Project Part 3

Devon Quaternik

ELEN 640

Image Processing

Professor Sally Wood

December 9, 2016

**INTRODUCTION**

In this final part of the project, I will look at the results I got from parts 1 and 2 of the project. I learned a lot about how difficult it is to nail down a good set of processing steps to successfully detect objects in an image. Gray and color were vastly different. Preprocessing as a whole worked about as I expected, but the end results differed far more than one might imagine. Some work fully for one image and get nothing on another. One thing that always stood out was the robustness of the Hough transform. It worked best in all cases.

**RESULTS**

**Questions**

1) After a lot of testing, I was able to get almost all the images to work with at least one processing method, if not multiple. Each set was different. Surprisingly, just turning a color image to grayscale does not give similar results. This really showed me how you have to be careful when dealing with images because everything is so subjective. For the grayscale image set, closing and thresholding helped proved to be most effective. The color images were more similar in opening and smoothing often helped improve accuracy. In both cases the Hough transform worked at least as well as the other methods attempted. This does not surprise me, as the Hough has very good basis sets. Matched filters work well on consistent surfaces, but perform poorly with rotation of the coins. Image pyramids tended to have issues with partial coins and other issues arising from inconsistent lighting.

I was often left with more false positives, but rarely did no coins show up when using Hough. Preprocessing really made the difference, though. The color images were more difficult to fine tune using only Hough parameters. When I started preprocessing further, my radii tended to become more accurate and missing coins would show up.

2) The image I would classify as most ‘robust’ would be image 3 of the grayscale set, 572. I could get nearly perfect results without any preprocessing from the Hough transform. Often performance was worse when preprocessed. Parameters used were a diameter range of [35 60], sensitivity of 0.96, and edge threshold of 0.1. These can be shrunk down, but other images had worse performance when changing them. The things that make it a good picture are it has relatively even lighting of the coins. Even lighting means shadows can evenly be dealt with. The coins are well lit preventing them from blending with the background. This helps when thresholding, ensuring all parts of the coin are above or below the threshold. When changing parameters, how you changed things mattered more than what you changed. For instance, turning down sensitivity would prevent some coins from being detected. If you increase it there would be false-positives. With edge threshold, decreasing increased false-positives, whereas increasing would reduce coin detection. Similarly, tweaking the threshold one way or the other would affect things depending on the lighting. Better lit coins will be more resistant to a poorly chosen threshold.

If given a similar image with a larger pixel size, it would work okay, but it really depends on the size of the pixels and how accurate you expect the diameter to be. The coins will still likely be picked up in an image ¼ of the current size, but you’ll be hard pressed to tell them apart. Closer coins may also merge together. You may be able to separate them with pre-processing, but it may be less expensive to just have a higher resolution image, performance wise. The center estimate would also suffer, depending on how small you shrunk it. It will help increase performance to shrink, as you are looking over less pixels, it does come at a cost.

I would recommend diffuse illumination for a few reasons. For one, coins are made of metal, a specular surface. This causes bright points that can cause issues when trying to detect. They also cut down on shadows. The color images especially suffered from false positives with regards to shadows being picked up as obscured coins by the Hough transform. A single, nearby, light source would cause both of these issues to be exacerbated and would greatly interfere. Multiple light sources would help to cut down on shadows, but may still suffer from bright spots. Clearly, that leaves a diffuse source as the best option.

For background there are a few good options, but it can really come down to how much processing power you are willing to throw at it. A solid background will work well, provided it is a color that is high contrast to the coins. Something like black works well as it would make for easy thresholding. White would probably not work so well as the coins may blend in if poorly lit. Trying to match some coins up to nearby objects, it appears as though blue would work as well. The coins tend to standoff better, even when in poor lighting. Smooth would be the preferred texture, as anything else would probably show up as noise in the sensor. Textures are too small to work well, but a background pattern could help, as its size will more closely match the coins. Using something distinct, like a + or X, would make them easy to identify. Since your pattern would be relatively known, there’s no reason you could not detect the pattern first and use that to estimate where to look for the coins. This could help speed performance, but it may not be totally efficient. You are effectively performing two image searches, one for the background then one for the coins, but this would not be a linear jump. Since you are calculating where to search for coins, it isn’t full image searches. But with sub-pixel processing you can increase accuracy of the coins’ edges by basing it on where your pattern is blocked. This again raises complexity, which is compounded by the extra image searching. Bad patterns would be something to worry about as well. You need something with a distinct shape from what you are looking for, or they will blend together in the results. Most backgrounds that have some noise need some sort of preprocessing to ensure that the noise is filtered out. The filters needed will have to depend on the types of noise involved.

