GIAVANNA ZAREMSKI, GZ2337

COMS W4735 ASSIGNMENT 1

VISUAL COMBINATION LOCK

Abstract

The goal of this project is to create a visual combination lock that takes three images as an input and returns a designation indicating whether the lock is unlocked or if the given combination fails. The images will be a hand performing a gesture in a location on the screen. Each image will be compared to a secret key held by the program. For each image, the gesture and location must match the secret key in the correct order. The program assumes each image is in the JPG format and is taken from the domain explained below. This paper documents the iterative process of creating this program from developing the domain, identifying different gestures, and finally performing its ultimate goal of evaluating a given set of images.

Part 1: Domain Engineering

In order to develop a standardized background with contrast between the hand making the gesture and the background I decided to use the color opposite of red, green. I used a yard of green cotton fabric and laid it across a table as flat as possible to minimize shadows. I was constrained by the only camera I currently own, which is my cell phone. It is an iPhone 12 mini and it has two 12 MP cameras, but I used the "wide" camera and not the "ultra wide" as I was able to contain the space within this camera. The "wide" camera has a f/1.6 aperture, and I captured each image as a JPG. Since I would be taking pictures of my own hand, I used a bendable tripod and chair to raise the phone and take the images from an overhead position. By taking the pictures from overhead with direct flash I was able to reduce the amount of shadows in the image. I also created a sleeve of the same green fabric to wear so that only the hand making the gesture would be visible in the image.

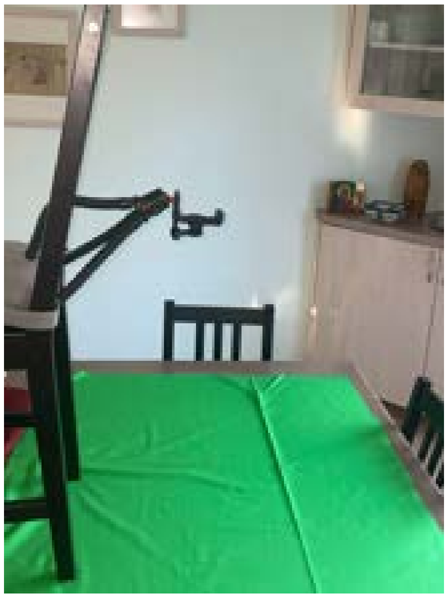


Figure 1: Environment Setup

The program environment used is a Windows Surface Studio operating a Windows 11 OS. The program is written in python using the IDE VSCode. The libraries used are OpenCV for computer vision functions, numpy for handling arrays and mathematical functions, and sys for passing arguments through the command line.

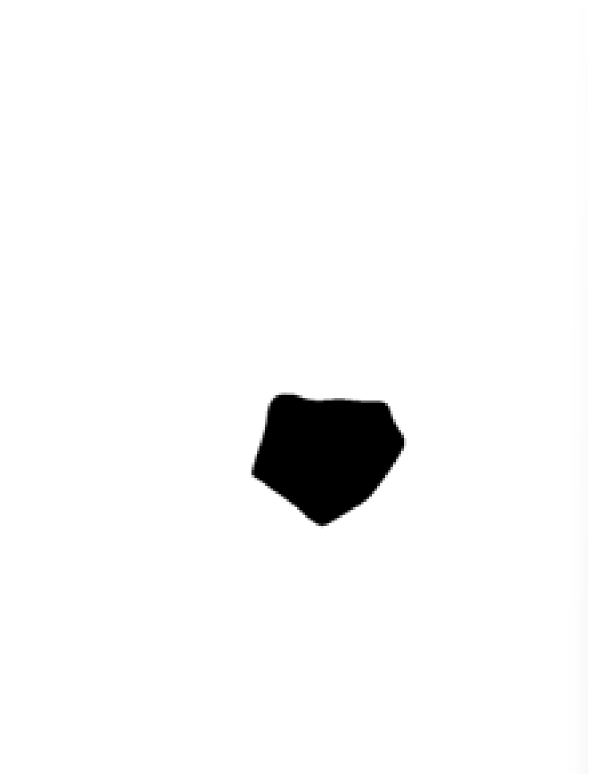
The initial library of images captured were intended to develop a robust program. The images consisted of a splay and a fist in each of the nine positions the program would be designed to identify: upper left, upper center, upper right, center left, center, center right, lower left, lower center, and lower right. Then, the library was expanded to include gestures other than splay or fist to analyze how the system would react. These images included a rock symbol, a tilted splay, a claw, and a peace sign. All the images were stored in JPG format which is compatible with the OpenCV library.

Part 2: Data Reduction

The first step to reducing the image to binary was to reduce the size of the images. The camera generated images of 3024x4032 pixels. These images were not displayable on the laptop used to run the program. So, the first step in the program was to reduce the image size by 25%. The images were still large and clear, but also could be viewed on the screen. Then, the program blurs the images. Originally, this step was not included in the program. However, initial testing revealed blurring the image reduced inaccuracy caused by shadows in the image. The next step is to filter the green from the image. The image is converted from BGR color channels to Hue-Saturation-Value (HSV) color channels. The parameters for Hue for the color green are from around 30-90. For my specific green fabric used, the most accurate filtering used a green Hue value of 36 - 100. The program then uses the OpenCV threshold function to convert the image to a binary image where the green values are mapped to 1 and the rest are mapped to 0. The result is an image with one channel where the gesture is black and the rest of the image is white.

