Project Part 2

Devon Quaternik

ELEN 640

Image Processing

Professor Sally Wood

December 1, 2016

**INTRODUCTION**

In this part of the report, I studied different methods for finding the center and diameters of circles in grayscale and color images, as well as what preprocessing worked when. I found that there was a lot more subjectivity in what worked best with color. It seems that even within one data set, large variation can make preprocessing difficult to predict. Again I have included all image results at the end of the paper.

**RESULTS**

**Image analysis methods**

1. The three methods I used to find the centers and diameters of my coins are matched filtering, circular Hough transform, and image pyramid to extract edges. Matched filtering is picking a filter that looks like what you want to find in the image. You take the normalized cross correlation between the two images and see where things line up. MATLAB’s built in tools can be used to find the centers of the circular region and the radius of each.

The Hough transform transforms the image in such a way that only circles are left and are easy to find. In matlab the function ‘imfindcircles’ implements this. You input an image and the radii range you are looking for and it will find circles. There other parameters to set that can change how sensitive it is to different features and affect the performance.

Image pyramids were used to extract the contours. Using MATLAB’s built in contour function would have been difficult due to the way they are expressed. From this I used a similar technique as matched filtering to generate the circle centers and radii.

1. For preprocessing, smoothing opening and using an edge detection algorithm helped improve results for some images. Some using the circular Hough transform did not need any processing to get all the coins. I did not get a chance to try any post processing to remove false positives, as getting the color images working took longer than expected. I can imagine a function that checks the centers returned for a matching color to help remove any extra circles. This may not help in all the images of set 2 as there are a few non-American coins mixed in, but for set 1 and any of the washers or buttons in set 2, it would easily be able to take out false positives.
2. For grayscale, Hough worked with no extra processing necessary for all pictures. This is shown in Figure 1. When I tried to preprocess my results ended up worse than before. Hough was the best method, but the other two were functional, mostly getting lots of false positives. The only method worth noting for accuracy for the color images was the circular Hough transform. Using edge filtering helped improve performace, best demonstrated by images 1, 5, and 6. These improvements are shown in figure 3, 4, and 5. The results of a selection of other results are given in Figure 2. Unlike in grayscale, the color images did not match filter well to anything I could come up with. I tried various means of preprocessing and all returned very few coins. The image pyramid was the opposite problem. It returned an extreme amount of centers and radii without any preprocessing. The image would be covered in circles to the point you would not recognize it at all. Morphological operations and thresholding helped, but left many false positives, including non-circular items. I imagine this has to do with my method of detection.
3. Color was not useful for what I attempted. All three of my circle detecting methods required grayscale images. Color would be more useful in post-processing for removing false positives, as I explained earlier in part B. If I were to use it, I would be using HSV. It would be far simpler and more reliable to match up the hue to check the coin color than check each RGB value. You would need to setup switch cases for each coin color, but overall HSV would be far simpler to implement.

**Response to objects that are not coins**

1. For matched filtering there was not much to be found, so there were no false positives. As mentioned earlier, image pyramids would give nothing but false positives. For the Hough, there were other objects separated in both sets of color images. Even with preprocessing, both the previous methods picked up too many false positives to counteract. Performance needs to be tuned before any post would help. In the first, the most common one was the coins’ shadow being picked up as an obscure coin. Following that was the inside of the bread ties. I’m not sure why, but it seems it picked them up as incomplete coins. Most of the non-circular objects were not picked up though. In the second image set, just about all the circles were found, including a shadow or two. Again this goes back to not having enough time to post-process, but most false-positives should be removed. Assuming most can be removed performance of the Hough is far better than the other two methods, for any given pre-processing method.
2. Processing sometimes did and sometimes did not bring false positives. Honestly this one is very subjective to the image. Some parameter sets would find one image perfectly and the next image would have double circles on all the coins. They seemed to be related to how I was using my morphological operations. When I opened or closed, certain objects would look more circular and end up being found. Sometimes coins would blend together into one large found center.