3) Compared with the more robust image, the Hough transform will struggle with finding the coins due to the brightnesses involved. Because a lot of the coins are much darker, the edge threshold will need to be different for these areas and sensitivity increased. It would increase false positives in the bright areas, as a tradeoff. For preprocessing of this image I considered a few approaches I that had been successful before. Edge detection after opening and smoothing, and thresholding a closed image. None of my preprocessing worked perfectly. This was the only image which I could not get to get all the coins. Edge detection with opening smoothing with the Hough transform would get me 3-4 coins. The only thing I had to change was lowering the edge threshold to 0.1. I typically left parameters alone for one set of images and preprocessing steps. I wanted to look for the most universal solution I could find. Post processing did not seem like it could fix the problems with my image. Unless I was to run another Hough transform for different parameters and a different set of pre-processing steps, there is not much I can do to find the coins in the shadow. Overall my methods all performed poorly on this image. Only a low percentage of the coins were found and I almost never could get the dark ones.



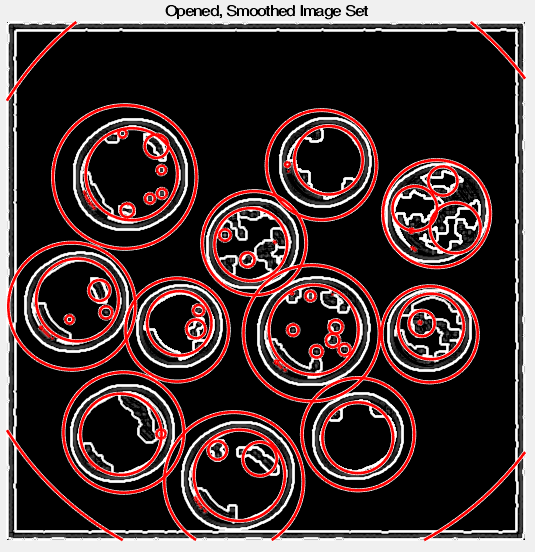
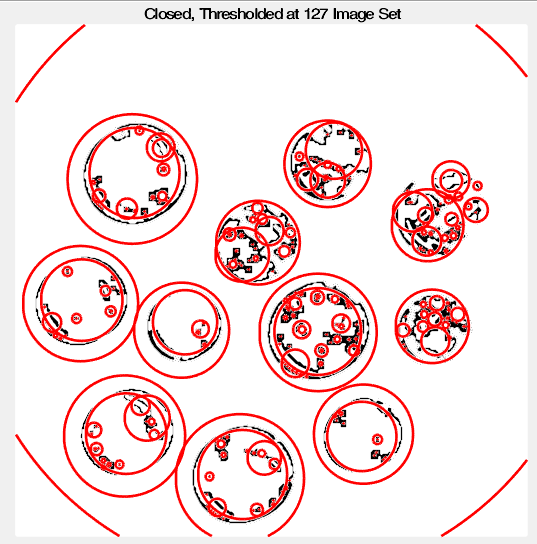


Figure : The results of closing and thresholding for Hough.

Figure : The results of opening, smoothing, and edge detection with Hough

Figure : The results of the Hough transform with no pre or post processing.

Figure : Image 569, the most robust image.



Figure : Image 622, Hough transform with opening, smoothing, and edge detection

Figure : Image 622, Hough transform with no preprocessing. Representative of first color set. Differences in color code mean only the original image is shown, no pre-processing as in the grayscale images.

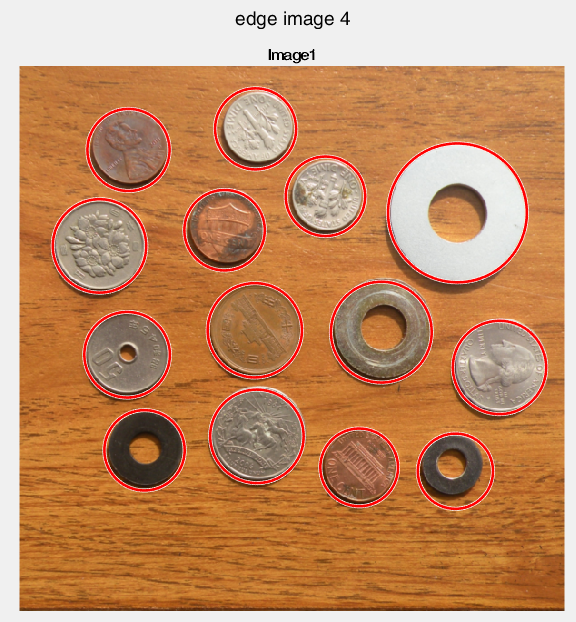


Figure : Image 595 with opening, smoothing and Hough transform. Representative of second color set. Best results shown.



