Graham Skeats
Final Project Report

My objective for this project was to build a fingertip tracker that would be able to identify and track it as it moved around. I was really interested in this project since I think that changing the way people are able to interact with computing devices is an interesting and important area of research. Since computing devices are becoming smaller and smaller, it seems like the traditional model of a large physical keyboard is becoming impractical. Moreover, even though computers are literally millions of time more powerful than they were even half a century ago, the keyboard as the standard input has essentially remained unchanged. As devices continue to shrink and evolve, a useful physical or digital keyboard can only shrink so far. Gesture recognition for wearable IOT devices or other advanced computing devices seems like a powerful way to quickly and easily enter information. Imagine for a second the failed google glass, a person could place their pointer finger down on a table and move it around as they would a mouse, moving a cursor to interact with a web browser in their display (or imagine minority report). Or even people with impaired motor skills, a digitally tuned finger tracker that is tolerate to shaking or erratic movement might be far easier to use than a trackpad or keyboard.

My solution to this problem used an excellent approach that I found online and was able to integrate into my own code. The program first draws nine rectangles on the screen as targets for the user to place their hand on. The program then samples the skin color of the hand and creates a histogram that can be used to identify patches that contain that color. Then the frame is filtered for all regions that don't contain that color. These first two steps result in an image like the ones below. It is important to note that the final image won't look like this, this is only so that later steps have less information to deal with as no critical regions are eliminated.

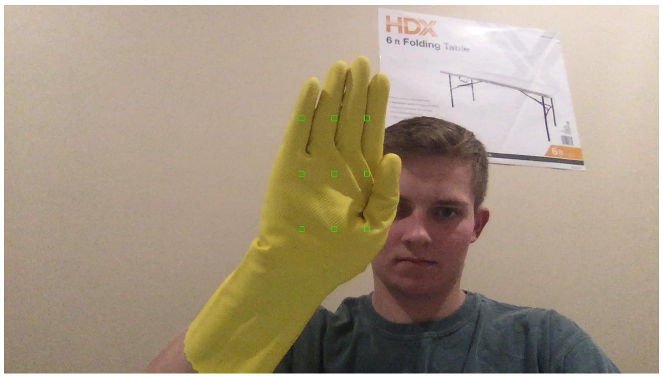

Figure 1 Unmasked image just before color capture, you can see the green rectangles drawn in the frame where the color is sampled.

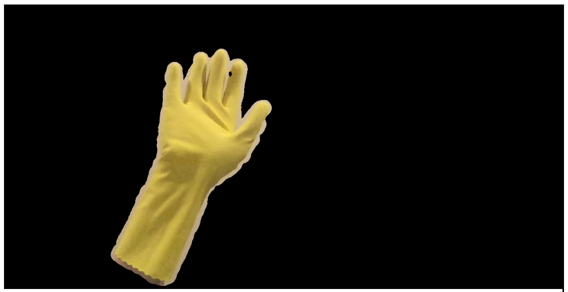
Figure 2 Masked image showing only areas including the color of the hand.

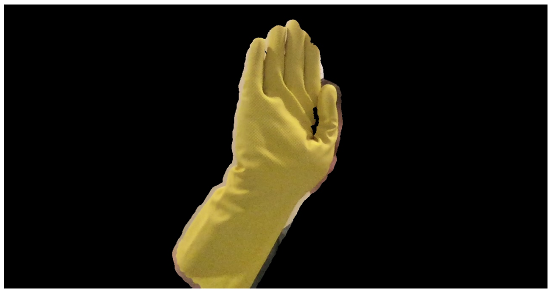
Figure 3 Masked image showing only areas that included the color of the hand.

As you can see in the images above, using the bright yellow cleaning glove I bought from CVS allowed the program to very easily delineate between areas that contained the desired color and those that didn't. Results using my barehand were also successful but I found that the program was sensitive enough that the color of the skin on my face would also be picked up if it were in the frame, skewing the results. This would be avoided if the camera was facing POV at the user's hand like it might from a wearable device like google glass since the users face wouldn't be in the field of view.

