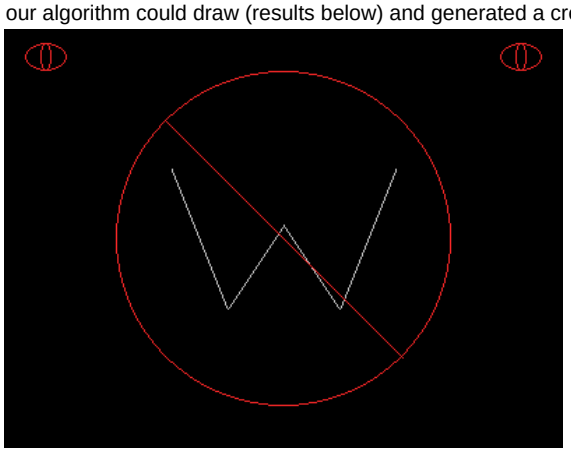
Russel_Lowenstein Assignment 3

CS 351 Assignment 3: Graphics Primitives

Justin Russell & Adam Lowenstein

Abstract

The goal of this assignment was to create four graphics primitives that will be used throughout the rest of the course as a means of generating new images: points, lines, circles, and ellipses. These primitives were based on and tested with the Image definitions created in the previous assignment. After creating each primitive, we tested our code with a provided source file and generated the red and black image seen below. We also calculated how many lines per second our algorithm could draw (results below) and generated a creative picture.



Description

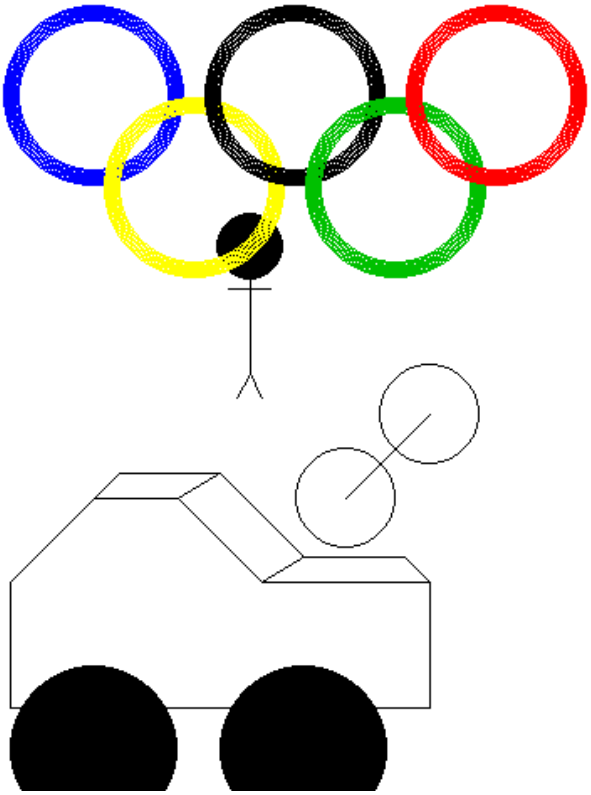
The point was the simplest to implement, because it only required a length-four array to hold location data. Lines were implemented using the Bresenham algorithm, which looked at the slope of the line relative to a test point in order to determine which pixels were "selected" in creating the line. With regard to circles and ellipses, we created functions to outline each shape using the algorithms provided in the class notes. At the end of the assignment, we generated a 3D version of a car crashing into an unlucky ~~Justin Russell~~ bike racer at the Olympics (see below for the image).

Algorithms & Pictures

Using the Bersenham algorithm repeatedly to generate a number of lines, we created the image to the right early on in the development of our code. It tested vertical, horizontal, and "regular" lines in all sections of the graphic plane. It may not look like much, but it was a monumental achievement at the time of its completion. We also calculated the number of lines that our algorithm could draw using a test program that was provided in the assignment. Our test resulted in an average line length of 103.9 pixels, with lines drawn at a rate of 328340 lines per second. The image to the left was generated (very efficiently) with this testing algorithm.

The circles and ellipses were created using provided algorithms that were modified to adapt a standard coordinate plane (where the center of the canvas is (0,0) and points are listed (x,y)) to the computer plane (where the top lefthand corner is (0,0) and points are listed (rows, cols) or (y, x)). We were able to fill these shapes by generating vertical lines of pixel-width 1 between every upper and lower boundary of the shape. These lines, when put together, created a "paint-bucket" effect on a hollow outline. The ellipse was somewhat trickier to fill than the circle--to fill the ellipse, we had to combine a partial vertical fill with some horizontal fill as well.

The image below is our 3D Olympic scene/tragedy:



Conclusion

After struggling through the creation of these graphic primitives, we now have a good understanding of the vast amount of work that is required to produce even a very simple shape. Points and lines are the most fundamental parts of any image, but that does not mean that creating and manipulating them is a trivial matter. However, now that we've attained this experience, we feel prepared to begin developing more complex images from the ground up.
