequence into a new compound transformation. To define this new transformation we click on the transforms in the left list that we wish to be included.

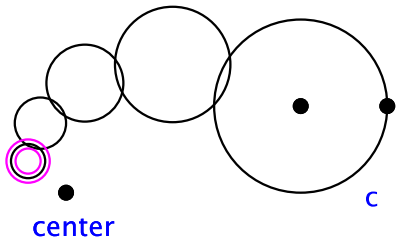


Click on the Rotate and Dilate entries in the left list. These will be automatically added to the list on the right. To finish the definition of our new transformation we need to give it a name so that we can refer to it later. In the Transform Name: text field type in "spiral". Then, hit Okay to finish the definition.



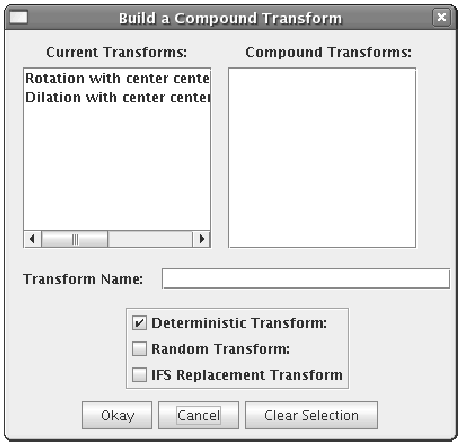
Our new transformation is now stored in the program and we can use it just like we use any of the four standard transformations. Let's go back to the Canvas and select circle *c*. The new transformation that we just created, spiral, can be accessed through the **Custom** menu in the Transform Panel. If we click on this menu we see that **spiral** has been added after the menu item **Iterated...**. New compound transforms will always appear under the **Custom** menu.

To rotate and dilate circle c in one action, we choose **spiral** from the **Custom** menu. Notice that the new rotated and dilated circle is created in a selected state. We can thus choose **spiral** again from the **Custom** menu, getting a third circle that is rotated 60 degrees from the first and dilated by a factor of $\frac{4}{9}$. We can repeatedly carry out the spiral transformation, getting a spiral of circles converging to the center point. In the figure at the right we have carried out the spiral transformation four times, beginning with the original circle c.



5.6 Compound Transformations - Random

Lets look at the Custom Transform Dialog Box again. Just below the text field where we name the new compound transform there is a group of three check boxes labeled "Deterministic Transform", "Random Transform", and "IFS Replacement Transform".



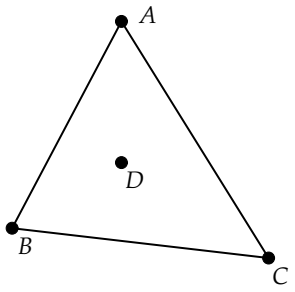
In the previous section we used a sequence of transformations to define a compound transformation. Every time such a compound transformation is applied to an object, the component transformations are carried out in the exact order they were selected when constructing the compound transformation. The sequence of transformations is thus completely determined for all time. A transformation that is fixed in this way is called a *determinis-*

tic transformation. Note that the four basic transformations—translations, dilations, rotations and reflections—are deterministic.

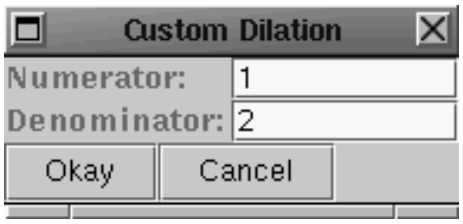
Modern geometric analysis of natural and fractal patterns has required *random* transformations as well as deterministic ones. When using a sequence of random transformations, we are not *guaranteed* that a transformation will be used, only that there is a likelihood of its being used.

To illustrate the use of random transformations, we will look at an example from Michael Barnsley's book *Fractals Everywhere* [1]. Barnsley calls this example the *chaos game*.

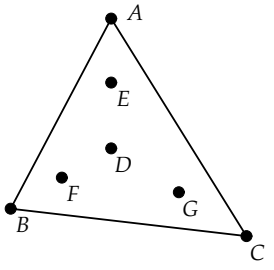
We start by constructing a triangle ΔABC and a point D inside the triangle.



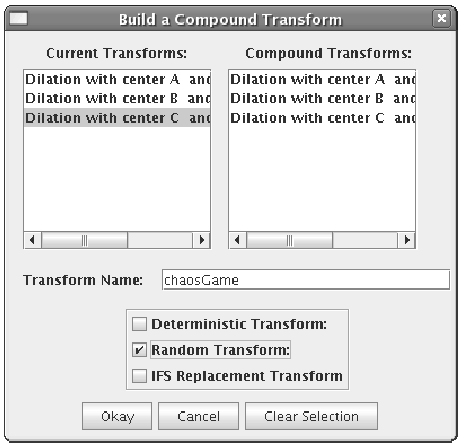
Next, we define a custom dilation of $\frac{1}{2}$ by choosing **Dilation** under the **Custom** menu in the Transform Panel and typing in "1" for the numerator and "2" for the denominator.



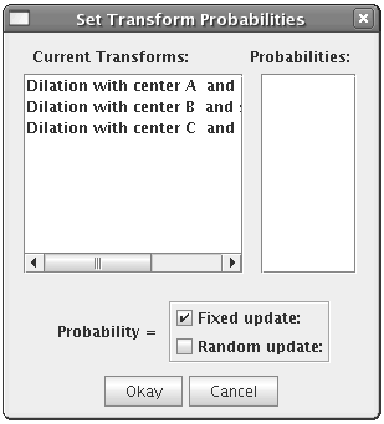
We will now dilate D towards the vertexes A, B, and C. To dilate D towards A, select A and choose **Center** from the **Mark** menu. Then, select D and click on the Dilate tool. To dilate D towards B, select B and choose **Center** from the **Mark** menu. Then, select D and click on the Dilate tool. Finally, set C as a center and dilate D towards C. We get three new (dilated) points E, F, and G as shown.



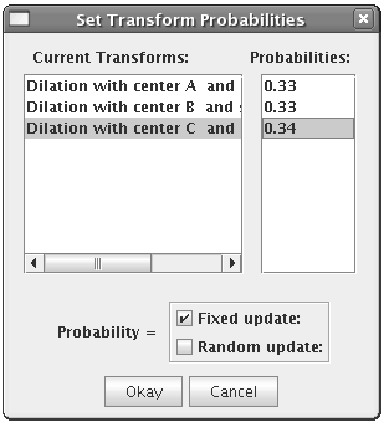
At this point, we will define a random compound transformation from these three dilations. Open up the Compound Transformation Dialog Box by choosing **Compound...** from the **Custom** menu. Select the three transformations in the list on the left, name the new transformation "chaosGame" and select the "Random Transform" check box, as shown on the right.



Once we hit the Okay button in the Compound Transformation Dialog Box, a new dialog box will pop up asking us to set the probabilities for each of the three transformations. This dialog box has two lists on the top: one for the transformations in the compound transform and one for the probability that each transformation will be applied.



We assign probabilities as follows. First, we select a transformation in the list at the left. Then, in the list labeled "Probabilities:" we type in the numerical value for the probability that the transformation will have. In the figure shown we have assigned probabilities of .33, .33, and .34 to the three dilations.

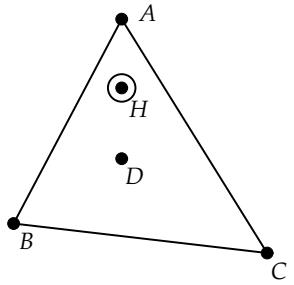


The check boxes in the lower right corner of the probability dialog box control how the new random compound transformation will be applied each

time the Canvas is re-drawn. If we wish one of the three dilations to be chosen at random a *single* time and then left fixed forever we would select "fixed update". However, if we want the dilation to be chosen at random *every* time we redraw objects in the Canvas, then, we would select "random update." For this example, select the fixed update option. Then, hit the Okay button to finish the definition of the new random compound transformation.

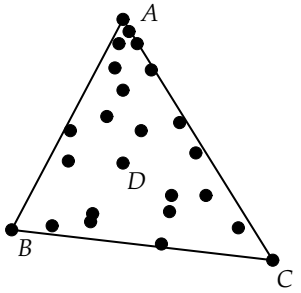
We have now defined a compound transformation that will be random in effect. Every time we apply the compound transformation, one of its three dilations will be chosen at random, with the assigned probability, and applied.

To see how this works, hide points E, F,and G, and select point D. Under the **Custom** menu you will see a new transformation listed, **chaosGame**. Upon choosing this menu item, D will be dilated by a factor of $\frac{1}{3}$ towards one of the three vertexes A, B, or C, creating a new dilated point H.



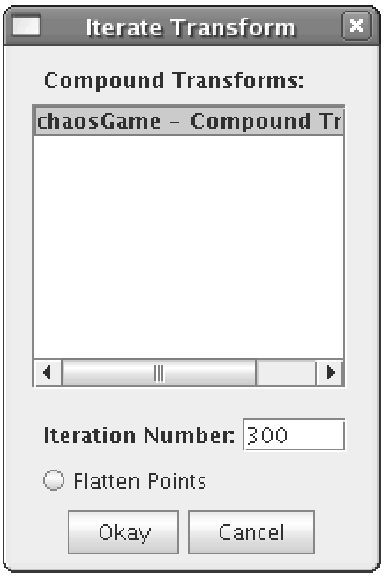
The choice of dilation will be random with about a .33 chance of each dilation being chosen. The new dilated point will be created in a selected state. If we choose **chaosGame** again we will dilate this new point randomly towards one of the vertexes. We can continue choosing **chaosGame** repeatedly to get a sequence of points that are randomly dilated by a factor of $\frac{1}{2}$ towards the three vertexes.

In the figure at the right we have carried out about twenty iterations of this random transformation, beginning with the single point D.

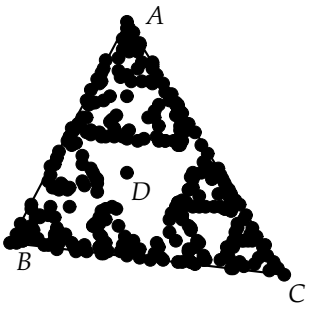


What will happen as we iterate this random transformation again and again? *Geometry Explorer* has a built-in capability to automate the iterations of a compound transformation.

Choose **Iterated...** from the **Custom** menu. A dialog box will pop up with a list of the currently defined compound transforms and a text field for the iteration level desired. (The field labeled "Flatten Points" will be explained in the next section.) Click on the list item labeled "chaosGame" and then type in 300 in the "Iteration Number" field. Click the Okay button to start the iteration. At this point the Okay button will switch to a Stop button. We can stop the iteration process at any point by clicking the Stop button.

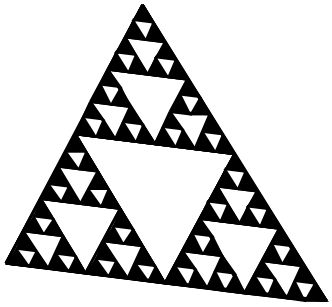


Geometry Explorer will carry out 300 iterations of the chaosGame random transformation. One possible result of this is shown at right. Notice that the pattern of points is not entirely random, as one might expect. There are discernible triangular gaps within the image that are very reminiscent of a famous figure called the Sierpinski triangle.



The Sierpinski triangle is produced by starting with a solid triangle. We then find the midpoints of each of the three sides of the triangle and connect these, forming four sub-triangles. Next we remove the middle triangle, leaving three solid triangles near the three vertexes. The process continues by applying what we just did (i.e. remove a middle sub-triangle) to each of the three solid little triangles. We then would have nine quite small solid triangles remaining. For each of these we again remove a middle triangle, and so on.

In the figure at the right we have carried out this process two more times, yielding 81 small triangles.



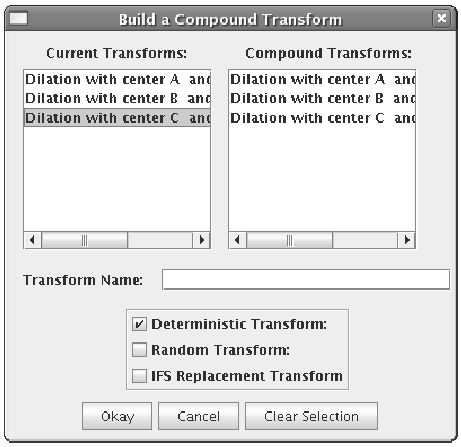
Note how similar the randomly iterated transformation process looks when compared to Sierpinski's triangle. In Barnsley's book [1] he shows that the random process above actually *converges* to Sierpinski's triangle. That is, if we iterate our random transformation an infinite number of times, and if our points could be represented as ever smaller dots, then the random collection of points we would get would exactly match Sierpinski's triangle. This is quite an amazing result—that an *ordered* geometric figure arises from an apparently *random* process. This is one of the great mysteries and delights of the theory of chaos and fractals.

5.7 Compound Transformations - IFS

There is one final type of compound transformation available in *Geometry Explorer*—a transformation built from a set of transformations to form an *iterated function system*, or *IFS*.

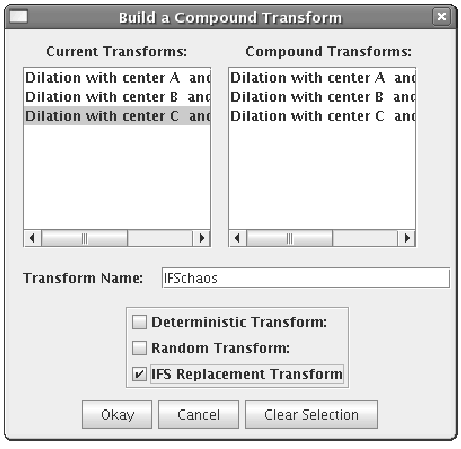
To illustrate this last type of transformation, will use some of the ideas from the construction in the last section. Carry out the steps of that construction up to the point of dilating point D to get the three points E, F, and G.

Choose **Compound** from the **Custom** pop-up menu in the Transform Panel. We see that the Custom Transform Dialog Box has the three dilations defined, just as it did in the last section.

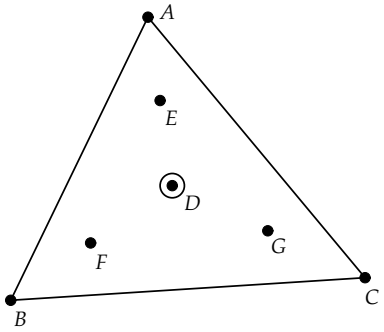


We will call these three dilations a *system* of functions that can be applied to a geometric figure, for example, a point. An iterated function system (IFS) is simple a system of functions that are applied repeatedly using *replacement* of the resulting figure. (For more on iterated function systems, review Chapter 9 of [6], or Barnsley's book [1].)

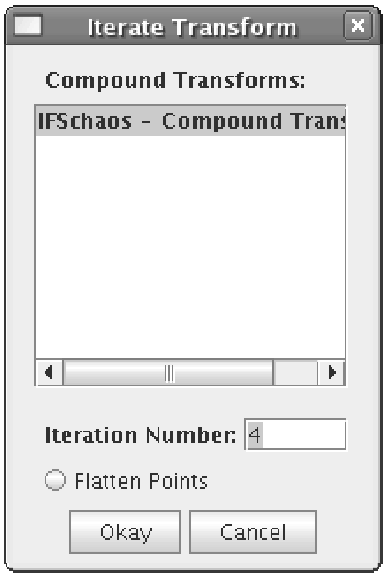
To illustrate how this works, let's define the three dilations as an IFS transformation. In the Custom Transform Dialog Box, click each of the three dilations to choose them as the component functions of the IFS. Label the new transformation "IFSchaos" and choose the check box labeled "IFS Replacement Transform." to set the compound transformation to the correct type. Then, click Okay.



To use this new transformation, select point D in the Canvas.

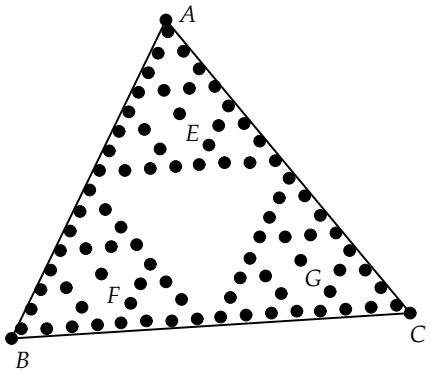


Then, choose **Iterated...** from the **Custom** menu in the Transform Panel. A dialog box will pop up as shown.

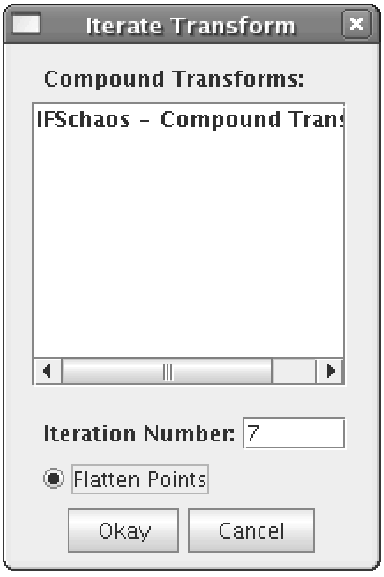


We will use this dialog box to define the *recursion level* for our transformation. The recursion level is the number of times the IFS system is applied to an object. In our example, we will apply the IFS to point D. The application proceeds with recursive replacement of geometric objects at each level. For example, the first time the IFS is applied, each of the three dilations will be applied to D, yielding three new points which *replace* D. That is, D no longer exists. The next time the IFS is applied, each of the three dilations is applied to each of the three new points, yielding nine points at level 2 that replace the three level 1 points. At level 3, we would get 27 points, at level 4 we would have 81 points, and so on. We can see that this recursive procedure grows *very* fast.

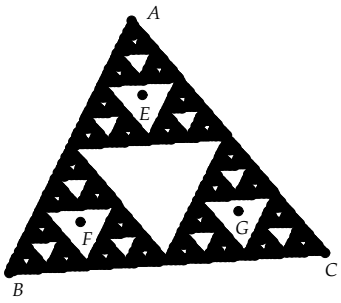
To illustrate the IFS system, click on the "IFSchaos" entry in the Transforms list of the dialog box and type "4" in the field labeled "Iteration Number." Then, hit Okay. The result is at right, showing 81 points at level 4. Note that D no longer exists.



Select one of the points inside the triangle and run the Iterated Transform dialog (choose **Iterated...** under the **Custom** menu) again. This time, click the toggle button labeled "Flatten Points." This will speed up *Geometry Explorer*'s drawing routines by making points non-interactive. Choose a higher level of recursion, say 7.



Click Okay in the dialog box to run the IFS. The completion of the process may take awhile, so wait until the Stop button returns to its Okay state. After the process finishes, we see that the IFS seems to have produced the Sierpinski Gasket.

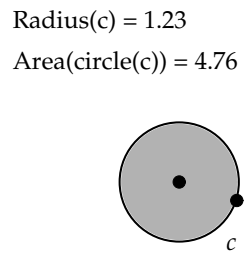


5.8 Transformations Based on Measurements

It is often useful to define transformations in terms of measurements such as distance, slope, area, etc. One can use measurements as the basis for rotations, translations, and dilations.

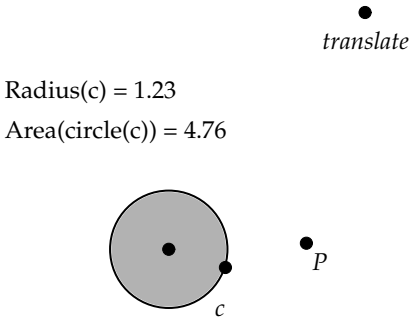
For example, two measurements can be used as the x and y translate values when defining a translation.

In the figure at the right we have created circle c and the filled area for c. We have also measured the radius of c and the area of c.

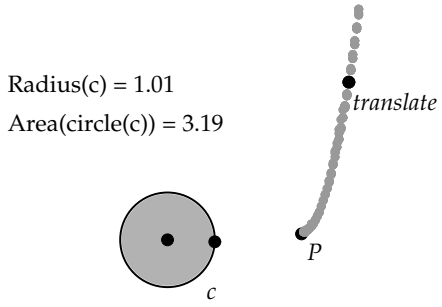


Suppose that we want to analyze the relationship between these two measurements. One way to study this relationship is to translate a point in the x direction by the radius measure and in the y direction by the area measurement.

To do this, first create a point P away from the circle. Then, select the radius and area measurements (in that order). Next, choose **Vector** under the **Mark** menu in the Transform Panel. A dialog box will pop up asking what type of translation is desired. Choose "Rectangular" (i.e. an x, y translation) Select point P. The Translate tool in the Transform Panel will now be active and clicking on it will cause a new point to be translated by the radius and area measurements, as shown.



As we move the radius point of circle c, the radius and area measurements will change and the translated point will change position accordingly. To see how the changes in c affect the translated point, we will use the trace capability of *Geometry Explorer*. Select the translated point and choose **Trace On** from the **View** menu. As we vary the radius point of c, we see a trace of the translated point. This trace appears to be parabolic, as would be logical since the radius and area are related by $A = \pi r^2$.



Transformations can be defined in terms of measurements in the following ways:

1. **Translate:** Translations can be defined in terms of a single measurement or in terms of two measurements. If a single measurement is selected and **Vector** is chosen from the **Mark** menu, then the translation will move a figure by that measurement amount in the x-direction, and will not move the figure in the y-direction.

   If two measurements are selected and **Vector** is chosen, then the translation will move a figure in one of two ways. The two measurements will either determine the x and y directions through which a figure is moved or will determine the angle of movement (from the horizontal) and the distance traveled in the direction of that angle. Upon choosing **Vector** a dialog box will pop up asking which of these two translation types is desired.

2. **Rotate:** Rotations can be defined by by first selecting a point to act as the center of rotation and then choosing **Center** from the **Mark** menu in the Transform Panel. Then, select a measurement to serve as the angle of rotation and choose **Angle** from the **Mark** menu. Note: The angle will be interpreted as measured in degrees.

3. **Dilate:** Dilations can be defined by by first selecting a point to act as the center of dilation and then choosing **Center** from the **Mark**

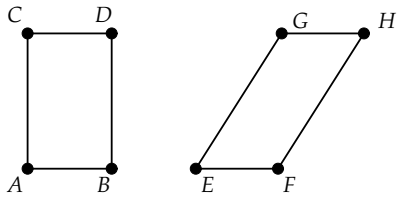
menu in the Transform Panel. Then, select a measurement to serve as the ratio of dilation and choose **Ratio** from the **Mark** menu.

Note that transformations defined by measured values are only available in the Euclidean Canvas.

5.9 Affine Euclidean Transformations

All of the transformations discussed so far—rotations, translations, reflections, and dilations—are specific examples of a larger class of transformations called *affine transformations*. Along with the four basic Euclidean transformations, the set of affine transformations includes other geometric transformations such as *shear transformations*.

In the figure at the right we have created a rectangle $ABCD$. The figure $EFGH$ is the image of $ABCD$ under a shear transformation in the x direction.



In general an affine transformation is an invertible transformation of points (x, y) in the Euclidean plane with the form

$$x = ax + by + e$$
$$y = cx + dy + f$$

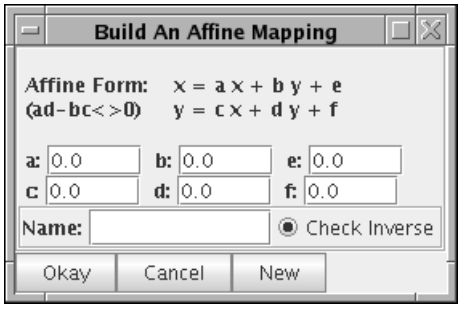
Here a, b, c, d, e, and f are real numbers and there is an additional restriction that

$$ad - bc \neq 0$$

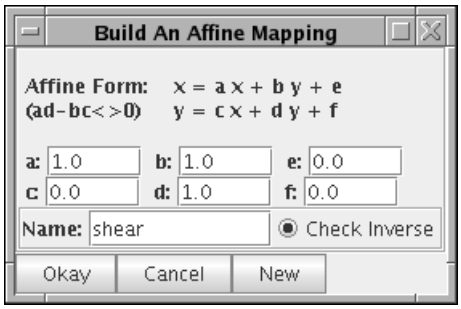
The restriction on $ad - bc$ guarantees that an affine transformation is *invertible*, i.e. that the effects of the transformation can be reversed. We require this condition for a general Euclidean transformation because we think of such transformations as movements or re-configurations of Euclidean objects that do not destroy any essential geometric properties of those objects. Thus, a transformation should be reversible.

From the definition above we see that an affine transformation is uniquely defined by a choice of the numbers a, b, c, d, e, and f (with $ad-bc$ non-zero). Let us look at how we can use affine transformations in *Geometry Explorer* to carry out the shear transformation discussed above.

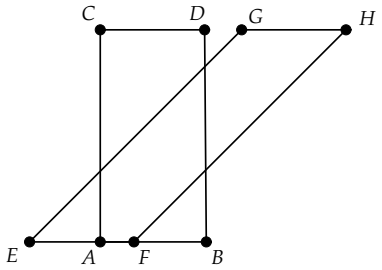
To begin with construct a rectangle $ABCD$ as shown in the figure at the beginning of this section. To define an affine transform, choose **Affine...** from the **Custom** menu in the Transform Panel. A dialog box titled "Build an Affine Mapping" will pop up as shown at the right.



We can put in values for a, b, c, d, e, and f in the appropriate text fields. (Note that we can quickly go from one text field to another by hitting the Tab key on the keyboard.) To define a shear transformation put in the following values: $a = 1.0$, $b = 1.0$, $c = 0.0$, $d = 1.0$, $e = 0.0$, and $f = 0.0$. Then, name the transformation "shear" and hit the Okay button.



At this point, the new transformation can be used to transform objects in the canvas. To transform rectangle $ABCD$, first select the rectangle by dragging a selection box around it. Then, click on the **Custom** menu. You will see the transformation **shear** listed near the bottom of the pull-down menu. Drag down to **shear** and select this menu item. The rectangle will be transformed as shown at the right.



Affine transformations include all of the standard *Geometry Explorer* transformations—rotations, reflections, translations, and dilations. An interesting exercise would be to determine the values of a, b, c, d, e, and f that give each of the four types of standard transformations.

Affine transformations are strictly Euclidean transformations and are not available in hyperbolic geometry. However, there is a general class of hyperbolic transformations called *Möbius Transformations* that act in a manner very similar to the class of affine transformations. *Geometry Explorer* has the capability to define and use Möbius transformations. Consult Chapter 7 for an example that uses Möbius transformations.

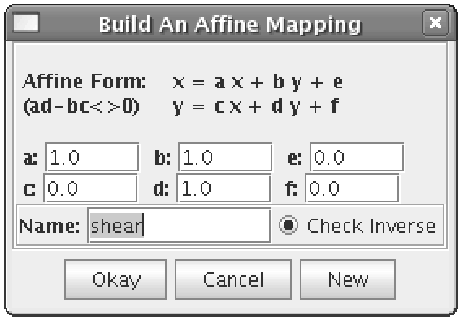
5.9.1 Affine Transformations on Circles

Circles do not transform as one would expect under affine transformations such as shears. Under a shear transformation if all of the points of a circle were to be transformed, then the circle would transform to an ellipse in some cases. However, for performance reasons, transformations in *Geometry Explorer* of objects such as circles, arcs, lines, filled areas, etc, only transform the defining points of these objects. Thus, when a circle is transformed in *Geometry Explorer*, the center and radius points are transformed and a new circle is constructed on these two transformed points. Thus, circles always transform to circles. Ellipses are planned for a future release of *Geometry Explorer*, at which time shears will transform circles in the way one would expect.

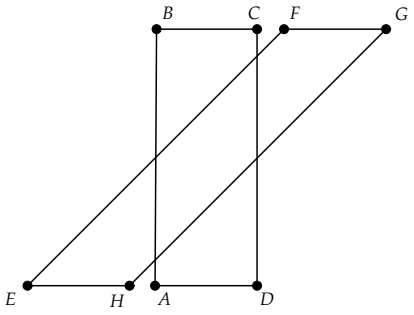
5.10 Editing Custom Transformations

When working with a custom transformation such as an affine transformation one may wish to alter the parameters of the transformation. This can be done using the transformation editing capability of *Geometry Explorer*.

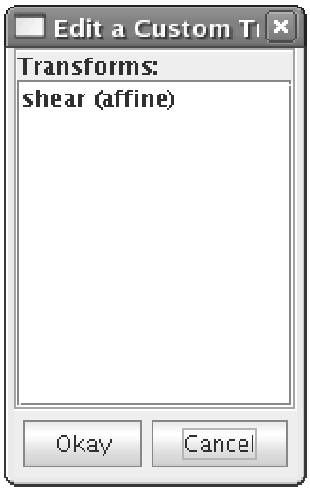
In the figure on the right we see the definition for the shear transformation used in the last section.



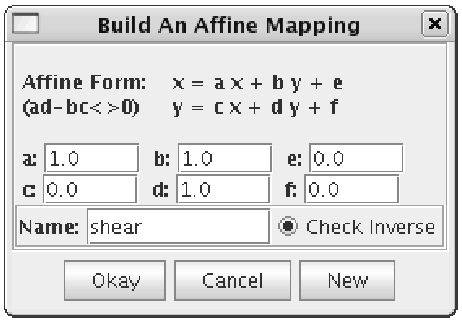
Applying this to $ABCD$ results in the transformed figure $EFGH$ as shown.



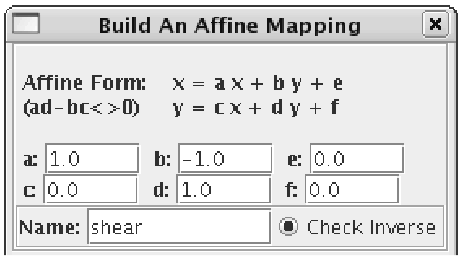
Suppose we wanted to investigate how a change in the parameter b in the definition of an affine transformation affects the behavior of that transformation. To edit the already defined transformation "shear', we first choose **Edit Transform...** from the **Custom** pop-up menu in the Transform Panel. The dialog at the right will pop up.



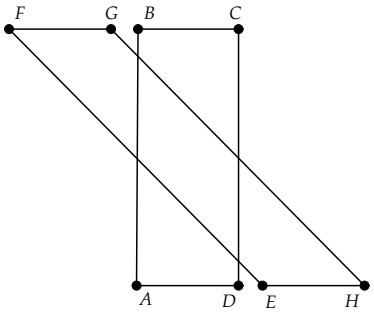
To edit a transformation we click on the transformation's name and then hit the Okay button. The affine transformation dialog box will pop-up with the values for the affine transformation in the input boxes.



At this point we can change any of the parameters for the affine transformation. Let's change parameter b to -1.0 and see what happens. Type in -1.0 for b and hit Okay.



Once we alter the definition of the "shear" transformation, the previously transformed figure $EFGH$ will transform to reflect these changes. We can see that making "b" negative switches the direction of the shear to the left.



Chapter 6

Analytic Geometry

> There was once a very brilliant horse who mastered arithmetic, algebra, plane geometry, and trigonometry. When presented with problems in analytic geometry, however, the horse would kick, neigh, and struggle desperately. One just couldn't put Descartes before the horse.
>
> —Anonymous

In 1637 Rene Descartes published the work *La Geometrie* in which he laid out the foundations for one of the most important inventions of modern mathematics—the Cartesian coordinate system and analytic geometry.

In classical Euclidean geometry points, lines and figures exist as idealized objects which are independent of any concrete context. Objects such as triangles and circles exist as unique objects with no apparent connection to each other or to other geometric figures.

The great insight of Descartes was to embed the study of geometric figures in a grid system, where a point is precisely located by its distances from two fixed lines that are perpendicular to one another. These two distances are called the *coordinates* of a point and are customarily labeled "x" and "y".

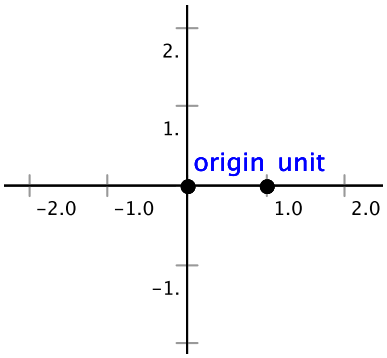
By studying the set of coordinates for a geometric figure one can identify patterns in these coordinates and, thereby discover the underlying relationship between x and y for the figure. For example, a line is a set of points (x, y) where x and y have a relationship of the form $ax + by + c = 0$, with a, b, and c being constants. Similarly the points making up a circle have their own x-y relationship. Descartes was able to unify the study of geometric figures under one philosophical principle—what we now call the *Cartesian* coordinate system.

This creation of Descartes ultimately led to the notion of *functions* and to the creation of calculus by Newton and Leibniz towards the end of the 17th century. The great achievement of analytic geometry is that it allows one to replace the study of collections of points by the study of the equations of the x-y relationships for those points. Thus, geometric questions can be analyzed algebraically.

6.1 The Coordinate System Used in *Geometry Explorer*

Analytic geometry is supported in *Geometry Explorer* through the use of a built-in coordinate system. This coordinate system is constructed from two lines meeting at right angles at a point called the *origin*. Additionally, there is a point on the horizontal axes (x-axis) called the *unit* point. The distance from the unit point to the origin is used as the unit distance in this coordinate system.

The built-in coordinate system is initially hidden. To see the coordinate axes choose **Show Axes** from the **Graph** menu (or right-click on a white space in the Canvas and clcik on **Axes** from the popup menu). To see the labels on the origin and unit points, use the Text/Label tool (the third one from the left in the Create Panel).



In the figure above the coordinate system runs from about -2.5 to 2.5 on the x and y axes. The unit point can be moved closer to the origin to increase the upper bounds on x and y. This effectively lets one *zoom* in on a plot. Similarly, by moving the unit point away from the origin one can zoom out from the origin.

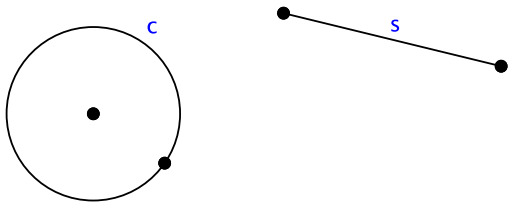
The origin and unit point can be treated just like any other point one may create in *Geometry Explorer*. These points can be selected, made part of other constructions, and moved about. Moving the origin has the effect of moving the entire coordinate system. Moving the unit point allows one to zoom in and out from the origin. Likewise, the two coordinate axes are lines and can be used like any other line.

The coordinate system can be used to plot points based on measurements and to plot the graph of a function $y = f(x)$.

