Morphological Antialiasing

As I began researching morphological antialiasing (MLAA), the first thing I set out to accomplish was deciding what language to write the algorithm in. I contemplated using C or C++, mainly because I thought it would be interesting, but ultimately went with java. Java has some incredible pixel reading capabilities that are easy to implement through java.awt. After this, I then began to prioritize finding an example, or a tutorial, of all the patterns that I would need to implement. I browsed many different articles and GitHub repositories, but ultimately, it was the original paper by Intel Labs that I ended up relying on. They reference three different categories of shapes: “L” shapes, “Z” shapes, and “U” shapes. They also mention, however, that any “Z” and “U” shape could be made by combining two “L” shapes. Thus, my journey was to implement the four different “L” shapes.

One interesting thing that I ran into was the problem of reading colors that had already previously been changed. This led to some confusion, as well as my pictures turning entirely one color. I resolved this by first finding all the patterns, and then blending the new pixels all at once. After completing my algorithm, for some reason, some lines didn’t look like they were being affected. This was because I misinterpreted the paper from Intel Labs, and in a section where they listed the “L” patterns they were using, I assumed those were all of them. I was incorrect, there are actually eight “L” patterns. The last main issue I had was getting the 16 different parts of my algorithm (eight sections for finding patterns, eight sections for writing those patterns) down to two. My goal was to create two parameterized loops to make my code more compact. I managed to accomplish this in regard to writing the patterns, however, it proved to be a bit more difficult when finding the patterns themselves. In addition to these, while this wasn’t really a

problem, the part I got stuck on for the longest period of time was finding the blending weights given the length of the pattern found. Dr. Yang once mentioned to me that while studying for his doctorate, he would work for only two hours a day on his research before going about the rest of his day. The thought process here is that one will continue to think about the problem(s) amidst the other activities that day holds. This ending up working, and later that night I solved the algorithm in my head while driving!

A good thing I learned from this project was the overall increase in enjoyment I had when I wasn't stressed about finishing in time. I tend to procrastinate on my projects, so for this one, I tried to get an early jump. It was a much more enjoyable experience! Lastly, something that I learned which wasn't great is that my algorithm would need to be written in a much more efficient way, as in 20 times faster, if it were to run on a machine at 60 frames per second.

Citations

Reshetov, Alexander. *Morphological Antialiasing.* Intel Labs. https://www.intel.com/content/dam/develop/external/us/en/documents/z-shape-arm-785403.pdf. Accessed 5 Dec. 2023.