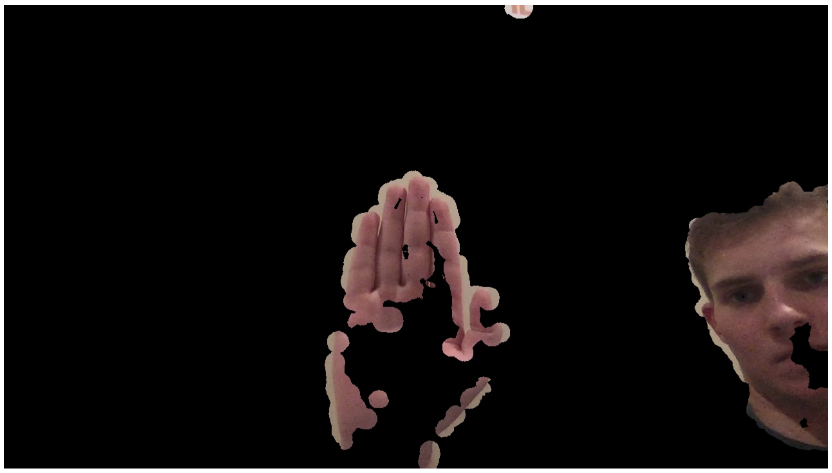

Figure 4 The program was able to identify regions that my skin was visible using my natural skin color.

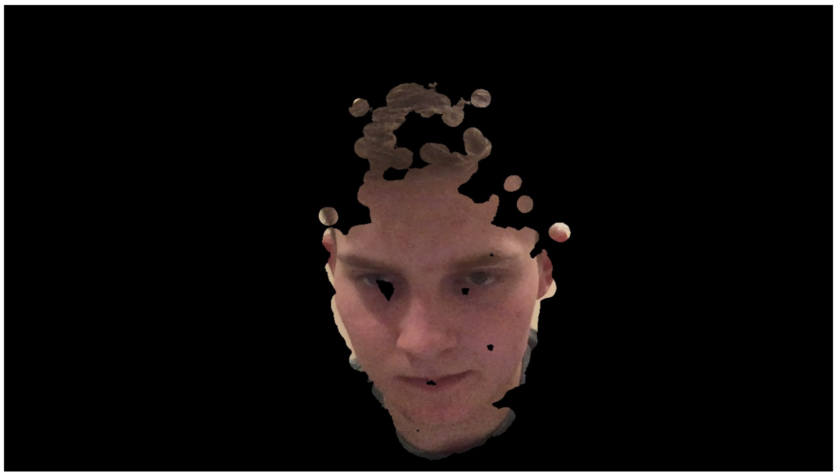

Figure 5 Since my hand is the same color as my face, the program picked out my face as well even when my hand isn't in the frame as it is here.

I also noticed that shading, for example in my palm in Figure 4 had a greater effect on my barehand than when I was using Once the image has been masked it is passed to cv2.findContours which calculates the contours in the frame. A contour is basically just a curve that joins all the points along a boundary that have the same color or intensity. This means that once the max contour in the image is found, which should be our hand unless users face is also in the frame, we have recognized the region that the user's hand is in. Now that only the hand is in play, it is easy to calculate the furthest point in within the contour from the center of it, which if you think about someone with all their fingers flat or are pointing with their index figure, are their fingertips. Finally, by storing the last twenty furthest points the movement of the finger over time can be seen in one frame. In the below images the yellow dots show where the program has identified the farthest point from the centroid of the contour. The pink dot also shows the current centroid of the contour.