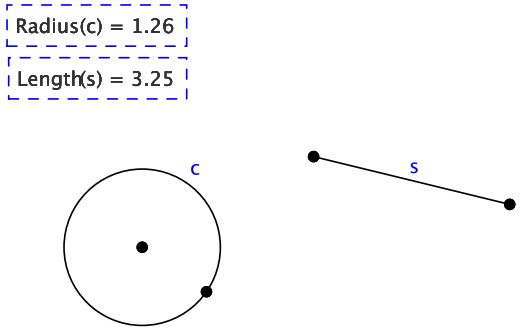
6.2 Plotting Points Based on Measurements

Points with coordinates (x, y) can be created in the coordinate system by selecting two measurements to serve as the x and y coordinates of the point.

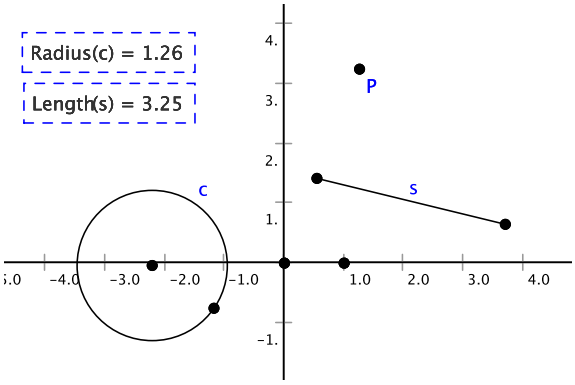
To illustrate this feature, let's plot a circle c and a segment s.



Now, let's measure the radius of circle c and the length of segment s. Select the circle and then choose **Radius** from the **Measure** menu. Then, select the segment and choose **Length** from the **Measure** menu.



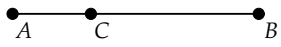
To graph these two measures as an (x, y) coordinate pair, we first show the coordinate axes by selecting **Show Axes** from the **Graph** menu. Then, we select the radius and length measures (in that order) and choose **Add As (x,y) Point from Measures** from the **Graph** menu. Point P will be created with x-coordinate equal to the radius of c and y-coordinate will equal to the length of s.



If we now alter either the size of circle c or segment s, point P will update correspondingly.

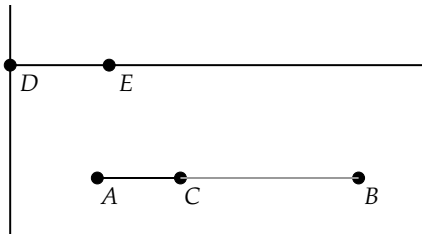
For another example of plotting measurements, let's consider the problem of finding the rectangle of maximal area subject to a fixed perimeter.

First, let's start with a line segment $\overline{AB}$ and attach a point C to this segment.

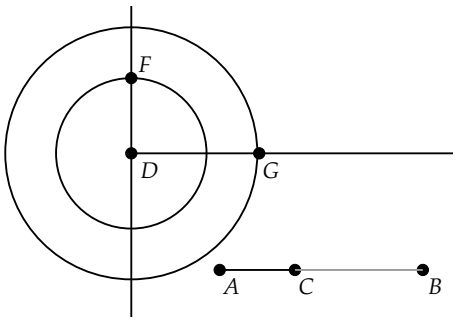


Now, hide $\overline{AB}$ and construct two segments $\overline{AC}$ and $\overline{CB}$. These will form the width and height of our rectangle. We will color these two segments differently so that we can keep track of which is which.

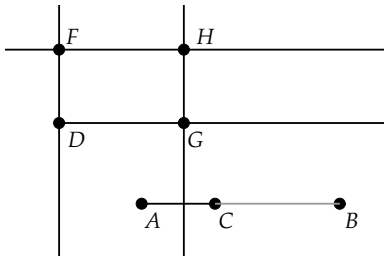
Let's create a horizontal ray $\overrightarrow{DE}$ and a perpendicular to $\overrightarrow{DE}$ at D. These will form a framework on which we will build our rectangle.



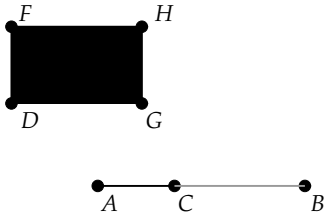
Hide point E and create a circle at D of radius equal to AC. To do this select D and $\overline{AC}$ and click on the Circle constructor tool in the Construct Panel (the second button from the left in the second row of buttons in the Construct Panel). Likewise, create another circle at D of radius CB. Finally, find the intersection of the first circle with the perpendicular (call this point F) and the second circle with the horizontal ray (call this point G).



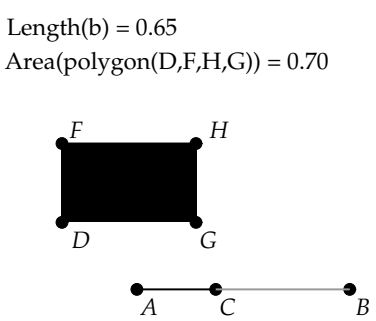
Next, hide the two circles and find perpendiculars to $\overline{DF}$ at F and to $\overline{DG}$ at G. Then, find the intersection of these two perpendiculars at point H.



To make things easier to analyze, hide all of the rays and lines. Then, select points D, G, H, and F and click on the Filled Polygon tool in the Construct Panel.

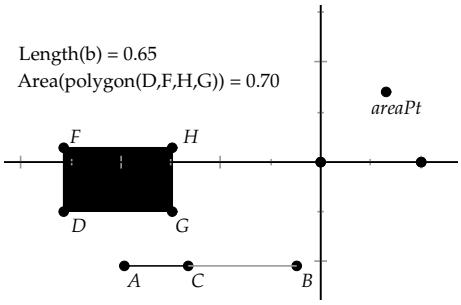


Select $\overline{AC}$ and choose **Length** from the **Measure** menu. Then, select the rectangle area (click inside the area) and choose **Area** from the **Measure** menu.

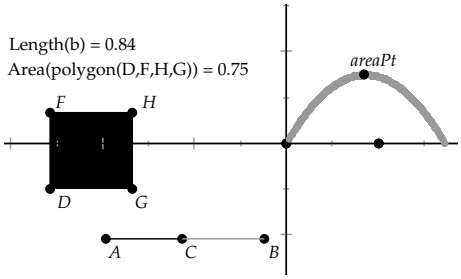


As we move point C back and forth along line $\overleftrightarrow{AB}$ we see that the area of the rectangle grows and shrinks. It appears that there is is a maximum value for the area. We will plot the length of $\overline{AC}$ and the area of the rectangle as a coordinate pair (x, y) to see what this relationship really is.

To see the coordinate axes, choose **Show Axes** under the **Graph** menu. Then, select the length measurement and area measurement (in that order) and choose **Add As (x,y) Point from Measures** from the **Graph** menu. A new point will be created (*areaPt* in the figure on the right) that is plotted with x-coordinate equal to the length measure and y-coordinate equal to the area measure.



Finally, let's put a trace on the plotted point. Select *areaPt* and choose **Trace On** under the **View** menu. Then, move point C back and forth.



Clearly, the relationship between the height of the rectangle and the area is a quadratic one. If we measured the x-coordinate at the maximal value on the traced curve we would see that the maximum area occurs where the height and width (AC and CB) are equal. Thus a square is the rectangle of greatest area for a fixed perimeter.

6.3 Analysis of Functions

6.3.1 Plotting $y = f(x)$

In the previous section we saw how one can use *Geometry Explorer* to plot the relationship between two measurements as a coordinate pair (x, y). In particular we saw that for the maximal rectangle problem, the relationship between the length of a side of the rectangle and the area is quadratic.

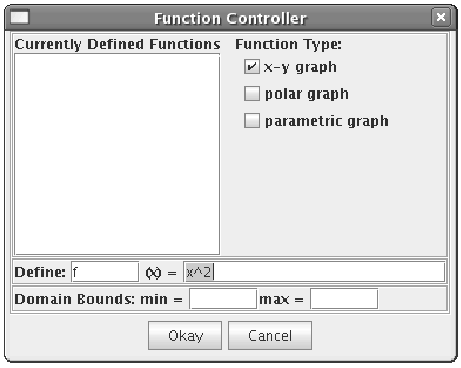
In many cases the relationship between two measurements will turn out to be a *functional* relationship. That is, for a given y-value there will only one corresponding x-value. We say that y is a function of x, or $y = f(x)$.

We can turn this process around and ask what the set of pairs (x, y) for a given functional relationship $y = f(x)$ will look like. For example, what

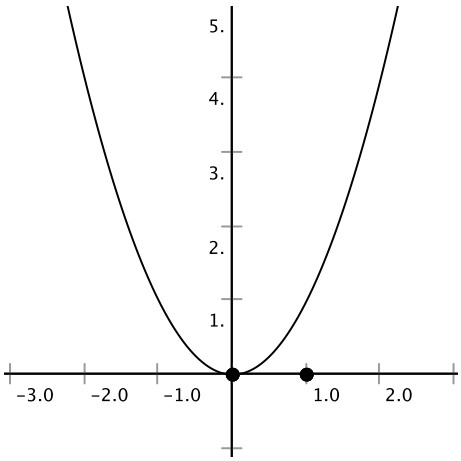
does the relationship given by $y = x^2$ look like? We will use the symbolic capability of *Geometry Explorer* to plot the set of points defined by this relationship. This set of points is called the *graph* of the function.

To begin make sure that the coordinate axes are visible. (If the axes are hidden, choose **Show Axes** the **Graph** menu.) To define a function choose **Add Function to Graph...** from the **Graph** menu.

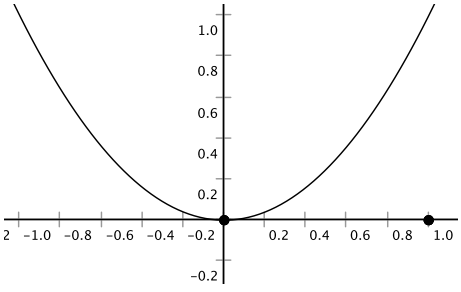
A dialog box like the one on the right will pop-up. In the field labeled "Define:" type in "f" for the name of our function and then type in "`x^2`" for the relationship between x and y. At this point we could also type in bounds for x if we so desired. Hit the Okay button to finish the definition of $f(x)$.



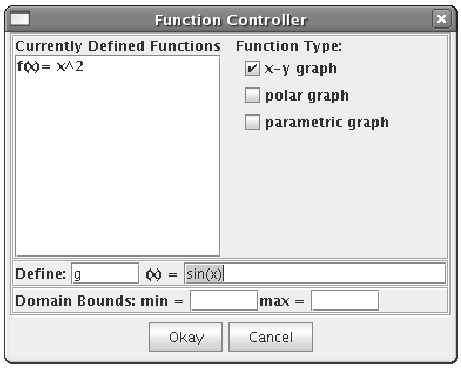
The graph of the function will appear in the Canvas. Note that when we do not define minimum and maximum bounds for x, *Geometry Explorer* will select the minimum and maximum bounds based on the x-values visible on the screen.



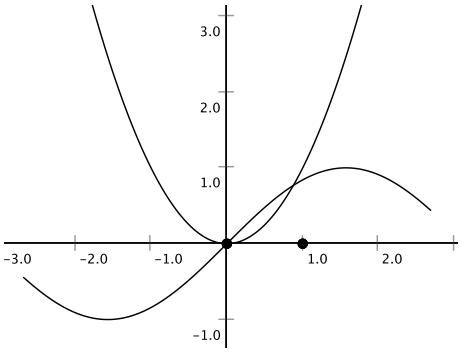
If we move the unit point away from the origin, we will zoom in on the origin, as shown in the figure on the right. Likewise, as we move the unit point toward the origin we will zoom out from the origin.



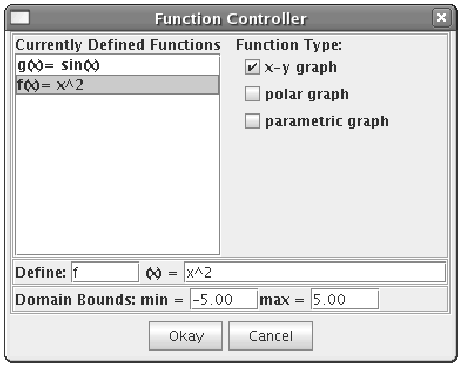
Suppose we now wanted to look at the graph of the function $y = sin(x)$. We would again choose **Add Function to Graph...** from the **Graph** menu. Type in "g" for the name of our function and "sin(x)" for the function and hit Okay.



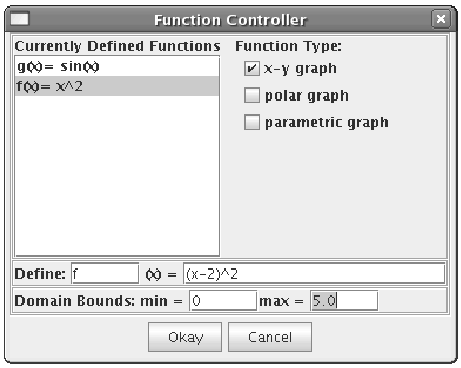
Now, both graphs appear in the coordinate system.



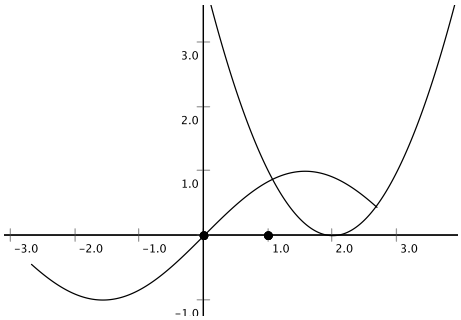
Suppose we want to alter the definition of $f(x)$. Instead of x^2 we want to look at $(x-2)^2$, i.e. a shift in the graph of $f(x)$. This can be done quite easily. Pop up the function controller dialog box again. You will see two functions in the defined list area. Click on $f(x)$. Instantly, the definition of $f(x)$ will appear in the lower portion of the dialog box.



Type in `(x-2)^2` for the new definition of $f(x)$. Let's also change the bounds on x for $f(x)$. Type in 0 and 5.0 for xmin and xmax. Then, hit Okay.



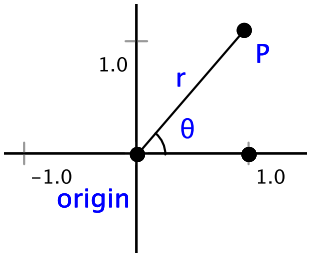
Note how the graph of $f(x)$ is instantly updated to reflect the changes we made in the definition of $f(x)$.



6.3.2 Plotting Polar Functions $r = f(\theta)$

In this section we will use the symbolic capability of *Geometry Explorer* to plot the set of points defined by the relationship between two variables r and θ, so-called *polar coordinates*.

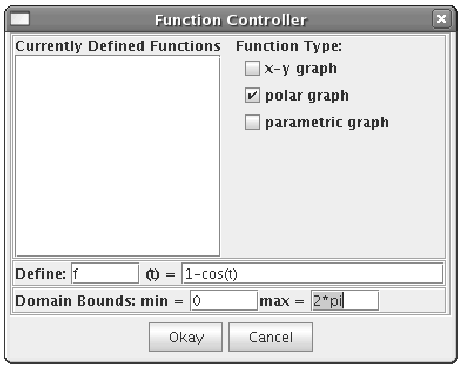
The coordinate r will represent the distance of a point P in the plane from the origin, and θ will represent the angle measure (in radians) of a point as determined from the positive x-axis counter-clockwise.



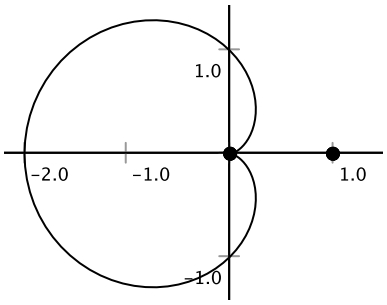
Geometry Explorer allows one to graph the functional relationship where r is a function of θ, that is, $r = f(\theta)$.

For example, suppose we want to graph the relationship $r = 1-\cos(\theta)$ for θ between 0 and 2π. We start by making sure that the coordinate axes are visible. (If the axes are hidden, choose **Show Axes** the **Graph** menu.) To define our function choose **Add Function to Graph...** from the **Graph** menu.

A dialog box like the one on the right will pop-up.Click the box labeled "polar graph" and type in "f" for the name of the function and "`1-cos(t)`" for the function. Then, type in 0 for the minimum bound and "2*pi" for the maximum bound and hit the Okay button.



The graph of the function will appear in the Canvas. Note that we type in t for the θ term in our polar function. Also note that *Geometry Explorer* can understand the symbolic expression "2*pi."



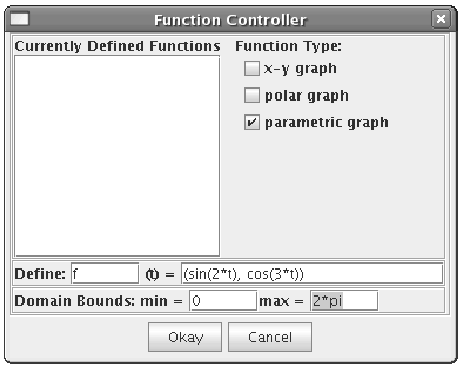
The graph of $r = 1 - cos(\theta)$ has a heart-like shape and is, in fact, called a *cardioid*.

6.3.3 Plotting Parametric Functions

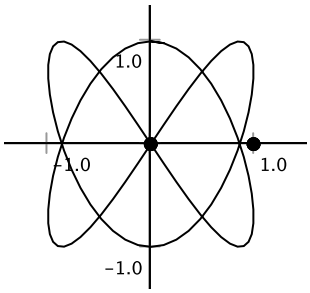
In this section we will see how *Geometry Explorer* can be used to plot the set of points defined by a *parametric* function $f(t) = (x(t), y(t))$.

As an example, suppose we want to graph the relationship $f(t) = (\sin(2*t), \cos(3*t))$ for t between 0 and 2π. We start by making sure that the coordinate axes are visible. (If the axes are hidden, choose **Show Axes** the **Graph** menu.) To define our function choose **Add Function to Graph...** from the **Graph** menu.

A dialog box like the one on the right will pop-up.Click the box labeled "parametric graph" and type in "f" for the name of the function and "`(sin(2*t), cos(3*t))`" for the function. Then, type in 0 for the minimum bound and "2*pi" for the maximum bound and hit the Okay button.



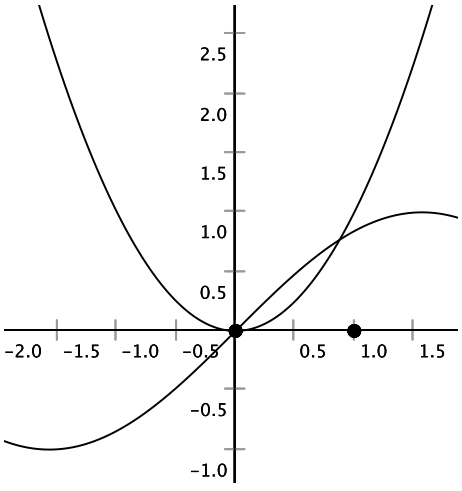
The graph of the function will appear in the Canvas. It is a quite interesting loopy curve. This curve is called a *Lissajous curve* or *Lissajous figure*.



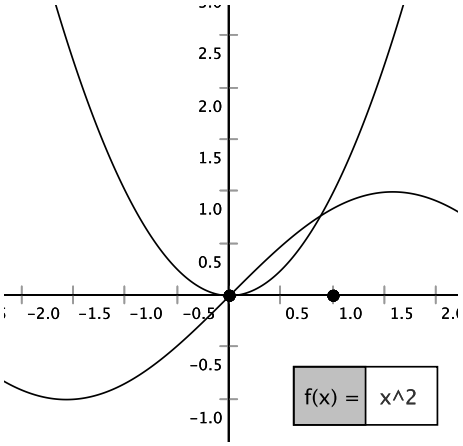
6.3.4 Adding Input Boxes for Functions

In the previous section we saw how we could change the definition of a graphed function by using the function controller dialog box. We can also change the definition of a function by adding an *input box* to the canvas. We can type a new definition for a function directly into this box and quickly modify an existing function.

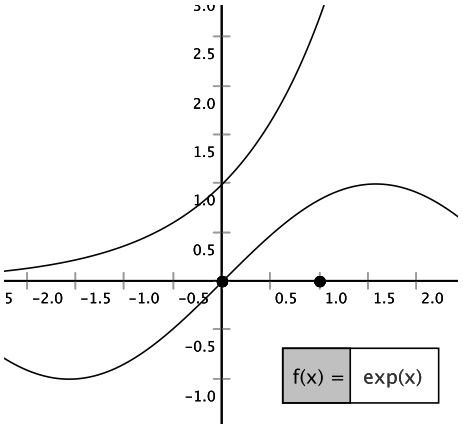
As an example, let's look at the two functions we graphed in the last section, that is $f(x) = x^2$ and $g(x) = \sin(x)$.



To add an input box for $f(x)$ we first select the graph of $f(x)$ (click somewhere on the graph) and then choose **Input Box for Function** in the **Graph** Menu.



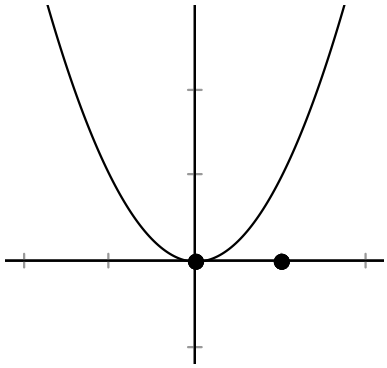
A text input box will be created on the canvas as shown. To change the definition of $f(x)$, click somewhere after the "=" and use the arrow keys and the backspace key to erase the given definition. We can then change the definition. In this case we will change the definition to $\exp(x)$, the exponential function. Type "exp(x)" in the input box and hit the return key. The graph will instantly reflect the change in the definition of $f(x)$.



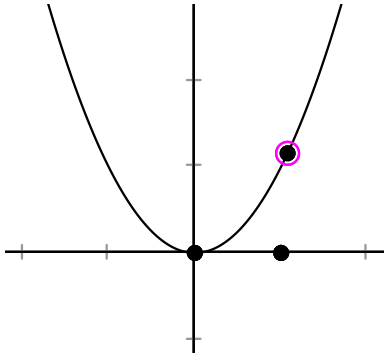
6.3.5 Attaching Points to the Graph of a Function

Points can be *attached* to the graph of a function so that when moved, they always travel along the graph. This attaching can be done in two ways.

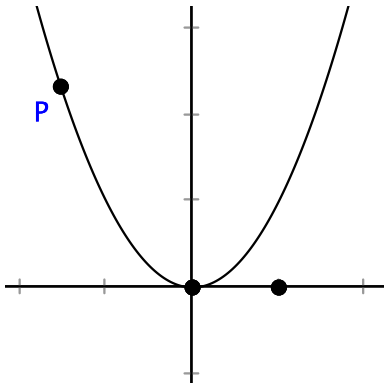
Points can be attached by using the Point tool and placing a point directly on top of the graph. In the figure at the right we have graphed $y = x^2$. (See the previous section for details on how to create this graph.)



Using the Point tool (second from right in top row of the Create Panel) we can click the mouse directly on top of the graph to attach a point to the graph. In the figure at the right we have placed a point directly on the graph near the point $(1, 1)$.

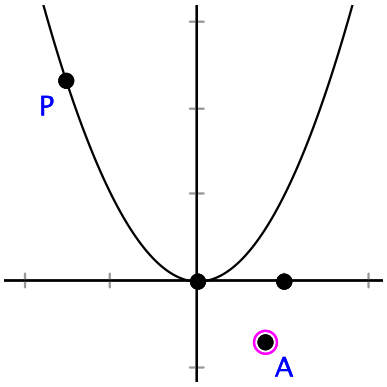


If we now select the point and try to move it, the point (labeled "P" here) will remain attached to the graph.

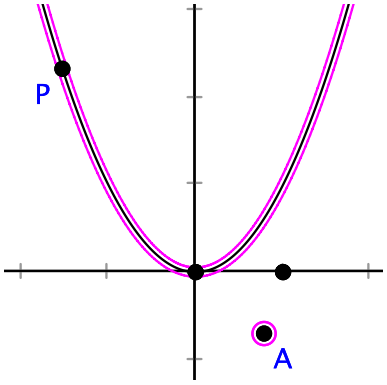


Another way that points can be added to the graph of a function is by defining a point on the graph in terms of the x-coordinate of another point in the Canvas.

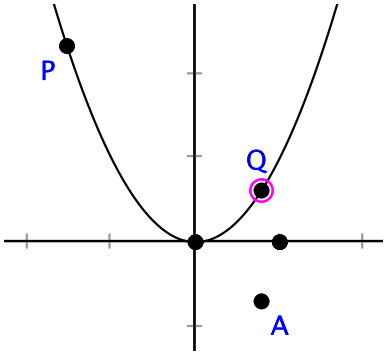
Here we have added point A to the example shown above.



To define a point on the graph using the x-coordinate of A we need to first select A and the graph of the function. (To select the graph just click on it with the selection tool.) Note that the graph is shown as selected when it is outlined in purple.



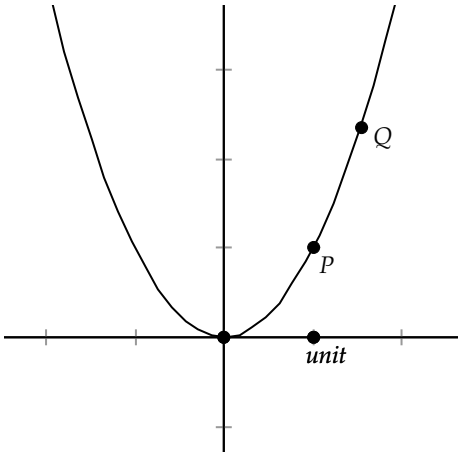
Once we have selected these two objects, we can choose **Add Point on Function from x-Point** from the **Graph** menu. Point Q will be created on the graph and will be linked to A. If we move A, point Q will move correspondingly along the graph.



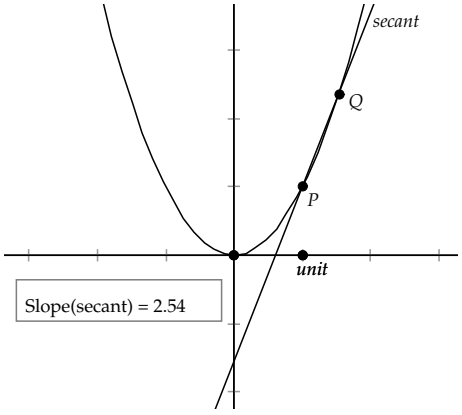
6.3.6 Tangents to Functions

One of the great contributions that calculus made to the study of functions was the discovery that the notion of *slope* could be extended from graphs that were straight (i.e. lines) to ones that were curved.

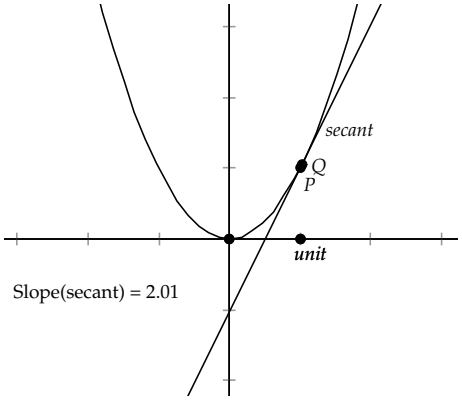
In the figure at the right we have constructed the graph of $f(x) = x^2$. We have also added point P to the graph by defining it in terms of the unit point on the x-axis. Lastly, we have attached point Q to the graph. (See the examples above for details on constructing graphs of functions and attaching points to functions)



Let's suppose that we want to define the slope of the graph at the point P (which is actually the point $(1, 1)$) We can get an average slope for the function between P and Q by constructing a line through P and Q and measuring its slope. The line connecting P and Q is called a *secant* line of the graph.

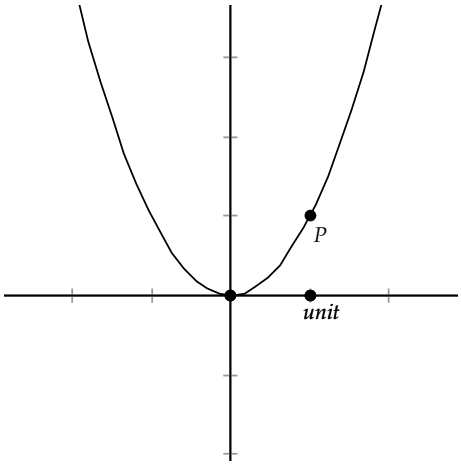


If we move point Q towards P the secant line slope will approach a specific value. In our case it appears that this value is about 2.0. We define the slope of the graph at P to be the limiting value of the secant line slopes. Also, a line through P with that value of slope will be called the *tangent* line to the graph at P.

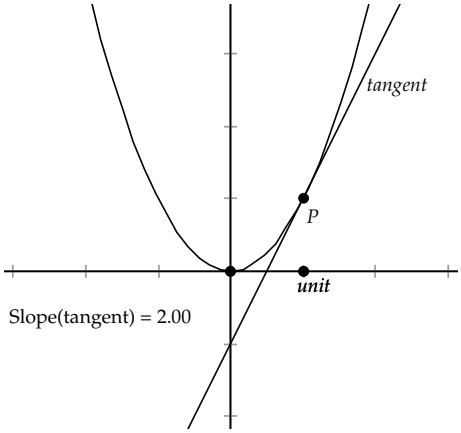


The construction of the tangent line to the graph of a function at a point on the graph can be carried out by *Geometry Explorer* without the limiting process described above.

Here again is the graph of $f(x) = x^2$ with point P as defined above.



To construct the tangent to the graph at P we need to first select P and then choose **Tangent to Arc/Circle/Function at Point** from the **Misc** menu. *Geometry Explorer* will calculate the tangent and construct it in the Canvas. In the figure at the right we have also measured the slope of the tangent line. Note the agreement with the limiting value of the secant line slope from above.



If we now move point P the tangent line will automatically be re-calculated and re-displayed.

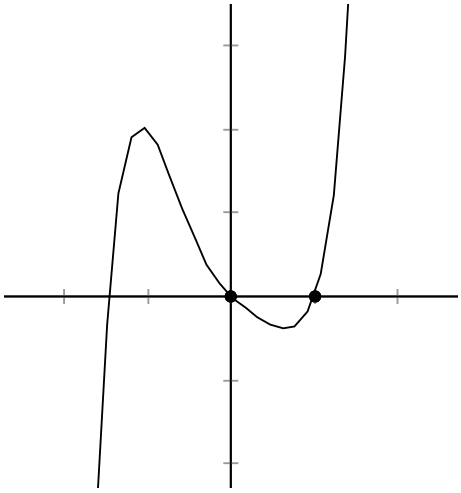
6.3.7 Derivatives of Functions

In the previous section we constructed the tangent line to the graph of a function at a point on that graph. Suppose we calculated the slope of this tangent line at every point on the function. We would get a new function which would measure the slope of the tangent line to $f(x)$ for each x-value for which $f(x)$ is defined. This function is called the *derivative* of $f(x)$ and is often written as $f'(x)$.

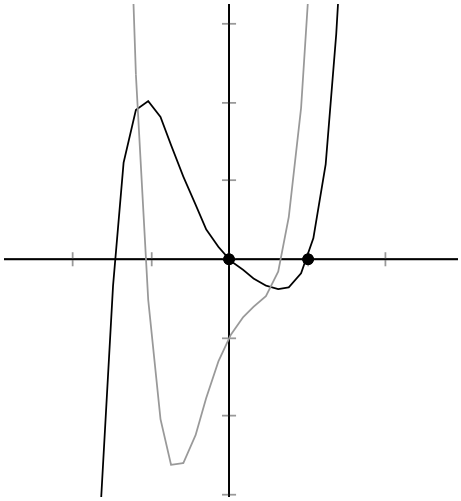
We can easily construct the derivative of a function in *Geometry Explorer* using the **Derivative of Function** option under the **Graph** menu in the main window.

Let's look at an example of using the derivative to find maximum and minimum values for a function.

In the figure at the right we have graphed the function $f(x) = x^5 - x^3 + x^2 - x$. (For a review of how to graph this function look at the preceding sections in this chapter)



To plot the derivative of $f(x)$ we first select the graph of $f(x)$ and then choose **Derivative of Function** from the **Graph** menu.



Note the relationship between where $f(x)$ reaches local maximum and minimum values and where the derivative (lighter color) crosses the x-axis. It appears, at least from this one example, that the derivative is zero where the function has local extreme values. Try changing the definition of $f(x)$ to see if this relationships persists.

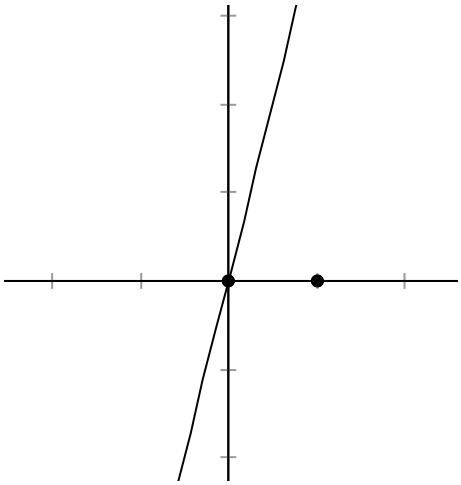
6.3.8 Iterated Functions and Dynamical Systems

One of the most fascinating new areas to be developed in mathematics in the 20th century is that of discrete mathematics and, in particular, the study of discrete dynamical systems. A *dynamical system* is a physical or mathematical system that changes over time. For a *discrete* dynamical system we require that we measure the change in time for the system in individual chunks, rather than continuously.

For example, a discrete dynamical system could be the population of moose in northern Minnesota each year. We will call P_n the size of the moose population n years after some initial recording of the population. We then get a series of population values P_0, P_1, and so forth, for the initial population, the population one year later, etc. We would have no information or data for the population at times other than these discrete values.

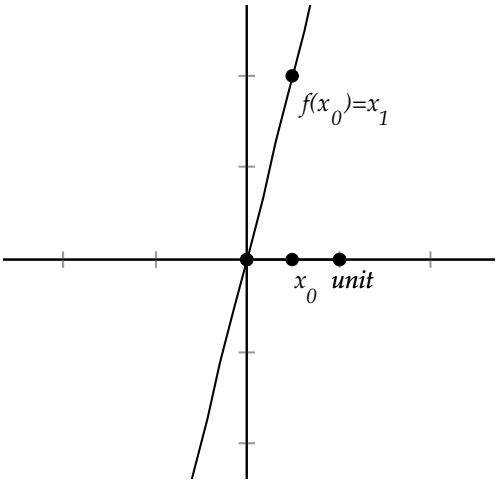
Suppose that we assume that the population at any given year is a constant multiple of the previous year's population. That is, $P_n = kP_{n-1}$. The multiplying factor k we can think of as a combination of the birth and death rates for that year. If the factor k is larger than 1.0, then it is easy to see that P_n will always be larger than P_{n-1} and that P_n will just keep getting bigger and bigger, without bound, as n grows. (This is assuming the initial population P_0 is bigger than 0.)

