Christine Moore
CIS4930

Project 1 Write – Up
Photometric Stereo

Introduction

This assignment was an introduction to graduate level type assignments to me. I enjoy the style of learning, because there is no pressure from tests or quizzes so during class I can focus on learning the material. This project really helped me understand the processes behind photometric stereo and the math behind complex image manipulation. Before this project, I had never heard of eigen vector decomposition and had no clue you could apply the concept of linear least squares to image processing. I pushed myself on this project because I truly think creating a normal map is such a cool concept and idea. I understood the general flow of events that must occur to obtain the normal map, but I was unable to successfully implement the whole system. I also had path issues in trying to display images which was very disheartening because I got very excited after figuring out how to implement a part of code but I could never test it with images to fix minor bugs or have a proof of concept image to report.

Attempted Implementation

While it is obvious I did not complete the implementation of the algorithm, I made a serious effort to attempt/complete as many parts of it as I could. Since I couldn't test out my code because of strange path issues, there may still be small, easy to change errors in my code that I wouldn't have picked up on without the ability to run it. But I am confident I accomplished or at least was extremely close to completing the WriteNormalMap functions, and the initial normal solving method, and the creation of an 80th percentile image to use in computing error. YAY! Sorry for the informality of that yay, but as of 6:34 PM on this glorious twelfth day of February, I got the Debug Normals method to accept my path! To run it though, debug normals data and image have to be in the same path as the matlab files.

The areas in which I had difficulty implementing were the least squared optimizer, the solving of normal through eigen vector decomposition, and the utilization of confidence weights in adjusting normal values. The least squared optimizer required the use of a struct, and I have never used structs before but I will learn them (hopefully) later this year in Data Structures. I understood where to get the values to fill the struct, however I did not know how to use the struct in the optimization method and eventually in solve normal using the eigen vector decomposition. I understood that the struct was being passed around from method to method and how to update it, but what to do with it was a struggle for me. Also, I understood the use of image confidence values. I understood how to initialize confidence values, but I couldn't figure out how to use the Twinkle algorithm or other similar algorithms to update confidences. I came up with an idea for calculating the confidence of an image though: using the error values for each color channel, if you divide the error by 255 you get a decimal value that indicates the relative level of error. I subtracted that decimal from 1 to yield a confidence value per pixel.

I also struggled to figure out how to relate the LSQ(), Lsq_Optimization, and Solve Normals function. Within the functions I wrote comments to describe my thought process but I felt confused what each method was supposed to distinctively take care of. Also, I understood the Linear Algebra behind solving for the normals and I solved for them in CalculateInitialNormals. I used these normal to calculate error values as well! Where I got hung up though was how to change the values of the normals in order to reduce error. I know this changing of the normals is accomplished by the lsq optimizer, but I just couldn't figure out how to implement it.

Overall, I think the hardest part of this assignment for me was just the overall difference in programming style. I've never had an assignment that utilized images with code and I have not encountered assignments that require mathematical computing. Coding an eigen vector decomposition is a little different then coding say like binary trees or linked lists so it was a new style of thinking. While it was challenging, I'm glad I had the opportunity to at least attempt this challenge because it helped me advance my skills.

Approach

The only real data structure I used in this assignment was a strcut to make the lsqData and simple array and matrices to hold images and image data. To cycle through image pixels, the use of nested for loops and matrix indexing through matlab were heavily used. For this assignment the actually coding was not too big of a challenge, the concepts were the challenge.

Running Code

As you can see when you look at my code, very few methods are 100% complete and ready to be used. Time wasn't necessarily the reason the other methods are incomplete, but rather that I got stuck during implementation. As bummed as I am that the code does not run to completion, I am happy with what I have because I really put every bit of effort I had into it. I'm excited to see the code that actually works so I can see what I needed to do differently and how far I got on my own. The one method I did get to work was DebugNormals. To run Debug Normals make sure the leaf_normals.jpg and the leaf_normal.data are in the same path as the .m files and just hit the run button and it should produce the correct normal map with white spaces.