The binary image can then be used to identify the gesture being made. In this initial iteration, the program identified only a fist or a splay. The main difference between a fist and a splay is the number of fingers that are raised, a fist has none and a splay has five. Each finger can be viewed as a contour, so the program needs to count the contours visible in the image. The OpenCV function is able to find contours in the image. The biggest contour in the image is the outline of the image, and the second largest is then the gesture. However, through image testing I determined if skin is offscreen the program will count this as one large contour and be unable to identify the gesture. However, the image was able to determine contours if the sleeve was visible in the image, i.e. the entire hand is visible in the image. After finding the contours, the program uses the OpenCV functions for finding the convex hull and convexity defects between the contour and the hull. These defects are the indents between the fingers. To determine if the program should regard these defects as a human finger the program determines if the angle is less than 90 degrees as most human fingers can not separate at an angle greater than 90 degrees. Then, the program checks if the defects are four (a splay) or zero (a fist).

The program then simply determines the location of the gesture by counting the number of black pixels in each segment of the image. The segment with the greatest number of black pixels is said to be the location of the gesture.



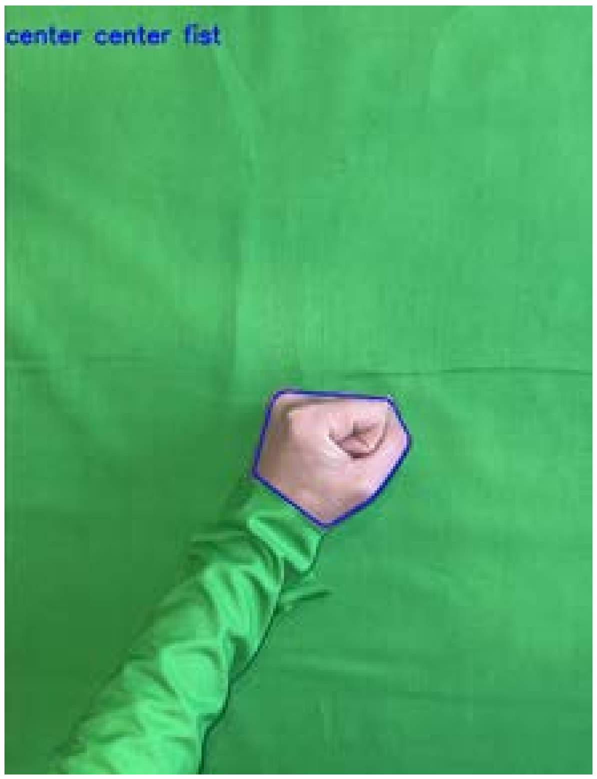


Figure 2.1: Binary Intermediate of Centered Fist. Figure 2.2 Identified Image of Centered Fist.

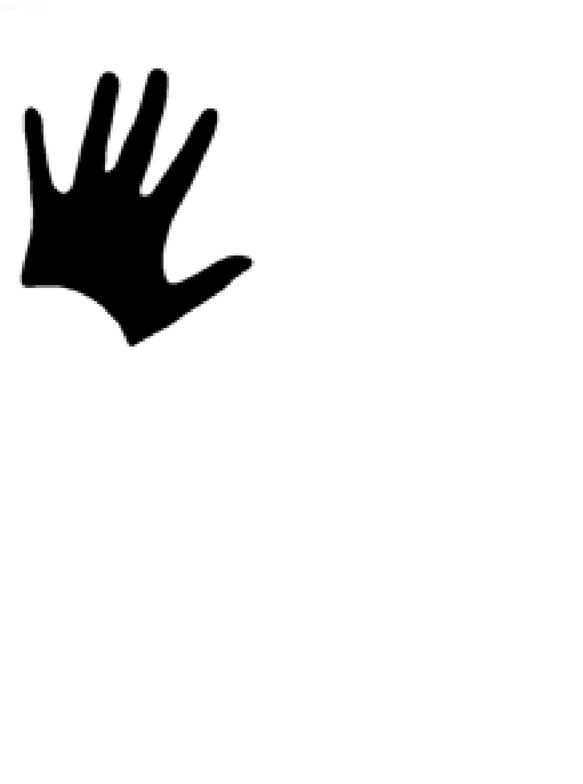
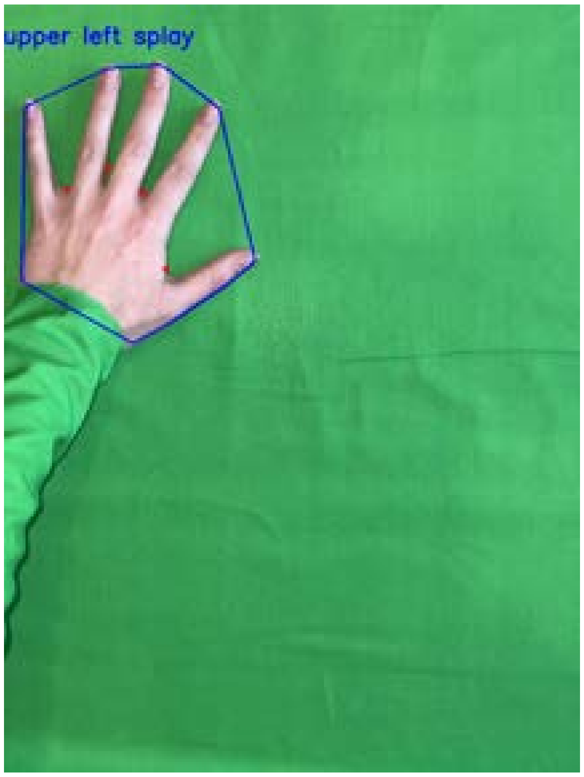