The discrete equation $P_n = kP_{n-1}$ has a close connection to the function $y = kx$. In the figure at the right we have graphed the function $f(x) = 4x$. (To do this make sure the axes are visible, choose **Add Function to Graph...** from the **Graph** menu, and type "f" for the function name and "4*x" for the function.)

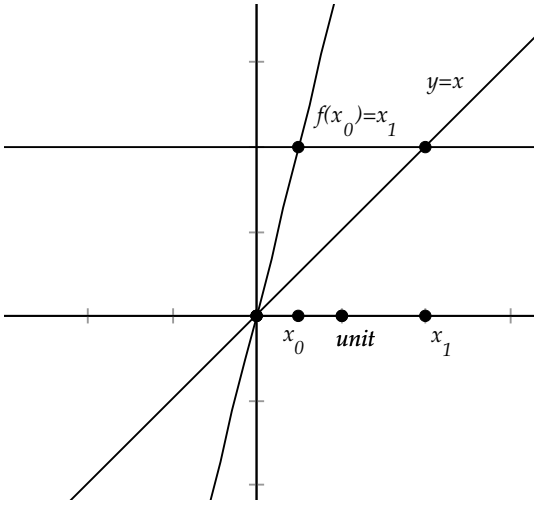


Now, consider the discrete equation $x_n = 4x_{n-1}$, and suppose that the initial value for x_0 is 0.5. Then, $x_1 = 40.5 = 2$ and x_1 will be the same value as $f(0.5)$ on the graph of $f(x) = 4x$.

To see this, attach a point "x_0" at the value of 0.5 on the x-axis. Select this point and the graph of $f(x)$ and choose **Add Point on Function from x-Point** from the **Graph** menu. The point on the graph of $f(x)$ corresponding to "x_0" is then constructed. This point will have y-coordinate equal to $f(x_0)$ which is equal to 2.0. Thus, the y-coordinate of this point will give us x_1.

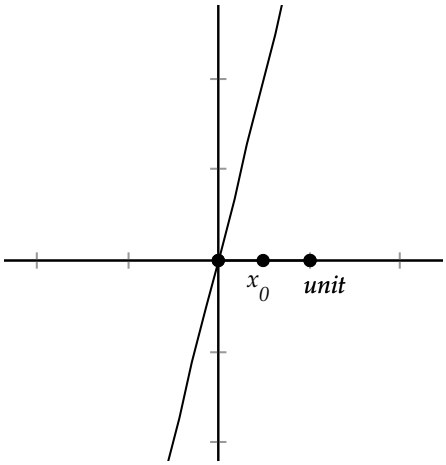


How would we get x_2 from x_1? We need to use the y-coordinate of the point we just calculated on the graph of $f(x)$ as a new x-value. One easy way to think about this is to follow a horizontal line from $y = f(x_0)$ over to the line $y = x$. Where these two lines intersect we can drop down to the x-axis to get the new x-value to use for $f(x_1)$. This is illustrated at the right.



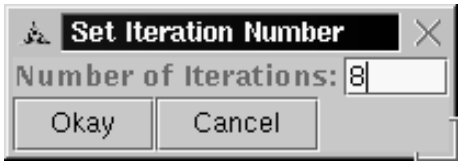
We could continue in this fashion, starting with an x-value on the x-axis, going up to the graph, moving across to $y = x$, dropping down to the x-axis and going back up to the graph again, yielding a set of x-values that are exactly the values x_1, x_1, etc for the discrete system $x_n = 4x_{n-1}$. This process of finding x_1, x_2, and so forth is called *iterating* the function $f(x) = 4x$ on the initial value of x_0. Carrying out these iterations can be quite tedious, so *Geometry Explorer* comes with a built-in tool to automate such calculations.

Let's start over with the graph of $f(x) = 4x$ and the initial value $x_0 = 0.5$.

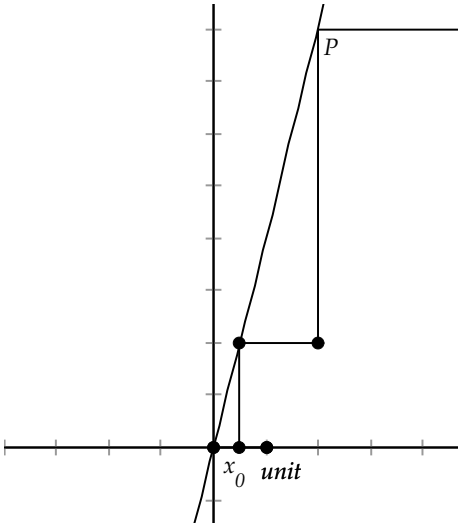


To iterate the process described above, select point `x_0` and the graph and then choose **Iterate Function From Point...** from the **Graph** menu.

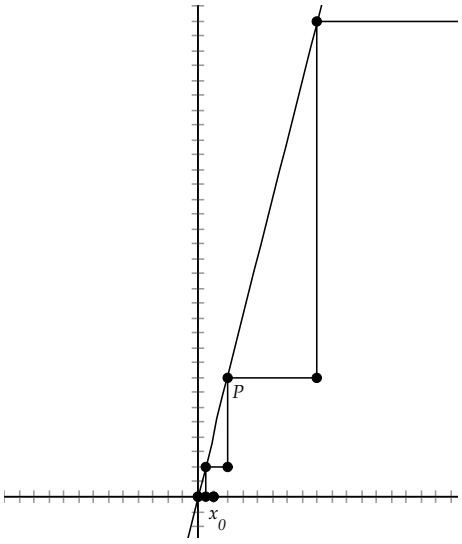
A dialog box will pop-up like the one on the right. We can set the number of iterations to carry out in the text field of this dialog box. In this example we will carry out 8 iterations. Type in "8" in the dialog and hit Okay.



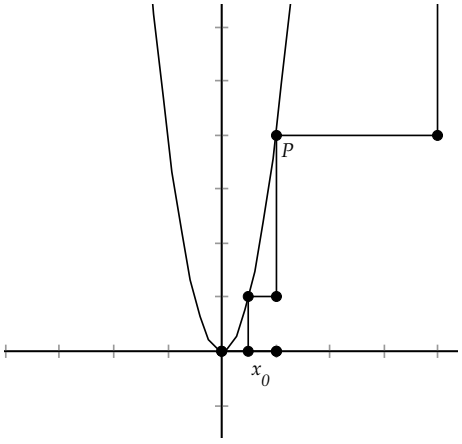
Geometry Explorer will then carry out the sequence of iterates starting at the initial point. From the initial value of 0.5 we can see that the first iterate will be 2 and the 2nd iterate will be 8 (the y-coordinate of point P). The next iterate goes off the screen.



If we move the unit point closer to the origin we will zoom out on the graph, allowing us to see more of the iterations of $f(x)$. We see that the third iterate will be 32.



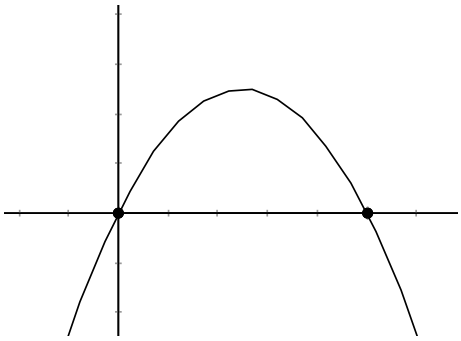
All of the iteration calculations are dynamically calculated based on the graph of $f(x)$. To see the power of this, suppose we change the definition of $f(x)$ to be $f(x) = 4x^2$. To make this change, choose **Add Function to Graph...** from the **Graph** menu, click on the definition of $f(x)$ in the defined functions list, and then type in "`4*x^2`" for the function. Hit Okay to make the change to $f(x)$. As you can see, the iterates of $f(x)$ also change correspondingly.



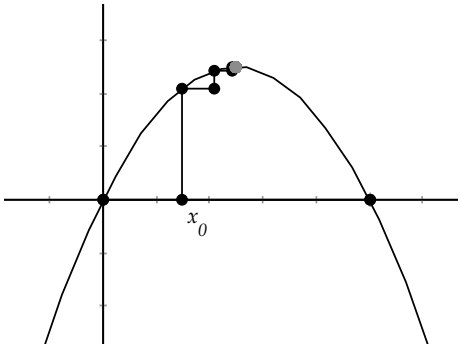
Iterated Functions and Chaos

In both examples above it is apparent that the iterates of the function grow larger and larger without bound or go to zero. This is not the case for all iterated functions.

Let's consider the function $f(x) = k * x * (1 - x)$ where k is a constant. In the figure on the right we have graphed the function for $k = 2$. (Define the function as in the examples above using the options under the **Graph** menu)

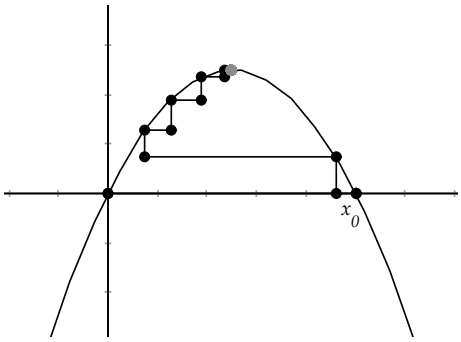


Now, we will attach a point "x_0" to the x-axis by using the Point tool and clicking on the x-axis between the origin and the unit point. Select this point and the graph and choose **Iterate Function from Point...** from the **Graph** menu. Type in "80" for the number of iterations and hit Okay to generate the iterations as shown.

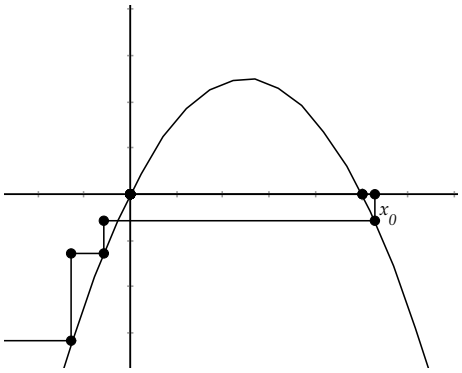


Notice how the iterations are always bounded, and actually seem to converge at the vertex of the parabola. (Note: As the number of iterations increases the color of the iterate point will fade. Thus, we can tell that the iterates are converging as all of the light colored points are bunched up at a single point)

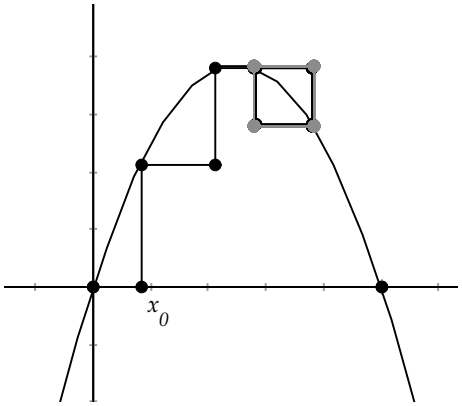
Does this convergence behavior occur only for this one choice of x_0? If we move x_0 anywhere between the origin and the unit point (not including these end points) we see that the iterates *always* converge to the vertex of the parabola. We call this convergence point an *attracting* point for the dynamical system since initial values are attracted to it under iteration.



If we move x_0 outside the interval between the origin and the unit point we see that the iterates decrease without bound.

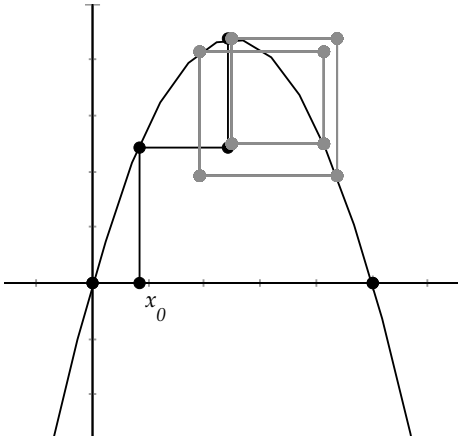


Now, let's alter the value of k slightly in our function. Edit the function by choosing **Add Function to Graph...** from the **Graph** menu, selecting $f(x)$ in the function list and typing in "3.1*x*(1-x)" for the definition. (You may have to move the graph a bit to get a good view of the iterates)

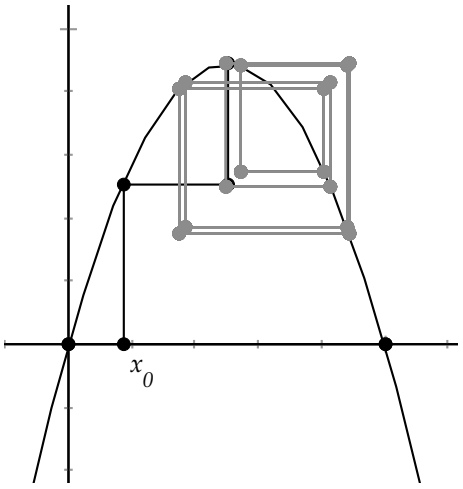


What we notice now is that the later iterates (the faded color points) appear to be bouncing between two attracting points which are no longer at the vertex of the parabola. The single fixed point has *bifurcated* into a cycle of two points to which the system has stabilized. We call this a *2-cycle* for the system.

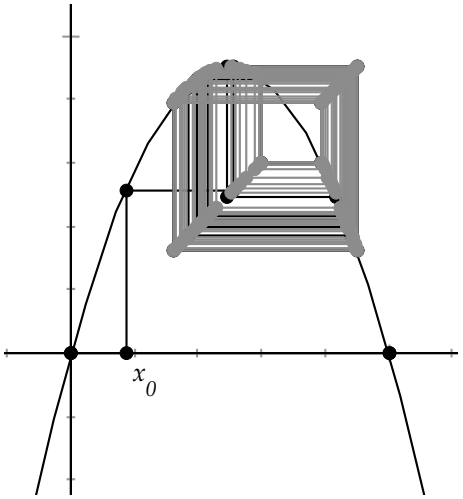
Let's alter k one more time to the value of 3.5. Amazing! The 2-cycle has bifurcated into a *4-cycle*.



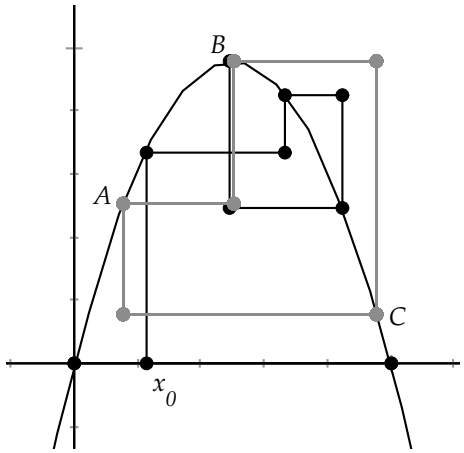
Now, set $k = 3.555$ and we see an *8-cycle*.



One might think that as we increase k we will get doublings of the cycle size, and this is what actually happens. However, looking at the pattern of k-values it appears that the next k value we would need to get a 16-cycle would be very close to 3.555. In fact, if we were to carefully plot the $k-$values as we moved k from 3.555 to 3.6 we would find a 16-cycle, 32-cycle, 64-cycle, etc until at $k = 3.6$ we get the figure at the right. At this value of k the cycle has become infinite in length and there is no way to follow it or predict what the dynamical system is doing. Mathematicians call this state of the system a *chaotic* state, for apparent reasons.



Interestingly enough, as we keep increasing k the system will periodically drop out of chaotic behavior. For example at $k = 3.83$ we find a three-cycle as described by the points A, B, and C in the figure at the right.



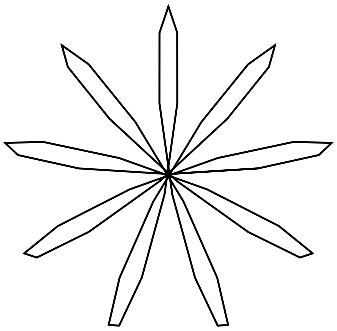
A very fine reference for the study of dynamical systems in chaotic behavior is the text *Does God Play Dice: The Mathematics of Chaos* [12] by Ian Stewart.

6.3.9 Controlling the Appearance of Plotted Functions

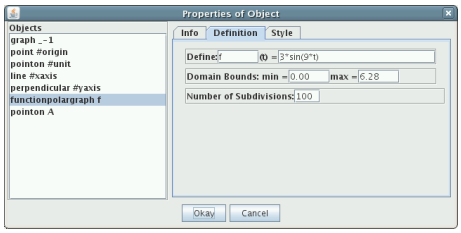
Making highly accurate plots of arbitrary functions is no trivial task. Many computer packages plot functions by subdividing the domain (interval on the x-axis) of the function into a certain number of sub-intervals, computing the value of the function at these points and then "connecting the dots" with line segments to get an approximation of the graph of the function. The default mode of graphing used in *Geometry Explorer* subdivides the interval of definition for the domain of a function to a level so that the resolution of the increments of the domain variable are less than one screen pixel. This gives a fairly good visual representation of most functions.

However, there may be some cases where one wants to increase the resolution for the graph of a function. For polar and parametric graphs, one can use the Properties Dialog Box to change the resolution.

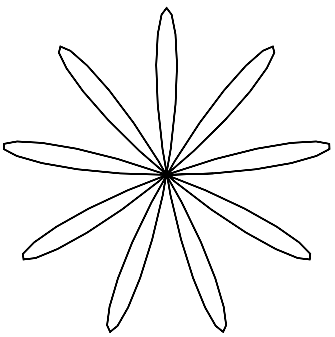
For example, here is the graph of the polar function $f(t) = 3 * sin(9 * t)$ with 100 subdivisions of the interval from 0 to 2π.



The graph looks fairly crude at this resolution. To change the resolution, right-click somewhere on the graph and choose **Properties...** from the popup menu. Click on the "Definition" tab in the Properties Dialog Box.

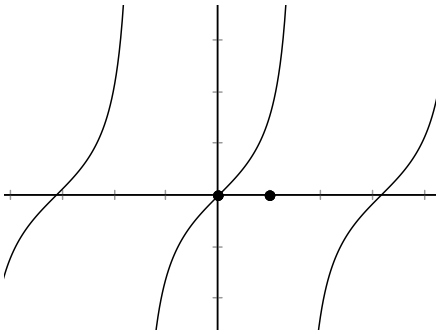


Type in "200" in the text field labeled "Number of Subdivisions" and hit "Okay." The graph looks much smoother now.



Handling Discontinuities

If we graph the function $f(x) = tan(x)$ we will see something like the figure shown at the right.



Note that the graph is correctly displayed in that $tan(x)$ has vertical asymptotes at periodic points on the x-axis. *Geometry Explorer* checks for discontinuities by looking at where the function is changing in an abrupt fashion. This simple algorithm works for most functions, but is not guaranteed to work for all functions having discontinuities.

Chapter 7

Hyperbolic Geometry

> I have discovered such wonderful things that I was amazed....out of nothing I have created a strange new universe.
>
> —János Bolyai (1802–1860), from a letter to his father, 1823

7.1 Background and History

Classical Euclidean geometry is an *axiomatic* system. In an axiomatic system one proves statements based on a set of agreed-upon statements called axioms, or postulates, which need no proof. Logically, one needs axioms in order to avoid an infinite regression of statements which depend on other statements which depend on others, etc. Axioms are supposed to be fairly self-evident and obvious to those working in the system.

In Euclid's axiomatic system there are five axioms/postulates:

1. Between any two distinct points, a segment can be constructed.
2. Segments can be extended indefinitely.
3. Given a point and a distance, a circle can be constructed with the point as center and the distance as radius.
4. All right angles are congruent.
5. Given two lines in the plane, if a third line l crosses the given lines such that the two interior angles on one side of l are less than two right angles, then the two lines if continued indefinitely will meet on that side of l where the angles are less than two right angles.

Postulates 1-4 seem very intuitive and self-evident. If one is going to have ruler and compass constructions, then one needs to construct segments, extend them, and construct circles. Also, geometry should be uniform so that angles do not change as we move objects around. Thus, the axiom on right angles is needed.

The fifth postulate, the so-called *parallel postulate*, seems overly complex for an axiom. It is not at all self-evident or obvious and reads more like a theorem.

In fact, many mathematicians tried to find simpler postulates, ones that were more intuitively believable, to replace Euclid's fifth, with the hope that the fifth postulate could then be proved from the first four postulates and the new postulate.

One of these substitutes is called *Playfair's Postulate*:

> Given a line and a point not on the line, it is possible to construct exactly one line through the given point parallel to the line.

It turns out that this postulate is logically equivalent to Euclid's parallel postulate; replacing Euclid's fifth postulate with Playfair's Postulate does not really simplify Euclid's axiomatic system at all.

In the 1800's several mathematicians experimented with negating Playfair's axiom to see if a contradiction to the first four Euclidean postulates could be reached. János Bolyai, Carl Friedrich Gauss, and Nikolai Lobachevsky all worked on negating Playfair with the following postulate:

> Given a line and a point not on the line, it is possible to construct *more than one* line through the given point parallel to the line.

Gauss, one of the greatest mathematicians of all time, was perhaps the first to work with this negated parallel postulate. Harold Wolfe [13, page 47] describes a letter Gauss wrote to a friend about his work on non-Euclidean geometry:

> The theorems of this geometry appear to paradoxical and, to the uninitiated, absurd; but calm, steady reflection reveals that they contain nothing at all impossible. For example, the three angles of a triangle become as small as one wishes, if only the sides are taken large enough; yet the area of the triangle can never exceed a definite limit, regardless of how great the sides are taken, nor indeed can it ever reach it. All my efforts to discover a contradiction, an inconsistency, in this Non-Euclidean Geometry have been without success

In fact all three mathematicians, Gauss, Bolyai, and Lobachevsky, developed non-Euclidean geometry (one which used the negated Playfair postulate) *without* finding any contradictions. This did not mean however that such a strange geometry was logically consistent. There was still the possibility that one of the three just missed looking at a statement that would lead to a contradiction.

The consistency of non-Euclidean geometry was demonstrated by Beltrami, Klein, and Poincaré in the 1800's to early 1900's. What these three did was to create a model inside of Euclidean geometry, with definitions of points, lines, circles, and angles, where Euclid's first four postulates where true and where the negated Playfair Postulate was true. Since the model was created within Euclidean geometry, then if non-Euclidean geometry had an internal contradictory statement, then that statement, when translated into its Euclidean environment, would be an internal contradiction in Euclidean geometry! Thus, if one believed Euclidean geometry was consistent, then non-Euclidean Geometry was equally consistent. Traditionally, the non-Euclidean geometry described by Beltrami, Klein, and Poincaré has been labeled *Hyperbolic* geometry.

Geometry Explorer employs three different models of Hyperbolic geometry, the Poincaré disk model, the Klein disk model, and the Upper Half-Plane model, as environments for the exploration of the "strange new universe" of non-Euclidean geometry.

7.2 The Poincaré Disk Model

In the Poincaré disk model the universe of points for the geometry is the set of points interior to a circular disk. This disk is called the *Poincaré disk*. Lines in this geometry are quite different than Euclidean lines. A line is defined as a circular arc within the disk that meets the boundary circle at right angles. Some lines in the Poincaré model of non-Euclidean geometry are shown in (Fig. 7.1)

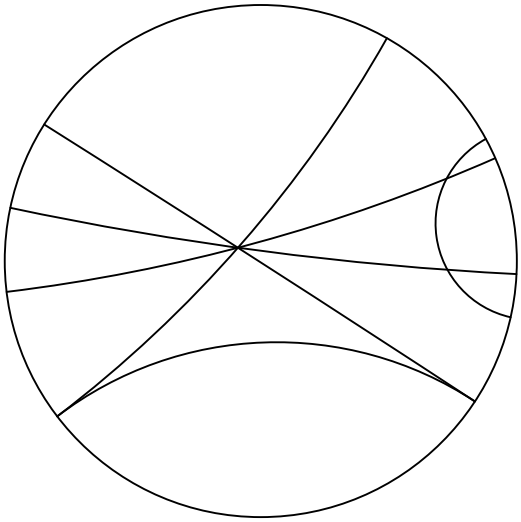
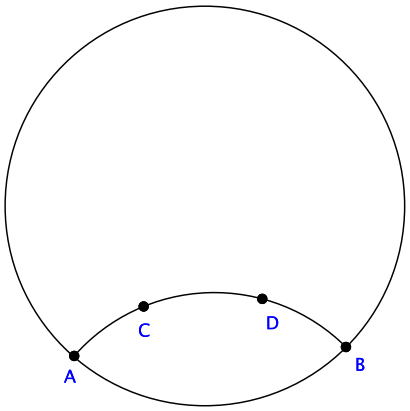


Fig. 7.1 Lines in the Poincaré Disk Model

To define circles, we need a notion of distance. Since the boundary of the Poincaré disk is not reachable in Hyperbolic geometry, we want a definition of distance such that the distance goes to infinity as we approach the boundary of the Poincaré disk.

In the figure at the right we have two points C and D in the Poincaré disk. There is a unique line (Euclidean arc $ACDB$) joining C and D that meets the boundary of the disk at points A and B.



We define the *distance* from C to D as

$$CD_{hyperbolic} = \left| \ln \left(\frac{CB/CA}{DB/DA} \right) \right|$$

where AC, CB, AD, and DB are Euclidean lengths.

As C approaches A, or D approaches B, the fraction inside the logarithm function becomes infinite, and thus the distance goes to infinity, as desired.

We now define a circle of radius R centered at a point P in the Poincaré disk as the set of points in the Poincaré disk whose hyperbolic distance to P is R. It turns out that this set of points is actually a Euclidean circle, but the Euclidean center does not match the Poincaré circle center P. Since distances go to infinity at the boundary of the Poincaré disk, then for any point in the disk, and any finite radius, a circle centered at that point with the specified radius exists.

Finally, angles are defined as they are in Euclidean geometry. We use the Euclidean tangent line to lines (i.e. Euclidean circular arcs) in the Poincaré model to determine angles.

Let us check Euclidean postulates 1–4 in this model. Lines can certainly be constructed for any two distinct points. Lines can be indefinitely extended because of the distance going to infinity at the boundary. Circles of any radius can be constructed. All right angles are congruent, as angles mean the same as they do in Euclidean geometry.

Thus, the Poincaré disk model satisfies the first four Euclidean postulates. In (Fig. 7.1) it is not hard to see that multiple parallels exist to a given line through a point not on that line. So, the Poincaré model satisfies the negated Playfair axiom as well.

We conclude that Hyperbolic geometry in the Poincaré disk is just as logically consistent as Euclidean geometry is. In this chapter we will see how to explore this new geometry. But, first we will consider two other models of Hyperbolic geometry.

7.3 The Klein Disk Model

In the Klein disk model of Hyperbolic geometry the universe of points for the geometry is the set of points interior to a circular disk, just as it was for the Poincaré disk model. The difference in the Klein model is that lines are defined as parts of Euclidean lines that intersect the interior of the disk. In (Fig. 7.2) we see the same set of lines as in (Fig. 7.1), but now viewed in the Klein model.

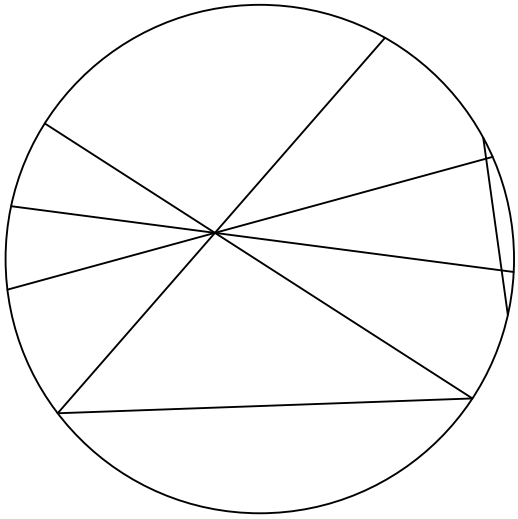


Fig. 7.2 Lines in the Klein Disk Model

Distance in the Klein model is defined almost the same way as it is in the Poincaré model.

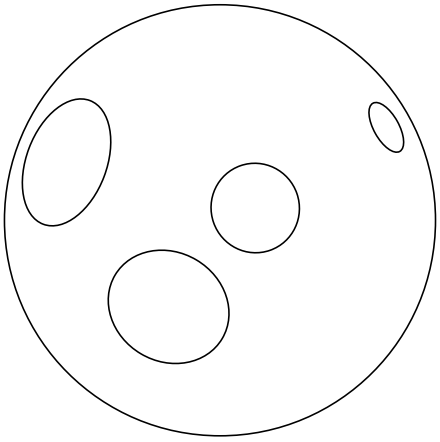
The Klein distance from C to D is defined as

$$CD_{klein} = \frac{1}{2}\left|\ln\left(\frac{CB/CA}{DB/DA}\right)\right|$$

where A and B are the points where the Euclidean line through C and D intersects the boundary circle, and AC, CB, AD, and DB are Euclidean lengths.

With the changes in interpretation of line and distance, it may not be too surprising that circles and angles turn out to be rather strange.

Here are a few circles in the Klein model. In general Klein circles will be Euclidean ellipses.



It turns out that we again show that the Klein model satisfies the first four Euclidean postulates. In (Fig. 7.2) it is not hard to see that multiple parallels exist to a given line through a point not on that line in this model. So, the Klein disk model satisfies the negated Playfair axiom as well.

7.4 The Upper Half-Plane Model

In the Upper Half-Plane model of Hyperbolic geometry the universe of points for the geometry is the set of points with positive y-coordinate, that is the set of points in the half plane above the x-axis. Lines will now be parts of circles that meet the x-axis at right angles. In (Fig. 7.3) we see the same set of lines as in (Fig. 7.1), but now viewed in the Upper Half-Plane model.

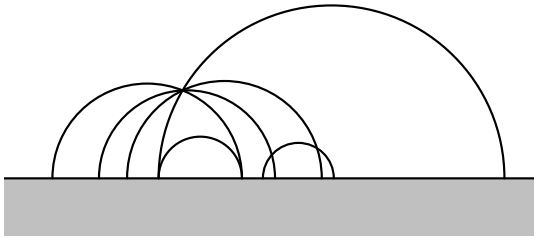


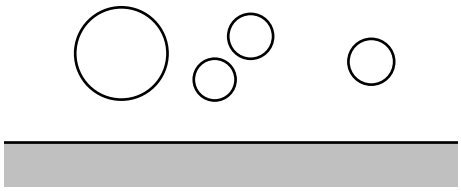
Fig. 7.3 Lines in the Upper Half-Plane Model

Distance in the Upper Half-Plane model is defined in a way that makes the distance function consistent with that of the other two models. In fact, one can define a function that maps points in the Upper Half-Plane model

to points in the Poincaré model in such a way that the mapping preserves distance. We say that the two models are *isomorphic*. For details on how this is done, consult the book by Hvidsten [6].

Circles in the Upper Half-Plane model turn out to be Euclidean circles, although with hyperbolic centers different from the corresponding Euclidean circle centers. Angles are defined in terms of Euclidean angles.

Here are a few circles in the Upper Half-Plane model.

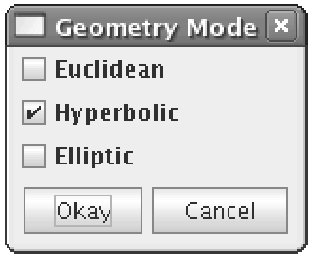


Due to the existence of the isomorphism mentioned above between the Upper Half-Plane model and the Poincaré model, the Upper Half-Plane model will satisfy the first four Euclidean postulates and also the negated Playfair axiom.

7.5 Working in the Hyperbolic Canvas

When one opens a new *Geometry Explorer* window (using the **New** menu option under the **File** menu) a dialog box pops up asking which of the three geometries — Euclidean, Hyperbolic, or Elliptic — will be used in the new window.

To start working in Hyperbolic geometry click on "Hyperbolic" in the dialog box and click Okay.



A *Geometry Explorer* window will pop up with a view of the Poincaré disk model of Hyperbolic geometry in the Canvas. (Fig. 7.4). The Poincaré disk model is the default model used by *Geometry Explorer*. To switch to the other models, just choose one of the three options listed under the **Model** menu in the main window. In the remaining sections of this chapter

we will illustrate constructions in the Poincaré disk model, even though all constructions are equally valid in all three models.

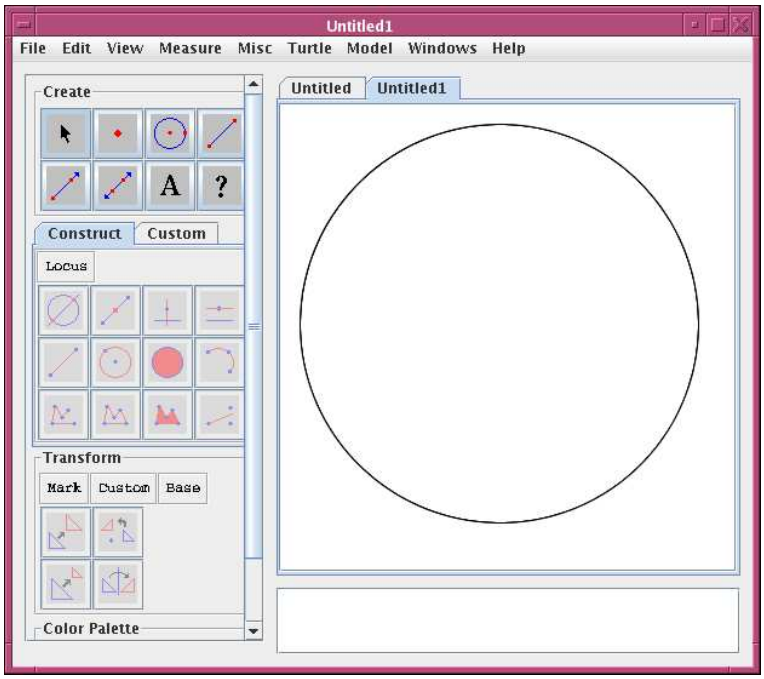


Fig. 7.4 The Hyperbolic Main Window

The hyperbolic window looks almost identical to the Euclidean window. Working in Hyperbolic geometry with *Geometry Explorer* is essentially no different than working in Euclidean geometry. Almost all of the tools work in both environments. Exceptions include:

1. In the Euclidean canvas the Parallel tool in the Construct Panel is used to construct the *unique* parallel for a line and a point off the line. In Hyperbolic geometry there are no unique parallels. In the hyperbolic environment, using the Parallel tool (with the same selection of a linear object and a point) will result in the creation of two parallels called *limiting parallels*. In (Fig. 7.5) we see the two (unique) limiting parallels to line a through point A (the parallels are the lines that are selected). These are parallels since they are lines through A that do not intersect line a (although they intersect at the boundary, they are still parallel, as the boundary is not in the hyperbolic plane)

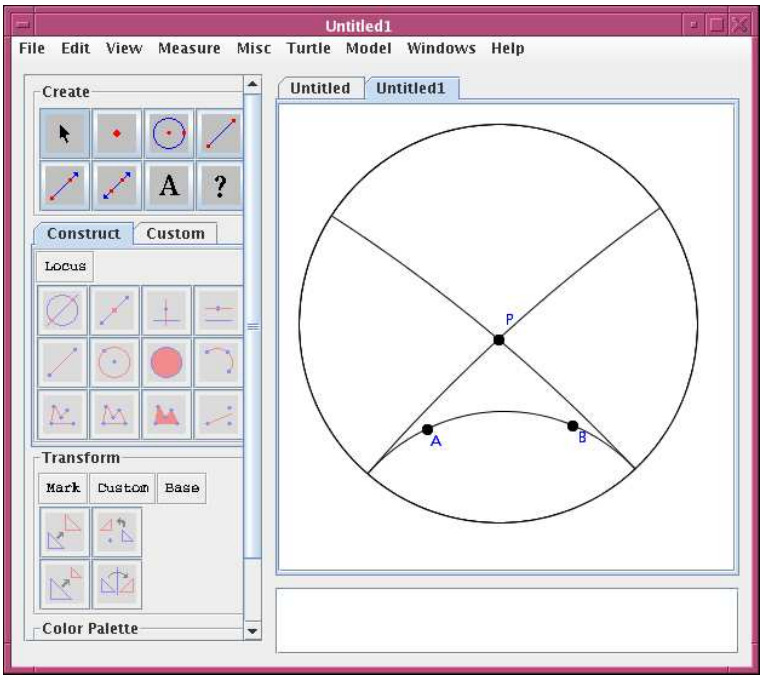


Fig. 7.5 Limiting Parallels to Line "a" through point A

2. In the Euclidean canvas, circles and arcs can be defined using three points. This construction depends on the Euclidean parallel postulate, (i.e. the uniqueness of parallels) and thus is not available in the hyperbolic canvas.

