Name: Nicholas Ung
NUID: 002336960

CS 5310
Summer 2024

Lab 4 Report: Scanline Fill

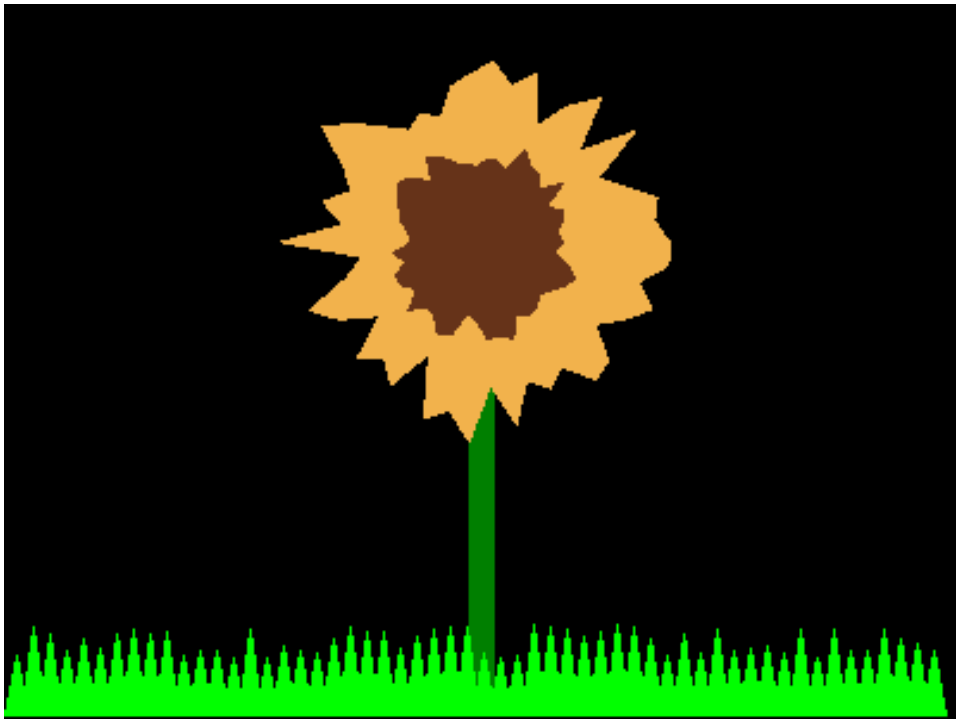
Summary

This project focused on implementing essential polygon functionalities and fill algorithms, which are pivotal for rendering systems. Primary objectives included developing a **polygon API**, implementing the **scanline fill algorithm** for general polygons, and creating the **barycentric fill algorithm** specifically for triangles. The outcomes are demonstrated through a series of required and creative images, showcasing the effectiveness and functionality of the developed algorithms.

Required Images

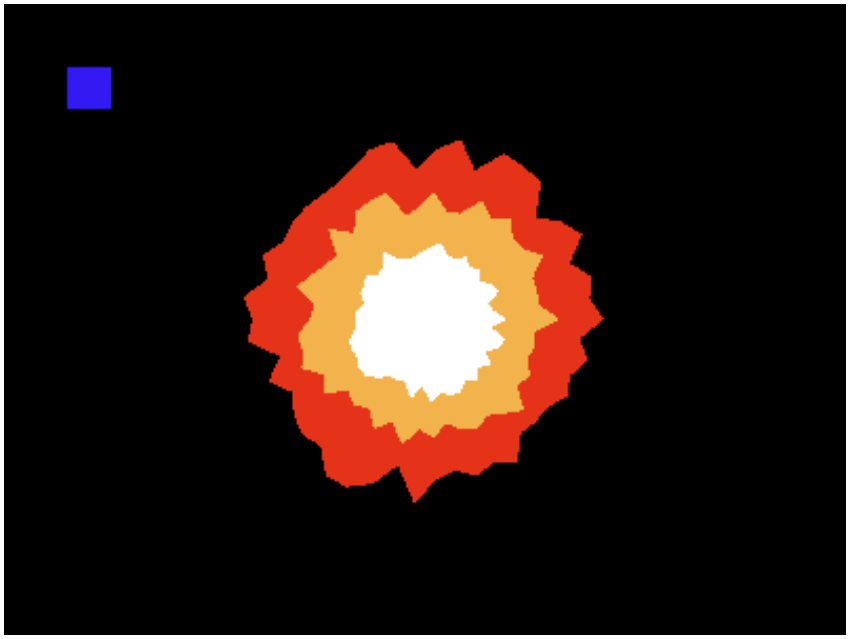
1. ***Creative Image - Sunflower in a Field of Grass***

   Description: This image depicts a lone sunflower in a field of grass. The flower itself is made of 2 polygons and a stem colored with the scanline fill algorithm. The algorithm iterates over each scanline, determining the intersections with the polygon edges and filling the pixels between these intersections to provide consistent coloring. Additionally the grass was made of triangles colored using the barycentric fill algorithm.