Figure 2.3 Binary Intermediate of Cornered Splay. Figure 2.4 Identified Image of Cornered Splay

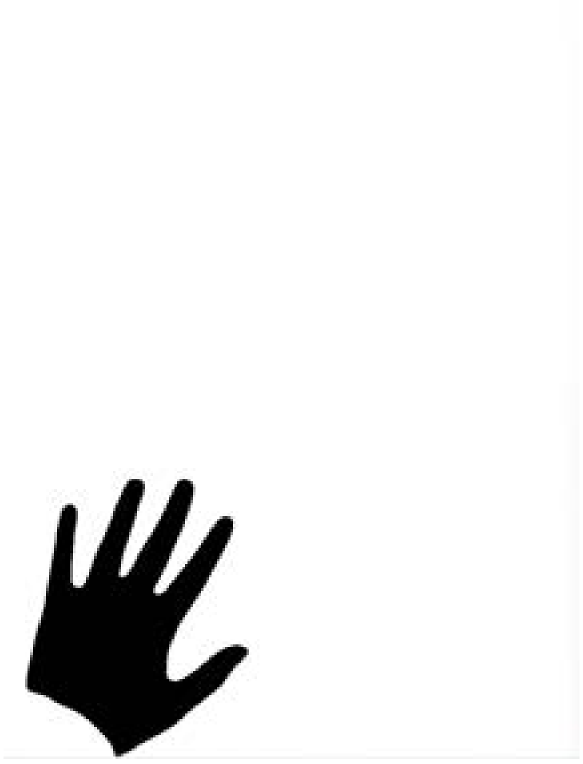
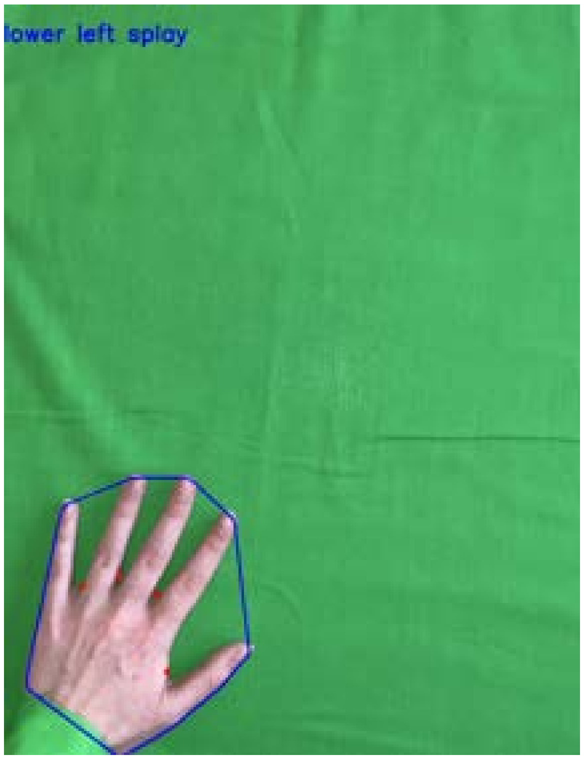



Figure 2.5 Binary Intermediate of Cornered Splay. Figure 2.6 Identified Image of Cornered Splay

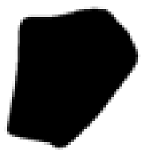
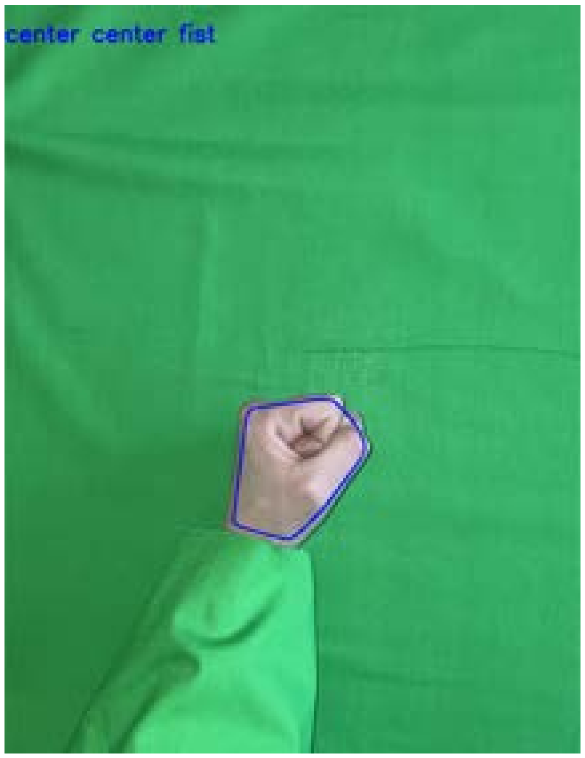



