Project Part 1

Devon Quaternik

ELEN 640

Image Processing

Professor Sally Wood

**INTRODUCTION**

The purpose of this report was to study the effects of different image processing techniques on images of coins. The end goal is to determine whether they would be suitable for finding coins, counting them, and figuring out the diameter. The techniques used were thresholding and edge detection. Each was tested with different preprocessing steps in order to find the best image possible. For simplicity in comparison, all images are given at the end of findings. All morphological filtering used the same structuring element, a disk with radius 3. This seemed to provide the best amount of noise reduction, without sacrificing for quality.

**FINDINGS**

**Thresholding**

A) For thresholding, I found that using the middle value of the entire gray level spread worked best, which was 127 in this case. I attempted a few others, but most were completely incomprehensible. I chose round numbers like 100, 150, or 175 to test, but were extremely poor and mixed the background in a lot. Using the median value of the image seemed like a good idea at the time, but after seeing the results it was clearly not as useful as I had thought. Overall the middle value gave the best performance, not allowing too much noise through. You can vary the number 127 by about 5 and still have something I would deem as good. Past that some noise starts to get dominant or the coins are too wiped out. When looking at the histogram, most of the background would be lumped to one portion of the histogram, with other smaller lumps where coins existed. Distinguishing the coins means taking gray levels that don’t appear in them out of the picture. As the coins are shiny, they will be brighter than most of the background. In my code, it is important to note that the background is white and coins are black.

B) Even the best results for image 1 were partial coins from pure thresholding. No matter how I tried to adjust it, they would only half appear. A few of the coins have some internal structures, but they are the most complete. For image 2 it is the opposite story. Almost all are filled in and a few are partial. Image 3 came out the best from this method, with almost all coins as complete circles. Images 4, 5, and 6 are all similar to Image 2, with varying levels of completeness. All the results are from the same strategy, as the others tried gave very bad results. I have only left one test in to demonstrate the inferiority compared to a threshold in the middle. Certain images improve if you vary the threshold slightly up or down, but mostly keeping it steady gave the best results across the board.

C) All seemed to help in one way or another, but applying more than one to thresholding rarely seemed to improve results. The best results came from closing the image. It reduced most any noise from the background, and created clear circles. You could apply smoothing to reduce some of the slight noise, but it was largely not worth the extra computation. If you run an image through the ‘edgetest.m’ function in the zip file, you will be able to see various results. In my images below, all were closed before being thresholded, which ended up yielding good results across the board.

D) Some of the basic problems with basic thresholding is lighting in the images. If I had to place the pictures on a scale of difficulty to draw out the circles, image 1 would be the hardest, 3 the easiest, and the rest pretty much interchangeable. That is because the coins in image 1 have a very bright or close light source, causing them to have a large brightness gradient. This leads to only part of the coin’s full circle showing up, as displayed most prominently in image 1.

E) The best results came from closing the image using the Matlab function ‘imclose’. This is followed by thresholding at 127, the midpoint of the range [0 255]. Cursory testing showed no difference in the order of filtering or thresholding. If I opened the original then thresholded, it seemed to be the same as thresholding then opening, for instance.

F) If thresholding had perfectly separated the coins, you could easily find the diameter. You would need to look for where the change happens, the edge, then mark the locations that are furthest apart. This would give you the diameter. You could then get the radius and use that to find the center from any known edge point. Because this is a circle you would need either sub-pixel processing or accept that there will be some error in the measure. This would require an algorithm that can ‘fill in’ circles, so to speak. Since some are not completed, assumptions will need to be made. A match filter could work well, but the Hough filter would probably be a more suitable candidate.

**Edge Detection**

A) I found Sobel and Prewitt to be most effective, with Sobel coming through slightly better overall. Both were effective in separating coins from the backgrounds, given the proper preprocessing. I have chosen Sobel as it seems to deal with noise slightly more effectively. I went with the automatic threshold that Matlab used. It seemed to give good enough results, that with a little noise reduction you can easily make out at least partial circles in all the pictures. To reduce a lot of the noise, I opened then smoothed the images before edge detection.

B) All of the images look similar in that they have the coins edges and not much else. 1, 4, 5, and 6 all have coins which show a lot of noise on the interior. This is due to the pattern of the coins, and would ideally be ignored when calculating coin placement and size. If there were noise outside the coins, things would be more disconcerting, but since it is only one side, the circles are still obvious.

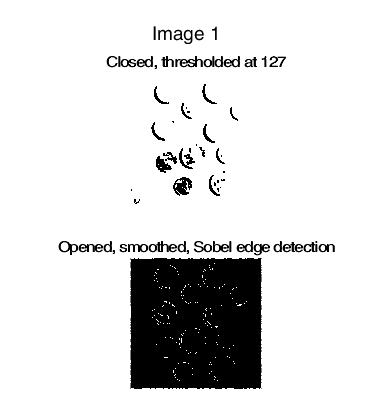
C) To get the best results I opened them images with the structuring element described in the introduction. Then I smoothed the images with a 5x5 averaging filter. This helps to blur the edges that some of the backgrounds had built in, as well as push some noise below the threshold. What it leaves intact are the strong edges of the coins. Another good thing is it only leaves noise on one side of the coin. This is helpful in identifying locations of the coins and ensures a good measure of the diameter.