3. There is no **Graph** menu in the hyperbolic window.

4. Some measurements are different. There is no x- or y-coordinate measure and no slope measure. These depend on a coordinate system. However, there is a new measure—the *defect* measure. The defect is the difference between 180 degrees and the angle sum of a triangle in Hyperbolic geometry.

5. There is a different set of options under the **Misc** menu. See Chapter 9 for more info on how to use the options in this menu.

Other than these few differences, one can move back and forth between Euclidean and Hyperbolic geometry quite easily. Recordings made in Euclidean geometry can even be played back in Hyperbolic geometry, as long as they avoid the parallel constructions described in the four exceptions above.

Since the two environments are essentially the same, one can look at other sections of this manual to learn how to do various constructions, measurements, recordings, etc, in the hyperbolic Canvas.

In the next two tutorials, we show how to use *Geometry Explorer* to explore two interesting aspects of Hyperbolic geometry.

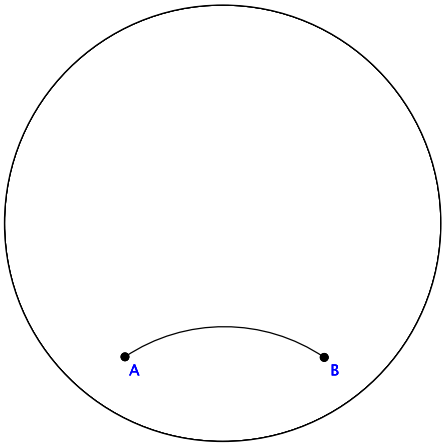
7.6 Saccheri Quadrilateral

Girolamo Saccheri (1667–1733) was a Jesuit priest who, like Gauss and the others mentioned above, tried to negate Playfair's Postulate and find a contradiction. He published a book entitled "Euclid Freed of Every Flaw" just before he died (modest fellow).

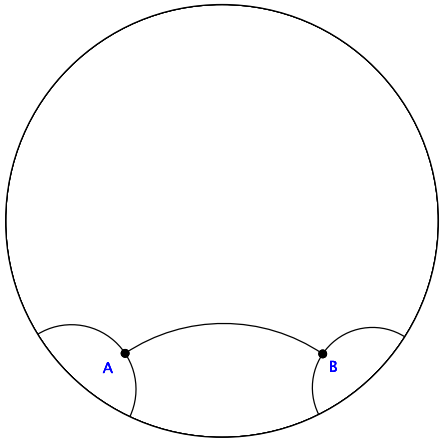
His work dealt with quadrilaterals $ABCD$ whose base angles are right angles and whose base-adjacent sides are congruent. We call such quadrilaterals *Saccheri Quadrilaterals*. Of course in Euclidean geometry, a Saccheri quadrilateral must be a rectangle, i.e. the top (or summit) angles must be right angles.

Let's look at Saccheri quadrilaterals in Hyperbolic geometry.

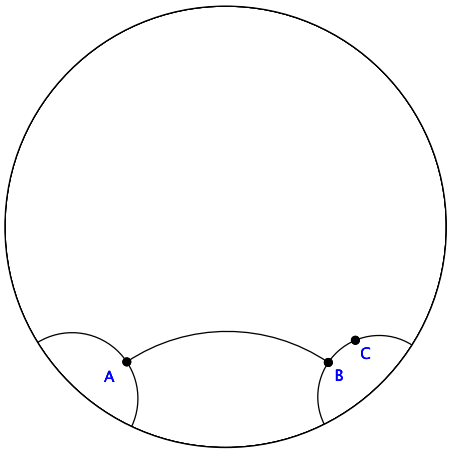
We start by creating a segment $\overline{AB}$ that will serve as the base of our quadrilateral.



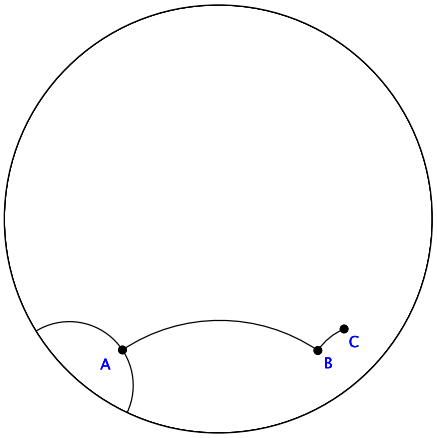
Next we construct perpendiculars to $\overline{AB}$ at A and B. Select $\overline{AB}$ and A and click on the Perpendicular tool in the Construct Panel. Then, select $\overline{AB}$ and B and click on the Perpendicular tool again.



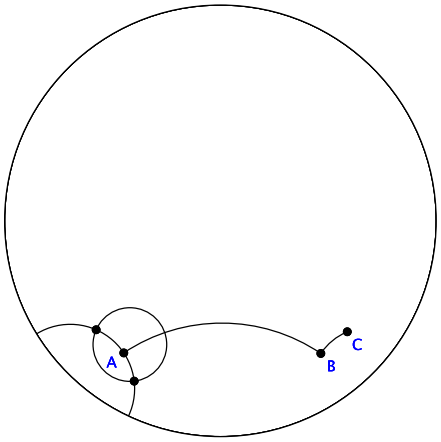
Now, attach a point C along the perpendicular at B above $\overline{AB}$ as shown.



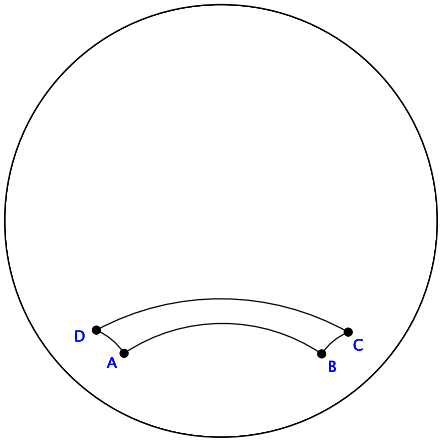
Hide the perpendicular line at B and connect B and C by a segment.



Select A and $\overline{BC}$ and click on the Circle tool in the Construct Panel, producing a circle centered at A of hyperbolic radius the length of $\overline{BC}$. Select the circle and the perpendicular at A and click the Intersect tool to find their intersection.

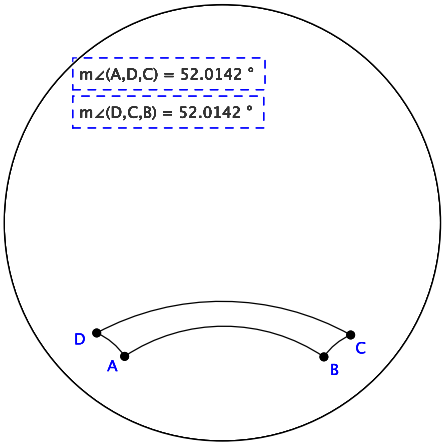


The upper intersection point is all we need, so hide the circle, the perpendicular, and the lower intersection point. Then, connect A to D and D to C to finish the construction of a Saccheri Quadrilateral.

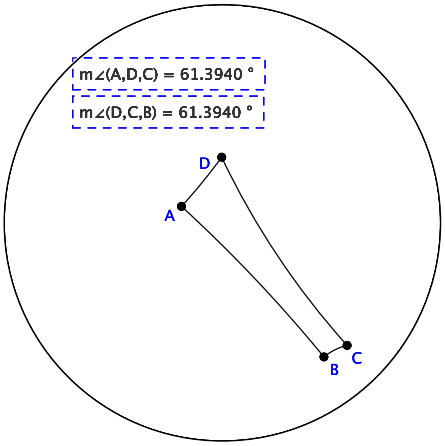


At this point we can use our Saccheri Quadrilateral to study many fascinating properties in Hyperbolic geometry. Let's look at a couple.

Measure the two upper angles in the quad. For this configuration they are equal and less than 90 degrees, which is what they would be in Euclidean geometry. Is this always the case?



It appears that this result holds for all configurations of our Saccheri Quadrilateral (except perhaps when orientations switch because our intersection point switches from above the rectangle to below it and the angles become greater than 180 degrees). In fact, this is a theorem in Hyperbolic geometry—that the summit angles of a Saccheri Quadrilateral are always equal and less than 90 degrees (i.e. are acute).



Saccheri himself could find no contradictions in assuming that the summit angles of this quadrilateral were acute rather than right angles. But, he could not believe what his own work was telling him. He ended up resorting to insults:

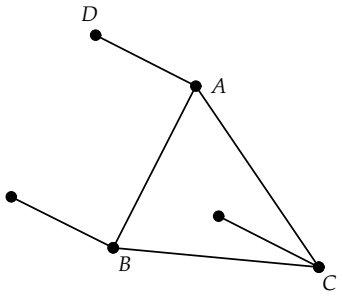
> The hypothesis of the acute angle is absolutely false, because [it is] repugnant to the nature of the straight line! [2, page 125]

7.7 Translation, Parallel Transport, and Holonomy

Translation in Euclidean geometry is a fairly simple process. To translate an object we essentially move it along a straight line. All parts of the object

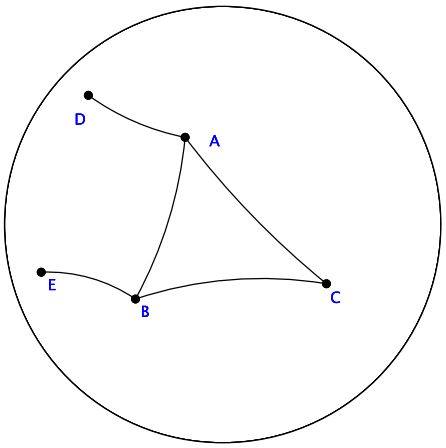
being translated must move along the line in a parallel fashion. To illustrate this idea, suppose that we are carrying a stick along a straight path. Then, a true translation of this stick along the path must keep the ends of the stick at fixed angles to the path. If the angle that the stick makes with the path is kept fixed as we move along the path, we say that we are performing a *parallel transport* of the stick along the path.

In the figure at the right we start with segment $\overline{AD}$ and triangle path $ABCA$ (start at A, move to B, then C, then return to A). Suppose we parallel transport $\overline{AD}$ to B, then again to C, then again to A. The orientation of the stick after returning to A is identical to the original orientation of the stick—it points in the same direction.

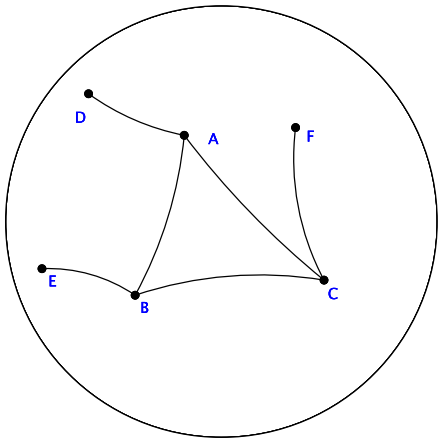


Translation in Hyperbolic geometry is not quite so simple. Let us look at what happens as we translate a segment around a triangle in the Poincaré model. Start by creating a triangle ΔABC and a segment $\overline{AD}$. Define a translation from A to B by selecting A and B (in that order) and choosing **Vector** from the **Mark** menu in the Transform Panel. A dialog box will pop up. Choose "Rectangular" for a simple translation and then hit Okay. We have now defined a hyperbolic translation from A to B.

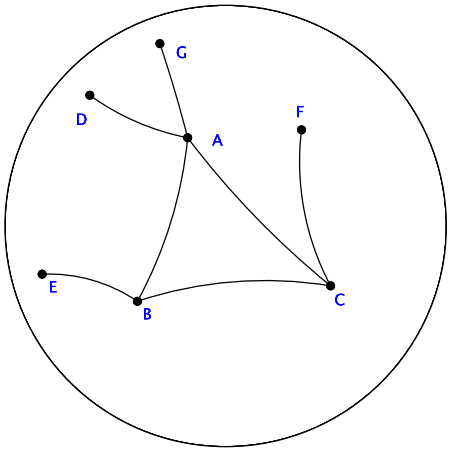
To translate $\overline{AD}$ by this translation, select $\overline{AD}$ and point D and then click the Translate tool in the Transform Panel. $\overline{AD}$ will be parallel transported to point B resulting in the segment $\overline{BE}$, as shown.



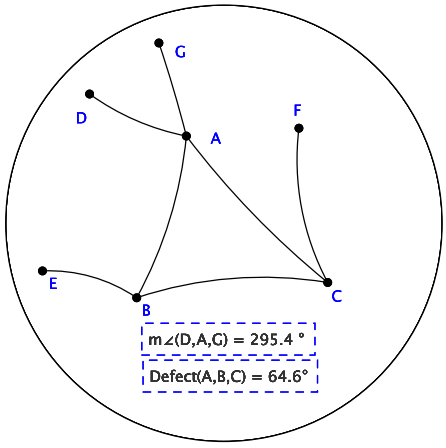
Next, define a new translation vector from point B to point C and translate $\overline{BE}$, resulting in $\overline{CF}$, as shown.



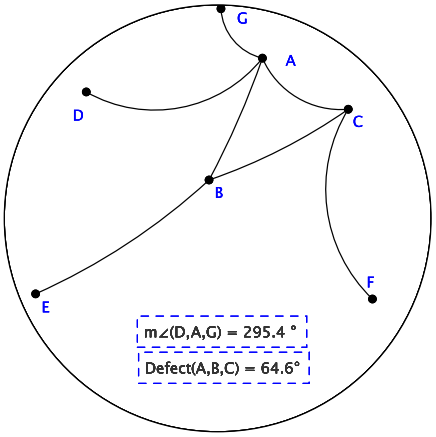
Finally, define a translation vector from C to A and translate $\overline{CF}$, resulting in $\overline{AG}$.



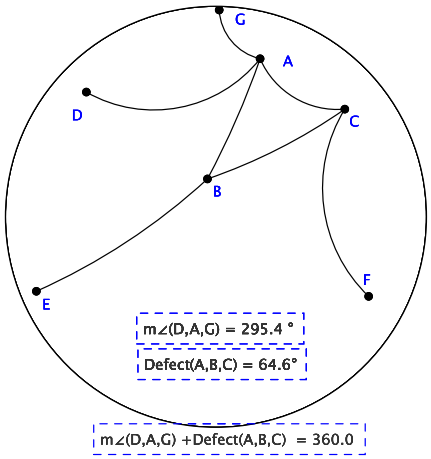
Clearly, the original segment $\overline{AD}$ and the parallel transported segment, $\overline{AG}$, are not in the same direction, as was the case in Euclidean geometry. To see how much the segment has changed, let's measure $\angle DAG$, as this measures the net rotation of $\overline{AD}$ to $\overline{AG}$. As a comparison value to this angle change let's also compute the defect of ΔABC.



The defect and the net angle change appear complimentary. Let's see what happens as we move the triangle around.



The complimentary relationship between the defect and angle change still holds. Let's use the Calculator to add these two measurements together. In the Calculator Window, double-click on the defect measure, hit the "+" key and then double-click on the angle measure. Hit the Evaluate button and then the Add to Canvas button to add this new measure back to the canvas. (If you need help on using the Calculator check Chapter 4.)



The relationship between the defect and the angle change described above is in fact connected with a very important idea in geometry, the notion of *holonomy*.

We define the holonomy of ΔABC as follows. Let $\overline{AD}$ be a segment and let $\overline{AG}$ be the parallel transport of $\overline{AD}$ around the triangle (counterclockwise). Then, the holonomy is the smallest angle measured between these segments.

In general, the holonomy and the defect of a hyperbolic triangle are linearly related, as was seen above. Actually, if we replace the measure of an angle greater than 180 degrees with that measurement minus 360 degrees, we get a negative angle measurement which would would make the holonomy equal to the negative of the defect. It is customary to use this notion of negative angle change to calculate holonomy.

Since the defect is proportional to the area of a triangle, we have that the holonomy and the negative of the area of a triangle are proportional. This fact is a direct consequence of a very deep formula in geometry called the Gauss-Bonnet Formula. This formula relates the area, holonomy, and Gaussian curvature of a region on a surface. For more information on holonomy, curvature, etc, one can consult any textbook on differential geometry. One good introductory work in this area is the text by Millman and Parker [8] . A text that introduces the notion of holonomy very nicely is the book by Henderson [3].

7.8 Möbius Transformations

We end this chapter with a consideration of a general class of transformations called *Möbius transformations*. Möbius transformations include the basic hyperbolic transformations of rotations and translations, as well as other geometric transformations which are compositions of these two types of transformations.

It is an interesting fact of Hyperbolic geometry that two translations, when combined together, do not necessarily make another translation, unlike in Euclidean geometry where the composition of translations is always another translation. This fact was illustrated in the previous section on holonomy and parallel transport.

Thus, the class of Möbius transformations is structured quite a bit differently than the class of affine transformations in Euclidean geometry. (see the Chapter 5 for more information on affine transformations.) Also, to be technically accurate, what we are calling Möbius transformations are really elements of a larger class of transformations in the complex plane. However, in Hyperbolic geometry we restrict these more general transformations to those which preserve the Poincaré disk (the boundary circle in the Poincaré model), since points on the Poincaré disk (points at "infinity") must stay on the disk. It is this restricted set of transformations on the complex plane that we will call Möbius transformations.

In general, a Möbius transformation is an invertible transformation of points $z = x + iy$ in the complex plane having the form

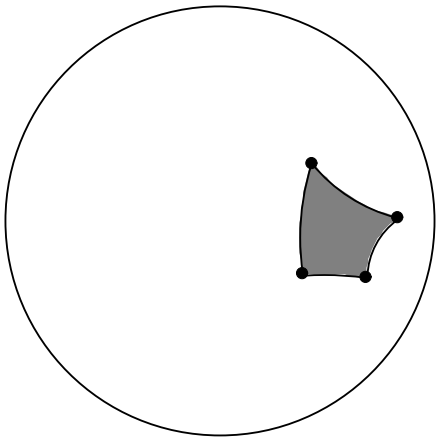
$$z = e^{it}\frac{z - z_0}{1 - \overline{z}_0 z}$$

In this expression, t is a real number and z, z_0 are points in the Poincaré disk represented as complex numbers. A complex number z represents a

point (x, y) in the plane by $z = x+iy$, where $i = \sqrt{-1}$. The bar over the term z_0 in the formula above represents the operation of complex conjugation. If $z = x + iy$ then, $\overline{z} = x - iy$. The term $e^i t$ in the formula can be expanded as $e^i t = \cos t + i \sin t$.

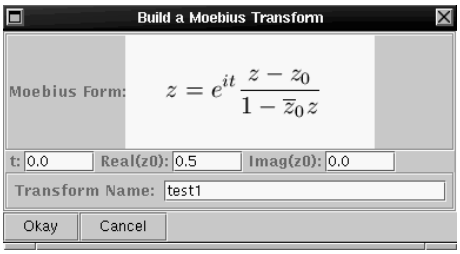
Let's look at an example of a Möbius transformation using *Geometry Explorer*.

In the Poincaré disk construct a quadrilateral to the right of the origin as shown. Then, fill the quadrilateral.



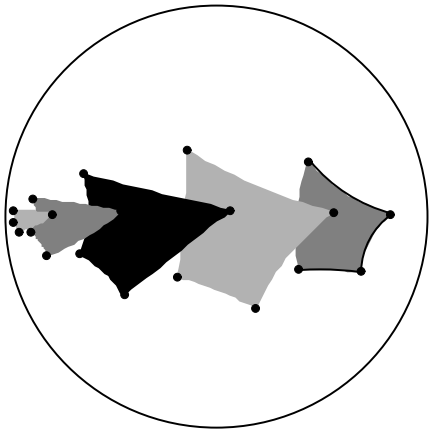
To define a Möbius transformation, we choose **Moebius Transform** from the **Custom** menu in the Transform Panel. A dialog box titled "Build a Moebius Transform" will pop up. We can put in a value for t, and for the x and y components of z_0 in the appropriate text fields in this dialog box. (Note that we can quickly go from one text field to another by hitting the Tab key on the keyboard.)

Put in the following values: $t = 0.0$, Real$(z0) = 0.5$, Imag$(z0) = 0.0$. Then, name the transformation "test1" and hit the Okay button in the dialog box to finish the definition of the Möbius transformation.

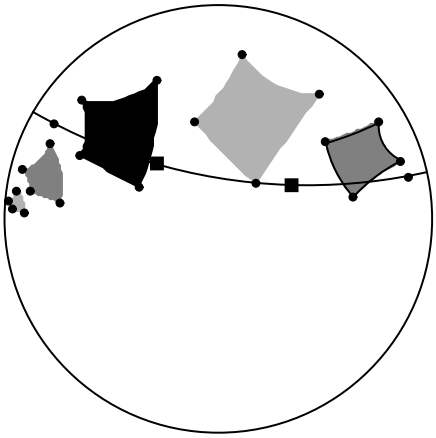


At this point, the new transformation can be used to transform objects in the canvas. To transform the quadrilateral, first select the interior of the quadrilateral by clicking the Select tool somewhere in the filled region. Then, click on the **Custom** menu in the Transform Panel.

You will see the transformation **test1** listed near the bottom of the pull-down menu. Drag down to **test1** and select this menu item. The filled quadrilateral will be transformed as shown at the right. Select **test1** several more times to get a sequence of transformed quadrilaterals.

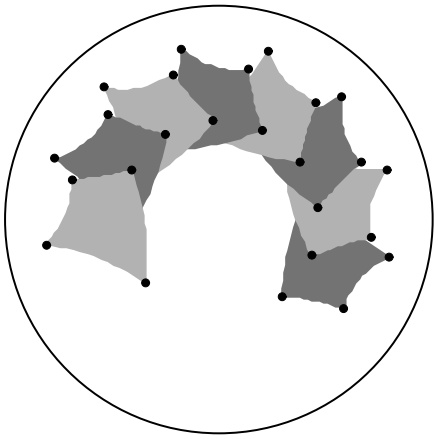


Note that it appears that the transformation is actually translating the filled region along a line in the Poincaré disk. If we would measure the area of each transformed region we would see that it is identical to the original. Generally, Möbius transformations in Hyperbolic geometry preserve areas. If we move the original quadrilateral up a bit, we see that we know longer have a simple translation. A translation would have to follow some hyperbolic line, but lines such as the one shown bend the other way from the motion of the quadrilateral.



The combination of our translation of the quadrilateral upwards using the mouse with the original Möbius translation is not a translation. What is it then? It is actually motion along a special circle called a Steiner circle of the first kind. For more information on such transformations consult the chapters on Möbius Transformations, Steiner Circles, and Hyperbolic Geometry in the text by Henle [4].

Undo the constructions until you are back to the original filled quadrilateral. Now, choose **Moebius Transform** from the **Custom** menu and put in the value of $t = 0.5$ and 0.0 for the other two values. Name this transformation "test2" and hit Okay. Select the filled quadrilateral and select **test2** under the **Custom** menu several times to transform the region under this new Möbius transformation. What is happening now? Clearly, this is a rotation of the quadrilateral around the origin.



Generally, Möbius transformations that have t non-zero and z_0 equal to zero will be rotations about the origin. Möbius transformations that have t zero and z_0 non-zero will translate a point along a line through the origin, if that point lies on the line to begin with.

Chapter 8

Turtle Geometry

> My conjecture is that the computer can concretize and personalize the formal. Seen in this light, it is not just another powerful educational tool.... Knowledge that was accessible only via formal processes can now be approached concretely.
>
> —Seymour Papert

> It's ironic that fractals, many of which were invented as examples of pathological behavior, turn out not to be pathological at all. In fact they are the rule in the universe. Shapes which are not fractal are the exception. I love Euclidean geometry, but it is quite clear that it does not give a reasonable presentation of the world. Mountains are not cones, clouds are not spheres, trees are not cylinders, neither does lightning travel in a straight line. Almost everything around us is non-Euclidean.
>
> —Benoit Mandelbrot

In turtle geometry we create geometric figures by directing an imaginary turtle to move and draw on a planar surface. The turtle understands a few simple directions such as move forward, draw forward, and turn clockwise or counter-clockwise. For example, to instruct a turtle to draw a square we would have it draw forward one unit, turn left 90 degrees, draw forward one unit, turn left 90 degrees, draw forward one unit, turn left 90 degrees, and finally, draw forward one unit.

Turtle geometry was originally created in conjunction with the development of the LOGO programming language. The turtle was a robot that moved on the floor. The robot was able to understand two basic commands:

move forward and turn right. The robot was also equipped with a pen that could be raised and lowered, thereby allowing the turtle to draw as it moved, if it was so instructed.

The classic work in the area of turtle geometry and LOGO is Seymour Papert's text *Mind Storms*. [9] This book has inspired thousands of teachers and children to use LOGO and turtle geometry in the classroom to explore computer programming and geometry in a way that is very accessible to young (and old) students. From Papert's earlier association with child psychologist Jean Piaget, he explains in *Mindstorms* how turtle geometry can be used to help students build on the concrete associations and patterns of a turtle moving about so that more formal structures involving abstract geometric ideas can be created in the student mind. For example the abstract notion of angle can be directly tied to the concrete motion of turning the turtle.

Turtle geometry has proved to be an ideal geometry in which to explore the fractal structure of both mathematical and natural objects. Fractals are geometric figures which are complex in nature at all scales. A circle when highly magnified looks nearly like a straight line (think of how flat the surface of the earth looks to us) A fractal, on the other hand, never flattens out in the way that a circle or sphere does. It looks equally bumpy at all scales of magnification. Fractals are infinitely complex and this makes their construction difficult. However, it is often possible to construct a blueprint for the complexity of a fractal. The blueprint can be sent to a turtle as a series of directions, thus providing for an *approximation* of a fractal shape.

In this chapter we look at how *Geometry Explorer* implements turtle geometry and how this geometry is used to model fractal objects.

8.1 Basic Turtle Geometry in *Geometry Explorer*

In a computer environment we envision a small turtle on the screen that we direct from the keyboard or mouse. Since the basic motions of a turtle are *forward* and *turn*, then to define a turtle we must specify how far it will move forward and how far it will turn. A simple way to define these two values is by specifying a distance and angle in the *Geometry Explorer* Canvas.

The items under the **Turtle** menu in the main window are used to define a turtle on the Canvas and also to control turtle movements. There are five items under the **Turtle** menu:

1. **Turtle Heading Vector:** This menu item will be activated once two points are selected in the Canvas. The direction determined by

the vector between these points will determine the direction the turtle moves forward and the distance between these points will determine how far the turtle moves in that direction. Once we select two points and click on this menu item, the vector defined by the two points will be stored for use in creating a turtle.

2. **Turtle Turn Angle:** This menu item will be activated once three points are selected in the Canvas. These points will be the initial, vertex, and terminal points of an angle. This angle will determine how the turtle turns when directed to do so. Once we select three points and click on this menu item, the angle defined by the three points will be stored for use in creating a turtle.

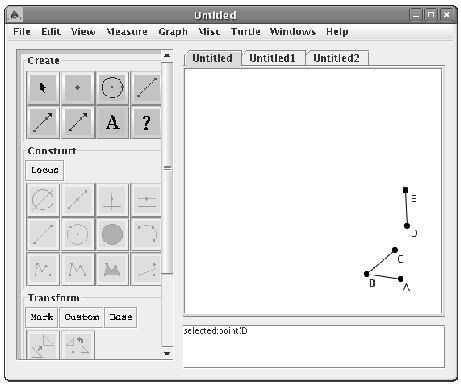
3. **Create Turtle At Point:** This menu item will be activated once a vector and angle have been defined (see the previous two items) and a point on the Canvas has been selected. This point will be the point at which the turtle will be located. Once we click on this menu item a turtle will be created in the Canvas at the position given by the point and a Turtle Controller Panel will pop up.

4. **Control Panel...:** This menu item will be activated once a turtle has been created, or when a turtle has been selected in the Canvas. After clicking this menu item a Turtle Controller Panel will pop up. This panel contains tools for controlling the movement of the turtle.

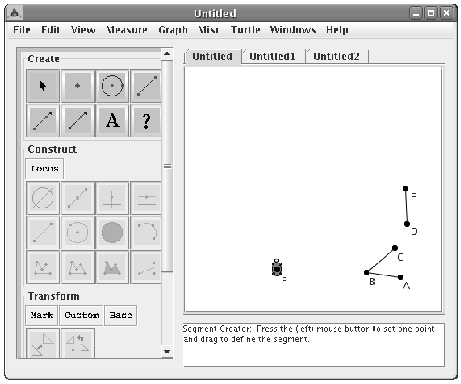
5. **Create Simple Turtle** This menu item is always active. After clicking on this item a turtle will be created in the center of the Canvas. It will have a default turning angle of 90 degrees and will move one unit in the Canvas coordinate system for each command to draw or move forward.

Let's look at an example of using the turtle in *Geometry Explorer*:

In the figure at the right, $\angle ABC$ and vector $\vec{DE}$ will determine the required angle and vector for a turtle. Select A, B, and C and choose **Turtle Turn Angle** from the **Turtle** menu to define the turtle angle. Then, select D and E and choose **Turtle Heading Vector** from the **Turtle** menu.

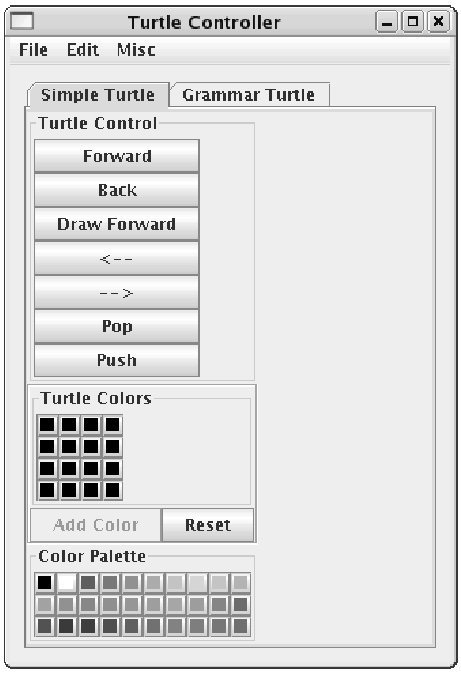


Next, we create a point F that will be the starting position of our turtle. Select F and choose **Create Turtle At Point** from the **Turtle** menu. A turtle will be created at F as shown.



The turtle is graphically displayed as a little green object. It looks a bit small in the previous figure. Here is a bigger version.

Once the turtle is created another window automatically pops up. This is the Turtle Controller Panel. This window will control the movement of our turtle. The Controller has two tabbed panels which can be used to control the turtle. The first panel which is visible at right is labeled "Simple Turtle". The hidden tabbed panel is labeled "Grammar Turtle". We will look at the first panel in this section.



The Simple Turtle panel consists of three areas labeled "Turtle Control", "Turtle Colors", and "Color Palette". In the Turtle Control section there are seven buttons: "Forward", "Back", "Draw Forward", "$<--$", "$-->$", "Pop", and "Push".

The function of each of these buttons is as follows:

1. **Forward:** Clicking this button will move the turtle forward a distance equal to the length of the vector defined when the turtle was created. The direction the turtle moves is the direction this vector points. No drawing will occur as the turtle moves.

2. **Back:** Clicking this button will move the turtle backwards a distance equal to the length of the vector defined when the turtle was created. No drawing will occur as the turtle moves.

3. **Draw Forward:** Clicking this button will move the turtle forward just as the Forward button would. However, as the turtle moves it draws a segment.

4. $<--$ **:** Clicking this button will rotate the turtle to the left (counterclockwise) through an angle equal to the angle defined by the three points used when the turtle was created.

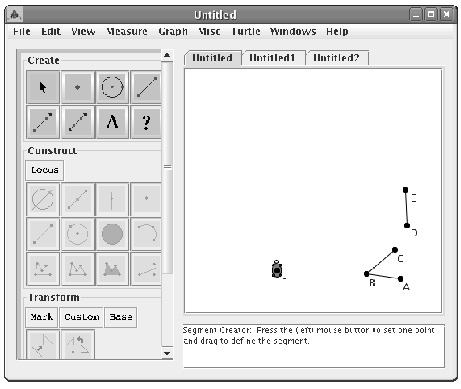
5. $-->$ **:** Clicking this button will rotate the turtle to the right (clockwise) through an angle equal to the angle defined by the three points used when the turtle was created.

6. **Pop:** Clicking this button will cause *Geometry Explorer* to restore any stored values for the heading and position of the turtle.

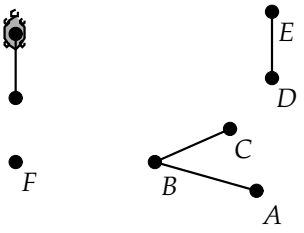
7. **Push:** Clicking this button will cause *Geometry Explorer* to store the current heading and position of the turtle.

Let's return to our example:

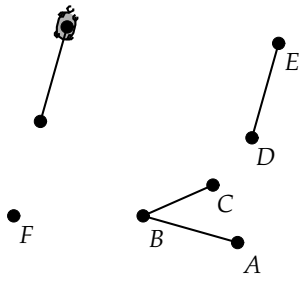
Here is where we left our little turtle.



