Benjamin Northrop
6/6/24
CS5310
Summer 24

Introduction

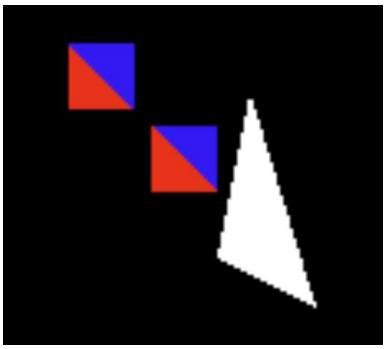
Homework 4 specialized on drawing and filling polygons. We had 2 primary means – a scanline fill polygon that used an active edge structure to determine if the scan line pixels were inside or outside the polygon. We also used barycentric triangles to draw triangles in order to quickly draw shapes and open the possibility of parallelization.

Projects

Test4a

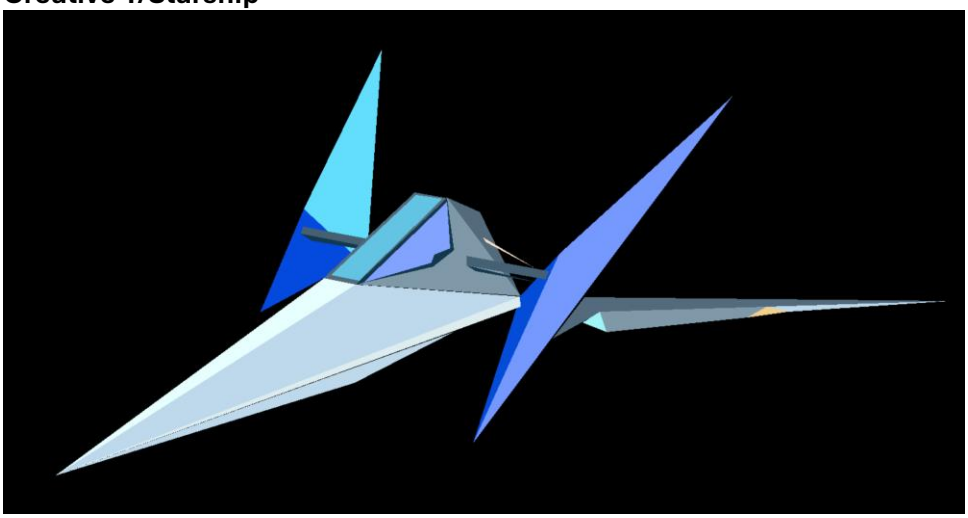
To complete the polygon fill program, I focused primarily on following the pseudocode closely. I then started with the values and formulas to ensure they worked correctly. After discussing with the TAs, I learned a great deal about the srand() and rand() functions in C, and how they can vary depending on the library in use on each computer.

Test4b



The barycentric triangle version at first took me a while to conceptualize, as it didn't click that the polygon length for these triangles *must* be 3. From there, I had drawn each triangle but there were a few pixels missing in each red triangle. Comparing the lecture notes to the code, I realized that the triangles all needed to be drawn Counter Clockwise. Research showed that the shoestring method of calculating the area of a triangle would be negative if done counterclockwise, and positive if done clockwise, so by doing this check I was able to manipulate the vertices so every polygon was drawn consistently.