Figure 2.7 Binary Intermediate of Centered Fist. Figure 2.8 Identified Image of Centered Fist

Part 3: Edge Cases and Extension

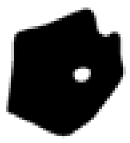
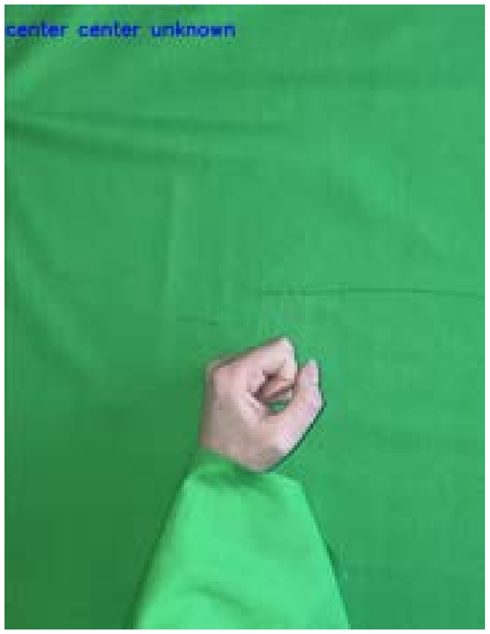



Figure 3.1: Centered fist that is not recognized as centered fist (False Negative)

Because this fist is slightly loose, the program detected the wrong number of contours and therefore classified the fist as unknown.

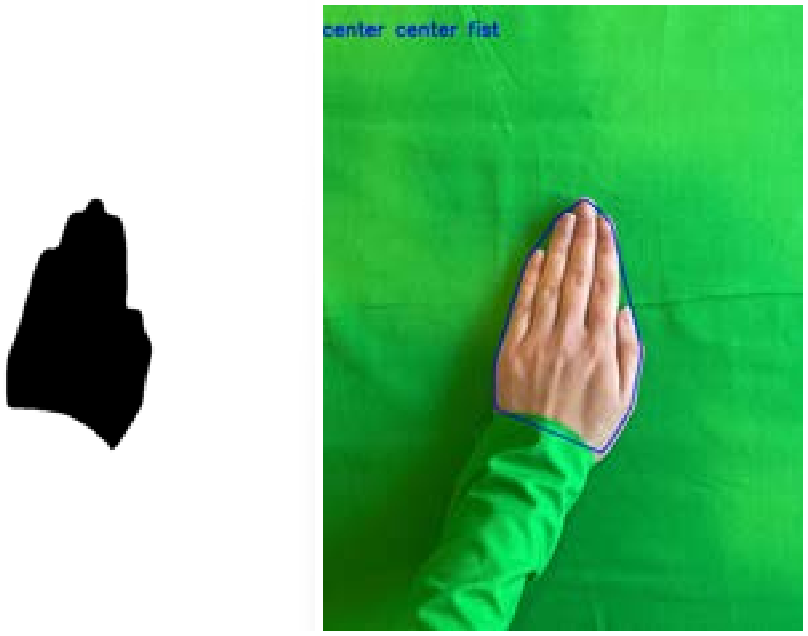


Figure 3.2: Non-centered fist that is recognized as centered fist (False Positive)

The program incorrectly recognized this flat hand as a fist because the fingers were very close together and it could not detect the defects necessary to consider it a splay. There were no defects so the program identified the gesture as a fist.

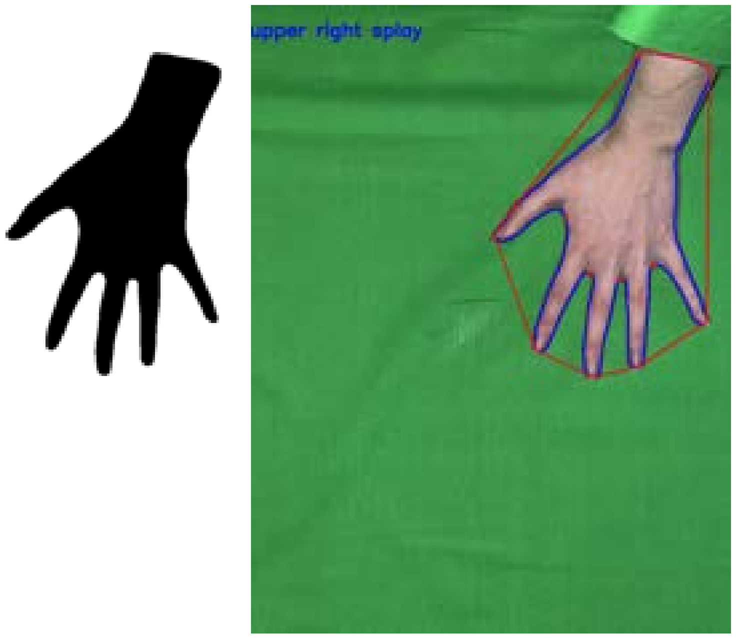


Figure 3.3: An image of a cornered splay incorrectly recognized as an upper right splay (False Negative)

This splay was intended to be a center right splay, however because more of the wrist was visible the program counted more black pixels in the upper right segment of the image, so it was regarded as an upper right splay.

