Seungmin Baek, Sungwon In

CSCI-B 456 Image Processing

Final Project

Finger Detection

A large number of human point with their finger the main point. By detecting the finger in the image or video, computer can figure out what people want to point out. Figuring out what user think most important point will improve quality of machine learning. And this efficient technique, detecting finger, requires three main steps. They are subtracting background, finding convex hull, and contouring.

STEP 1-1: Background Subtraction

To subtract the background, first we need to subtract current frame from previous frame. In this project, current frame will be an image including finger and previous frame will be an image without finger with same background. And we need to convert this image type to binary image for median filtering. To make this image to binary image, we need to put condition which checks each subtracted image pixel is greater than threshold or not. Formula for subtracting image will be **'abs(current_frame – previous_frame) > threshold = subtracted_image'.** The result of this formula will be

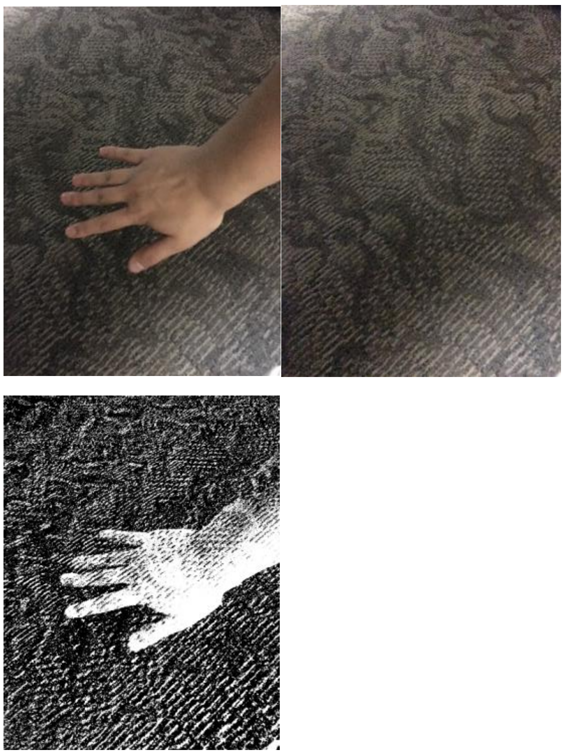


Fig 1) From top left: A,B, and C

A: Current frame.

B: Previous frame.

C: Subtracted Image.

Abs(A-B) > Threshold = C.

We test with 4 different threshold values: 20, 40, 50 and 100. Results are below.

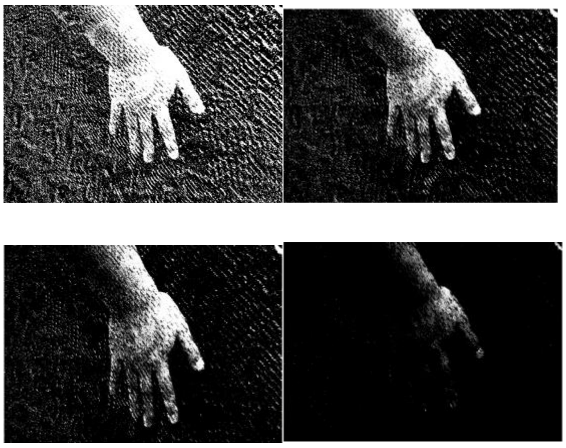


Fig 2) From top left: A,B,C, and D

A: with threshold 20.

B: with threshold 40.

C: with threshold 50

D: with threshold 100.

From this test, we could figure out that as threshold values increasing, noise is decreasing. However, as threshold values increasing, target figure lose its shape. So we decided to use threshold value to 20.

STEP 1-2: Image filtering

After converting image type, image have to be filtered for smoothing image. We have two option for filtering image, and they are median and Gaussian.

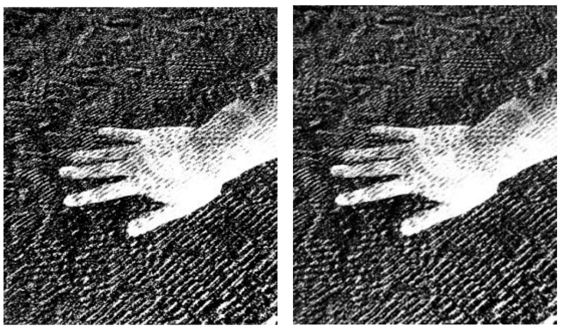


Fig 3) From left: A, and B

A: Image filtered with median filtering.

B: Image filtered with Gaussian filtering.