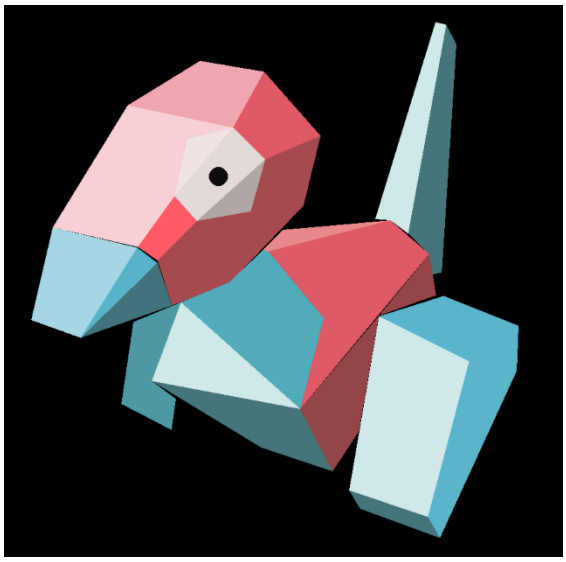
Creative 1/Starship



For the first creative picture, I decided to go back to a childhood game that I remember was full of polygons. Starfox 64's starship here is built entirely with barycentric triangles. I downloaded the source image and loaded it to gimp, then plotted each point of each vertex into excel. I then sampled the color for each triangle. I organized the vertices so that I could reuse shared sides as often as possible to reduce the overall size of the points array. Once all of the points and colors were plotted, I concatenated all the points to import into the code, and consolidated the colors into a single array, then used a loop to initialize and assign the points in the array.

Point	X	Y	r	g	b	rf	gf	bf	Note	Pts	Color	Color Index				
0	825	590	230	255	255	0.90	1.00	1.00	White top	0-2	White1	15	point_set(8	825 ,	590)	
1	401	901														
2	875	600											{825, 401, 875, 840, 724, 824, 793, 868, 895, 79			
3	840	623								1-3			{590, 901, 600, 623, 644, 590, 481, 550, 525, 48			
4	724	644	4	74	221	0.02	0.29	0.87	Left dark blue	4-6	DarkBlue1	1				
5	824	590								5-7						
6	793	481	100	221	252	0.39	0.87	0.99	Left light blue	6-8	LightBlue1	6				
7	868	550														
8	895	525											X array	{825, 401, 875, 840, 724, 824, 793, 8		
9	793	481	100	221	252	0.39	0.87	0.99	Left light blue	9-11	LightBlue1	6	Y array	{590, 901, 600, 623, 644, 590, 481, 5		
10	895	525														
11	916	230											Colors	White1, DarkBlue1, LightBlue1, LightI		
12	805	531	24	43	73	0.09	0.17	0.29		12-14	DarkGray1	4	Red Values	{0.9, 0.27, 0.39, 0.39, 0.09, 0.27, 0.3		
13	875	543							upper left cross	13-15			Green Valu	{1, 0.84, 0.87, 0.87, 0.17, 0.38, 0.42,		
14	788	520	69	98	116	0.27	0.38	0.45		14-16	MidGray1	10	Blue Value:	{1, 1, 0.99, 0.99, 0.29, 0.45, 0.49, 0.2		
15	880	538								15-17						

From there, it was simply a task of finding the correct start and stop polygons for each triangle.



Creative 2: Porygon

Similar to the starfox ship, I also did one that was almost all polygons. In a similar manner, I plotted each vertex and recorded the vertex and color into excel, then organized it all together in the same manner as before. This time, the challenge was also keeping track of how many points were in each polygon as I wrote them to the image.

Extensions

In this project, I did not do anything above and beyond with the programs themselves. Instead, I had a

lot of fun putting some major effort into how to organize vertices and store them for drawing later. Each image contained over 75 vertices and 15 colors, and required a deliberate process to ensure each polygon was being drawn appropriately.

Reflection

In both cases, there was a chance to develop a tool that would take a csv file of points and read them into a data structure, we could then also read in a separate csv file with the color values. With these two, and a third csv that had the start position and size, you could essentially export this image out into 3 csv files with vertices, colors, and the drawing instructions. Overall, I spent about 6 hours on these two images to get them as realistic and accurate as I could, so the csvIO program will need to wait until I have a little more bandwidth.

Acknowledgements

The skeleton code for the draw_polygonFill program was provided by the instructor. The images were retrieved from Google images in order to determine the vertex locations.
Starfox Ship: 63ba876456b94e119b08b5cf04cf585a.jpeg (1920×1080) (sketchfab.com)
Porygon:
https://images.wikidexcdn.net/mwuploads/wikidex/2/2d/latest/20190407231904/Porygon.png