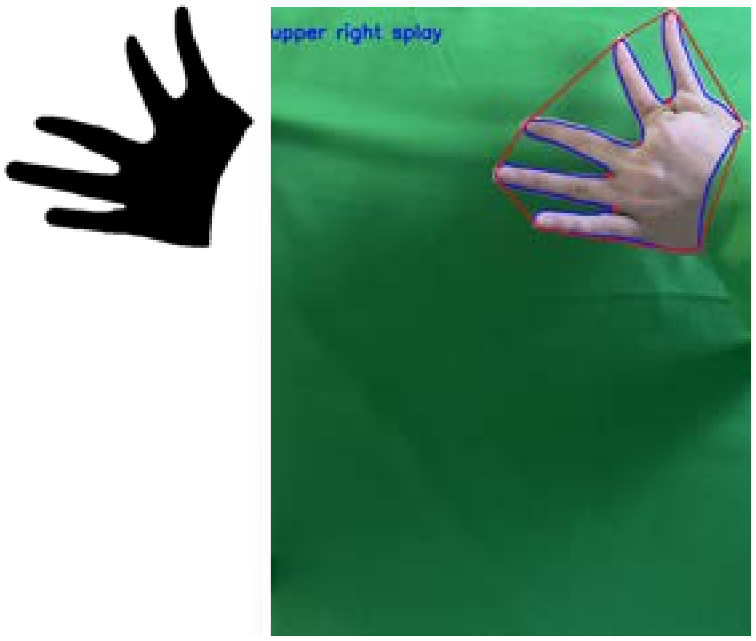


Figure 3.4: An image that is incorrectly recognized as splay upper right (False Positive)

The gesture in this image uses two hands to create four defects which causes the system to incorrectly identify the image as an upper right splay.