Figure : Image 622, Hough transform with closing and thresholding at 127.

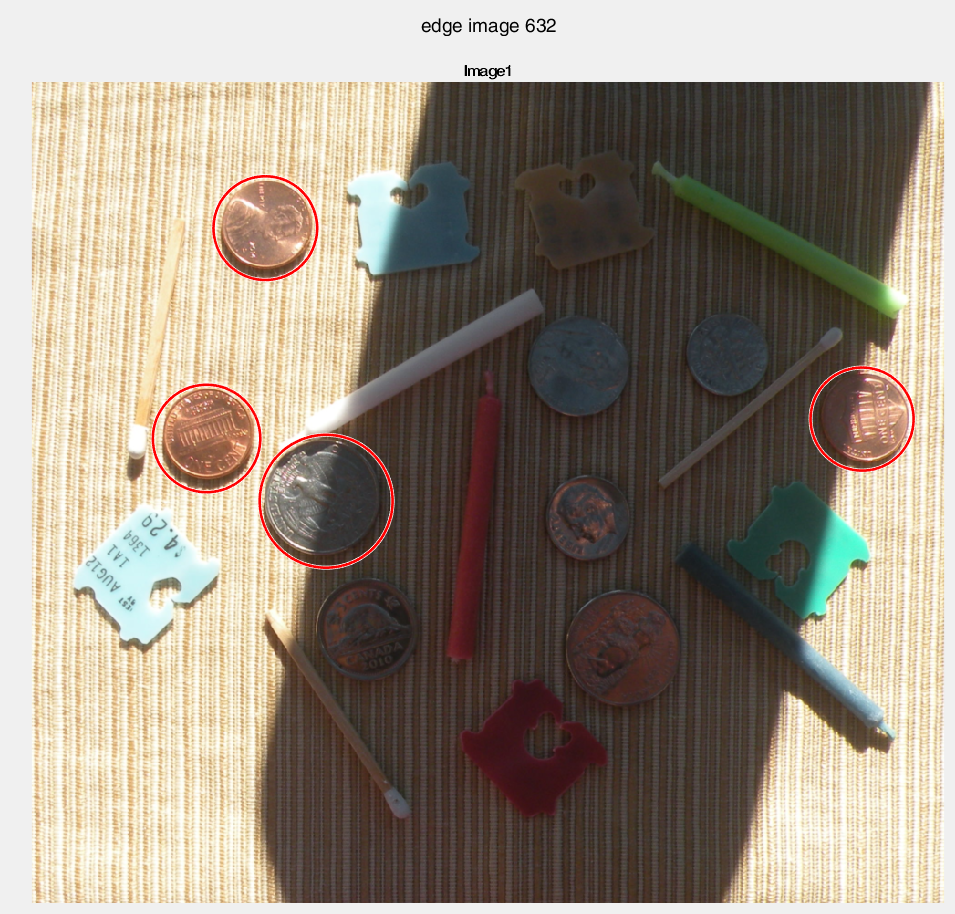


Figure : Image 632 with opening, smoothing, and edge detection for Hough transform.