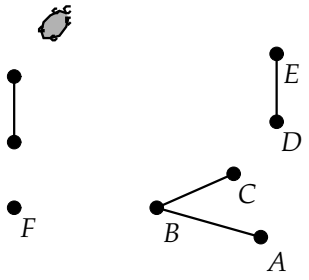
Now let's hit the Forward button and then the Draw Forward button.



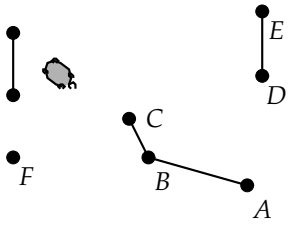
By dragging point E we see how the vector $\vec{DE}$ determines the direction and length of turtle movements.



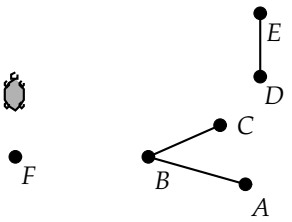
Lets put E back to its original position. (Use the **Undo** sub-menu from the **Edit** menu.) Then, hit the Turn Right button ("$--\ >$") followed by the Forward button.



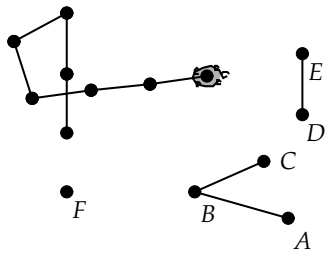
If we now move point C, we will alter the angle by which the turtle turns.



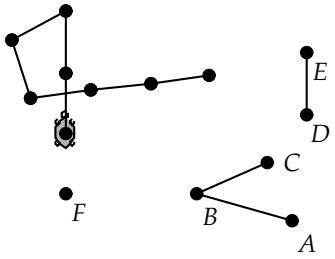
Let's put the turtle back to its original position by choosing **Undo** from the **Edit** menu enough times to undo all the turtle movements. Then, Move Forward once.



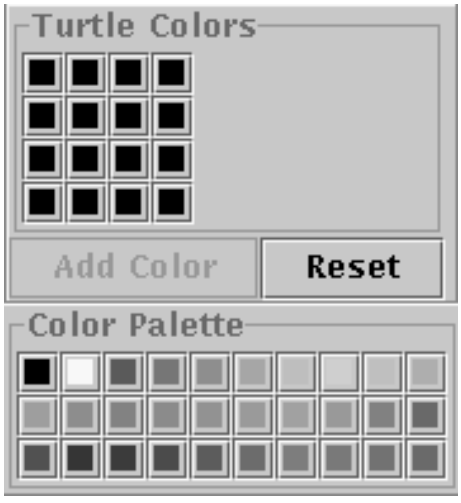
Click the Push button to store this state of the turtle. Now, carry out a series of turtle actions. In the figure at right the turtle has been directed to do a number of draws and rotates starting from the position in the previous figure.



At this point suppose that we wished to go back to the position where we did a Push. All we have to do is click the Pop button to restore the turtle to this position. Note how the turtle's position and heading direction are exactly those that we stored when we first clicked the Push button.



Suppose that we wished to change the turtle drawing color. This is done via the Turtle Colors section of the Simple Turtle Panel. For now, ignore the buttons labeled "Add Color" and "Reset". To change the turtle drawing color, just click on one of the colors in the Color Palette section and the color change will take effect the next time the turtle draws.



8.2 Turtles, Fractals, and Grammar Re-writing

As Mandelbrot states in the quote at the beginning of this chapter, fractals are not ordinary Euclidean objects like lines and circles. Broadly speaking, the main difference between a fractal and a Euclidean object like a circle is that the circle is *smooth*. By smooth we mean that as one magnifies smaller and smaller sections of the circle, the curviness of the circle flattens out. In small regions smooth curves look like line segments.

Fractals, on the other hand, *never* flatten out in this way. At any level of magnification they are jaggy and bent. Most objects in the natural world are fractal-like because of this property. For example no matter how close one gets to a cloud, it always looks fuzzy and vaporous. It never looks like the surface of a sphere.

Thus, to study natural geometry one must look at the geometry underlying fractals. However, there is one problem. Since fractals are complex at every level of magnification, how do we specify such shapes in a formula? Lines can be specified by two points because the line's shape is entirely uniform. This is not true of a fractal.

To specify a fractal we need a *procedure* that can encapsulate the complexity of the fractal at every level of magnification. There are several ways to do this. One way is by a *recursive* procedure that defines the shape of a fractal in terms of the shapes of its sub-parts. In Chapter 10 there is an example of creating the Koch curve that is recursive in nature (uses looping in the Recorder window).

Another way to represent the levels of a fractal is by the use of a *grammar* in which symbols represent geometric operations like movement, drawing, etc. In this grammar a set of symbols will define the overall shape of the fractal and there will be rules for how the shape of the fractal changes as we magnify sections. These rules will be called *production rules*, as they tell us how to produce the fractal.

We will use the notion of a grammar to construct *approximations* of fractals. We can only construct approximations since a fractal is by its nature an infinitely complex object, at all levels, and a finite computer cannot create an infinitely complex object.

We will need a set of symbols to control our turtle. These symbols will be the following:

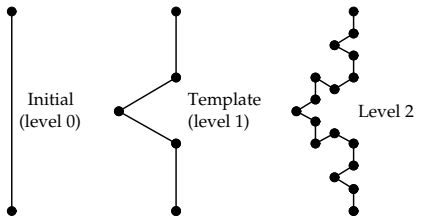
1. **f** Forward
2. | Back

3. **F** Draw Forward
4. + Rotate Left
5. − Rotate Right
6.] Pop
7. [Push

This set of symbols is used in the classic work by Prusinkiewicz and Lindenmayer on grammar-based fractal systems in nature titled *The Algorithmic Beauty of Plants* [7]. This beautiful book describes how one can use grammars to model plants and plant growth. This method of describing natural fractal shapes has been called *Lindenmayer Systems*, or *L-Systems*.

To illustrate how to use this grammar-based fractal description system, let us look at the Koch curve example.

The Koch curve is a fractal that is constructed as follows: begin with an initial segment. Replace that segment with a template curve made up of four segments as shown at the right.



The angles inside the peak are all 60 degrees, making the triangle formed by the peak an equilateral triangle. The initial segment will be called the Koch curve at level 0. The template will be the Koch curve at level 1. If we replace each of the segments in the template with a copy of the template at a reduced scale we get the third curve—the Koch curve at level 2.

To model the Koch curve using our set of symbols, we could say that the initial segment is basically a turtle Draw Forward, or an F. The template is (forgetting for now the problem of scaling) a draw forward followed by a turn left of 60 degrees, then a draw forward followed by two turn rights of 60 degrees, then a draw forward followed by a turn left of 60 degrees, and finally another draw forward. Thus, the sentence that describes the template is: $F + F - -F + F$. (Assuming our turns are always 60 degrees). Study the template curve and convince yourself that this is the correct sentence for the curve.

Thus, the Koch curve is *grammatically* defined by an initial sentence F, which we will call the *axiom* and a template sentence $F + F - -F + F$

which governs how the initial segment is replaced. This template sentence we call a *production rule*. At level 0, the Koch curve is just F. At level 1 we replace F by $F+F--F+F$. At level 2 we need to replace all segments by the template again. Well, this is equivalent to replacing all occurrences of the axiom F in the level 1 sentence $F+F--F+F$ by the production rule sentence $F+F--F+F$. If we do this we get a level 2 sentence of $F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F$. (Convince yourself this is right).

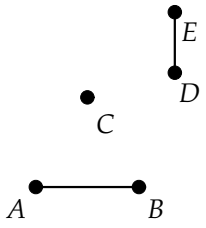
In other words each succeeding level of the Koch curve is represented by *re-writing* the sentence for the current level by using the production rule. The sentence for level 3 would be $F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F+F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F--F+F--F+F$.

Thus, the Koch curve can be completely defined by just two sentences: the starter axiom of F and the production rule of $F+F--F+F$. The Koch curve is then the curve you get by carrying out the re-writing of the axiom to an *infinite* level, having the turtle do the geometry.

A major difference between this grammar re-writing system of describing the Koch curve, and the recursive description given in Chapter 10, is that in the grammar-based system we do not re-scale the template, whereas in the recording we shrink everything by a factor of $\frac{1}{3}$ before looping.

Let's look at how all of this works in practice. First, we create a turtle to interpret sentence symbols. We start by defining a 60 degree angle. Construct segment $\overline{AB}$. Select A and set it as a center of rotation by choosing **Center** under the **Mark** menu in the Transform Panel. Then, click on **Rotation** under the **Custom** menu in the Transform Panel and type in 60 for the angle and hit the Okay button. Then, select point B and click the Rotation tool in the Transform Panel.

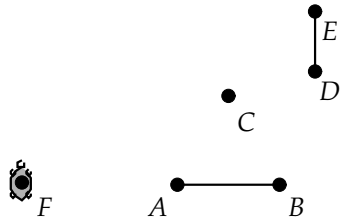
We now have a 60 degree angle $\angle BAC$, as shown at the right. Next, create segment $\overline{DE}$ to be used as the turtle heading vector.



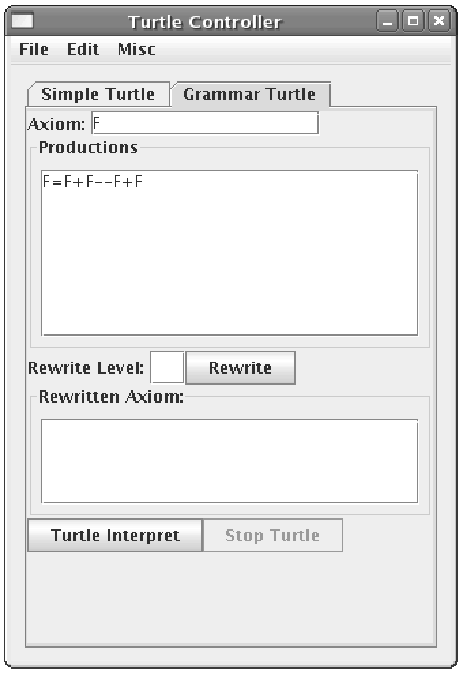
To store the angle we select B, A, and C (in that order) and choose **Turtle Turn Angle** from the **Turtle** menu in the main window. Then

select D and E (in that order) and choose **Turtle Heading Vector** from the **Turtle** menu.

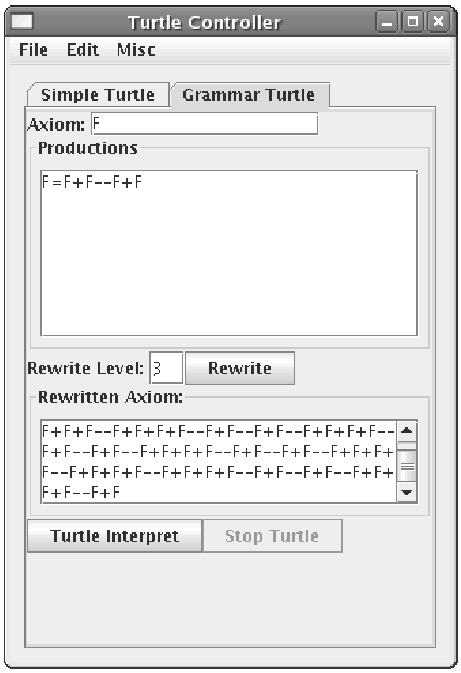
Create point F as shown and select it. At this point the **Create Turtle At Point** item under the **Turtle** menu will be active. Choose this item to create a turtle at point F.



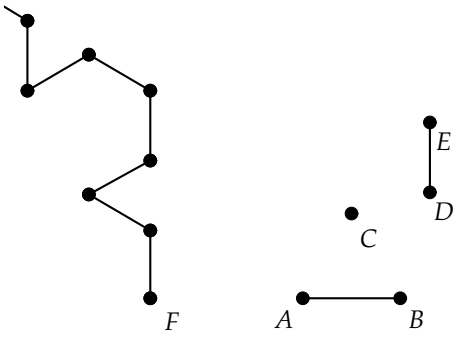
The Turtle Controller window will pop up whenever we create a turtle. We will make use of the tabbed panel labeled "Grammar Turtle". Click on that tab. In the box labeled "Axiom:" type F for our axiom. In the area labeled "Productions:" type in the production rule in the form $F = F + F - -F + F$. We use the equal sign to designate that the symbols on the left side of "=" will get replaced by the symbols on the right side.



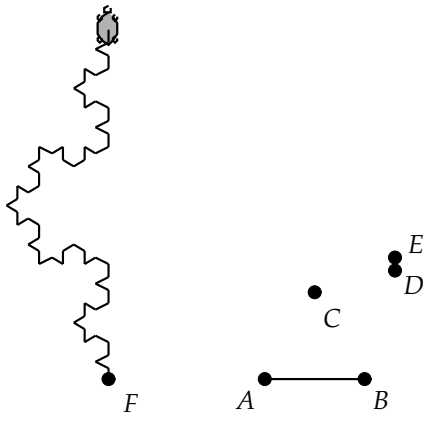
In the box labeled "Rewrite Level:" we can now put in the level of re-writing we desire. Once we have specified this and hit return, the re-writing will take place. Note that nothing will happen with the turtle however. In the example shown we have re-written the sentence to level 3. Note that the Rewritten Axiom text box now has too much text to display. To see the rest of the new sentence, just scroll down in this text box.



Now we are ready to have the turtle carry out the drawing for us. If we hit the Turtle Interpret button, whatever is in the Rewritten Axiom text box will be interpreted and drawn to the Canvas by the turtle. However, the turtle will take some time to carry out the level 3 sentence (anywhere from a couple of seconds to several minutes, depending on the speed of your computer). You might notice that the turtle leaves the Canvas, as shown in the figure at the right.



To bring the turtle back in view we just need to move point E closer to D, thus shrinking the turtle's move forward length.We have hidden most of the points in the figure at right for a better view of the turtle's movements.



At any point, as the turtle carries out the interpretation of the sentence, we can stop the turtle by just hitting the Stop Turtle button.

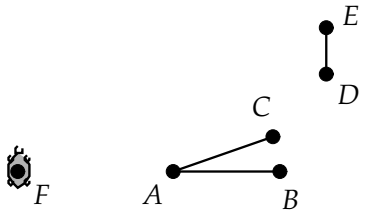
As one can imagine, this grammar re-writing system can be very powerful in representing complex fractal shapes.

8.3 Plant Grammar

What makes many plants fractal-like is their branching structure. A branch of a tree often looks somewhat like the entire tree itself, and a branch's sub-branches look like the branch itself, etc. To model branching we will need to use the push and pop features of our turtle. This is necessary to efficiently carry out the geometry of a branch and then return to where the branch was attached.

The analysis of complex branching patterns is quite an interesting subject, but one that we will not go into in any detail. For more information, see the Prusinkiewicz text [7]. To illustrate how to use turtle geometry to create a branching pattern we will look at the following example.

Create an angle $\angle ABC$ and a segment $\overline{DE}$ on the Canvas. Then, create point F and create a turtle at F. (Review the two sections above if you need help in doing this.)

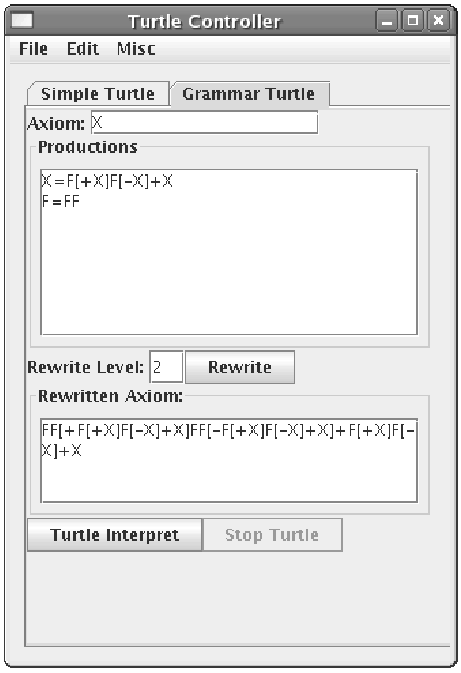


The grammar we will use to model a branching pattern consists of the turtle symbols described in the last section plus one new symbol X. We can

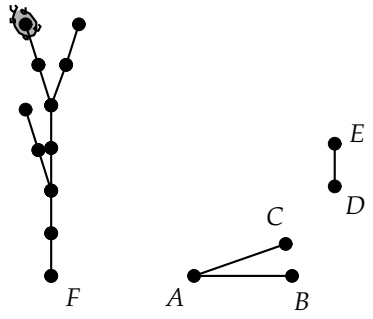
think of X as being a *virtual* node of a plant that we are creating. Initially, we start with the axiom being just X, signifying the potential growth of the plant. Then, we replace X by a set of branches and new nodes using the production rule $X = F[+X]F[-X] + X$. We also put in a production for growing branches longer in succeeding levels. This is the production $F = FF$.

At level 0 the sentence will be the axiom X. At level 1 we will have $F[+X]F[-X] + X$. At level 2 we will have $FF[+F[+X]F[-X] + X]FF[-F[+X]F[-X] + X] + F[+X]F[-X] + X$.

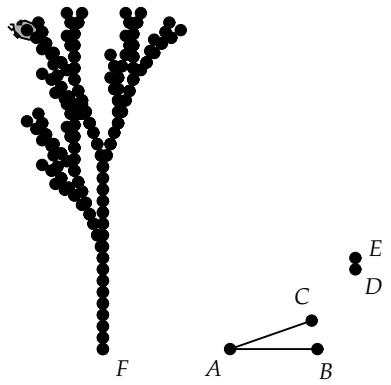
Go to the Turtle Controller window and type in the axiom and two production rules in the Grammar Turtle section of the dialog window, as shown at right. Then, create the new sentence at level 2 by typing in "2" in the Rewrite Level box and hitting return.



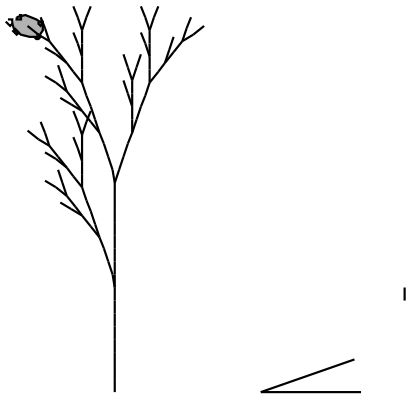
Now, hit the Turtle Interpret button. The turtle will draw the shape shown at the right. The figure is clearly a branched structure, but is not really much like a plant. To more fully develop the branching pattern we need to re-write the axiom to a higher level.



Undo all of the turtle actions by choosing **Undo to Start** from the **Misc** menu in the Turtle Controller Window). Move E close to D, so that the turtle will move only a short distance each time it moves. Change the rewrite level to 4 in the Turtle Controller and hit Return. Click the Turtle Interpret button and wait for the turtle to interpret the re-written sentence.



You will be able to tell when the turtle is done by the state of the Stop Turtle button. If the turtle is still drawing the button will be active. Once the turtle completes drawing the button will become inactive. The image is very blotchy with all of the points visible that were drawn by the turtle. Let's hide these points by choosing **Hide All** and **Points** from the **View** menu. Now, the figure (shown at right) looks like the bushy branch structure of a plant (minus the leaves).



For other branch patterns consult the text by Prusinkiewicz [7].

8.4 Color Index Tables

As described in the first section of this chapter, one can change the drawing color of the turtle by clicking on one of the colors in the Color Palette area of the Simple Turtle panel of the Turtle Controller.

Additionally, one can change colors grammatically. This is done by specifying a table of colors and then accessing the table by the use of the symbols ‘ and ’. The symbols function as follows:

1. ‘ (forward quote) Move one position forward in the color table. If the

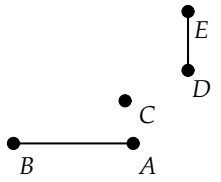
next index is beyond the end of the table, cycle around and start at the beginning of the color table.

2. ' (back quote) Move one position back in the color table. If we go past the first color in the table, cycle to the end of the table.

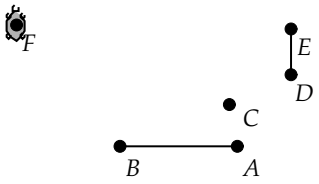
As an example let's look at drawing a multicolored regular 17-gon. The construction of the 17-gon by ruler and compass was a problem that was tackled by one of the greatest mathematicians of all time, Carl Friedrich Gauss. Before Gauss's time, it was known how to construct regular polygons with a prime number of sides when the number of sides was 3 and 5, but no other results were known for polygons with a prime number of sides. At the age of nineteen Gauss discovered a Euclidean construction for the regular seventeen-sided polygon. In fact, Gauss was so proud of this accomplishment that he requested a regular 17-gon to be carved on his tombstone. However, the stonemason who eventually carved the stone refused to carve the figure, stating that it would be indistinguishable from a circle. Eventually, a monument in Brunswick, Germany, where Gauss grew up, was created with the 17-gon carved into the surface.

The angle needed to construct the regular 17-gon is that of 360/17 degrees, or approximately 21.176471.

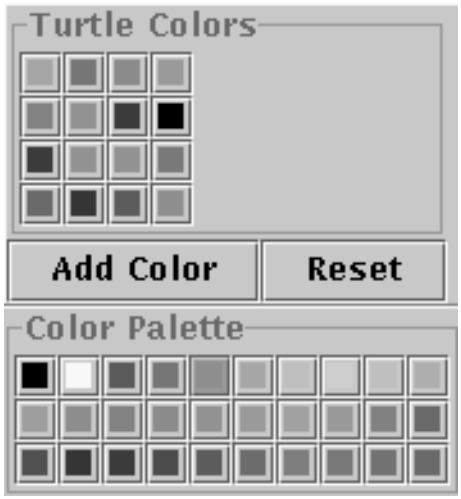
We begin by creating segment $\overline{AB}$. Then, select B as a center of rotation (use the **Mark** menu in the Transform Panel) and create a custom rotation of 21.176471 degrees (use the **Custom** menu). Next, select A and click the Rotate tool in the Transform Panel to get the desired angle. Also, create $\overline{DE}$ to use as the turtle heading vector.



Create point F and set $\angle ABC$ as the turtle turn angle and points D and E as the turtle heading vector. Select point F and create a turtle at F. (Review the first section of this chapter, if needed, for help on defining and creating a turtle.)

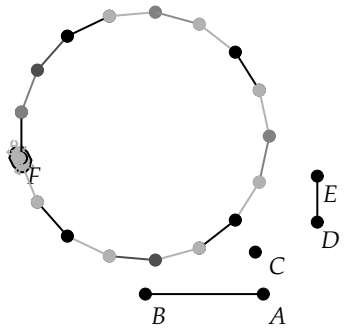


To define a table of colors we use the Color Palette in conjunction with the Add Color button in the Turtle Controller Window. Click on one of the colors and then on the Add Color button. The color will appear in the square table above the Add Color button labeled "Turtle Colors." Click on another color and add it to the table. Fill out the rest of the table as you wish. Note that the table has a maximum of 16 colors.



This table will serve as an indexed color table for the ‘ and ’ symbols as described above. To draw a 17-gon we can use a sequence of 17 draw and turn commands as described by the sentence $F-F-F-F-F-F-F-F-F-F-F-F-F-F-F-F-F$. We will also change color at each step by inserting a ‘ after each F, yielding $F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F$. The turtle will then move forward one index in the color table each time it reads the ‘ symbol.

Type in $F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F-\text{‘}F$ in the Axiom box in the Turtle Controller, Type in 0 in the Rewrite Level box and hit Rewrite. Then, click the Turtle Interpret button to get a multicolored 17-gon. You may have to move F or shrink $\overline{DE}$ to keep the polygon in view.



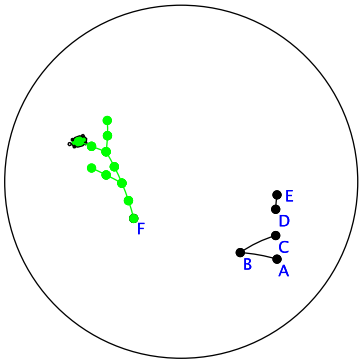
8.5 Saving, Opening, Printing Grammars

To save a defined axiom and set of production rules, just choose **Save** from the **File** menu in the Turtle Controller window. To open an already saved grammar definition, just choose **Open**. If one chooses **Print** then the axiom and productions will be printed out.

8.6 Turtle Geometry in Non-Euclidean Environments

Turtle Geometry is also available in Hyperbolic and Elliptic Geometry. The creation of a turtle follows the same rules as for the creation of a Euclidean turtle – we first define a turn angle and heading vector and the create a turtle at a point.

For example, here is a turtle defined at point F with turn angle given by A, B, and C and heading vector given by D and E. The turtle has carried out the level 2 re-write of the plant grammar example from earlier in this chapter.



Chapter 9

Tessellations

> A long time ago, I chanced upon this domain [of regular division of the plane] in one of my wanderings; I saw a high wall and as I had a premonition of an enigma, something that might be hidden behind the wall, I climbed over with some difficulty. However, on the other side I landed in a wilderness and had to cut my way through with great effort until - by a circuitous route - I came to the open gate, the open gate of mathematics.
>
> —Maurits Cornelis (M. C.) Escher (1898–1972)

Much of the renewed interest in geometric design and analysis in the modern era can be traced to the artistic creations of M. C. Escher. While he did not prove new theorems in geometry, he did use geometric insights to create fascinating periodic designs like the design in Fig. 9.1.

A beautiful book that describes Escher's artwork, and the mathematics behind the art, is *M. C. Escher Visions of Symmetry* [10] by Doris Schattschneider.

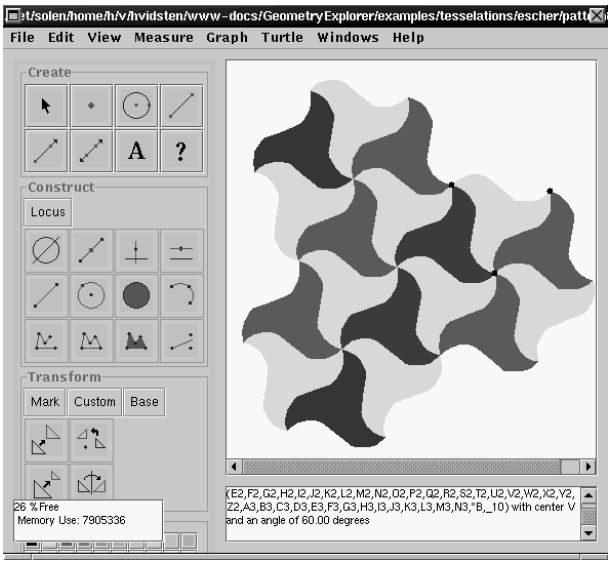


Fig. 9.1 A Tiling in the Spirit of Escher

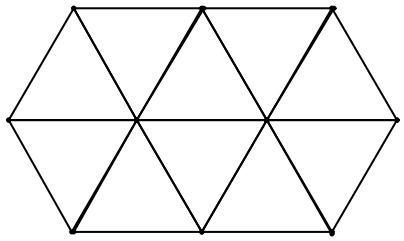
One of Escher's favorite themes was that of a *tessellation* of the plane by geometric shapes. A tessellation (or tiling) is a covering of the plane by repeated copies of a shape such that there are no gaps left uncovered and the copied shapes never overlap. In Fig. 9.1 we see the beginnings of a tessellation of the plane by a three-sided shape.

9.1 Regular Tessellations of the Plane

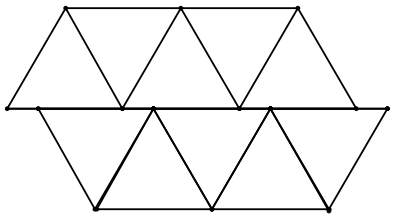
The simplest tessellations of the plane are those which are built from a single tile in the shape of a regular polygon. A regular polygon has the property that all sides have the same length and all interior angles created by adjacent sides are congruent. We will call a regular polygon with n sides a *regular n-gon*.

A regular 3-gon will be an equilateral triangle, a regular 4-gon will be a square, etc. We can tile the plane with regular 3-gons in more than one way.

In the figure at the right we see a tiling by equilateral triangles in which all triangles meet at common vertexes.



In the new tiling at the right we have shifted the top row of triangles a bit. This configuration will still lead to a tiling of the plane, although all triangles no longer share common vertexes.

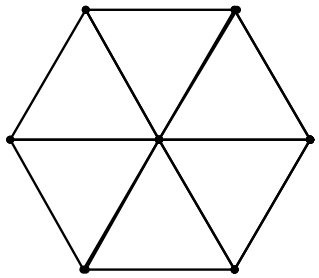


We will call a tessellation *regular* if it is made from copies of a single regular n-gon with all n-gons meeting at common vertexes. Thus, the second triangular tiling above is not a regular tiling.

How many regular tilings are there? Clearly, the example above shows that there is a regular tiling with triangles. It is also clear that we can tile with squares. The triangle tiling above shows that regular hexagonal tilings are possible. In fact, these three are the only possible regular tilings.

It is not hard to see why this is the case. At a common vertex of a regular tiling suppose that there are k regular n-gons meeting at the vertex. Then, an angle of 360 degrees will be split into k parts by the edges coming out of this vertex. Thus, the interior angles of the n-gon must be $\frac{360}{k}$. On the other hand, suppose we take a regular n-gon, find its central point and draw edges from this point to the vertexes of the n-gon.

In the example at the right we have done for this for the regular hexagon. The point A is the central point of the hexagon.



For a regular n-gon, this triangulation will yield n isosceles triangles, each with angle sum of 180 degrees. Thus, if we add up the sum of the angles in all of the triangles in the figure, we would get a total angle sum of $180n$.

On the other hand, if we add up only those triangle angles defined at the central point then the sum of these will have to be 360. For each of our isosceles triangles the other two angles at the base of the triangle will be congruent. (the angles at B and C in the hexagon example) Let's call these angles α. Then, equating the total triangle sum of $180n$ with the sum of the angles at the center and the sum of the base angles of each isosceles triangle we get

$$180n = 360 + 2n\alpha$$

and thus,

$$2\alpha = 180 - \frac{360}{n}$$

Now, 2α is also the interior angle of each n-gon meeting at a vertex of a regular tessellation. We know that this interior angle must be $\frac{360}{k}$, for k n-gons meeting at a vertex of the tessellation. Thus, we have that

$$\frac{360}{k} = 180 - \frac{360}{n}$$

If we divide both sides by 180 and multiply by nk we get

$$nk - 2k - 2n = 0$$

If we add 4 to both sides we can factor this as

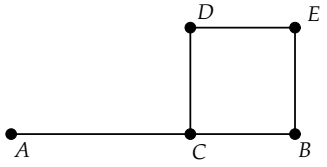
$$(n-2)(k-2) = 4$$

There are only three integer possibilities for n and k, namely 6, 4, and 3. These three possibilities directly correspond to the three regular tessellations with equilateral triangles, squares, and regular hexagons.

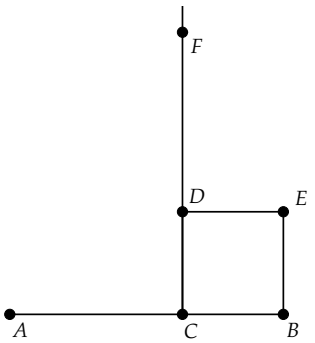
9.2 A Tessellation Construction

Escher received inspiration for his work in tiling from a visit to the Alhambra, a 14th century palace in Grenada, Spain. In this Moorish palace he found almost every wall, floor, and ceiling surface covered with abstract geometric tilings. We will look at one of these tilings now—a tiling in the shape of a dart.

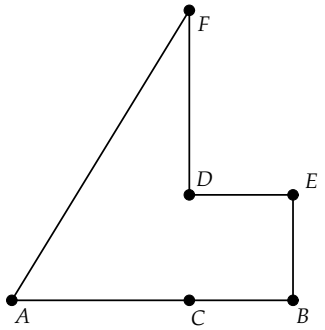
Create segment $\overline{AB}$ attach a point (C) to this segment. Next, construct a square on $\overline{CB}$. To do this first mark point C as a center of rotation (use the **Mark** menu in the Transform Panel). Then, define a custom rotation of 90 degrees (use the **Custom** menu). Select B and click the Rotate tool in the Transform Panel, producing point D. Likewise, rotate C 270 degrees about point B, producing point E. Then, connect segments as shown.



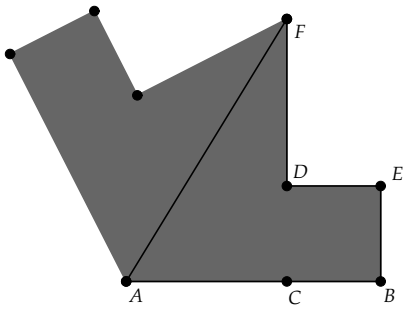
To make the point of our dart, draw a ray from C vertically through D and attach a point F to this ray.



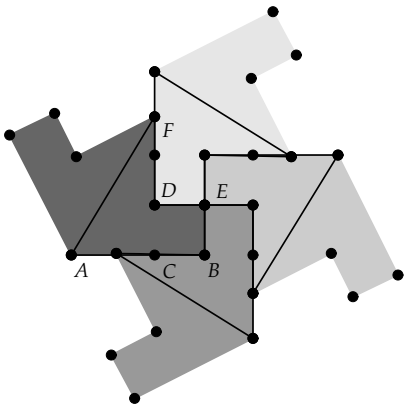
Next, hide the ray, hide $\overline{CD}$, and connect segments as shown.



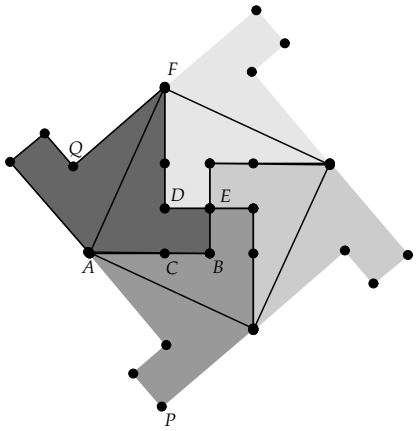
We have now made half of our dart. Select points A, B, E, D, and F and click the Filled Polygon tool in the Construct Panel to color our half-dart. Select $\overline{AF}$ and choose **Mirror** from the **Mark** menu in the Transform Panel. Select the half-dart by clicking inside the filled area and click the Reflect tool to reflect it across $\overline{AF}$ to get the entire dart.



Select E and set it as a center of rotation. Define a custom rotation of 90 degrees. Select the entire dart by clicking and dragging with the Selection tool to enclose the figure in a selection box.Then, rotate the dart three times (click the Rotate tool three times). In the figure at the right we have changed the color of each component dart so that we can see the pieces better. Also, we have rescaled the Canvas as the image grew too large. (To rescale the Canvas, choose **Rescale Geometry in Canvas** (**View** menu).



