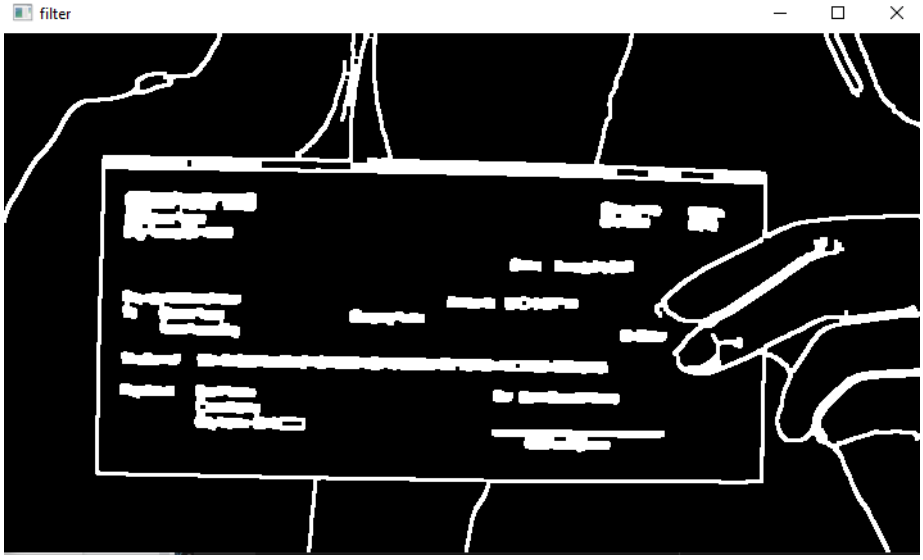
Ansheng Li

Check Detection Project 3

For the preprocessing of the image, I used a similar approach for lane detection. I first resized the image to a standardized size as some images have different dimensions. The new resized image is then turned into a grayscale with blur using kernel size of 5. Kernel size 5 had the best results out of 3 and 7. The canny edges I used are the low end of 10 and high end of 100. This reduced most of the larger noise around the check. I used dilation and erosion to further emphasize the edges of the check so that contouring would provide better results.



This is the image after all the preprocessing steps are completed.
Contouring the canny edges provided a stronger outline of the check but for cases where the hands are in the image, the contour does not connect. I used a convex hull to encase the contours and used the convex hull to detect the four corners of the check. This provides four positions on all polygons of size 4 while the area is bigger than a specific number.

From the contours, the convex hull is used to calculate the area with the biggest rectangle. The algorithm returns the four positions of the contour with the biggest rectangle. By sorting them in order, we can use it for our warp perspective.

Some issues with my approach is that images that have a background similar to the check are unable to be detected. Due to my preprocessing, the edges seem to be lost after the filter is completed. While my approach worked for most cases, the similar check and background color is difficult to compute.