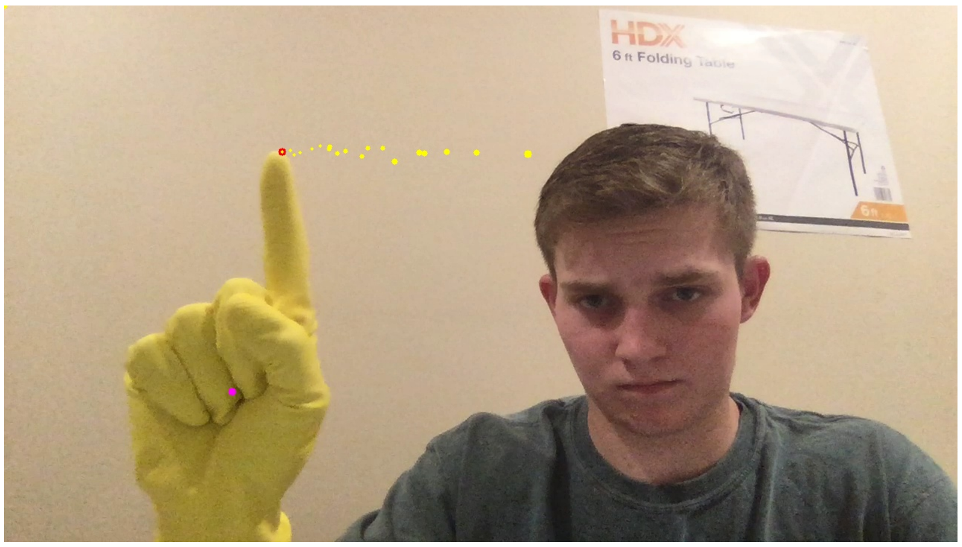

Figure 7 Dragging my finger in a line in the air.

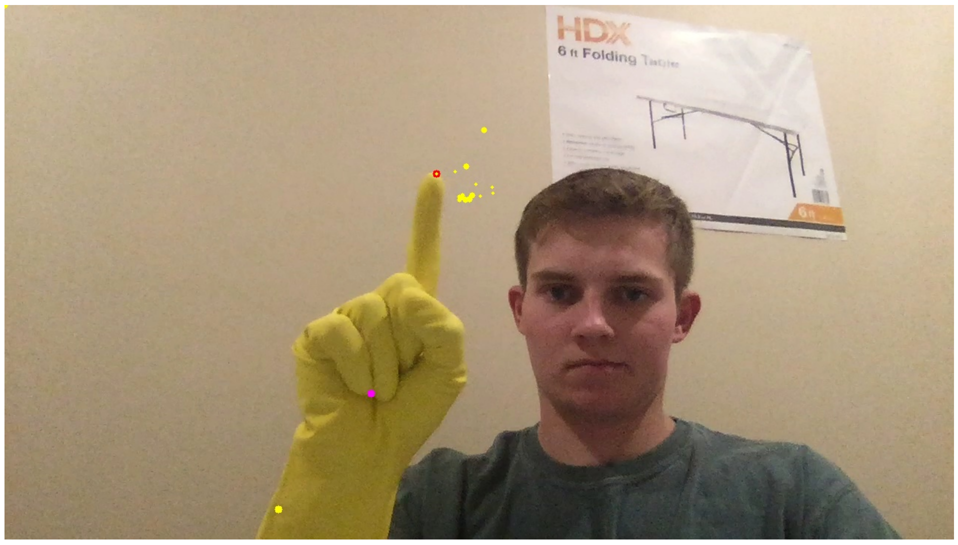

Figure 6 Moving my finger around in a circle you can see one dot below the right side of my wrist where too much forearm was in the frame.

Results with my barehand were successful but again not as good as with the glove. What I found was that if the users face is in the image then that could be the max contour instead of the hand which would lead the program to find the furthest point on the users face rather than their finger tip. Additionally, and this was a problem with the glove as well, if too much of the forearm was in the frame then the furthest point from the center of the contour, which would include the forearm, would be somewhere on the bottom of the forearm.