To make this tiling (no gaps) we move F (the "point" of the dart) to a position where it directly matches the base of the dart to its right, as shown in the figure.



We have now constructed a basic "tile" that can be translated to completely cover the plane. The points labeled "P" and "Q" will be used to define a translation vector for our 4-dart region.

To translate this basic tile we select P and Q and set these as a rectangular vector of translation (choose **Vector** from the **Mark** pop-up menu). Then we select the whole 4-dart region and click the Translate tool in the Transform Panel. Fig. 9.2 shows this two-tile configuration. It is clear that the 4-dart region will tile the plane if we continue to translate it vertically and horizontally.

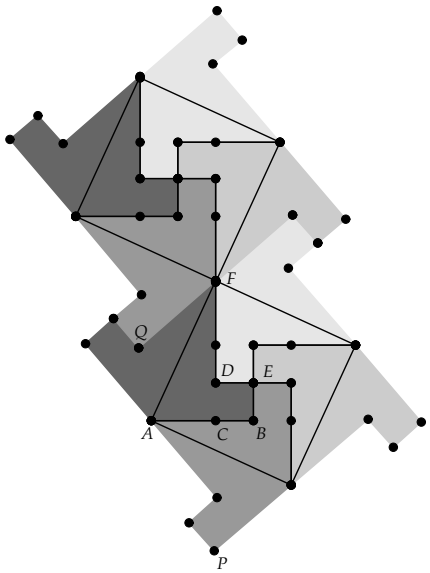


Fig. 9.2

9.3 Hyperbolic Tessellations

In Euclidean geometry there are just three different regular tessellations of the Euclidean plane—the ones generated by equilateral triangles, squares, and regular hexagons. We can look at the same question in hyperbolic geometry, namely how many regular tilings are there?

We can argue in a similar fashion to what we did in the Euclidean case above. If we have k hyperbolic regular n-gons meeting at a common vertex of a tiling, then, we would have that the interior angles would again be $\frac{360}{k}$. Also, we could again find a central point and triangulate each n-gon. The angles around this center point will still sum up to 360. However, the total triangle sum will now be *less* than $180n$. Thus, our equation would now look like

$$180n > 360 + 2n\alpha$$

and thus,

$$2\alpha < 180 - \frac{360}{n}$$

and since the interior angle of the n-gon must be 2α we get

$$\frac{360}{k} < 180 - \frac{360}{n}$$

Dividing both sides by 360 and by nk we get

$$\frac{1}{n} + \frac{1}{k} < \frac{1}{2}$$

We see that if there is a regular tessellation by n-gons meeting k at a vertex, then $\frac{1}{n} + \frac{1}{k} < \frac{1}{2}$. On the other hand, if this inequality is true then a tiling with n-gons meeting k at a vertex must exist. Thus, this inequality completely characterizes regular hyperbolic tilings. We will call a regular hyperbolic tessellation of n-gons meeting k at a vertex a *(n,k)* tiling.

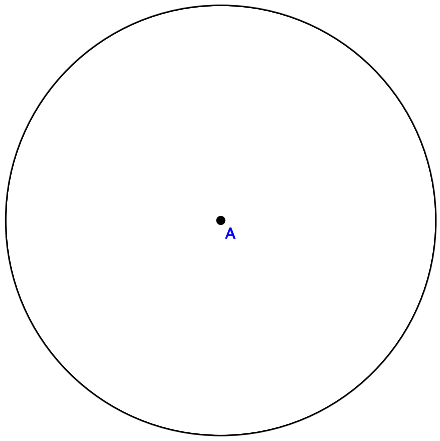
As an example, let's see how to generate a $(5, 4)$ tiling. In a $(5, 4)$ tiling we have regular pentagons meeting four at a vertex. How do we construct regular pentagons of this kind? First, it is clear that the interior angle of the pentagon must be 90 degrees ($\frac{360}{4}$). If we take such a pentagon and triangulate it with triangles to a central point, as we did above, we see that the angles about the central point are all 72 degrees and the base angles of the isosceles triangles are 45 degrees (half the interior angle). Thus, to build the pentagon we need to construct a hyperbolic triangle with angles of 72, 45, and 45.

In Euclidean geometry there are an infinite number of triangles that have a specified set of three angles, and these triangles are all similar to each other. In hyperbolic geometry, two triangles with congruent pairs of angles must be congruent themselves!

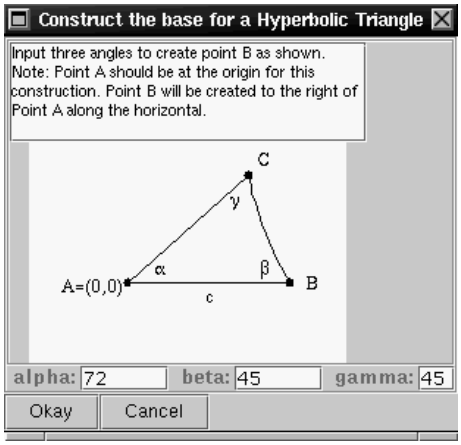
So, we know that a hyperbolic triangle with angles of 72, 45, and 45 degrees must be unique. *Geometry Explorer* has a built-in tool for constructing hyperbolic triangles with specified angles.

To build the interior triangle for our pentagon we will first need a point at the origin. *Geometry Explorer* has a special menu in the hyperbolic main window titled **Misc**. We will use two options under this menu—one to create a point at the origin and the other to assist in creating a 72, 45, 45 triangle.

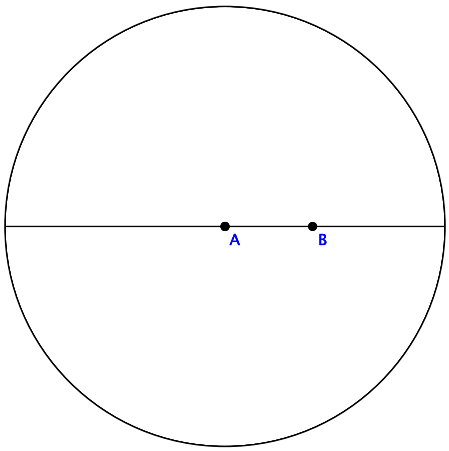
Choose **Point At Origin** from the **Misc** menu to create a point at the origin in the Poincaré plane.



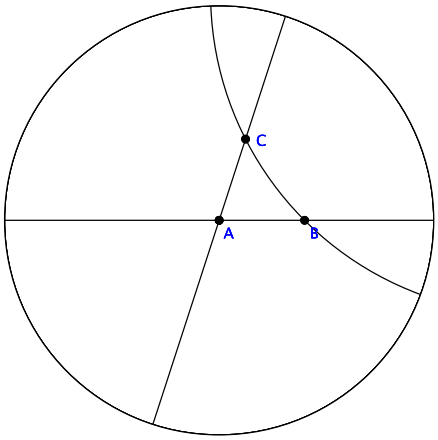
To create our triangle we will first create the base segment of a triangle with angles of 72, 45, and 45. Select the point at the origin and choose **Base Pt of Triangle with Angles...** from the **Misc** menu. A dialog box will pop up as shown. Note how the angles α, β, and γ are designated. In our example we want $\alpha = 72$ and the other two angles to be 45. Type these values in and hit Okay.



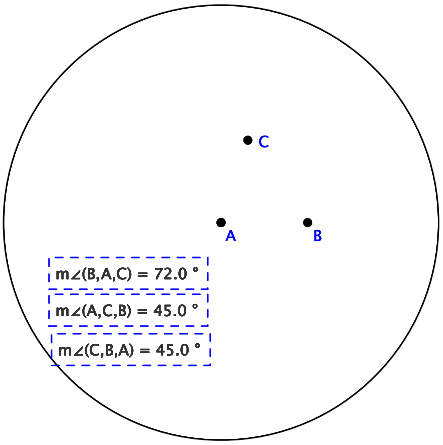
In the Canvas, a new point will be created that corresponds to point B in the dialog box just discussed. Create a line through points A and B as shown.



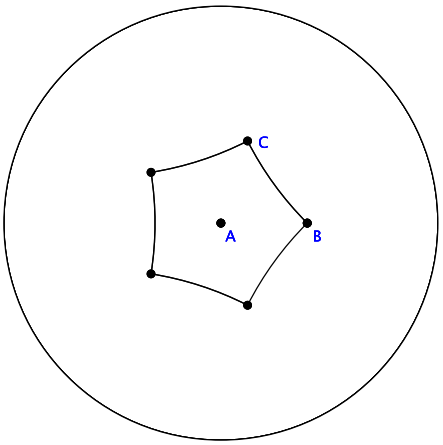
To find the third point of our triangle, we need to rotate line $\overleftrightarrow{AB}$ about point A by an angle of 72 degrees and rotate $\overleftrightarrow{AB}$ about B by an angle of -45 degrees. Carry out these two rotations and then select the two rotated lines and construct the intersection point, point C.



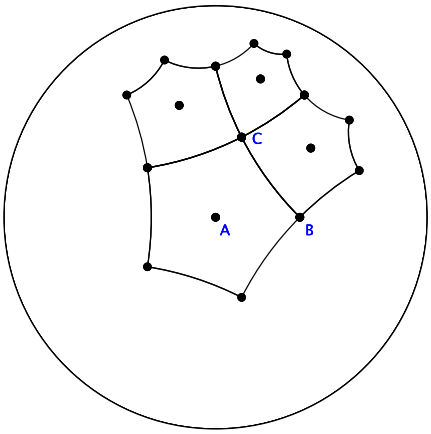
Now, hide the three lines and any extraneous line points. Let's measure the three angles just to verify that we have the triangle we want.



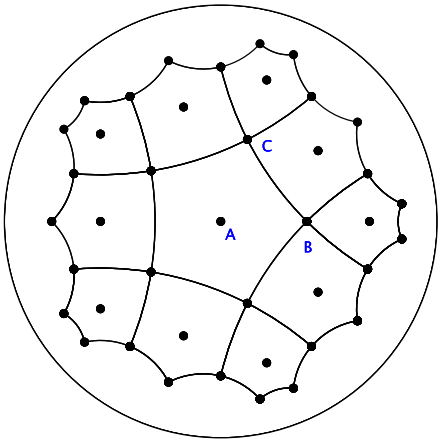
Looks good. Next, hide the angle measurements and connect C and B with a segment. Then, select A as a center of rotation and define a custom rotation of 72 degrees. Rotate segment $\overline{BC}$ (along with its endpoints) four times to get a regular pentagon.



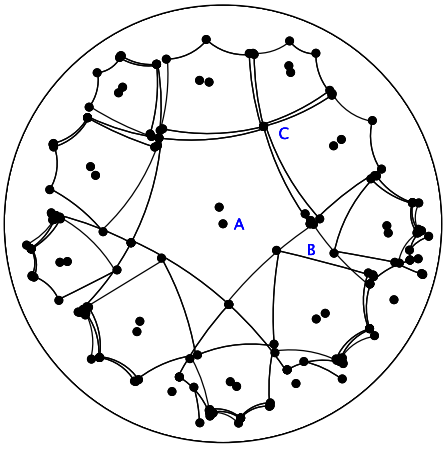
Finally, select point C as a center of rotation and define a new custom rotation of 90 degrees. Then, rotate the pentagon three times, yielding four pentagons meeting at right angles!



If we continue to rotate the pentagon about exterior points in this figure, we see that a tiling of the hyperbolic plane is indeed possible with regular pentagons meeting at right angles.



However, if we move the point at the origin, we see that the tiling breaks up in a quite nasty way. Why is this the case? The problem here is that by translating the origin point we have created compound translations and rotations for other parts of the figure. In hyperbolic geometry compositions of translations are not necessarily translations again as they are in Euclidean geometry.



Chapter 10

Recording Constructions

> It is the glory of geometry that from so few principles, fetched from without, it is able to accomplish so much.
>
> —Isaac Newton (1643–1727)

In many software applications such as word processors and spreadsheets the ability to make *macros* is extremely useful. A macro is a way to encapsulate a series of user actions into a single action. The new action might be a complex formula for a spreadsheet or a special type of textual filter for a word processor.

If one makes frequent use of a series of steps it makes sense to try to encapsulate those steps into a new user action that can be called at any time with a single user command.

To make this encapsulation possible, a program must provide a process whereby a series of steps can be *recorded*, and then *played back* at a later time.

Geometry Explorer provides this mechanism in two ways – through the use of a Recorder window and by the creation of Custom Tools.

10.1 Using the Recorder Window

In *Geometry Explorer* the Recorder window (Fig. 10.1) can be used to record a series of steps that a user carries out in a geometric construction. These recordings can be played back to generate similar constructions. They can also be saved, opened, printed, etc.

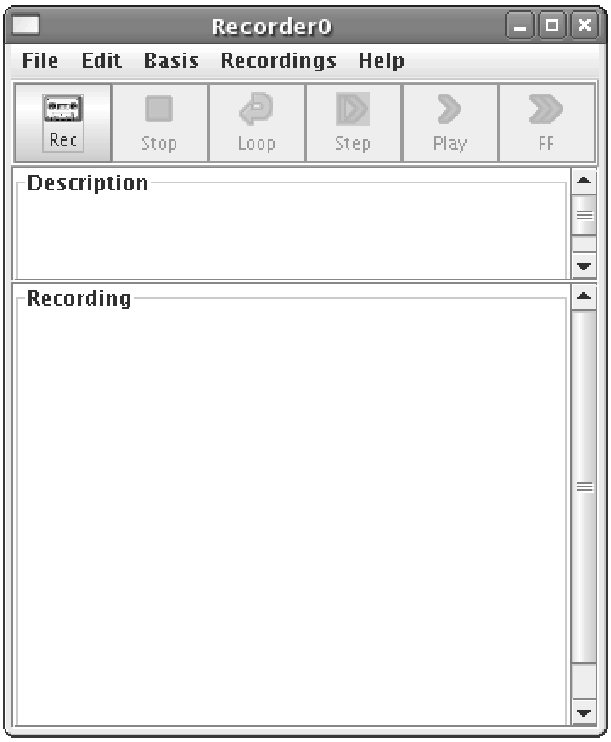


Fig. 10.1 The Recorder Window

To make a recording, one creates a new Recorder by choosing **New Recording** from the **File** menu. A Recorder window like that in Fig. 10.1 will appear. This window is organized into three sections: the Record/Playback buttons, a Description text area, and a Recording text area.

Initially, the only button that is active is the Rec button. This is because no steps have been recorded to play back.

10.1.1 Starting a Recording

To start a recording click on the Rec button. Once this button is pressed the Recorder will listen in on your actions, recording each step in your construction of a geometric figure.

Basis vs Non-Basis Recorded Objects

As you record geometric objects the Recorder makes a distinction between so-called *basis* objects and non-basis objects. An object will be classified as

a basis object if it is not built from any previously constructed objects.

For example, in Fig. 10.2 we have started a new Recorder and have made a segment on the screen. The Recorder records that we have made three objects—the two endpoints and the segment. The endpoints are basis elements since they are not dependent on any other object. However, the segment itself is not a basis element as it depends on the two endpoints for its definition.

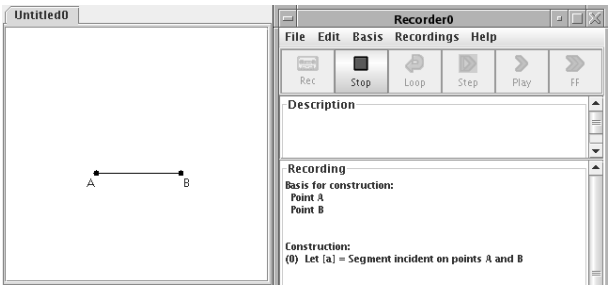


Fig. 10.2 Recording of a Segment Construction

A *construction* object will be an object that depends on basis objects that the recorder has already recorded.

Let us look at another example to clarify this distinction between basis and construction elements. In Fig. 10.3, the segment was created *before* we started up the Recorder. Then, after clicking the Rec button the segment was selected and the Midpoint tool in the Construct Panel was clicked. The Recorder recorded the segment as a basis element, since it does not depend on any previously *recorded* elements, although it does depend on previously constructed elements—points A and B.

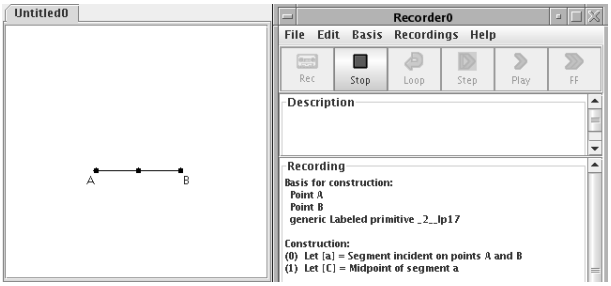


Fig. 10.3 Recording of a Midpoint Construction

Thus, in one case the segment gets recorded as a construction object and

in another case it gets recorded as a basis object. This may seem confusing, but the critical thing to remember is that basis objects are those objects that do not depend on previously *recorded* objects.

10.1.2 Playing a Recording

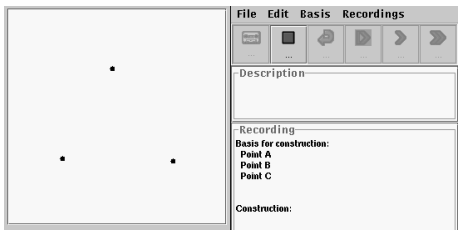
The distinction between basis and construction objects outlined above is critical in understanding how a recording can be played back in the Canvas. To illustrate the Recorder playback capability let's look at an example.

The Centroid of a Triangle

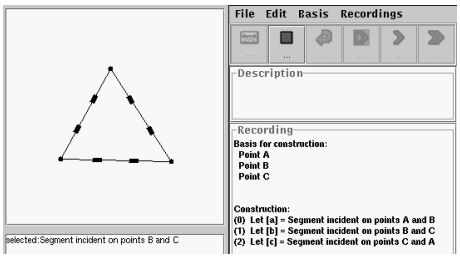
A *median* of a triangle is a segment that joins one vertex to the midpoint of the opposite triangle side. It is a theorem in classical plane geometry that the three medians of a triangle are *concurrent*, that is they meet at a single point. This common point is called the *centroid* of the triangle. This point is also known as the center of gravity of the triangle. If we thought of the triangle as being made of very light, homogeneous material, then the triangle would perfectly balance at the centroid.

Let us make a recording of the median construction of the centroid of a triangle. We could then play this recording on other triangles to find their centroids. Since a triangle is uniquely defined by its three vertexes we will want the basis of our recording to be just the three vertexes of a triangle. We proceed as follows:

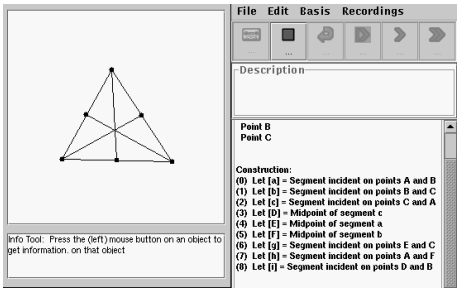
First we create a new Recorder window and start with a blank Canvas. We start with a blank Canvas as we want our basis to be just the first three points that define a triangle. The first step then will be to start the Recorder (click the Rec button) and make three points on the Canvas. Note that the Recorder classifies the points as basis elements.



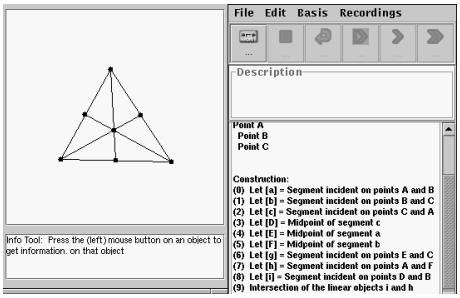
Now select the three points and click on the Closed Polygon tool in the Construct Panel (Second from left in bottom row of the Construct Panel). Note how the three segments get recorded as construction elements.



Next, select the three segments and click on the Midpoint tool in the Construct Panel. Connect each midpoint to the vertex opposite it in the triangle. Note how these medians appear to intersect at a common point.

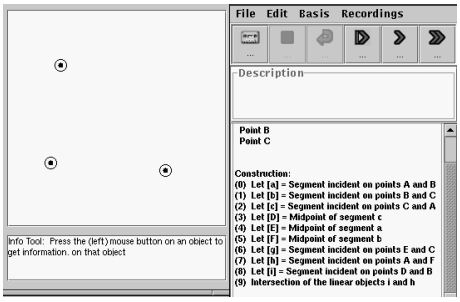


Finally, select two of the medians and click the Intersect tool in the Construct Panel to construct the centroid of the triangle. Then, stop the recording by clicking Stop in the Recorder window.

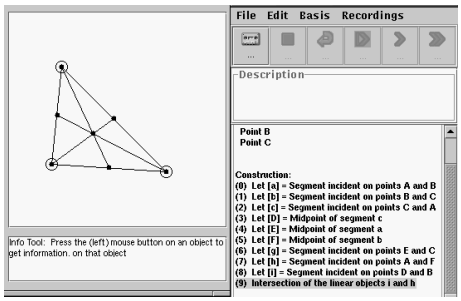


At this point our centroid recording (or macro) is complete. To play this recording back we need to specify a set of geometric objects that exactly match the number and type of our *basis* elements. Then the Recorder can substitute these new basis elements for the old basis elements and play the recording on them, because the only thing that recorded construction elements depend on are basis elements.

Clear the Canvas and create three points on the Canvas. Select all three. The Recorder will check to see if the selected objects match its recorded basis objects and if so, it will activate the three playback buttons Step, Play, and FF. These will be discussed in detail below.



Click on the Play button. The steps in the recording will be played back sequentially, with each step being highlighted as it is played back. When the playback is done, the Recorder will inactivate all playback buttons and will re-activate the Rec button. If we hit Rec at this point our recording would be erased for a new recording to start.



10.1.3 Recursive Recordings

So far we have looked at how to record a sequence of steps used in constructing a geometric object. *Geometry Explorer* also provides a mechanism for recording the construction of a *fractal* geometric object, namely one that is recursive or *self-similar*. We have already looked at fractals in Chapter 5 and Chapter 8.

A fractal is equally complex at all levels of magnification. Thus, to construct a fractal is a bit tricky, since one needs to make jaggy objects down to an infinitely small range!

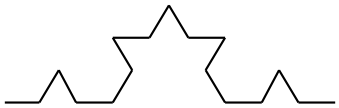
However, there is one class of fractals that are constructible, if one uses a recursive (or looping) process. These are the *self-similar* fractals.

Self-similarity can best be described by an example—the Koch curve. (For a different take on the Koch curve see Chapter 8).

The Koch curve

To construct the Koch curve we start with a segment $\overline{AB}$. We remove the middle third of the segment and replace that third with the two upper segments of an equilateral triangle of side length equal to the middle third we removed. The resulting curve is shown at the right.

Now, think of this curve as a template that can be placed on any segment. We can make a copy of the template, shrink it by a factor of $\frac{1}{3}$, and replace one of the small segments with the scaled down copy of the template. If we do this for each of the four small segments in the template we get the curve at right.



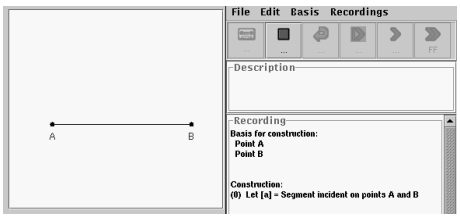
At this point we have a curve with 16 small segments, each of length $\frac{1}{9}$ the original segment $\overline{AB}$. We can now replace each of these segments with a $\frac{1}{27}$ scale copy of our template, yielding a new curve with 64 segments each of length $\frac{1}{27}$ the original.

Apply this template reducing-copying procedure again to each of the 64 segments, yielding a new curve with 256*segments*, each of length $\frac{1}{81}$ the original.

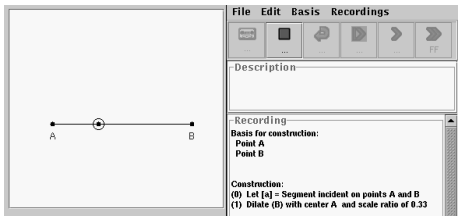
The Koch curve is the curve that results from applying this template replacement process an *infinite* number of times. The curve is self-similar in the sense that if you took a piece of it and magnified that piece by a factor of 3, you would see basically the same curve again. Each small part of the curve essentially is identical to the curve itself. This qualifies the curve as a fractal since no matter how much we magnify the curve we never get rid of the jaggedness of the curve.

Let's see how to make a recording of the Koch curve.

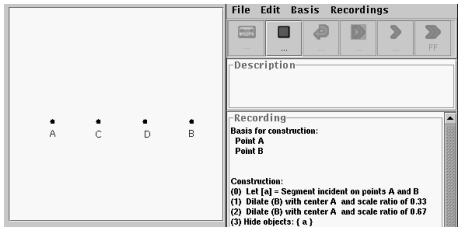
First, we need to record the construction of the template. Pop up a new Recorder (under the **File** menu) and start with a clear Canvas. Click Rec in the Recorder window to begin recording. Create a segment $\overline{AB}$ on the Canvas.



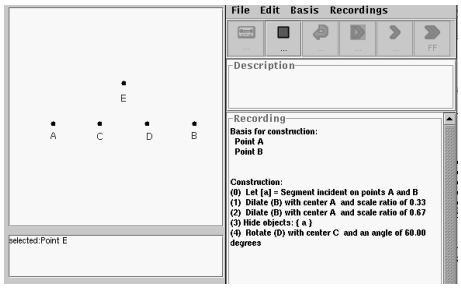
Divide $\overline{AB}$ into three equal parts as follows: Select A and choose **Center** from the **Mark** pop-up menu in the Transform Panel. Define a custom dilation of $\frac{1}{3}$ by choosing **Dilation** from the **Custom** pop-up menu in the Transform Panel. Select B and click the Dilate tool in the Transform Panel.



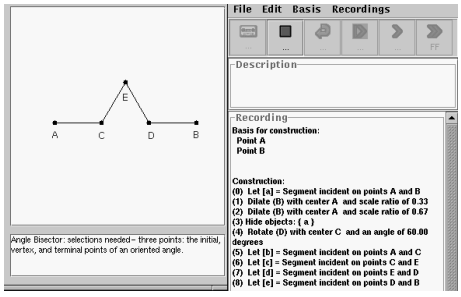
Similarly, carry out the steps to dilate B by a ratio of $\frac{2}{3}$ towards point A. Then hide $\overline{AB}$. We have now split $\overline{AB}$ into equal thirds at C and D.



Now we will create the "bump" in the middle of the template. Set C as a center of rotation/dilation and define a custom rotation of 60 degrees. After this is done, select D and click the Rotate tool in the Transform Panel.



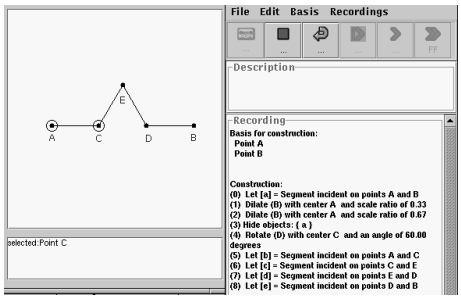
Finally, select points A, C, E, D, and B (in that order) and click on the Open Polygon tool in the Transform Panel (first from left in third row). At this point our template curve is complete. However, do not stop the Recorder yet.



At this point in the Koch curve construction, the template should be used to replace each of the four segments that are in the template itself. When a process is defined in terms of the process itself (or a part of that process) that process is called a *recursive* process. Thus, we need to recursively apply

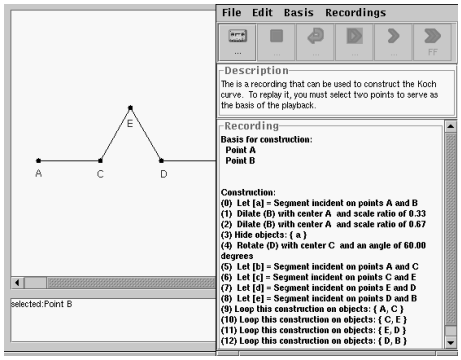
the same recording that we just completed to each of the four segments of the template curve that is currently in the Canvas. We do this using the Loop button in the Recorder window.

Select points A and C. The Loop button (the one with the arrow looping back on itself) will now be active. In general the Loop button will become active whenever a selection is made that matches the set of basis elements used in the recording. Since this recording has two points (A and B) as basis elements, then any selection of two points will activate the Loop button.



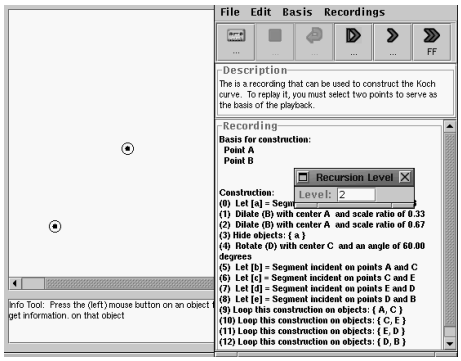
Now, click the Loop button to record the fact that we want the recording to recursively play itself back on A and C. Then, make the Recorder loop on each of the other three segments. Do this by 1) selecting C and E and clicking Loop, 2) selecting E and D and clicking Loop, and 3) selecting D and B and clicking Loop. Then, stop the Recorder as our Koch curve construction is complete.

At the right we have added a short explanation of the recording in the text area labeled "Description". This description will tell anyone using the Koch recording what its purpose is and what basis elements are needed to playback the recording.



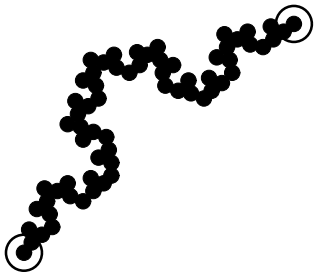
To playback this Koch curve recording we need to create two points in the Canvas to serve as the basis for playback. Once we create the correct basis points, the Recorder playback buttons will become active and we can begin playback.

In the figure on the right we have created two points, selected them, and hit FF. At this point the Recorder needs to know how "deep" the recursive looping should be played back. A dialog box will pop up asking for this level of recursion.

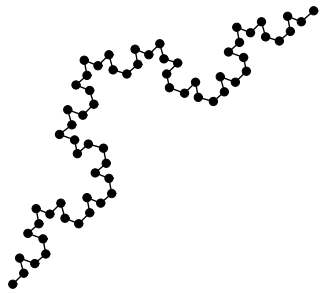


A recursion level of 0 would mean to just play the recording and not loop at all. A level of 1 would mean to play the recording back and loop the recording on all subsegments. Level 2 would mean that the sub-sub-segments would be replaced with templates, etc.

Let's see what happens at level 2. Enter "2" in the dialog box and hit Okay. The recording will playback, recursively descending down segment levels as the template replaces smaller and smaller segments. Watch closely how the recording gets played back to get a feel for this recursive process.



The curve is densely packed together so lets stretch it out a bit by grabbing one of the endpoints and dragging it.



10.1.4 Saving, Opening, Printing

Under the **File** menu in the Recorder window there are menu items that allow one to save the current recording, open an existing recording, print

the text of a recording (i.e. what appears in the Recorder text area), open a new Recorder window, hide a Recorder window, and close a Recorder.

These operations are fairly standard file handling operations. It should be noted that if one has not saved a recording and attempts to close the Recorder, then the user will be asked if they want to save the recording or not before closing the window.

Hiding a Recorder window does not close the window. The window still exists, but is not visible. Use the **Recordings** menu in a visible Recorder window (or the **Windows** menu in *Geometry Explorer* to choose the hidden Recorder window and pop it back up.

10.1.5 Playback on Sets of Basis Elements

Suppose that we want to play a recording back on multiple basis sets simultaneously. We can store away sets of basis elements by selecting a basis set (for example two points in the Koch curve recording) and choosing **Store Givens set** under the **Basis** menu in the Recorder window. Once we select the correct basis set, the **Store Givens set** menu item will be activated.

For example, suppose that we have a triangle ΔABC and an open Recorder window containing the Koch recording. Then, we can select A and B and choose **Store Givens set** to store this set of points. Likewise, we can select B and C and choose **Store Givens set**, and finally select C and A and choose **Store Givens set**. At this point, we have stored three sets of basis elements that match the recording's basis. If we hit FF and choose a level of 1 we would get the Koch Snowflake curve. (Fig. 10.4)

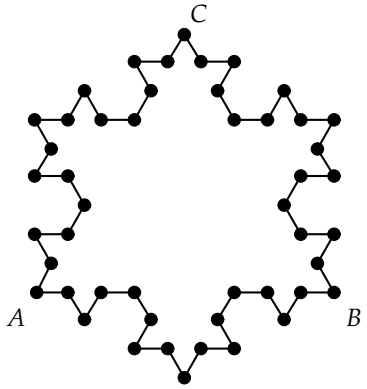


Fig. 10.4 The Koch Snowflake Curve

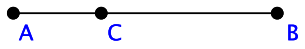
10.2 Custom Tools

The second method provided by *Geometry Explorer* to record a user's geometric construction is thorough the creation of a Custom Tool. The difference between recording a construction using a Recorder Window versus a Custom Tool is that a Recorder Window "listens in" as a construction is carried out and then stores the result. A Custom Tool is created *after* you have finished a construction.

To illustrate this, we will create a Custom Tool to carry out the Koch snowflake construction described in the first part of this chapter.

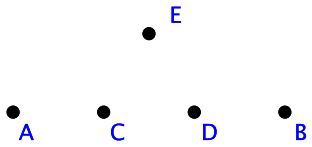
We start with the construction of the template for the curve. Create a segment $\overline{AB}$. We divide $\overline{AB}$ into three equal parts as follows:

Select A and choose **Center** from the **Mark** pop-up menu in the Transform Panel. Define a custom dilation of $\frac{1}{3}$ by choosing **Dilation** from the **Custom** pop-up menu in the Transform Panel. Select B and click the Dilate tool in the Transform Panel.

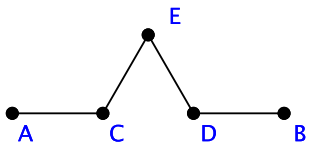


Similarly, carry out the steps to dilate B by a ratio of $\frac{2}{3}$ towards point A. Then hide $\overline{AB}$. We have now split $\overline{AB}$ into equal thirds at C and D.

Now create the "bump" in the middle of the template by setting C as a center of rotation/dilation and defining a custom rotation of 60 degrees. After this is done, select D and click the Rotate tool in the Transform Panel.



Finally, select points A, C, E, D, and B (in that order) and click on the Open Polygon tool in the Transform Panel (first from left in third row). At this point our template curve is complete.