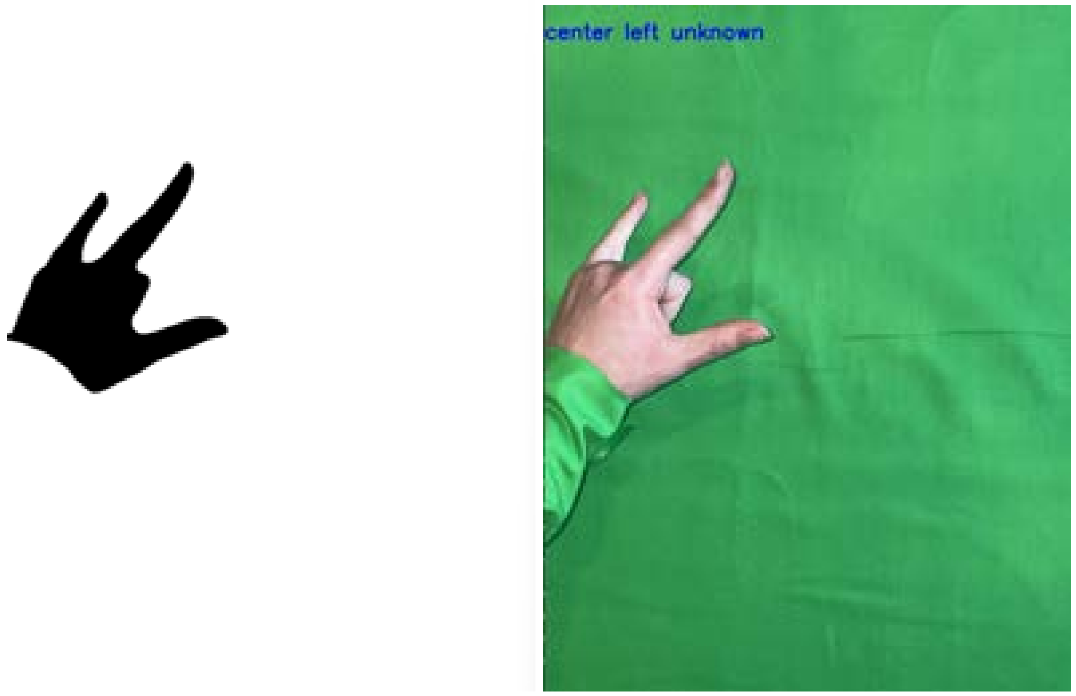


Figure 3.5.1: An image accurately recognized as unknown

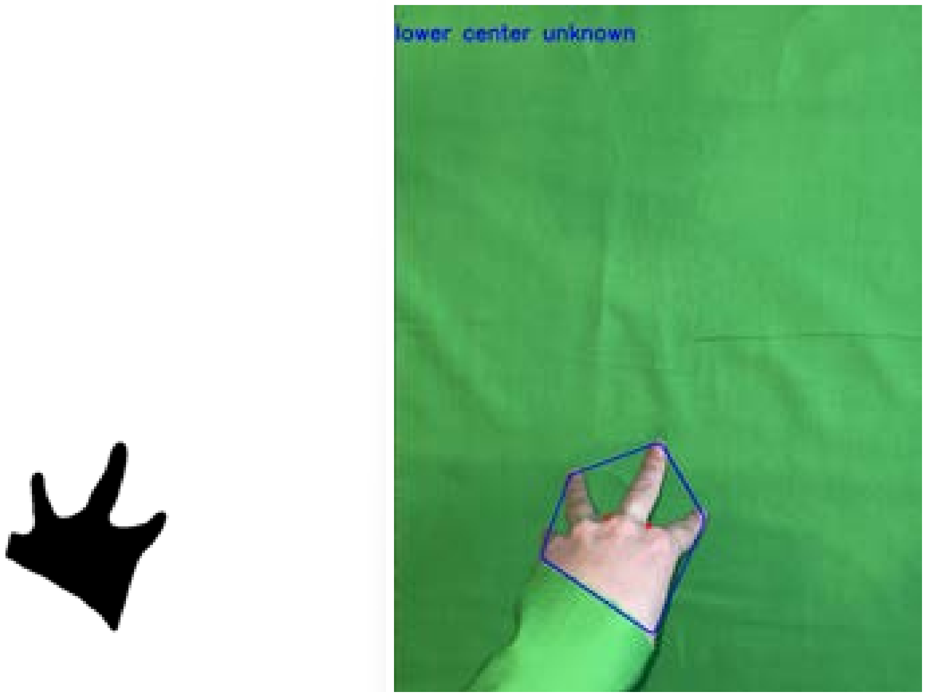


Figure 3.5.2: An image accurately recognized as unknown

To expand the vocabulary to include a palm gesture, the program needed a way to define the difference between a closed fist and palm which both can have zero defects. So, for each image the program now calculates the area of the convex hull of the gesture. Fists are smaller than a palm so this information is used by the program to define a palm vs. a fist. The values used to classify fist vs palm were defined based on two different hand sizes.

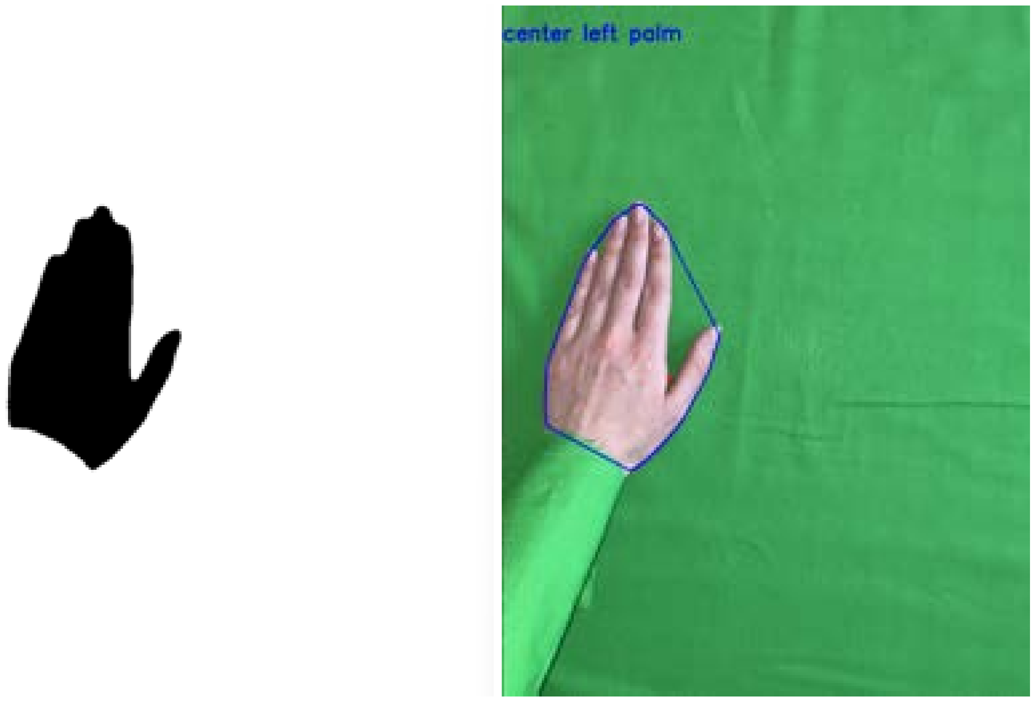


Figure: 3.6.1: An image of a palm correctly recognized as a palm (True Positive)