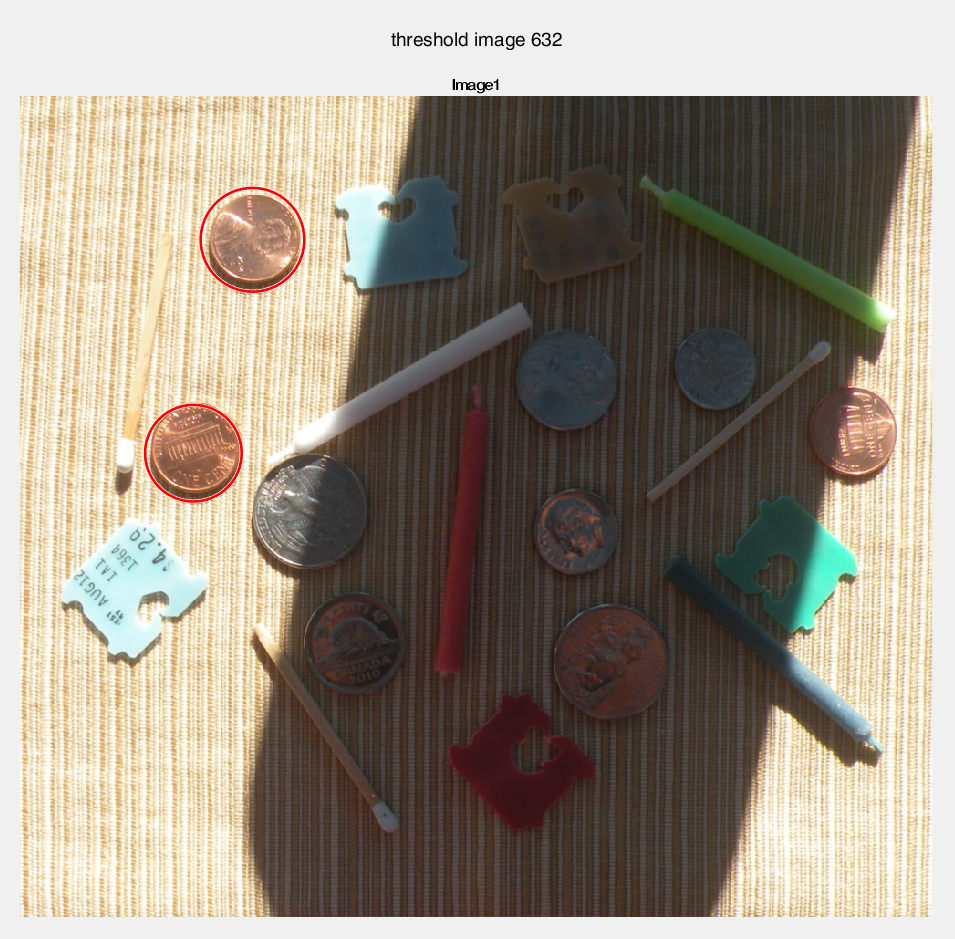


Figure : Image 632 with closing and thresholding at 127 for Hough transform.