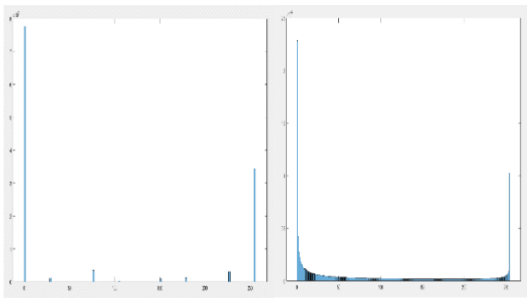


fig 4) From left: A, and B

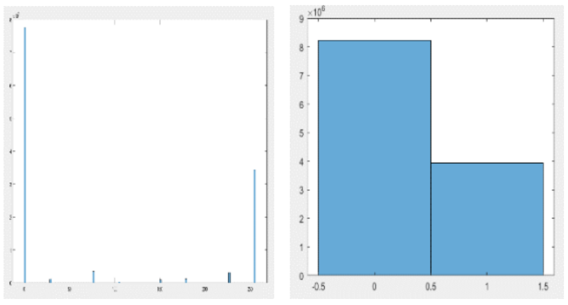
A: Histogram of median filtered image.

B: Histogram of Gaussian filtered image.

We test two options with the same image, and figure out median filtering returns better smoothed image. Histogram of image prove that median filtering better eliminate pixels in range between 1 and 254.

STEP 1-3: Converting Image Type

Even after median filtering, we could see filtered image still has pixels between 1 and 254. So we used the im2bw function to convert those pixels to 0 or 1.



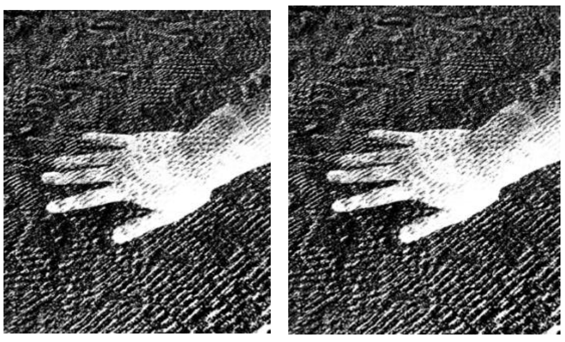


Fig 5) From top left: A, B, C, and D

A: Histogram of median filtered image.

B: Histogram of image after im2bw function

C: Image filtered with median filtering

D: Image converted to binary image

STEP 1-4: Find connected component

As we can see from the ***step 1-3: image filtering***, there are still lots of noise on image. To eliminate all the noises and get the hand feature, we need one more step which is finding connected component. To find connected component. Extracting feature that are above specific number of connected components. It will find main connected component in the image. In this project, main connected component will be hand figure. We found connected component with the function bwareopen(connected_components, threshold).

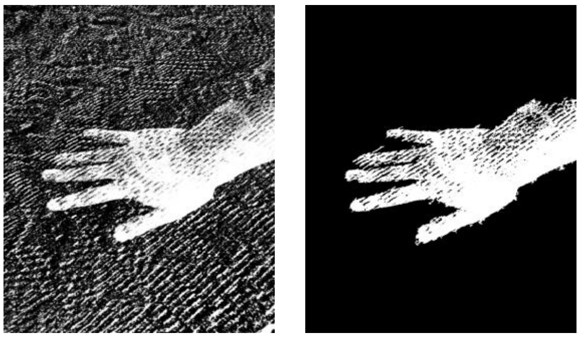


Fig 6) From left: A, and B

A: Binary image.

B: Image after bwareopen function

Following the image 'B', we can se e noise is totally eliminated. Because bware open(binary_image, threshold) is the functio n which removes specific connected compo nents which have less pixels than threshold, we could get main object.

STEP 2: Convex Hull

Computing the convex hull has a lot of algorithms to access; Graham's Scan is most representative algorithm. Here is how it works.

1. Find the smallest y-coordinate in the image. And let's say this y-coordinate 'pivot'
2. Except pivot, sort all the pixels respect to ascending order of x-coordinate and save it to an array. And let's say that array is called 'P'
3. Push pivot and P(1) into stack. And let's say stack is called 'S'
4. Iterate through index 2 to number of elements in 'P'. And repeat the step 5 and 6.
5. Check if 'S' has more than 2 elements or not. And if 'S' has more than 2 element, pop the element P(i) until P(i) can be pushed into 'S'
6. Push P(i) into 'S'
7. Mission complete.

And this algorithm can be computed by

builtin function ‘regionprops’. Regionprops returns Area, BoundingBox, Centroid, ConvexArea, ConvexHull, ConvexImage, Eccentricity, EquivDiameter, EulerNumber, … etc.