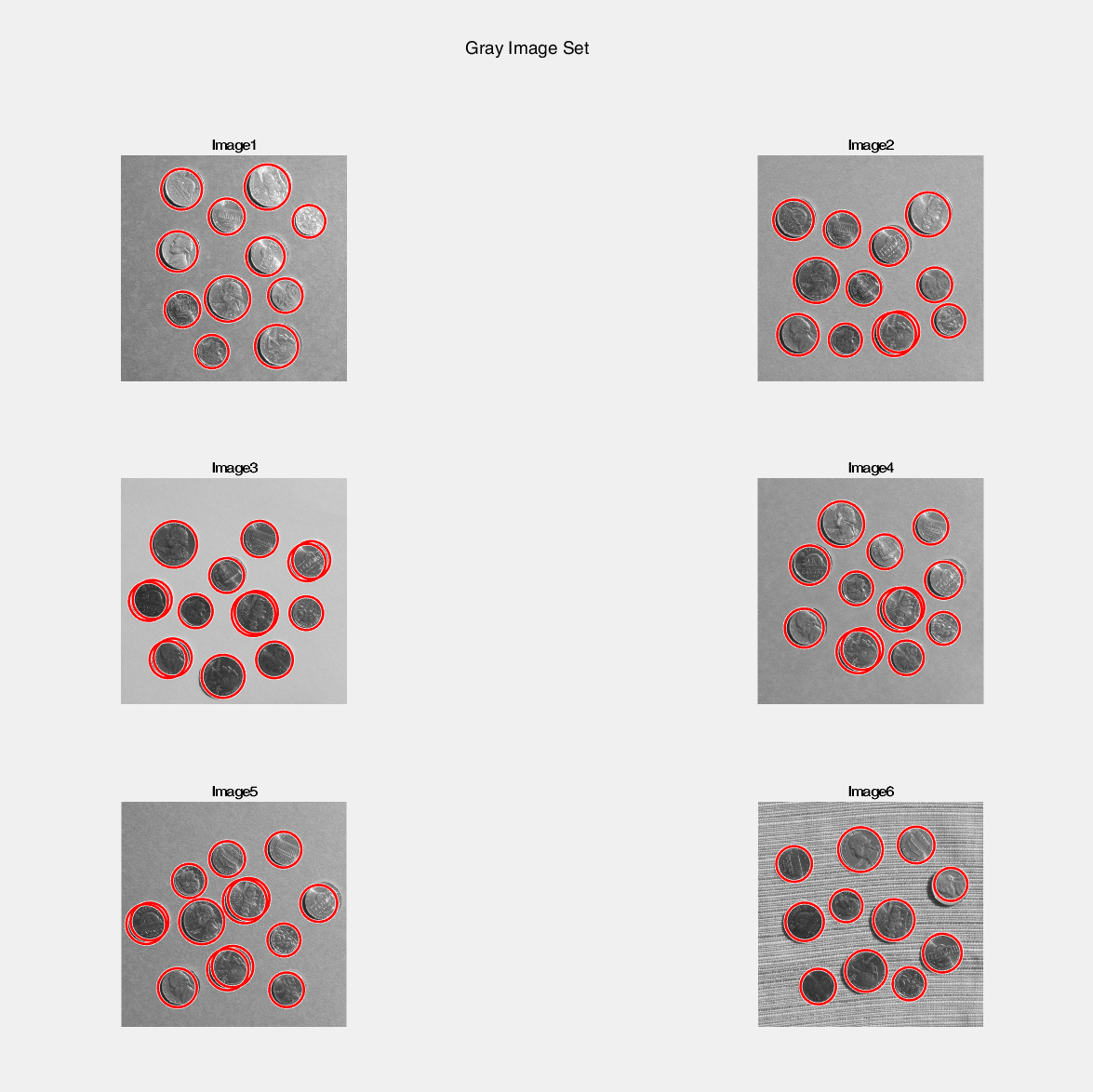


Figure 1: Hough Transform on gray image set

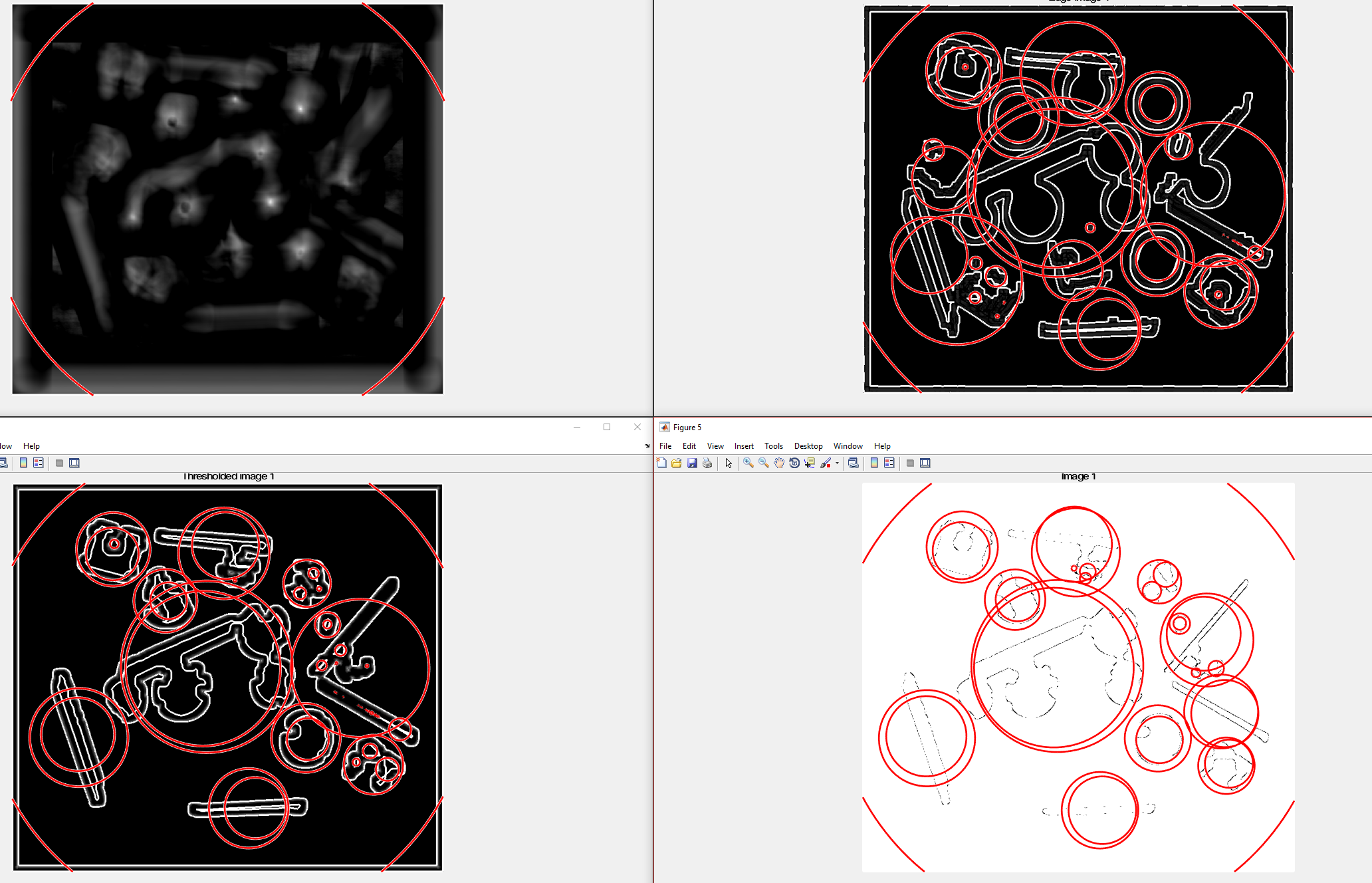


Figure 2: All use image 1 of colored set 1. Top left: Matched filter for color image with penny. Bottom left: Matched filter of image thresholded at 127. Top Right: Matched filter of edge image. Bottom right: Image pyramid of thresholded image.



Figure 3: Left: Image 1 results after opening and edge detection. Right: Results with no preprocessing. Note the inaccuracy of the radii and the double on the nickel.