**Block Diagram**

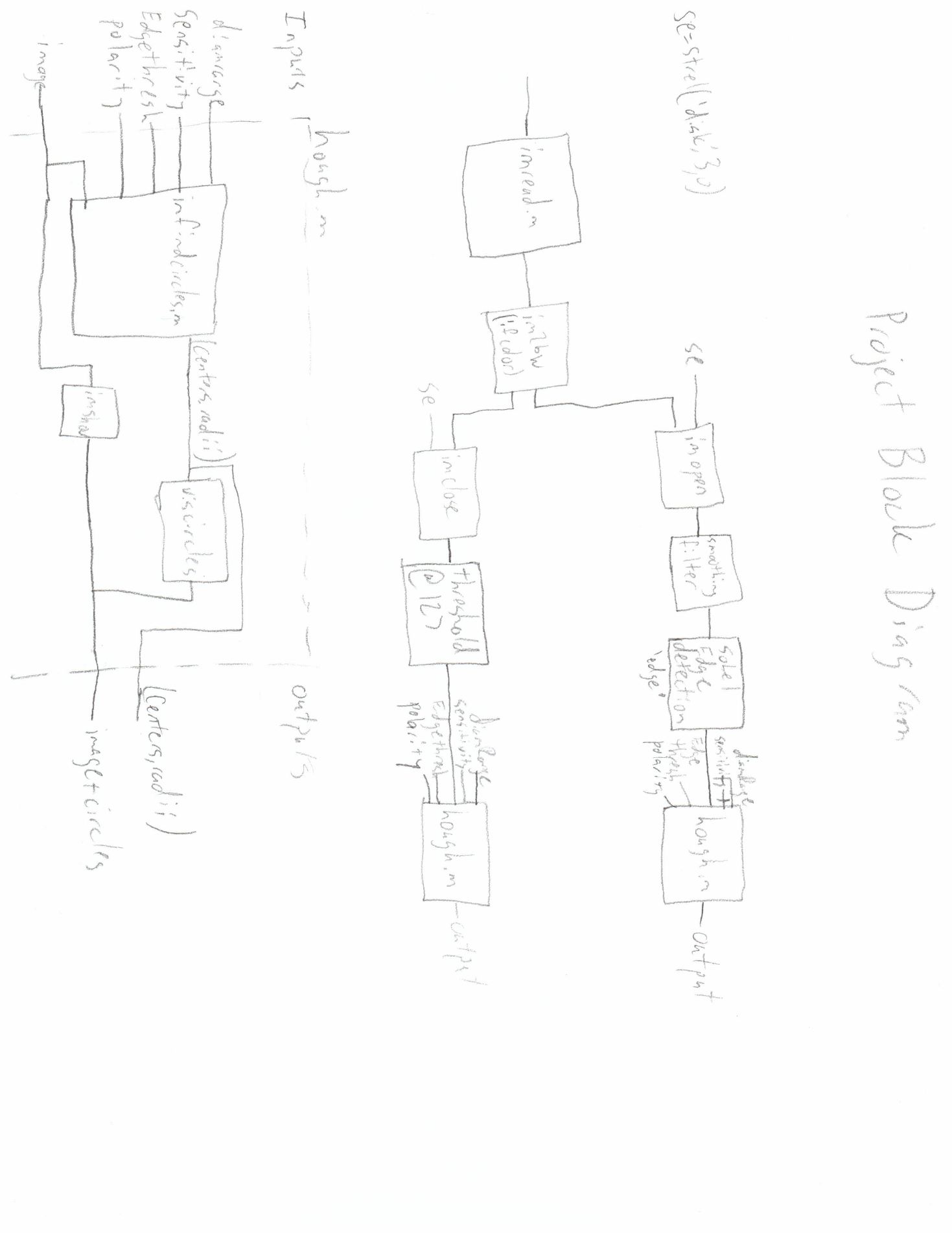


Figure : Block diagram of final chosen steps

My block diagram shows my final chosen processing steps. As color images were performing so poorly in the other two, they were of little value to show in the block diagram. They were also less flexible and had fewer parameters to tweak. Most blocks are my own, but a few are not. First is imread, which takes a PNG, in this case, and converts it to either a grayscale or RGB matrix. Next im2bw will convert an RGB image to grayscale. Imopen and imclose perform morphological filtering on an image according to their name. Se is the size and shape of the filter used to dilate and erode. Within hough.m, imfindcircles is the MATLAB function that performs the Hough transform and returns the centers and radii. It uses the parameters shown. DiamRange sets the size in pixels to look for. Sensitivity is how quickly it will react to an edge. Turning sensitivity up will increase positives, both false and true. Edgethresh sets the threshold that determines when an edge is detected. Lowering this allows for obscured coins to be detected, but can result in false positives, as seen when shadows are picked up as second coins. Polarity is bright or dark and determines whether to look for circles brighter or darker than the background. Viscircles will show the circles on the image that are taken from imfindcircles. This is displayed over the original input image as the pre-processed ones lose visual meaning.

**CONCLUSION**

If there is one thing I have learned from this project, it is that there is no one universal solution that will give you great results. Each different set responded best to different pre-processing, parameters for processing, and had I accomplished it, different post-processing as well. Something that sounds simple, ‘find coins in an image’, quickly becomes difficult when lighting is played with or a poor background is used. Even the coins’ own shadows could interfere. All in all, I got most of the images to work in at least one case. However, I was surprised at how much you have to let through to get all the correct circles. Often tuning parameters word remove true positives before false ones. I learned plenty more, but that was the lesson that shown through at every turn.

**REFERENCES**

Documentation. (n.d.). Retrieved December 12, 2016, from https://www.mathworks.com/help/images/examples/detect-and-measure-circular-objects-in-an-image.html

imread. (n.d.). Retrieved December 12, 2016, from https://www.mathworks.com/help/matlab/ref/imread.html

imopen. (n.d.). Retrieved December 12, 2016, from https://www.mathworks.com/help/images/ref/imopen.html

imshow. (n.d.). Retrieved December 12, 2016, from https://www.mathworks.com/help/images/ref/imshow.html

imclose. (n.d.). Retrieved December 12, 2016, from https://www.mathworks.com/help/images/ref/imclose.html

imfindcircles. (n.d.). Retrieved December 12, 2016, from https://www.mathworks.com/help/images/ref/imfindcircles.html

viscircles. (n.d.). Retrieved December 12, 2016, from https://www.mathworks.com/help/images/ref/viscircles.html