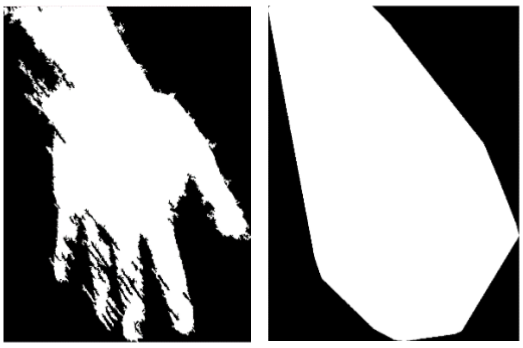


Fig 7) From left: A, and B

A: Image of biggest ConvexArea’s image.

B: Image of biggest ConvexArea’s ConvexHull.

A is the image that largest filled image from the subtracted background image above. Also, B is the image of the convex hull of the A. Computing convex hull is the process that helps to get vertices of the hand image. As you can see, vertices of finger and vertices of convex hull seems match each other. However, we cannot make sure. It is necessary process to detect finger and for the next step “Contouring”.

STEP 3: Contouring

As we mentioned above, Contouring is the process to make sure finger in the image fits to vertices of convex hull. However, since the convex hull image is filled with white color, we need to get edge of convex hull. We used the canny edge detection. By using the **imfuse** function in Matlab, we combined two images hands and edges of the convex hull.

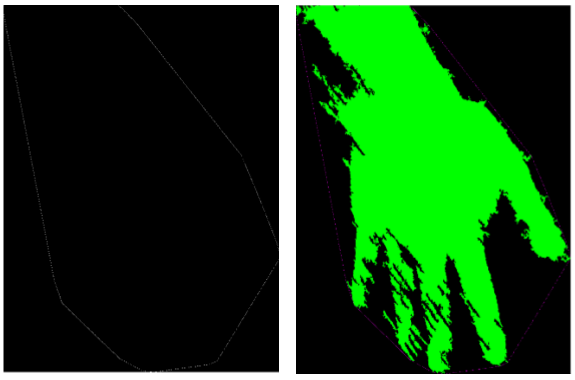


Fig 8) From left: A, and B

A: Canny edge of convex hull.

B: Combined image of edge and hand.

A is the canny edges from the convex hull image, and B is the image that edges and hand image are combined. It proves that vertices of convex hull image are perfectly matched on finger tips.

Results

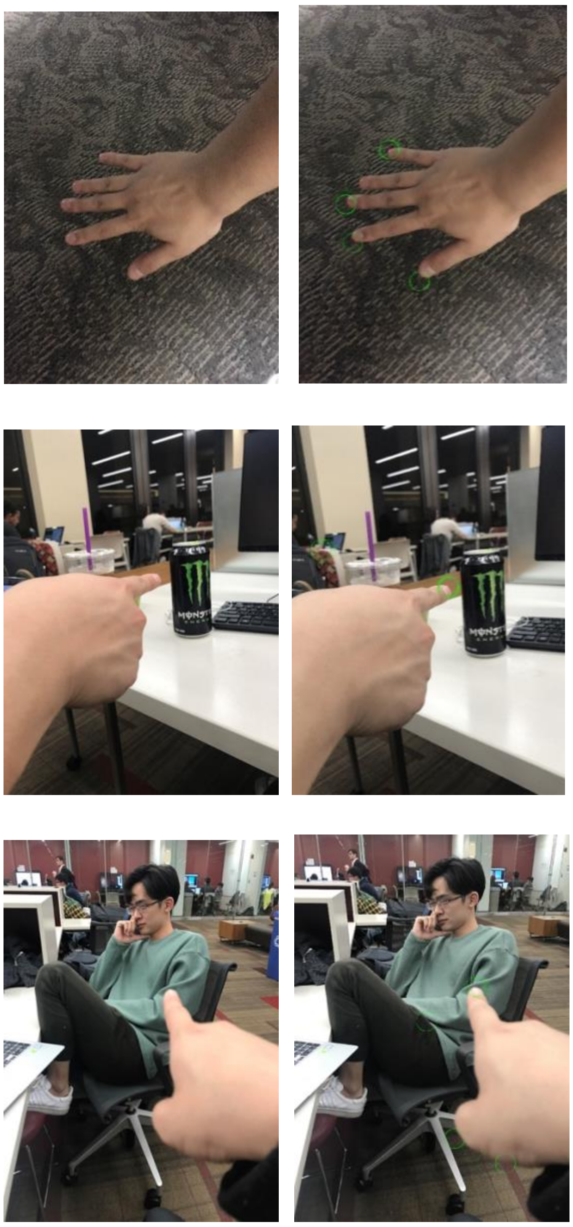


Fig 8) From left: A and B

A: Current frame.

B: Image after finger detection.

This what we got from the input image. We can find that green dotted circle is marked on the finger tip in each image. From this results, it is not hard work to make computer to know what human consider the important point is.