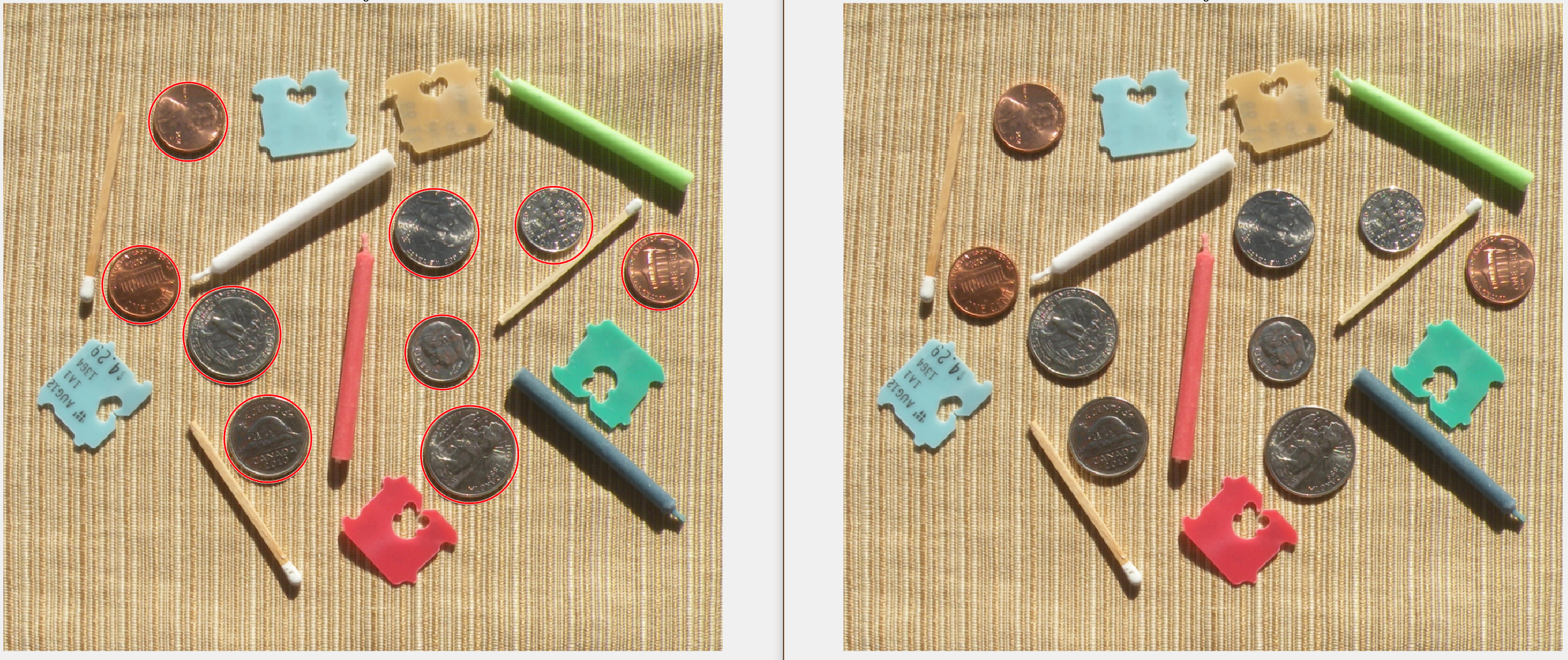


Figure 4: Left: Image 5 with opening and edge detection. Right: Results with no preprocessing. Goes from detecting no coins to detecting all of them.