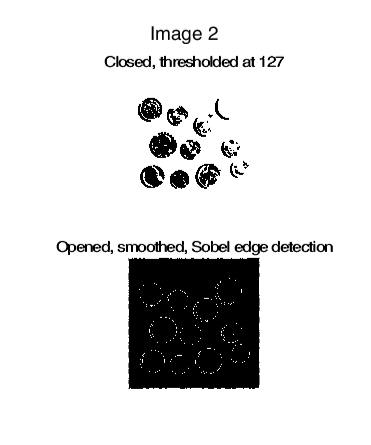
D) The basic problems of edge detection are picking up background edges, picking up patterns on the coins, and noise in general. These images seemed to come out noisier, with most edges not being connected. Although not a problem for a human, it could be an issue with computers. A technique like a match filter would not work well unless you set the threshold for a match very low, due to lack of complete circles. Orientation of the gradient would be helpful for this, as you would know whether you were entering or leaving a coin.

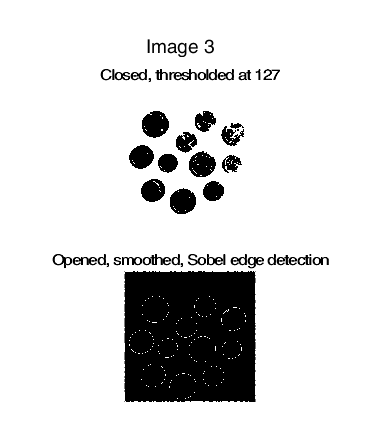
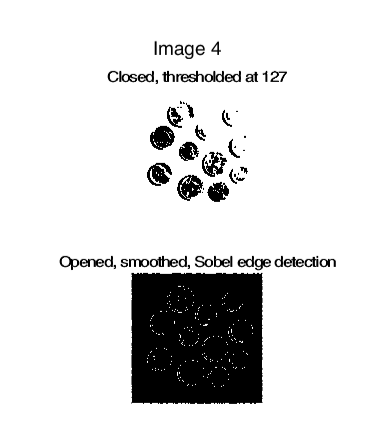
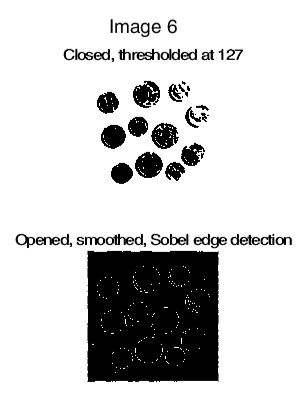
E) Unlike thresholding, order of filters and edge detection mattered. I filtered original images, then passed the filtered image to the ‘edge’ function. I used automatic thresholds, as my results from changing did not seem to help much. I tried 4 different methods; Canny, Sobel, LoG, and Prewitt. Canny was overall too noisy. LoG worked decently, but Sobel and Prewitt were a lot better at reducing interior noise from the patterns of the coins. None were great at completing the edges, but Sobel and Prewitt left the most relative to reduction in other noise.

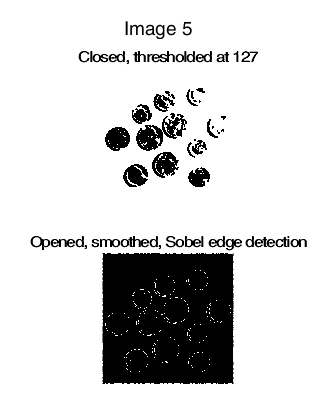
F) If the edge detection was perfect, a match filter for the expected coin size would work very well. A Hough transform that detects circles at the correct ratio to fit all the coins could also work. More noise is removed from the middle here so it should perform better than thresholding. Again this requires an algorithm that can fill in circles that are incomplete, as unlike with simple thresholding, almost all of them are here.

**IMAGES**









**Plans for Improvement**

I would like to explore tweaks to the two methods I have described here that help to improve the completeness of the circles, and, if possible, remove interior noise. Completing the edges as far as thresholding may be exploring different types of thresholds, including ranges or based on some non-linear idea like median. Closing worked best overall for this, so I will likely stick with that and try to tweak the threshold.

Edge detection is a similar story, with types of filtering seeming good, but tweaks in other places needed. I want to see the effect that a different structuring element would have on both thresholding and edge detection. For only edge detection, I want to try and change the shape of the Sobel filter and see if the results can improve further.

**CONCLUSION**

From what I could tell, closing in combination with thresholding at the middle of the gray value range worked well. Opening, smoothing and edge detection did as well. To pick which would be better overall would have to come down to computation time, which would easily go to simple thresholding. Both would rely on an algorithm that can complete a circle so that becomes a non-factor. The preprocessing is less with simple thresholding. Of course this is for this set of images. Another set would probably have a very different answer. But testing thus far would point to simple thresholding being enough to detect the coins, their size, and placement.