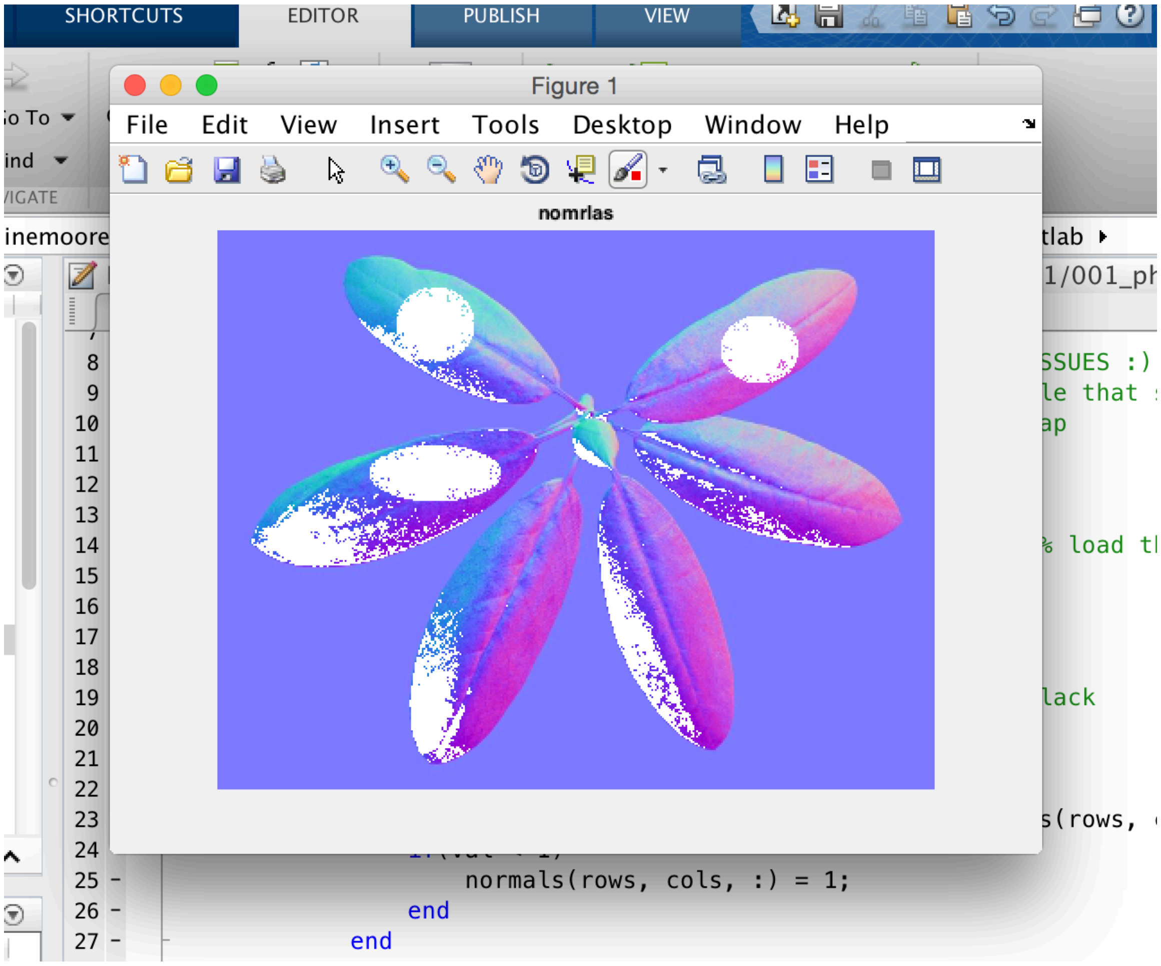
Closing Remarks

While I did not successfully complete the project, I'm happy with the amount of effort I put into it. What is the point of college if not to learn from your mistakes? I am extremely excited to eventually get a working system to solve for normals. Also, THANK YOU so much for your patience and help over the whole project. It was very much appreciated! I know I sent you a bunch of e-mails, so thank you for staying with me!

Normal Map Results

*Only debugNormals successfully produced a normal map, but wowwee when it finally worked I was so pumped haha!

DebugNormals Result:



```
SSUES :)
le that
ap

% load tl

lack

s(rows,
            normals(rows, cols, :) = 1;
        end
    end
```