To create a Custom Tool for our Koch curve we first must select the objects we want the tool to create when it is executed. In out case, we will have the tool construct only the segments. Select the four segments in the curve (and not the endpoints). Then, select the tab labeled "Custom" in the Construct area of the Tool Panel. Clicking on the button labeled "Custom Tool" brings up a popup menu as shown in Figure 10.5)

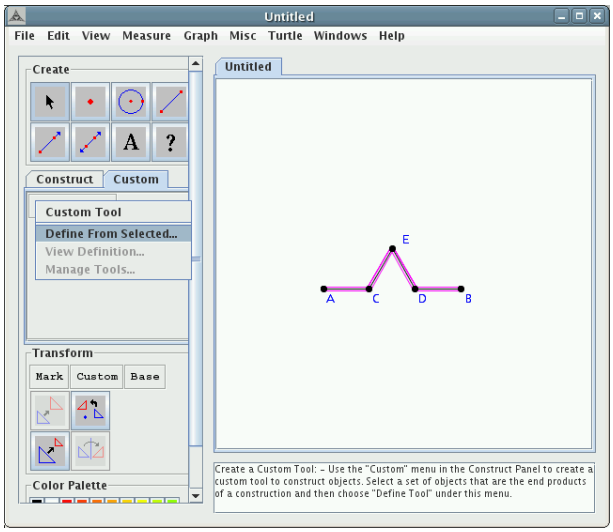
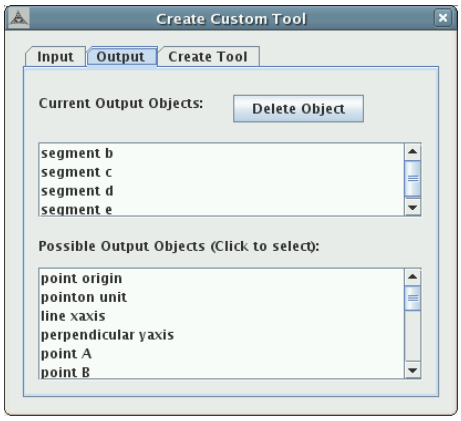
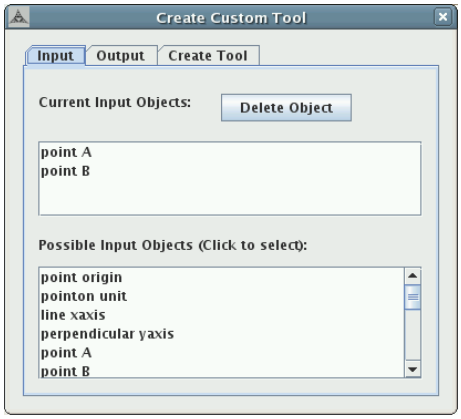


Fig. 10.5 Custom Tool

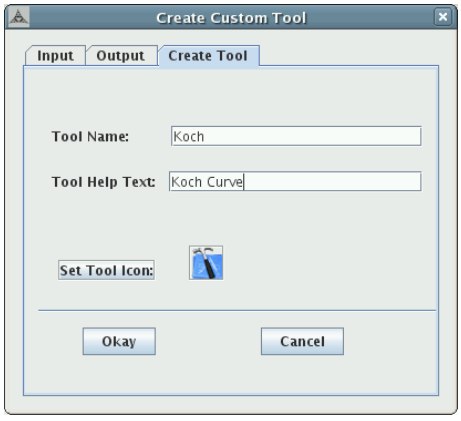
Select "Define from Selected...". A dialog box will pop up with three tabbed panels labeled "Input", "Output", and "Create Tool." The Output panel will be showing. In this window we see a list with the four segments we want the tool to create. These are the *ouput* of the tool.



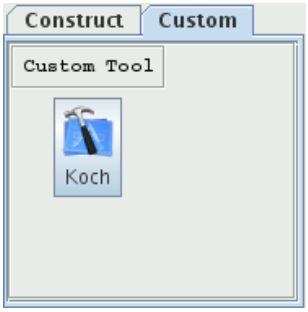
If we click on the "Input" tab we see a different list of objects. *Geometry Explorer* calculates all parent objects which the four segments depend on. In this example, there are two – points A and B. The deepest common ancestors on which everything depends are these two points A and B. These will be the necessary *input* to the new tool.



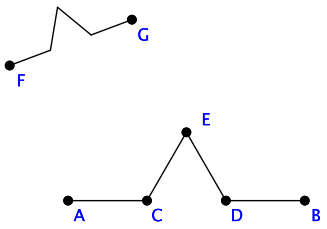
To finish the creation of the tool, click on the "Create Tool" tab. In this panel, we set the tool's name, any help text, and an icon for the new tool button that will be created. In our case, we name the tool "Koch" and have help text describing what the tool creates. We can also set an icon for the tool. An icon is an image file that can be used as a picture for the tool button that will be created. For this example, we will just use the default icon as shown. Click "Okay" to define the tool.



Once the tool is defined a new button will be created in the Custom panel in the Tool Panel as is shown here (the button labeled "Koch"). Every time a new tool is defined it will be added to this subpanel of the Tool Panel.



To use the tool, first click on its button (the one labeled "Koch") and then click twice in the Canvas. Two points (F and G will be created and then the saved construction of the Koch curve will be automatically carried out, beginning with points F and G. Note that all intermediate objects (such as the points) will be hidden.

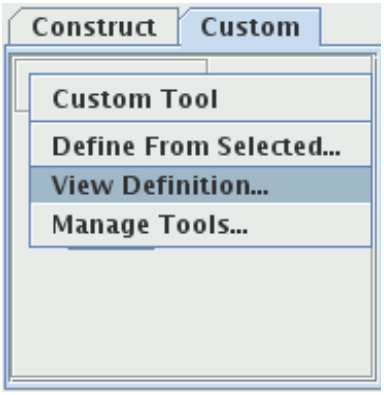


10.2.1 Managing Custom Tools

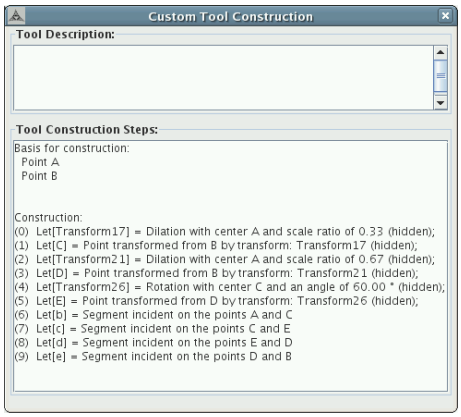
Once a custom tool is created we can change its properties and view a summary of its construction by using the menu options under the **Custom Tool** menu in the Custom Panel.

For example, consider the previous example, where we created a tool to carry out the construction of the Koch curve template.

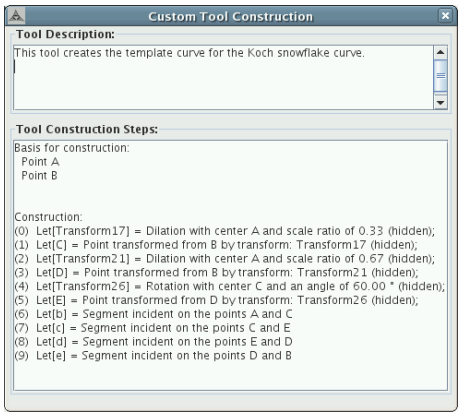
If we click on the **Custom Tool** menu in the Custom Panel we see three menu options.



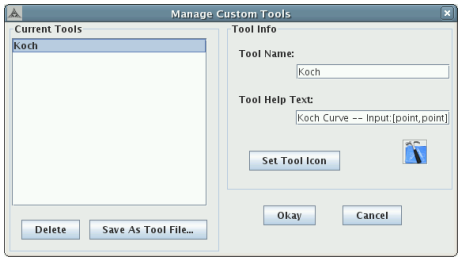
If we select **View Definition...** a dialog box will pop up describing the steps that the tool carries out when it is used.



Note that the upper text area is empty. We can use this area to give users more information about the tool, as shown here.



Now, suppose we select **Manage Tools...** from the **Custom Tool** menu in the Custom Panel. A dialog box will pop up as shown.



On the right side of this dialog box there is a panel labeled "Tool Info." In this panel we can change a tool's name, help text, and icon. On the left side of the dialog box, there is a list of all currently defined tools. We can select a group of these and either hit the "Delete" button or the "Save As Tool File..." button. These two options do as their names suggest. In the

first case, the selected tools will be deleted from the Custom Panel. In the second case, we can save all selected tools to a file.

Chapter 11

Animation

> Our nature consists in movement; absolute rest is death.
>
> —Blaise Pascal (1623–1662)

It is often the case that a good animation of a geometric concept will help the viewer better understand that concept. *Geometry Explorer* provides the capability to animate almost any object that one constructs, including points, lines, rays, segments, circles, arcs, areas, and input parameters.

11.1 User Interface for Animation

Animations in *Geometry Explorer* are controlled by using the five menu items under the **View** menu in the main *Geometry Explorer* window. These menu items are:

1. **Animate Object**: To animate an object we first select the object in the Canvas. Once the object is selected the menu item **Animate Object** will be enabled. Choosing this menu item will result in a dialog box popping up. The dialog box, titled "Animation Manager" will allow for the control of various animation parameters such as speed and direction.

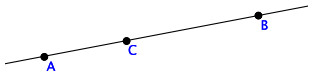
2. **Speed Up Animation**: Use this option to speed up the motion of an animated object. The object must be selected for the menu item to to be enabled.

3. **Slow Down Animation**: Use this option to slow down the motion of an animated object. The object must be selected for the menu item to to be enabled.

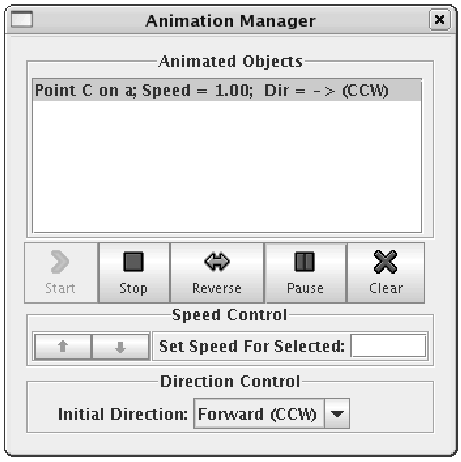
4. **Stop All Animators**: Use this menu item to stop all currently defined animations.

5. **Show Animation Panel...**: Use this menu item to bring up the Animation Manager dialog box. This box can be used to control animation parameters for objects that are currently animating.

As an example, in the figure on the right suppose we wish to animate point C, which is attached to the line $\overleftrightarrow{AB}$.

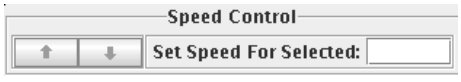


To animate C we select C and choose **Animate Object** from the **View** menu (or right-click on the object and choose **Start Animating** from the pop-up menu). The point will start moving and the dialog box on the right will appear.



Note the various sections in this dialog. At the top there is a large area where currently animating objects are listed, along with their speed and initial direction. In the middle we see a row of motion control buttons. Below this is a speed control area and finally a place to set an object's initial direction of motion.

Suppose that we want to speed up the motion of C. We can either type a speed into the field labeled "Set Speed For Selected" or we can use the up and down arrow keys to change speeds.



The direction control area is used to set the initial direction of motion of our object. This only applies to objects that are animated along a path, for example points on a line or circle.

The motion control buttons can be used to start and stop the animation, to reverse direction of motion, to pause all animations, and to clear all animations.

When animating a point attached to an object that has a boundary point (segment, ray, or arc), once the attached point reaches the boundary point the animation of that point will switch direction.

If the animating point is on a line or ray then the point could conceivably continue moving forever in either direction on the line or in a direction away from the endpoint on the ray. In *Geometry Explorer*, however, once the point hits the boundary of the *Geometry Explorer* drawing area (the Canvas) it will switch directions.

If the animation is of a point on a circle then the point will move forward or backward around the circle indefinitely. The direction of motion will switch every time the point reaches a position on the circle directly horizontal from the center of the circle on the right side of the circle.

11.2 Animation in the Euclidean Plane

In the Euclidean geometry environment of *Geometry Explorer*, animation is designed so that all moving objects move an equal distance in equal time. Thus, if point A moves one unit along a segment in one animation cycle, and point B is moving along a circle in the same cycle, then the distance traveled by B on the circle will also be one unit. This synchronization allows for the accurate prediction of the behavior of multiple animators.

11.2.1 Animating Circles along Segments - The Cycloid

To illustrate how animation can be used to define interesting Euclidean figures, let's look at the construction of the *cycloid*. The cycloid is a curve that is defined by plotting the position of a point attached to a circle as the

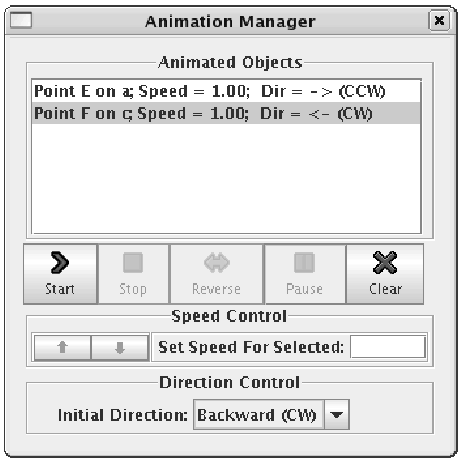
circle "rolls" along a line. A cycloid is basically the curve traced out by a point on the rim of a bicycle tire as the tire rolls along a surface.

To start the construction of a cycloid we create $\overline{AB}$ to serve as the road along which our circle will rotate. Actually, we will have our circle roll along this road with the center of the circle on the road level. (Think of the road having a deep groove in it and the axle rolling on the road surface) We need to construct a circle of fixed radius so we create $\overline{CD}$ to serve as this fixed radius.

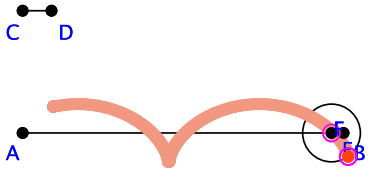
Now, attach a point E to $\overline{AB}$, and construct a circle at E of radius CD by selecting E and then $\overline{CD}$ and clicking on the Circle construction tool in the Construct panel. Attach a point F to the circle and change the color of F to red, so we can keep track of it. Also, set F to be traced by selecting F and choosing **Trace On** from the **View** menu.

To create the cycloid we need the circle to roll along $\overline{AB}$. We cannot re-create the *actual* physics of this rolling in *Geometry Explorer*, but we can mimic the rolling by having point E move to the right along $\overline{AB}$ while simultaneously moving point F clock-wise around the circle. Since animation is designed so that both points move equal distances in equal times the net effect of the two motions is the same as the motion of the circle itself rolling along $\overline{AB}$.

Select points E and F choose **Show Animation Panel...** from the **View** menu. When the motion dialog box pops up, select Point F from the list. Then, set the initial direction for Point F to be backward or clock-wise, using the Direction Control list.

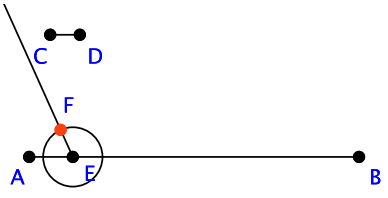


To start the animation, make sure the entries for E and F are both selected in the Animation Manager (hold down the shift key to do a multiple selection in the list). Then, click "Start" in the animation dialog box to see a curve like the one at the right. Click "Stop" before E reverses direction back along $\overline{AB}$.

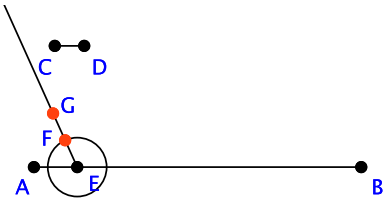


What would happen if the point that is being traced was inside the circle as it rolled along? What would happen if it was outside the circle?

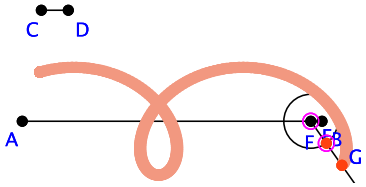
Undo the animation by choosing **Undo** from the **Edit** menu. Remove the trace on F by choosing **Clear All Traces** from the **View** menu. Construct a ray from point E through point F as shown.



We now create a point G on $\overrightarrow{EF}$ that is double the radial distance of F from E. Select E and set it as a center of rotation by using the **Mark** menu in the Transform Panel. Then, create a custom dilation with ratio $\frac{2}{1}$ by using the **Custom** menu in the Transform Panel. Dilate point F by this dilation and create a trace on G (as we did above for F).



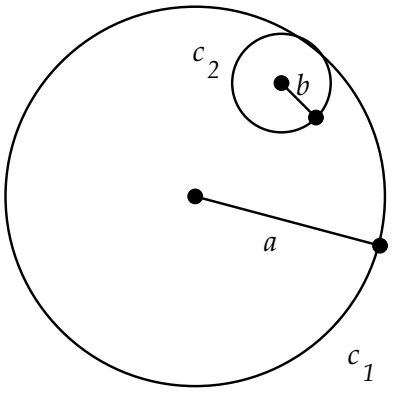
Re-start the animation and the curve at the right will be traced. This curve is called a *trochoid*.



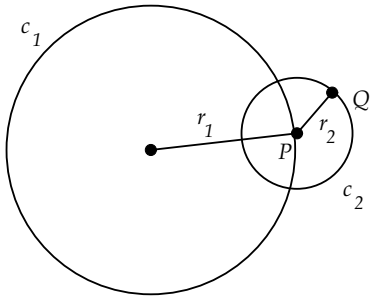
11.2.2 Animating Circles along Circles - The Hypocycloid

Many children love to play the game of Spirograph. In this game, one places a pen in a circular gear that rolls within a larger circular gear. The pen in the inside gear traces out elegant curves as the gear rotates. Let's see how we can create this effect in *Geometry Explorer*.

In the figure at the right we have a circle c_2 of radius b inside a larger circle c_1 of radius a. If c_2 rolls along c_1 until it returns to the starting position shown, then it would have rolled a total distance of $2\pi a$ and if it rotated a total of K times about its center point, then $2\pi a = K2\pi b$ and thus $K = \frac{a}{b}$. That is, if c_2 rolls along c_1, it rotates $K = \frac{a}{b}$ times as it makes one circuit around c_1.



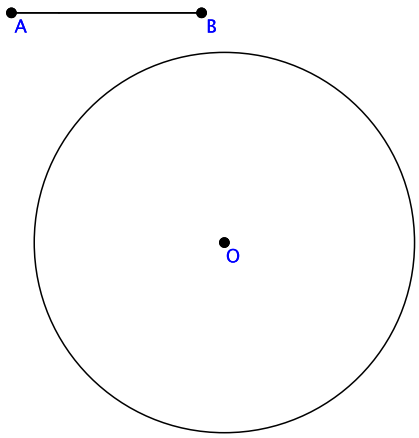
In *Geometry Explorer*, we can create a circle c_1 along which the center of another circle c_2 can move, as shown. Also, if we animate points P on c_1 and Q on c_2, then P and Q will move in such a way that the distance traveled by both in a unit of time will be the same. How does this motion relate to the physical rolling motion of the previous figure?



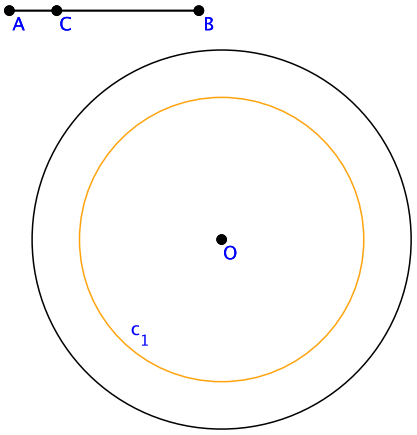
To analyze this, let's consider the net distance traveled by both P and Q as P makes one complete circuit around c_1. P will travel a distance of $2\pi r_1$ and Q will travel $2\pi r_2 N$ for some N, with N being the number of times c_2 rotates about its center. Thus, $N = \frac{r_1}{r_2}$. However, the motion of P also transforms Q as P rotates about O, and it transforms Q exactly once as P rotates once about O. If c_2 was actually rolling along the circle of radius $r_1 + r_2$ then, it would go through $\frac{r_1+r_2}{r_2} = N + 1$ rotations. Thus, the animation of circle c_2 on c_1 in *Geometry Explorer* actually models the physical rolling of c_2 inside a circle of radius $r_1 + r_2$.

Let's put this all together in an example.

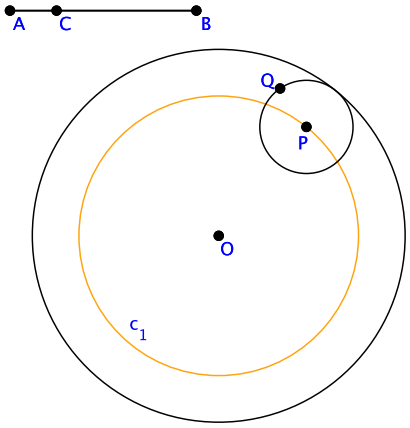
In the figure at the right we first create $\overline{AB}$ to serve as the fixed radius for the circle that the small circle will roll along. Create this circle by creating the center point O, and then selecting the center and $\overline{AB}$ and using the Circle construction tool in the Construct Panel.



Next, dilate point B by a factor of $\frac{1}{4}$ towards point A, yielding point C, and construct two segments – $\overline{AC}$ whose length is a quarter the length of $\overline{AB}$ and $\overline{CB}$ whose length is $\frac{3}{4}$ the length of $\overline{AB}$. Construct circle c_1 with center O and radius given by segment $\overline{CB}$.

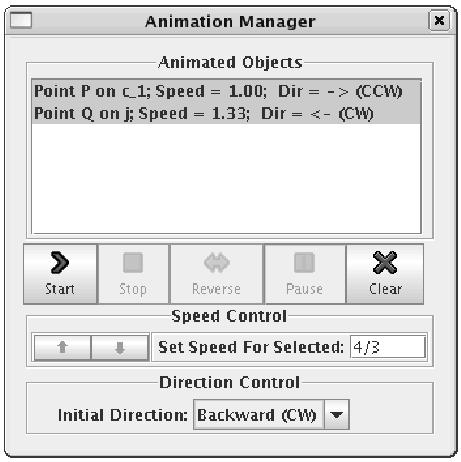


Next, attach P to c_1 and construct circle c_2 with center P and radius AC. Finally, attach a point Q to c_2 and set Q to be traced by selecting Q and choosing **Trace On** from the **View** menu.

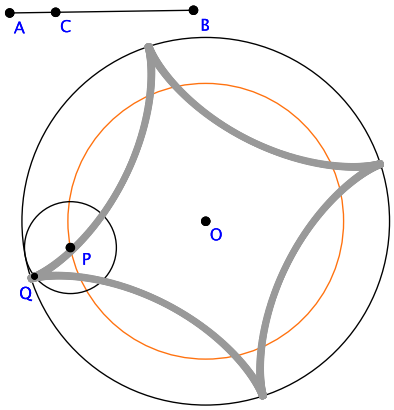


Now, the ratio of radii of c_1 to c_2 will be the number N in the analysis above. In this case $N = 3$, so Q will rotate four times around c_2 as P makes a complete circuit around c_1.

Select points P and Q choose **Show Animation Panel...** from the **View** menu. When the animation dialog box pops up, select Point Q from the list. Set the initial direction for Point Q to be backward or clock-wise, using the Direction Control list.



At this point we are ready to animate the rolling circle. Make sure that both entries for P and Q in the animation dialog box are selected and hit "Start". The curve traced out is known as a *hypocycloid*. It is also known as an *astroid*.



11.3 Animation in the Hyperbolic Plane

Animation in the hyperbolic plane is accomplished by the use of *Möbius transformations*. In general a Möbius transformation is an invertible transformation of points in the hyperbolic plane which are represented by complex numbers of the form $z = x + iy$, where $i = \sqrt{-1}$. Möbius transformations have the general form:

$$z = e^{it}\frac{z - z_0}{1 - \overline{z}_0 z}$$

(For more information on Möbius transformations consult Chapter 7.)

Möbius transformations include the basic hyperbolic transformations of rotations and translations. *Geometry Explorer* uses Möbius transformations

to animate points along circles, lines, etc. Simple animations of this type are quite similar to animations in Euclidean geometry of points along lines, circles, etc. An animating point in hyperbolic geometry will have constant speed in the hyperbolic sense, even though it will appear to move at different speeds to someone viewing the motion from our three-dimensional Euclidean space.

Compound animations, where a point is moving on a circle which is itself moving on another object (like the cycloid example above), are harder to construct and analyze in Hyperbolic geometry. The basic problem is that a composition of translations or rotations do not necessarily make another translation or rotation. This is quite different from Euclidean geometry where the composition of translations is always another translation and the composition of rotations is another rotation. Since compositions of transformations are not as uniform in Hyperbolic geometry, as compared to Euclidean geometry, when one object is moving along another object which is itself moving, the net result of the compound motion is not entirely predictable.

A second problem comes with the relationship between segments and circles in Hyperbolic geometry. In the Euclidean plane, there is a nice linear relationship between the length of the radial segment of a circle and the circumference of the circle. In the hyperbolic plane, this relationship is no longer linear, in fact the circumference of a circle of hyperbolic radius r is given by $C = 2\pi sinh(r)$.

This non-linear relationship can be seen in (Fig. 11.1) below. Here we have done the same sequence of construction steps in the hyperbolic plane that we would do to construct a hypocycloid of three cusps in the Euclidean plane. We start with a segment $\overline{AB}$ and a point C that is the dilation of point B towards A by one-fourth of the length of $\overline{AB}$. Then, we create two circles with the same center – a circle with radius AB and a circle of radius CB. Next, we attach a point P to the smaller circle and create a circle with center at P of radius AC. Finally, we attach point Q to this new circle and animate P counter-clockwise around its circle and point Q clock-wise around its circle. If this had been done in the Euclidean plane, point Q would trace out a four-branched path as P moved around the circle, and Q would return to the exact position at which it started. This is due to the linear relationship between the radius and circumference of a Euclidean circle. However, it is not at all clear whether the path traced out by Q does in fact close up as P moves around the larger circle.

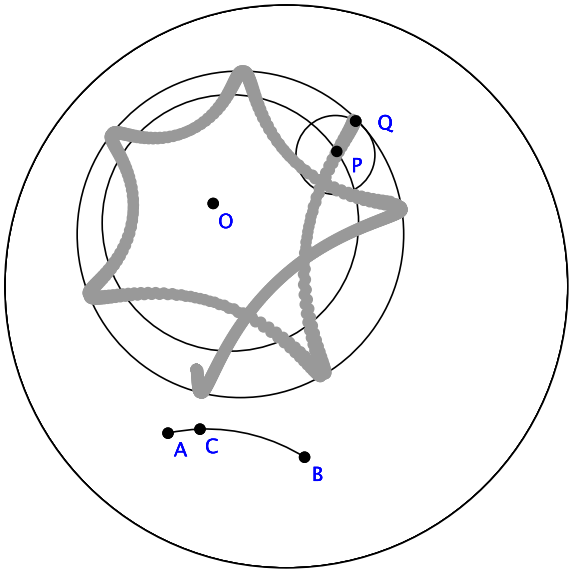


Fig. 11.1 Hypocycloid Construction in Hyperbolic Geometry

While animations can be set up as easily in Hyperbolic geometry as they are in Euclidean geometry, one cannot directly assume that the curves traced out by compounded motions will have any direct relationship to the curves traced out by similar Euclidean motions.

11.4 Animation in Elliptic Geometry

Many of the comments mentioned above for Hyperbolic geometry are also true for Elliptic geometry. Again there is no nice linear relationship between the length of a radial segment in a circle and the circumference of a circle. Also, there is again the problem of a composition of translations or rotations not necessarily producing another translation or rotation.

However, if we set up an animation like we did for the hypocycloid in the Euclidean plane, it appears that we get a hypocycloid with four arcs, as shown in (Fig. 11.2)

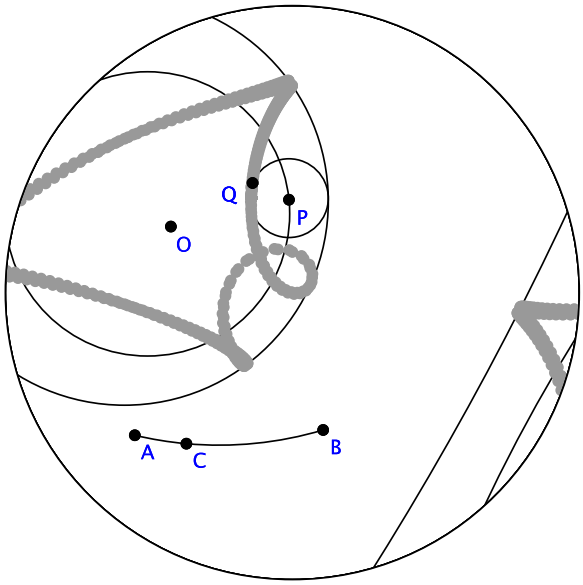


Fig. 11.2 Hypocycloid Construction in Elliptic Geometry

Chapter 12

Geometry Explorer and the Internet

> It can be shown that a mathematical web of some kind can be woven about any universe containing several objects.
>
> —Bertrand Russell (1872–1970)

12.1 The *Geometry Explorer* Web Browser

Geometry Explorer comes with a built-in web browser for viewing pages in the on-line help system and in making web links on the Canvas.

12.2 The *Geometry Explorer* Help System

There is an extensive on-line help system that can be accessed via the **Help** menu in the main *Geometry Explorer* window. The help system is designed as a series of web pages that are organized into categories that roughly correspond to the visual areas in the *Geometry Explorer* window—panels, menus, etc.

In (Fig. 12.1) we see the main help page for *Geometry Explorer*. From this starting page one can navigate to other pages discussing various capabilities of the program. There are also many geometric construction examples included in the help system.

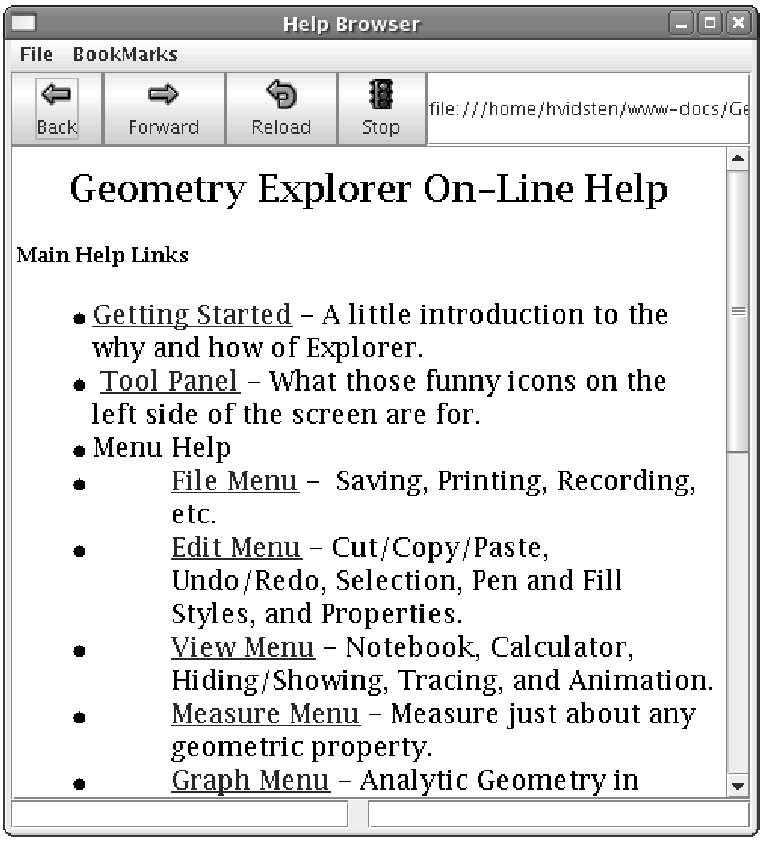


Fig. 12.1 The *Geometry Explorer* Help Browser

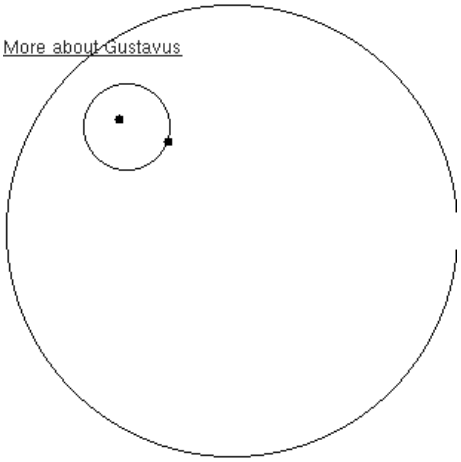
12.3 Using Web Links Directly from the Canvas

It is often useful to incorporate additional information on a geometric construction along with the construction itself. *Geometry Explorer* allows one to create an Internet hyper-link to an Internet web page *directly* from the Canvas. With this capability there is a virtually unlimited amount of information available to augment and enhance a given geometric construction.

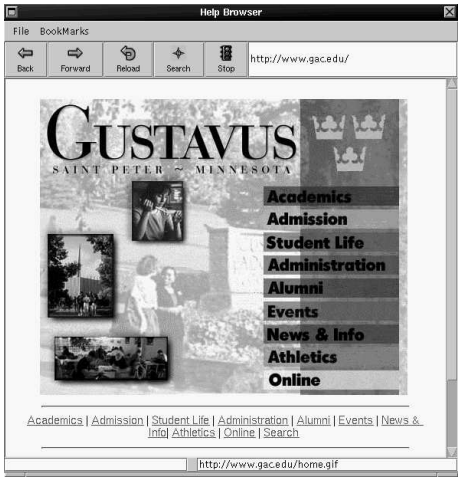
For example, to create a web link in the Canvas choose **Set Web Link...** from the **View** menu. A dialog box like the one at right will pop up.

In the first text area we type in the link text that will appear in the Canvas. In this case, we will create a link to the home page of Gustavus Adolphus College, so we label our link "More about Gustavus". In the second text area we type the *actual* address (URL) of the page to which we are linking. In this case, the URL is "http://www.gac.edu."

Once we hit Okay a text box will appear in the Canvas with the desired link text. We can position this text box anywhere in the Canvas we choose. The link text will be displayed as blue, underlined text. This is consistent with the way most browsers display links.



Now, to link or *surf* to this page we use the Info tool (the one with the question mark in the Create Panel). Click on the Info tool and then click on the text link in the Canvas. A web browser will pop-up and take you to the linked page.

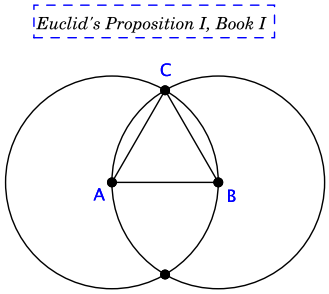


Note that the built-in browser can handle only simple web pages. Web pages that utilize scripting languages, like JavaScript, or that have multiple forms, will not be displayed correctly.

