Scott Gordon

4/25/2021

COMP 5460

Final Project Report

The purpose of this paper is to show the issues faced, lessons learned, and any remaining bugs in my final project, as well as list any extra effort that was put into the assignment past the requirements.

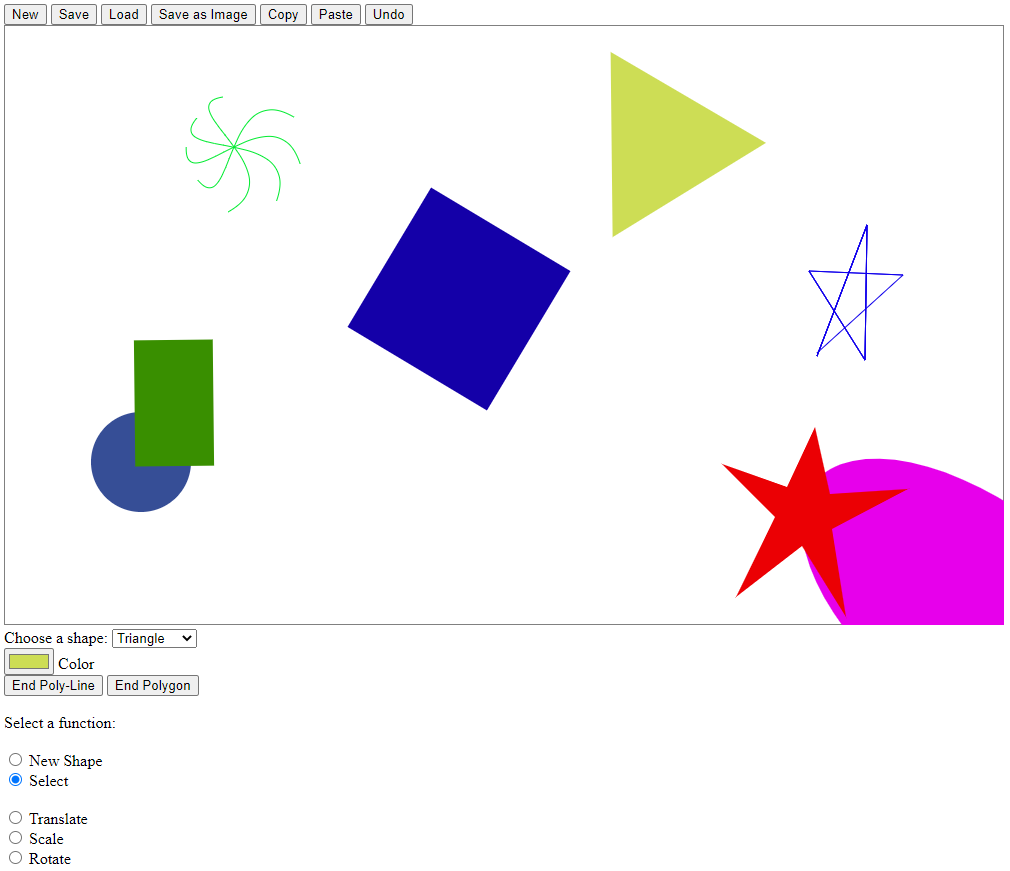
First, how my program functions from the user’s perspective. The user is shown an empty canvas, a menu of buttons above, and some options below. The menu above performs the actions listed on the buttons, the items below the canvas in order of appearance are to select a shape type, choose a color, complete a polyline or polygon, or select a function. The only functionality not listed is the ability to use control+c, control+v, and control+z for the functions copy, paste, and undo, respectively. When ‘New Shape’ is selected, the user can click on the canvas to create the shape with its center at the point of click. For curve, the user must click twice to give the line endpoints, then the next two clicks define the way the curve curves. For polyline and polygon, the user clicks to create lines and when they are done, they can click end polyline or end polygon to create the shape, for polygon the end is connected to the beginning to close off the shape. For each of the transformations, the user can just drag their mouse right to increase the scale or rotation or left to decrease. Translation is directly tied to mouse movement when holding down the mouse button.

To get started on this assignment, I started with my assignment three code, and adapted it to work for different shapes. I also took some of the code from my assignment five code to have the curve functioning. I knew I would need data structures later to perform different transformations on different objects while keeping track of everything. I created the ‘shapes’ array that would hold the matrix of points to define each shape, the ‘centers’ array that was to hold the center points for each shape, and a colors array.

When defining these arrays and trying to make the transformation work is when I ran into the biggest problem, I needed to create separate arrays for curves and polylines since they needed to be treated differently from the others. I could have used an object-oriented approach but the approach I took came to me first and I got it to work properly. The downside is that I must check multiple times in my code whether I am dealing with a polyline, curve, or in some cases an ellipse/circle. I can do this because the way I chose to define these objects, I can test for it. For example, a polyline is defined so that the ‘shapes’ array only contains a single pair for that shape, and it is [0, 0]. A curve is the same but contains [0, 1]. The only other shapes that only have one point in their ‘shapes’ representation are circles and ellipses. I can differentiate these two apart from the others because a circle should have [x, 0] where x is something other than 0 because it will be the width. An ellipse is [x, y] where x and y are the width and height, respectively. From this I can determine the appropriate function calls when preforming a transformation on any object in the ‘shapes’ array.

I obviously cannot talk about every function and every shape in this paper but one other problem I ran into was needing to create a special case for rotating an ellipse. For circle I just ignored rotation because the circle looks the same with any rotation, but the ellipse requires a special case. I created a map, to map shape indices to rotation values, that way if I identify a shape as an ellipse before rotating, I can instead use the function call to rotate it.

For extra credit in this project, I utilized rubberbanding on each of the functions, and they work no matter the shape. I also created a feature that highlights the currently selected shape so that the user knows what shape will be transformed. The last bit of extra credit I went for was to add color. I originally thought that color was required because it makes sense for a shape editor, it was surprising to me to find out that it is considered extra credit, but I implemented it anyway.

There are no known bugs remaining in my program, everything works as intended. For lessons learned, I regret not taking more time before starting the project to consider my approach. I believe an object-oriented approach would have saved me some time in figuring out confusing solutions, specifically ones revolving around the curves and polylines. This was a very interesting a fun project to work on. An image of sample output is placed below to show the program functioning. I also attached the image output of my program of the same artboard.

A picture containing text, businesscard, vector graphics

Description automatically generated