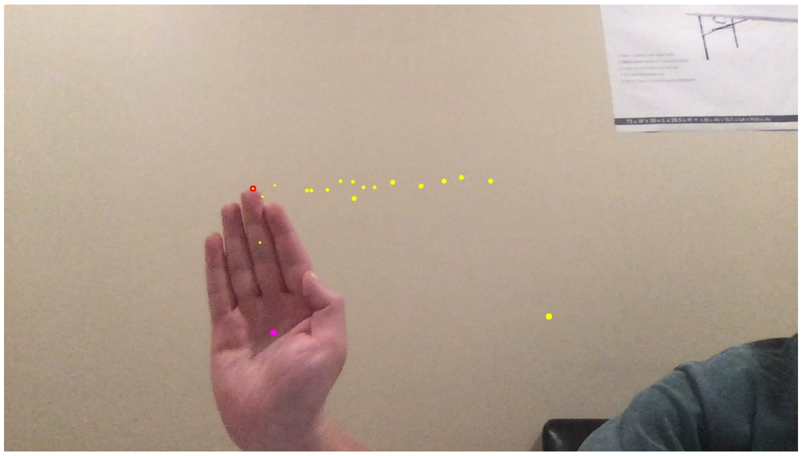
Figure 10 Dragging a line with my hand.

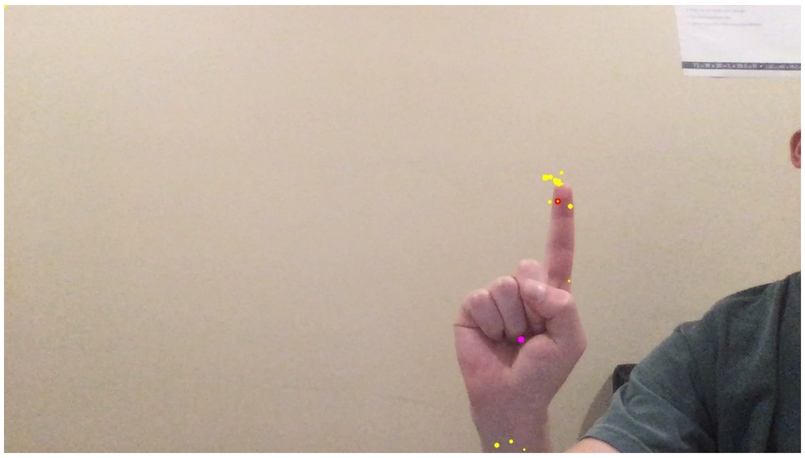
Figure 9 Identifying the fingertip when held still.

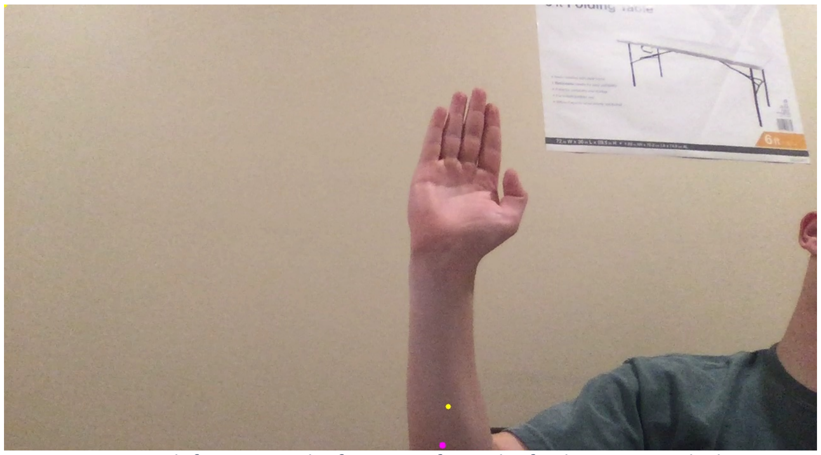
Figure 8 Too much forearm in the frame confuses the furthest point calculations.

This project was very satisfying to complete and I think that there are many possible areas of improvement that could make this even better. For example, I think that using a neural net that was trained to identify hands could use a bounding box that is more robust than our color matching masking strategy since this would be able to always ignore or cut out faces and forearms. Similarly, I think using this approach as a way to feed gesture features for complex hand movements to a neural network that could recognize them would be relatively simple way to extract even more information from these hand gestures and extend the functionality of this hand tracking program.

I also worked on extending this approach with optical flow, however I found that it was actually an inferior approach since I was extremely sensitive to noise when no fast motion was occurring in the frame. Nevertheless I was able to obtain the below image when moving my head side to side.