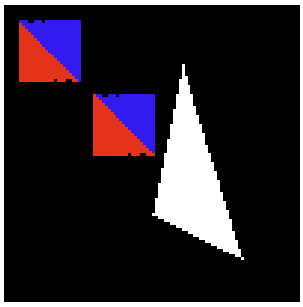
2. *Test Image B*

Description: Generated using ***test4a.c***, this image illustrates the scanline fill algorithm. The algorithm scans each line of the polygon, identifies intersections with polygon edges, and fills the span between these intersections. The result is a filled polygon without any outline, demonstrating the proper implementation of the scanline fill method.



3. *Test Image C*

Description: This image is produced using ***test4b.c*** and showcases the barycentric fill algorithm for a triangle. By scanning the bounding box of the triangle and calculating barycentric coordinates for each pixel, the algorithm determines which pixels lie within the triangle and fills them accordingly. The implementation takes into account the center of each pixel, ensuring accurate filling based on barycentric coordinates.

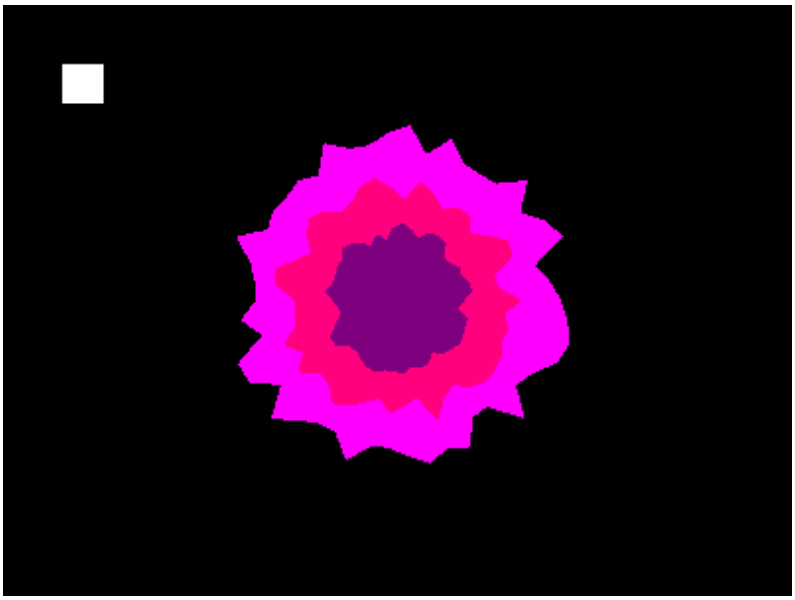


Name: Nicholas Ung
NUID: 002336960

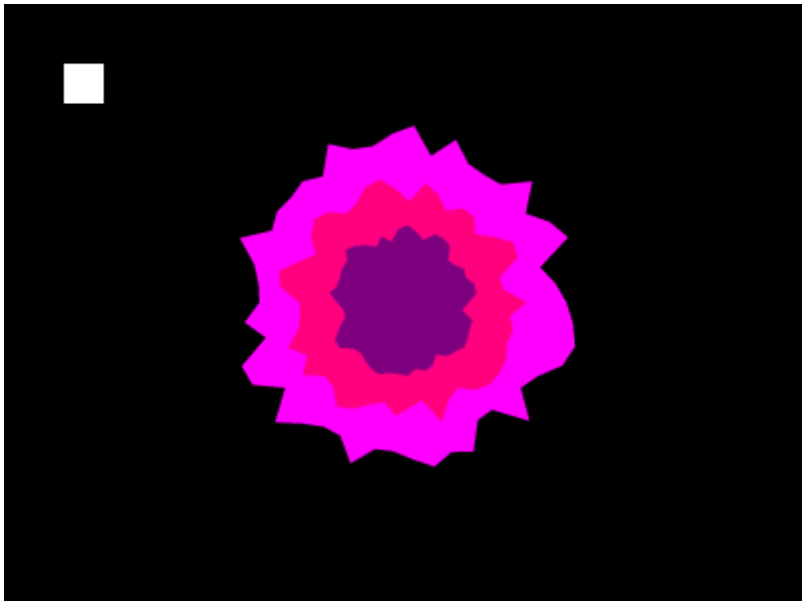
Extensions