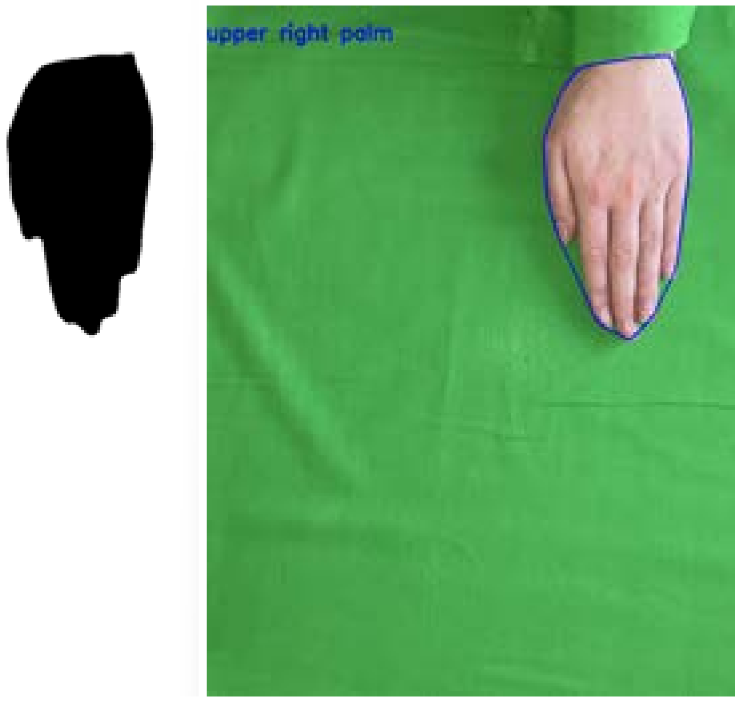


Figure: 3.6.1: An image of a palm correctly recognized as a palm (True Positive)

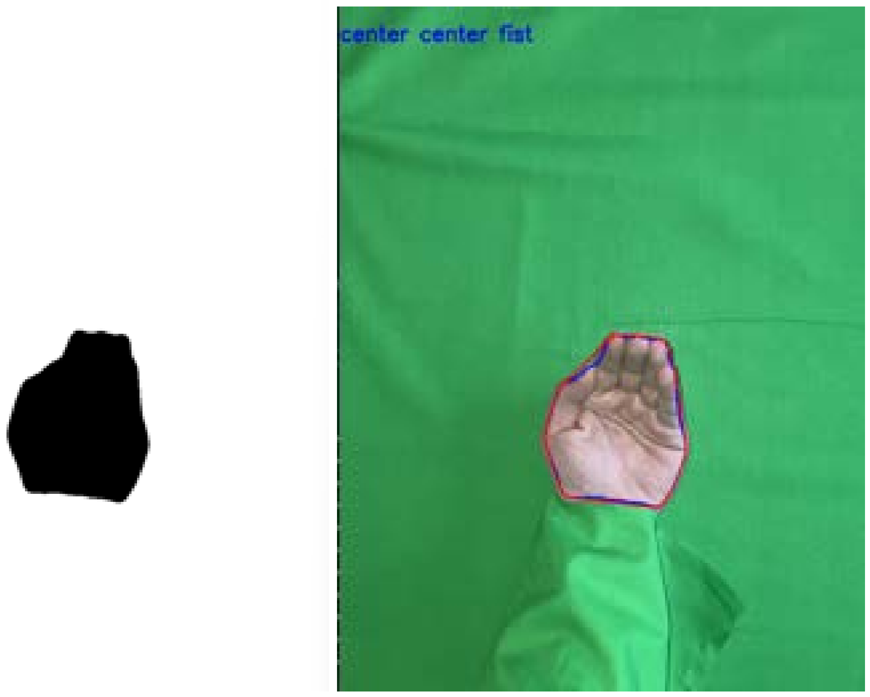


Figure 3.7: One image of a palm incorrectly recognized as a fist (False Negative)
The palm is slightly curved in this image, reducing the area of the convex hull. The reduction in area causes the program to classify the image as a fist instead of a palm.

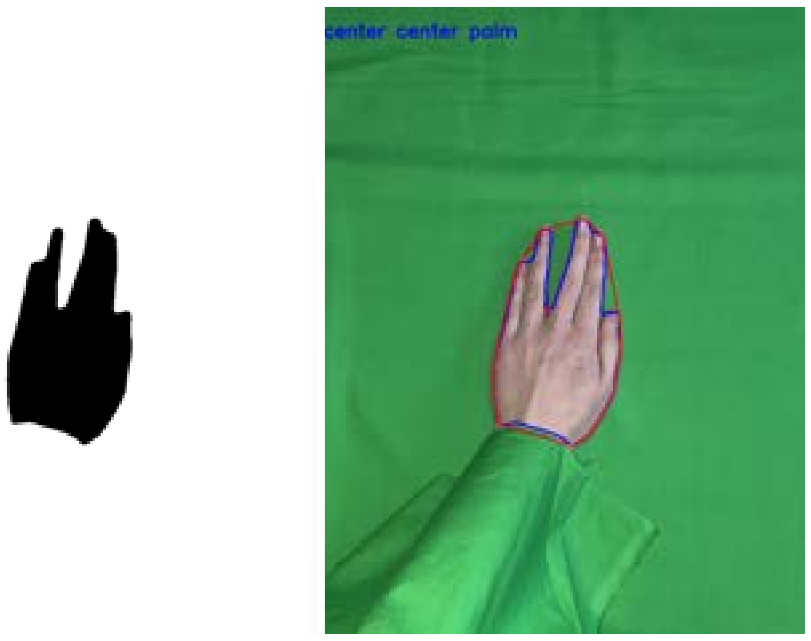


Figure 3.8: Image of splay incorrectly recognized as palm (False Positive)

If there is one finger defect recognized the system assumes this is a thumb for the palm. In this case the middle fingers are spread, so the system recognized it as a palm.