12.4 Saving Constructions as HTML Files

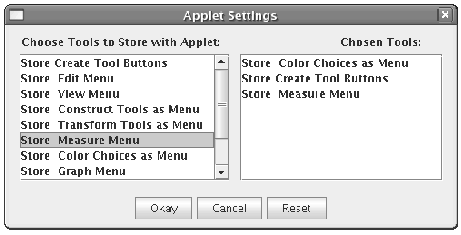
Constructions made with *Geometry Explorer* can be saved as HTML files which can be then played back by anyone with a web browser capable of running Java applets.

For example, suppose that we had just created the construction which illustrates Euclid's first proposition of Book I of *The Elements*. In this proposition Euclid shows how to construct an equilateral triangle.

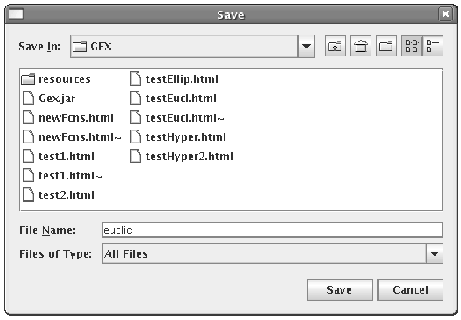


If we want to save this construction as an HTML file we choose **Save as HTML...** from the **File** menu in the main window. A dialog box labeled "Applet Settings" will pop up. We can use this dialog box to specify the functionality that the saved applet will exhibit.

For example, suppose that we want to save this applet so that users can have access to the Create panel tools, can change colors, and create measurements. Then, we would select these three items, as shown in the figure at the right, and click the Okay button.



A file dialog box will pop up asking us to name the file and to place it in a directory. In this example we choose to store the file as "euclid.html" under a directory titled "GEX". Note that we do not need to type in the suffix ".htm" for the file. This will be added automatically.



Once we click on the Save button, the construction will be saved to an HTML file. In order for this file to work correctly, we must also make sure

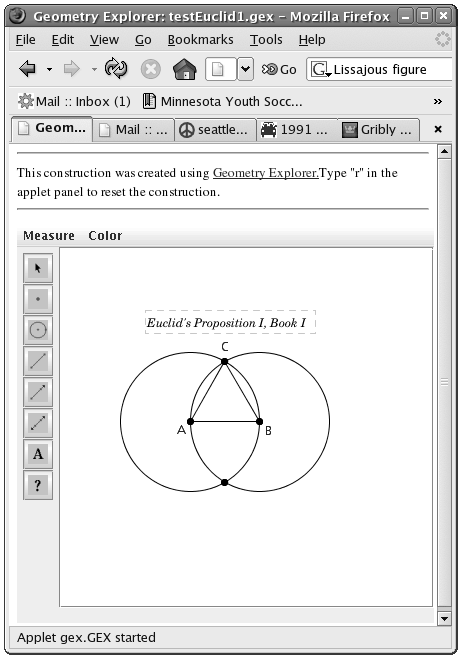
that a copy of the *Geometry Explorer* Java applet is in the same directory where the HTML file is to be stored. The *Geometry Explorer* applet is a scaled-down version of the main program that runs as a Java applet under web browsers. This applet is stored in the file Gex.jar in a directory titled "GEX" in the installation directory that was created when the *Geometry Explorer* program was first installed on your computer.

For our example we have already located the "GEX" directory and the directory does indeed have a copy of the file Gex.jar. If we save the file to this directory, then it will function correctly when opened by a web browser.

If the directory that we choose to save our file to does not contain a copy of Gex.jar, then it is necessary to make a copy of the "GEX" directory and put this copy (it must still be named "GEX") in the directory in which we are working. Then, we save our construction to the (copied) "GEX" directory.

After saving our construction to the appropriate directory, we can access the construction over the web from any Java-capable browser. For example, if we open an Internet browser and go to **Open Page** under the **File** menu in the browser window, we can open the HTML file that we just created.

Here we have opened the file under the Firefox browser. The applet will appear on the web page along with some information on how to use the applet. Note that there there is a group of eight buttons on the left side of the window. These buttons have the same functionality as the buttons in the Create Panel of the program. Also note the two menus labeled **Measure** and **Color**. These provide the functionality of the **Measure** menu and the ColorBar from *Geometry Explorer*.



All constructions, measurements, etc that one can carry out in the main *Geometry Explorer* Canvas can be saved to HTML files. Note however that

the zip file needed for running the web applet is about 1.6 megabytes in size. For those with slow Internet access (56k modems or slower) the applet may take a minute or two to start up.

The ability to store geometric constructions as HTML files and share them over the web *interactively*, and not just as static images, can be a very powerful educational tool. One can edit web pages created by *Geometry Explorer* to add explanatory text and other links. Consult a good reference on HTML for information on how to edit HTML files.

Chapter 13

Other Features

> To err is human, but to really foul things up requires a computer.
>
> —Anonymous

This chapter serves as a catch-all for various features that do not deserve a chapter all of their own.

13.1 The Edit Menu

13.1.1 Undo/Redo

Choose the **Undo** menu item from the **Edit** menu to undo the effect of the last action you performed. You may undo any creation, construction, measurement, color change, or transformation using this item. Each time you choose **Undo** another action is undone. In other words, the first time you choose **Undo** the last action you performed will be undone, the second time the second to last will be undone, and so on until you are left with a blank canvas again.

Choose **Redo** to undo the last **Undo** action. By using undo and redo one can create a geometric demonstration which can be re-wound, thus showing the steps of construction.

There is an additional menu item under the **Edit** menu labeled **Manage Undo/Redo...**. Choosing this menu item will result in a dialog box popping up listing all objects constructed since you started working with *Geometry Explorer*. By clicking on an object in the list you can immediately undo the construction back to the place where that object was created.

13.1.2 Cut/Copy/Paste

Choose **Cut** from the **Edit** menu to completely remove the currently selected geometric objects from the Canvas and place them in a memory buffer. In order to use this item, you must first select something on the Canvas. Once you cut a group of objects, you can paste the cut objects anywhere on the Canvas. Note: if the object to be cut has other objects which depend on it (e.g. a point may be an endpoint of a segment) then, the dependent objects will also be cut.

Choose **Copy** to create a second, identical version of the current selection in the Canvas. This item has no effect on the original group of selected objects. Copied objects are placed into a memory buffer and can be pasted anywhere on the Canvas as many times as you would like.

Choose **Paste** to paste the contents of the cut/copy/paste memory buffer at a slight offset from the location of the original selection that was copied. (Note: Pasting from one window to another is not available at this time)

13.1.3 Clear, Select All

Choose **Clear** to clear the screen of all constructions.

Choose **Select All** to bring up another menu of types of objects to select. These include **Points**, **Segments**, **Rays**, **Lines**, **Circles**, **Arcs**, and **Objects**. Choosing any of these sub-menu items will result in all objects of that type to be selected in the Canvas.

13.1.4 Point Size, Pen and Fill Styles

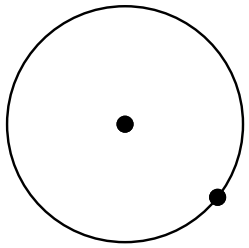
Point Size

Choose **Point Size** to change the size of the currently selected point.

Pen Styles

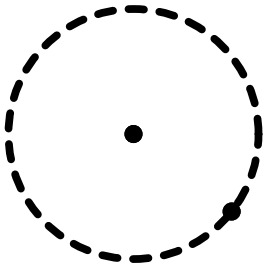
Choose **Pen Style** to change the current style of drawing of one dimensional objects such as lines, circles, etc.

For example, suppose we have drawn a circle in the Canvas and want to draw the circle with a dashed outline.

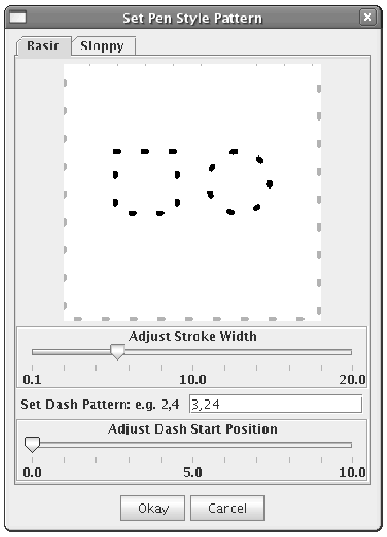


To change the current line drawing style, we choose **Pen Style** from the **Edit** menu and the submenu at the right will appear. Note the possible choices of thickness and dash styles for drawing. We will choose **Medium-Dashed**.

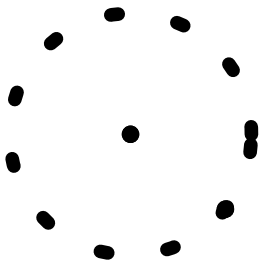
Once we choose this option the new style will be used to re-draw the circle as shown.



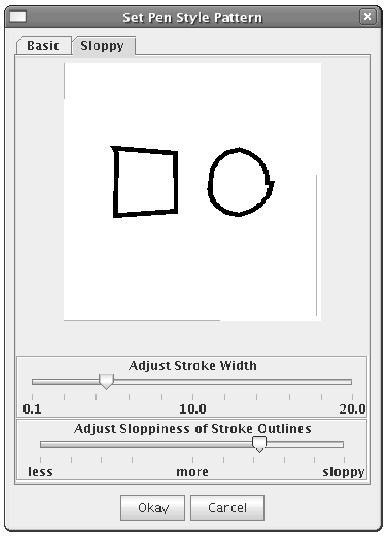
Suppose we are still not happy with the look of the circle. We can choose the submenu titled **Create/Edit Pen Style**. A dialog box will pop up as shown. Note the sliders for setting the stroke width (how wide of a line the pen makes as you draw) and where the dash patter starts. Also, note that we can specify the dash pattern. In the figure we have set the dash pattern to "3,24." This means that we have a pattern of 27 units, the first three are drawn and then the next 24 are skipped.



Let's use this pattern. Click Okay in the dialog box. The circle now looks as shown.



There is another tabbed panel in the Pen Style dialog box. If we select the circle yet once more, choose **Create/Edit Pen Style** and then click on the "Sloppy" tab we get the dialog box as shown at the right. The slider for width controls the width of the line just as with the previous example. The slider for "sloppiness" controls how sloppy the curve will be drawn. Here, sloppy means the lines don't quite connect together at the right angles, and are jaggy.



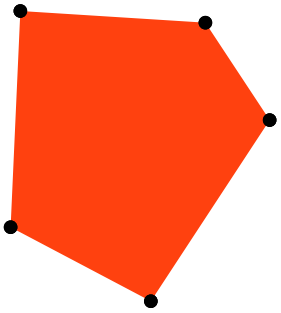
After clicking okay in the sloppy panel, we see the sloppy circle drawn on the canvas. "Sloppy" figures are also dynamically sloppy, as we move the mouse around, they shimmy and gyrate in their sloppiness.

There are two menu items under **Pen Style** labeled **Copy Pen Style** and **Paste Pen Style**. These can be used to copy an existing object's pen style and use it for another object. To do this, select and object, choose **Copy Pen Style**, select the other object and choose **Paste Pen Style**.

Fill Styles

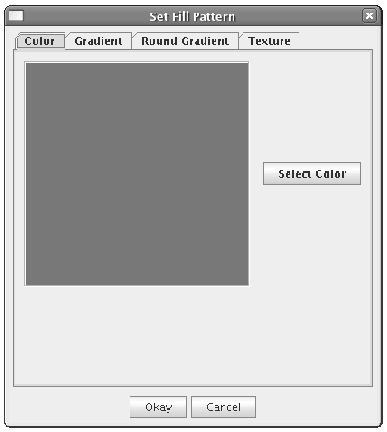
Choose **Fill Style** to change the current style of filling of areas such as polygonal and circular areas.

For example, suppose we have constructed a red-colored polygonal area in the Canvas and want to change how the area is filled in.

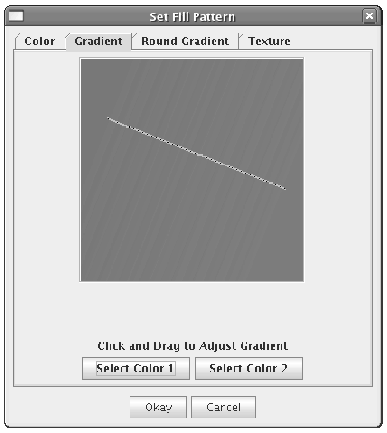


To change the current fill style, we choose **Fill Style** from the **Edit** menu and the sub-menu at the right will appear.

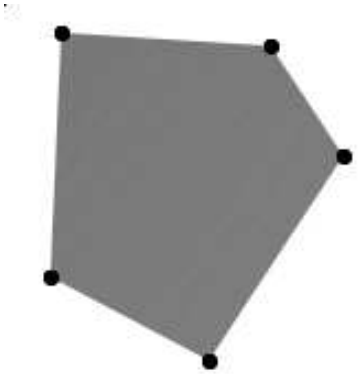
To change the fill style, we select **Create/Edit Pattern**. The dialog box shown will pop up. Note that there are four tabbed panels available – "Color", "Gradient", "Round Gradient", and "Texture." The Color panel has one button that allows you to select a solid color for the area. We will try something a little different.



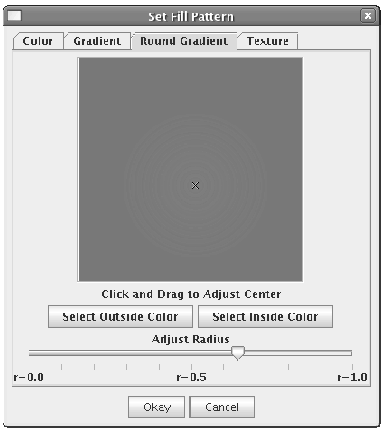
Click on the "Gradient" tab. The window now has two buttons on the bottom. Click on "Select Color 1" and choose a shade of blue. Note that the colored square in the middle of the panel now shows a gradual change in color from red to blue - a *gradient* of color. Also, note the white line in the colored square. We can click and drag to control how quickly the two colors change. Experiment with this a bit to see the effect. Then click Okay.



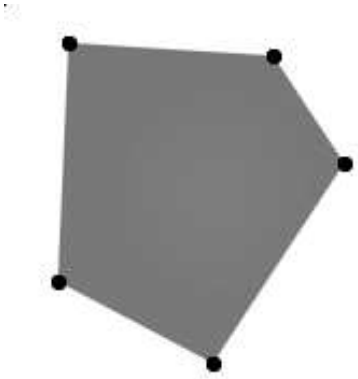
The area is now colored with the gradient fill pattern.



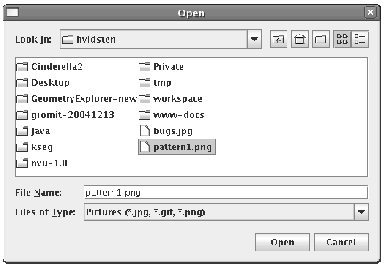
Now, let's select the area and choose **Create/Edit Pattern** again. This time click the tab labeled "Round Gradient." We again have the choice of two colors, but now the gradient is set up so that the color changes radially out from a central point. We can adjust the center of the gradient by clicking and dragging the white x in the colored area, and by adjusting the radius slider. Experiment with this to see the effect. Then click Okay.



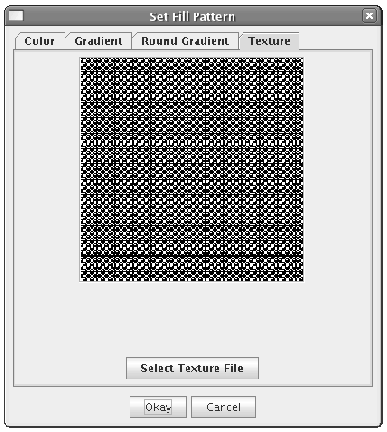
The area is now colored with the round gradient fill pattern.



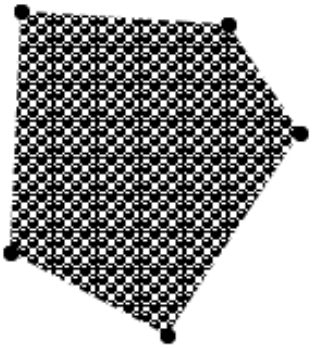
Select the area and choose **Create/Edit Pattern** one more time. This time click the tab labeled "Texture." Before the panel switches, you will see a dialog box open asking you for a file to open. This file should be either a jpeg, gif, or png image file. This image file will be used as a background to fill the area. In our case we will use a simple image called "pattern1.png." We choose this file and click Okay.



The Texture panel will then open up with this image as the background for the area in the middle of the panel. If we are not happy with this, we could click on the button labeled "Select Texture File" to get a new image. If we are happy with this image, we click Okay.



The area is now filled with a fill pattern based on the image.



There are two other menu items under **Fill Style** labeled **Copy Pattern** and **Paste Pattern**. These can be used to copy an existing object's fill style and use it for another object. To do this, select an object, choose **Copy Pattern**, select the other object and choose **Paste Pattern**.

13.1.5 Properties

Use the **Properties** menu item from the **Edit** menu to pop up a dialog box that allows one to get information about an object and to set certain properties of an object. Information includes a description of the object, the parents of the object (other objects that the given object depends on), and children of the object. Properties which can be set include visibility of the object, whether the object is selectable, the label of the object, the color of the object, and other properties which may be specific to the particular object. To enable the **Properties** menu, and object must be selected.

The menu item labeled **Manage Properties...** can be used to find a

particular object for properties management. Choosing this menu item will result in a dialog box popping up which lists all objects. After selecting an object and hitting Okay, the Properties dialog box will pop up for that object.

13.1.6 Setting User Preferences

Use the **Preferences...** menu item from the **Edit** menu to pop up a dialog box that allows one to change user preferences for the program. These preferences include the level of precision for measurements, whether to show labels or not, what the default size of points will be, what type of Canvas (Euclidean, Hyperbolic, or Elliptic) the program uses by default, and other preferences.

13.2 The View Menu

The items in the **View** menu deal with the appearance of objects on the screen and also with other windows and visual items that one can access from the main Explorer window. The items in this menu are grouped in five categories: Helper Windows, Hiding and Showing, Tracing, Animation, and Miscellaneous.

13.2.1 Helper Windows

1. **Notebook...** Clicking on this menu item will pop up a window into which one can put textual information that might not fit into a text box within the geometry canvas of the main window. For example, several pages of background information could accompany a demonstration on the Pythagorean Theorem. Note: This is a text-only window. Graphics can not be inserted along with the text. If you want to have a richer document accompany your geometry construction, create a web page and link it into your geometry canvas using a web link as described in the chapter on using the web in *Geometry Explorer*.

2. **Calculator...** Clicking on this menu item will pop up a calculator for creating complex expressions involving geometric measurements (as well as doing basic arithmetic). See the chapter on Measurements and Calculations for more information.

3. **Font...** Clicking on this menu item will pop up a dialog box that allows one to select a custom font for a given selected textual object

(label or text area in the Canvas).

13.2.2 Hiding and Showing

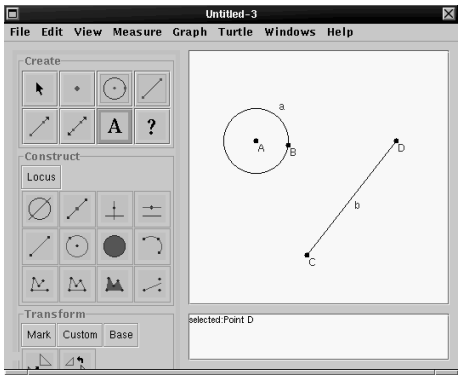
1. **Hide Object** Once you have selected an object or a group of objects, you can hide the objects by choosing this menu item. The objects are still effected by any changes made on the Canvas, but they are just not visible.

2. **Hide All** Use this option to hide all objects of a particular type. When we click on this item a sub-menu will appear with sub-menus **Points**, **Segments**, **Rays**, **Lines**, **Circles** and **Arcs**. After choosing one of these options, all of the objects of that type will be hidden.

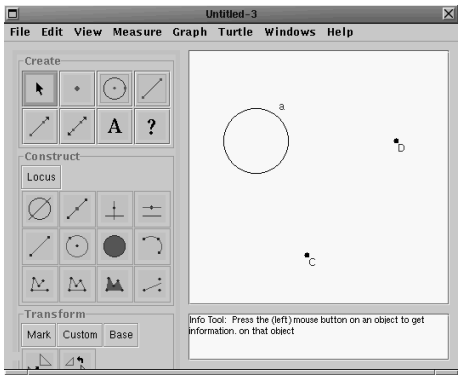
3. **Show All Hidden Objects** Choose this item to make all hidden objects visible on the Canvas.

4. **Show Hidden Object...** Use this option to find a specific object that has been hidden. A list of all hidden objects will be displayed and you can choose an object to show. Note that objects will be listed in the order that they were created.

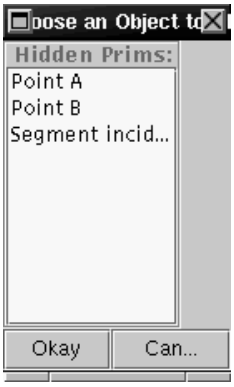
For example, in the figure at the right we have created a circle and a segment.



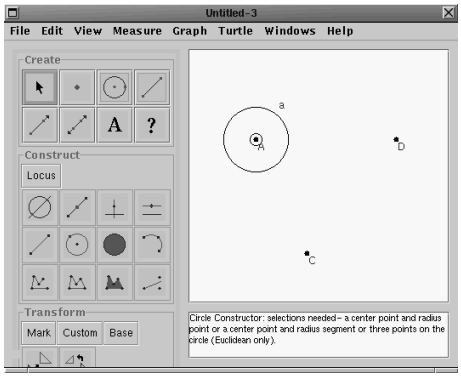
Now, suppose that we hide points A and B and segment b.



If we now select **Show Hidden Object...** from the **View** menu the dialog box shown at the right will pop up.



By clicking on entries in the list we can show objects in the Canvas. If we click on "Point A" and hit Okay point A will now be visible.



5. **Hide All Labels** Use this option to hide all labels on the Canvas.

6. **Show All Labels** Use this option to show the labels for all objects on the Canvas.

13.2.3 Tracing Objects

There are three menu items which control the tracing of objects: **Trace On**, **Trace Off**, and **Clear All Traces**.

The **Trace On** item will be active only when a *traceable* object has been selected. Traceable objects include all of the curvilinear objects (lines, segments, rays, arcs, circles) and points. After choosing **Trace On**, the object will leave a dim trace of itself as it moves.

If we select an object that is currently being traced then the **Trace Off** menu item will become active and we can then turn the trace of that object off if we wish. (Note: Use the Info tool to find out if an object is being traced or not)

To stop the tracing of *all* currently traced objects use the **Clear All Traces** option.

13.2.4 Animation

A very nice way to visualize certain geometric objects is to animate their construction. *Geometry Explorer* has the capability of animating most any object that has freedom of movement on the screen, including points, lines, segments, rays, circles, arcs, areas, and even parameters. For more information on the use of animation and the associated menu items under the **View** menu, consult the chapter on animation.

13.2.5 Miscellaneous View Options

Use the menu item **Image...** to import an image into the Canvas. Choosing this menu option will cause a file dialog box to pop up. Locate the image file desired and then hit the Okay button. Many image file formats are currently supported in *Geometry Explorer*. These include popular formats such as GIF, JPEG, EPS, PCX, PNG, and TGA formats. The image will be rendered on the screen with the upper left corner of the image positioned at the selected point. If the point is subsequently transformed the image will also be transformed.

Use the menu item **Set Web Link...** to create a web link on the Canvas. Refer to Chapter 12 for more information on this capability.

13.2.6 Zooming and Panning the Canvas

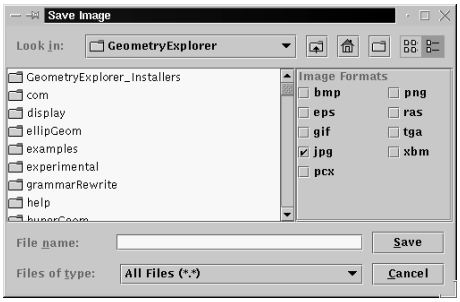
We can rescale the Canvas through the use of the keyboard. To zoom in on the Canvas, use the comma key, and to zoom out use the period key. These

keys were chosen because the "<" and ">" characters are also on these keys, signifying zooming in and zooming out.

We can also move the Canvas up/down and left/right by using the four arrow keys on the keyboard.

13.3 Saving The Canvas as an Image

Use the menu item **Save As Image...** from the **File** menu to save the current construction in the Canvas as an image file. Many popular image file formats are supported. When choosing to save the current construction as an image, the dialog box on the right will pop-up. One must select the image format and then type in the file name to save the image.



One can also save just a portion of the Canvas as an image. Hold down the Shift and Control keys and select a portion of the screen. Then, choose **Save As Image...** from the **File** menu as above.

13.4 Control Buttons

There are several types of on-screen buttons that can be added to the canvas to control actions such as hiding/showing objects and starting/stopping animations. The creation of these buttons is done via the **Misc** menu in the main program window. Under this menu is a sub-menu titled **Control Buttons**. If we choose this sub-menu a list of five items will pop up corresponding to the five types of buttons that we can create. Once a control button is created, we can make it carry out its function by clicking on the button. The five types of buttons are as follows:

1. **Show/Hide Objects** To create a button to control the visibility of objects, first select an object, or a group of objects, and then choose this option under the **Control Buttons** menu. A button will be created in the upper left corner of the canvas. This button will control the visibility of the selected objects. Each time the button is clicked

the object's state of visibility will be changed – from visible to invisible, etc.

2. **Move Points** To create this type of button, first select points in groups of two and then choose this menu item. A button labeled "Move" will be created in the canvas. When this button is clicked each first point in the groups of two will move towards the second point until they are coincident.

3. **Animate Objects** To create a button to animate an object first select the object to be animated. Then choose this menu option. A dialog box will pop-up asking for the choice of certain animation properties for the object. Once these have been chosen, a button labeled "Animate" will be created in the canvas. When this button is clicked the object will begin moving. To stop the animation click on the button a second time.

4. **Iterate Function** To create a button to iterate a point on a function, first select the graph of a function that has been defined and choose this menu option. A button labeled "Iterate" will be created in the canvas. In order to use this button we first select a point that is attached to the graph. Then, when this button is clicked the function will be iterated on this point. (For more information on iterated functions review this concept in Chapter 6)

5. **Sequence Buttons** To create a button to carry out a sequence of already defined button actions, first select the series of buttons and choose this menu option. A button labeled "Sequence" will be created in the canvas. When this button is clicked each button in the set of buttons will be activated in turn.

Note that a control button can be moved by clicking and dragging the thin gray area that surrounds the button. Also, the title of the button can be changed by changing the button's label using the Properties dialog.

13.5 The Info Tool

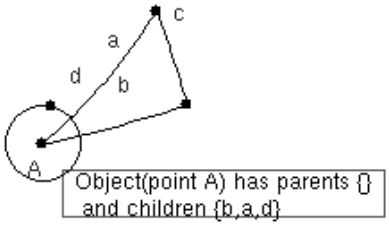
The Info tool is the button with the question mark in the Create Panel in the *Geometry Explorer* window. To use this tool, click on the question mark and then click on an object in the Canvas.

The Info tool is used to get information about the current state of an object. For example, it tells you the type of object, its label, and what its parents and children are. It also tells you if the object is being traced and if it is being animated.

13.5.1 Parents and Children

An object on the screen can have dependencies on other objects. For example, when we draw a line segment, we first plot a point and then drag the mouse to define the other endpoint. The segment depends on these two endpoints; we say that the segment is a *child* of the endpoints and the endpoints are *parents* of the segment. Objects on the screen can be placed in a hierarchy of relatedness, with points generally being at the top of the hierarchy. Points generally have no parents, except when they are midpoints or other constructed points. The Info tool can be used to get an idea of where an object fits within this hierarchy.

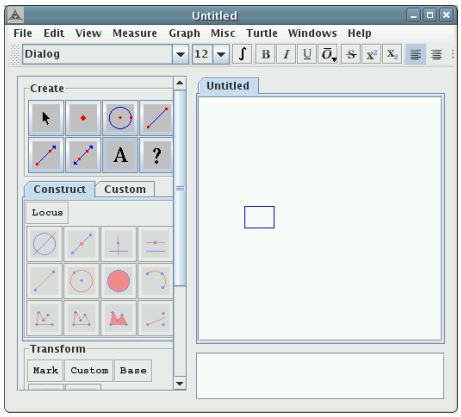
For example, in the figure at the right we have clicked on the Info tool in the Create panel and then clicked on point A. A small text box outlined in red pops up giving information about point A. Since A is not dependent on any other object it has no parents. This is specified by the empty brackets . However, it does have children which depend on it, segments a and b and circle d.



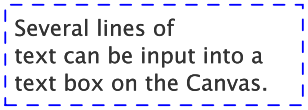
13.6 Editing Text Areas in *Geometry Explorer*

Text areas can be created within the Canvas by using the Text Tool in the Construct Panel (the one with an “A”).

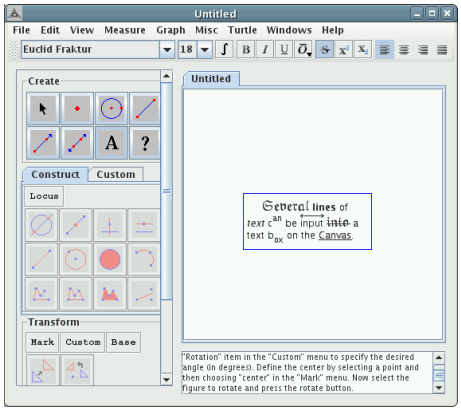
For example, here we have clicked on the TextTool and then clicked the mouse in the Canvas. A text area appears with a blue box surrounding it. Also note the Text Tool Bar that is now visible above the Canvas.



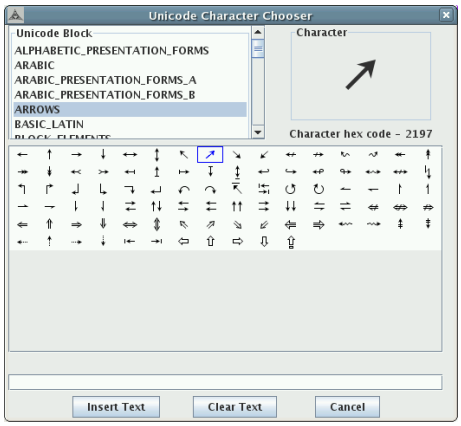
We can now insert text into the text area by clicking the mouse inside the box and typing on the keyboard.



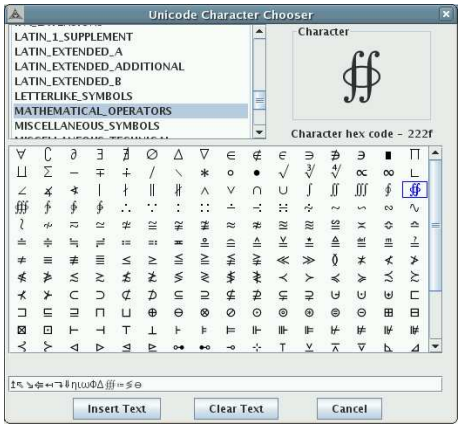
By using the Tool Bar we can change the style and justification of the text in the text areas. Here is an example of some of the styles.



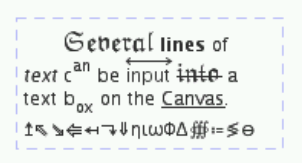
Note the button in the Tool Bar that looks like an integral sign (or squiggly "S"). This is the "Special Character" button. By clicking on this button, we can access all Unicode characters that are available with a given font. For example, if we select the Dialog font and click the Special Character button, we will see the dialog box at the right.



From this dialog box we can select a wide variety of symbols and foreign characters. By clicking on a character, we can add it to the textfield at the bottom of the dialog box. Here we have added characters from the Arrows, Greek, and Mathematical Operators groups of characters.



Once we click the button labeled "Insert Text" the special characters will be added to the text area at the text insertion position.



There are many other text areas used in *Geometry Explorer*. For example, in the Calculator there is a text area that shows the current mathematical expression and another text area showing results of evaluating the current expression. In the Notebook, there is a text area for additional comments and information that one may want to include with a geometric construction. In the Turtle Controller window there are several text areas that deal with defining and using grammars to control a turtle.

In most cases one can cut/copy/paste from and to these text areas by using the **Edit** menu in the window that contains these areas. In all cases,

the structure for cut/copy/paste, including keyboard equivalents, is the same from window to window within *Geometry Explorer*. The only exception to this is that one cannot cut or paste in a text area that is not editable, for example in the Turtle Controller window the text area that holds the result of re-writing a particular axiom is not editable by the user and thus one cannot cut or paste in that area, although one can copy from it.

When one cuts or copies text from a text area, *Geometry Explorer* places the text on the user's computer-wide clipboard. Thus, text that is copied in *Geometry Explorer* can be pasted into a user's word processor or spreadsheet program or any other program that allows for cut/copy/paste of text. Likewise, text copied from other programs can be pasted into text areas in *Geometry Explorer*.

Bibliography

[1] Michael Barnsley. *Fractals Everywhere*. Academic Press, San Diego, 1988.

[2] Marvin Jay Greenberg. *Euclidean and Non-Euclidean Geometries*. W. H. Freeman and Company, New York, 1980.

[3] David Henderson. *Experiencing Geometry of Plane and Sphere*. Prentice Hall, Upper Saddle River, New Jersey, 1996.

[4] Michael Henle. *Modern Geometries*. Prentice-Hall, Upper Saddle River, New Jersey, 1997.

[5] H. E. Huntley. *The Divine Proportion: A Study in Mathematical Beauty*. Dover Books, 1970.

[6] Michael Hvidsten. *Geometry with Geometry Explorer*. McGraw-Hill, Boston, 2004.

[7] Astrid Lindenmayer and Przemyslaw Prusinkiewicz. *The Algorithmic Beauty of Plants*. Springer Verlag Book, New York, New York, 1990.

[8] Richard S. Millman and George D. Parker. *Elements of Differential Geometry*. Prentice Hall, Englewood Cliffs, New Jersey, 1977.

[9] Seymour Papert. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Book Inc., New York, New York, 1980.

[10] Doris Schattschneider. *M. C. Escher, Visions of Symmetry*. W. H. Freeman and Company, New York, New York, 1990.

[11] James R. Smart. *Modern Geometries*. Brooks Cole Publishing Company, Pacific Grove, California, 1988.

[12] Ian Stewart. *Does God Play Dice: The Mathematics of Chaos.* Basil Blackwell, Inc., New York, New York, 1989.

[13] Harold E. Wolfe. *Non-Euclidean Geometry.* Henry Holt and Company, New York, 1945.

Index