1. *Anti-Aliased Polygons*

Description: I implemented the anti-aliasing extension to smooth my polygon edges by writing a function to blend the edge pixels with their surroundings. This reduces the visual jaggedness and creates a more polished appearance for rendered polygons. You can see noticeable visual differences between the first image (without anti-aliasing) and the second image (with anti-aliasing) thanks to the smoothing effect.



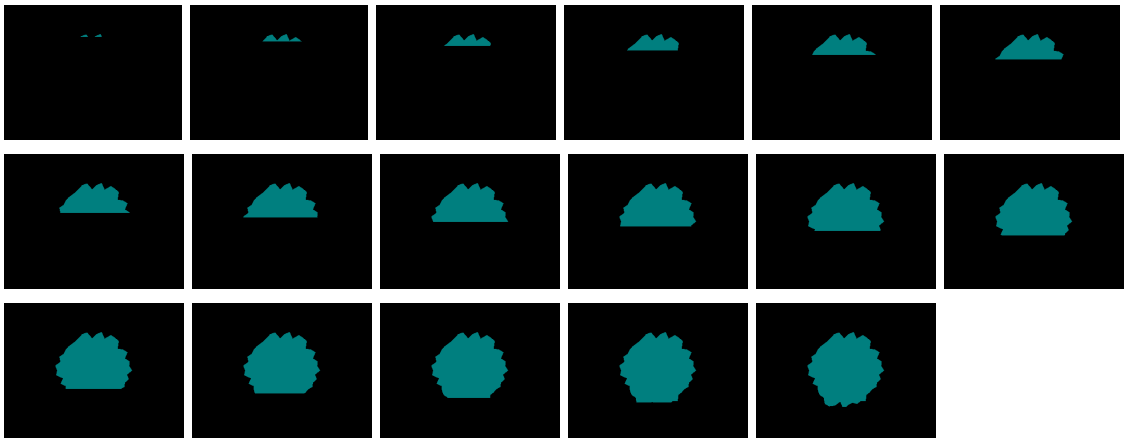
Without anti-aliasing (jagged edges)



With anti-aliasing (smooth edges)

2. *Scanline Fill Algorithm Demonstration*

Description: For my second extension, I created an animated GIF tutorial to demonstrate the scanline fill algorithm in a step-by-step visual format. It provides a clear understanding of the algorithm's process, illustrating how it constructs one layer at a time to complete the polygon. I accomplished this by implementing a function to record frames while the algorithm colors in the polygon line by line.