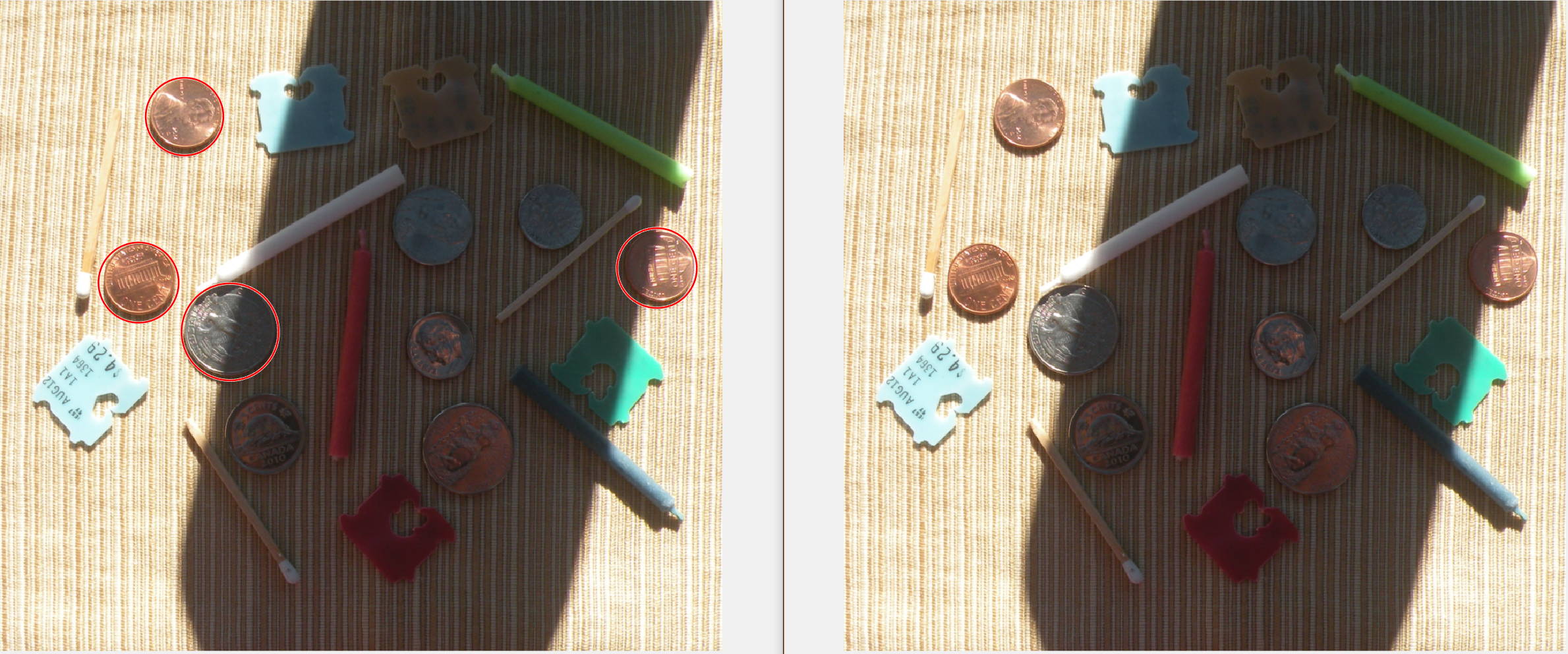


Figure 5: Left: Image 6 with opening and edge filtering. Right: Image with no preprocessing. While not the same improvement in performance as image 5, it picks up almost half the coins.

**Plans for improvement**

As I mentioned earlier, I was not able to add any post-processing to the images to remove false positives. I can find the coins of most of the images using the Hough transform and some preprocessing, with the exception of image 6 of set 1. The shadow really makes life difficult, but the rest seem to work really well. As for the other two methods, I would need to explore different preprocessing entirely, as color images do not lend themselves to image pyramids and matched filters as I had expected. For grayscale I would think some post processing would be able to reduce the number of false positives by using a radius range and possibly different size morphological operations. The Hough transform worked so well I really do not think that is anything but wasted computations. By tuning the parameters, Hough found all the coins in the grayscale, with few false positives related to shadows. Removing extras would be a matter of blocking two circles from showing up within a specific distance of each other.

**CONCLUSION**

The Hough transform is the best method for detecting circles. It has the ability to find obscured or incomplete circles and makes preprocessing a lot easier because of it. When it came down to it, I was surprised to find how little thresholding worked in comparison to edge detection when it came to color. Edge detection was not perfect, as I showed in my last report that it tends to not give full circles. This lead to a lot of false positives, but still performed far better than thresholding. Surprisingly the image with no preprocessing worked very well. If you are looking for computational efficiency, you may be able to get away with using most any halfway decent image for the Hough transform detection.

**References**

"Documentation." Detect and Measure Circular Objects in an Image - MATLAB & Simulink Example. Mathworks, n.d. Web. 02 Dec. 2016.