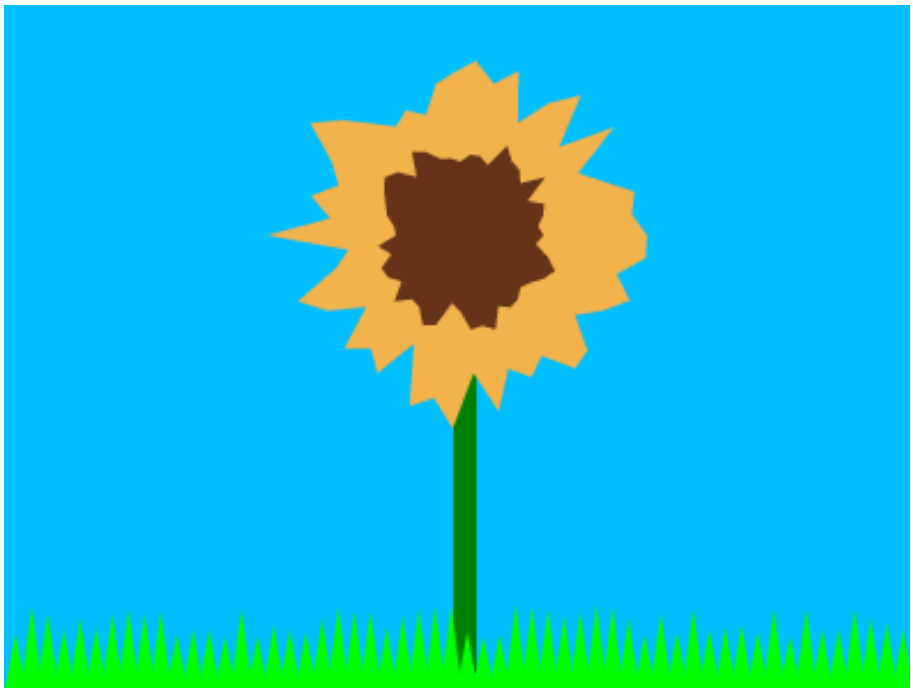


Name: Nicholas Ung
NUID: 002336960

Portfolio Images

1. ***Sunflower with Anti-Aliasing and Blue Sky***

   Description: I used the ***image_fillrgb*** method from the previous lab to change the background to sky blue, then redrew my sunflower image and applied my anti-aliasing extension to smooth out the edges. The result was a much smoother and more natural looking sunflower in nature.



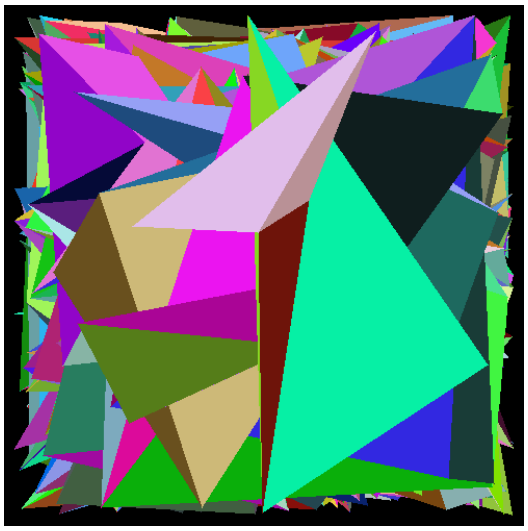
2. ***Remy from Ratatouille next to the Sunflower***

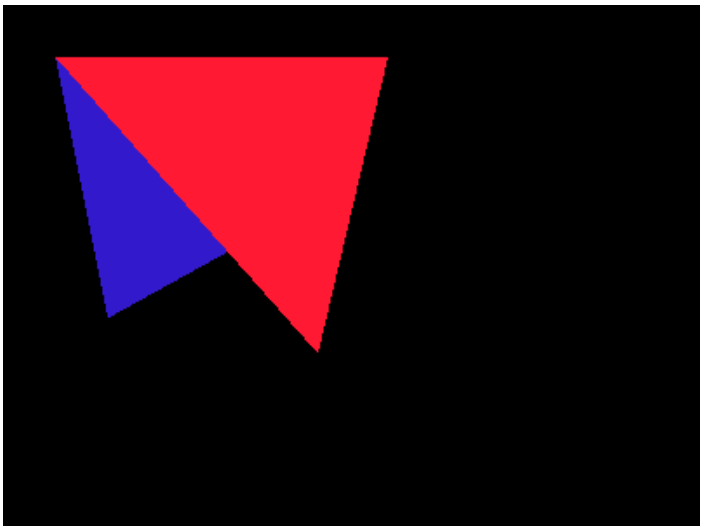
Description: For my second creative image, I went all the way back to the first lab and used the mask-creation and green screen technique to place Remy from Ratatouille next to the sunflower as a frame of reference for how big it is. The addition of the cartoon rat adds some more character and life to the scene.

Name: Nicholas Ung
NUID: 002336960

Other Images

I generated the images below by running ***test4c.c*** and ***polyspeed.c***. The terminal output from test C matched the expected results in ***polystress.txt*** as well.





Reflection

Through this project, I gained a deeper understanding of polygon rendering and filling techniques. Implementing the scanline fill algorithm highlighted the importance of handling edge cases and managing memory efficiently. The barycentric fill algorithm taught me about coordinate transformations and their applications in graphics rendering. Additionally, the extensions provided insight into enhancing basic algorithms to support advanced graphical effects like anti-aliasing and gave me a visual understanding of how the scanline algorithm works. Overall, this project was instrumental in building foundational knowledge for graphics programming and rendering systems.

Acknowledgements

I would like to recognize the following for helping me complete this assignment:

- **Instructor and Course Material:** Professor Maxwell's lecture notes and videos provided me with guidance and reference materials for implementing the algorithms.
- **Classmates:** Jiafang Du discussed conceptual challenges with me and provided feedback on my implementation approach.
- **Online Resources:** Various online tutorials and articles from sites like W3schools on scanline and barycentric algorithms contributed to my understanding of the concepts.