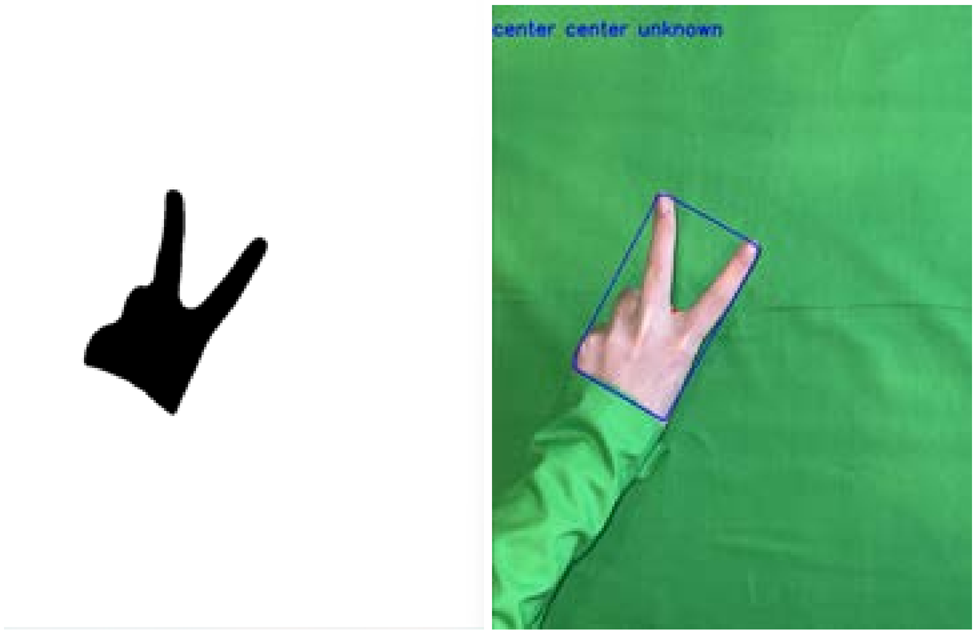


Figure 3.9.1: An image accurately recognized as unknown (True Negative)

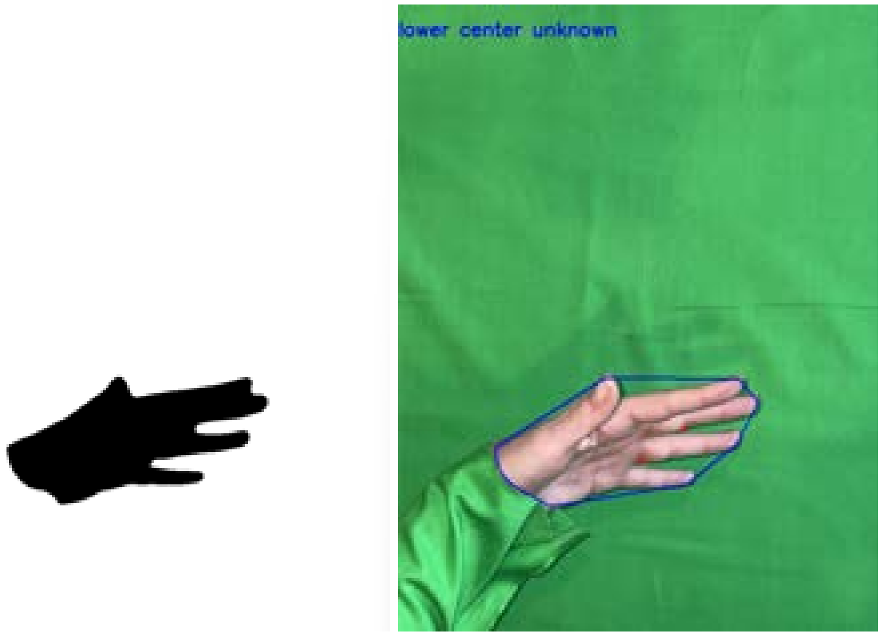


Figure 3.9.1: An image accurately recognized as unknown (True Negative)

Part 4: Evaluation

Key Sequence 1: Simple < center center splay, upper right fist, center center fist>

The first key sequence to evaluate the system is intended to be simple, both for the user to recreate and the system to identify. A splay is generally easily identified by the system since there are few gestures other than a splay that would cause four defects. However, since the splay requires stretching of the hand, the location could cause the user to place their hand outside the field of view of the camera. So, the first key image in the sequence is a centered splay. The fist is also usually identified well by the program as it is constrained by zero defects and the convex hull area being below a certain value. Since the fist is also a smaller gesture (by area) it would be unlikely for the user to place a portion of their gesture off-screen. Similarly, the fist should be easy to identify by the system in the center for the same reasons above.

Key Sequence 1: Designer Results

Three new images were presented to the system with the system designer providing the images. In the below results it can be seen the system passed, correctly recognizing each image and consequently "unlocking" the virtual lock.

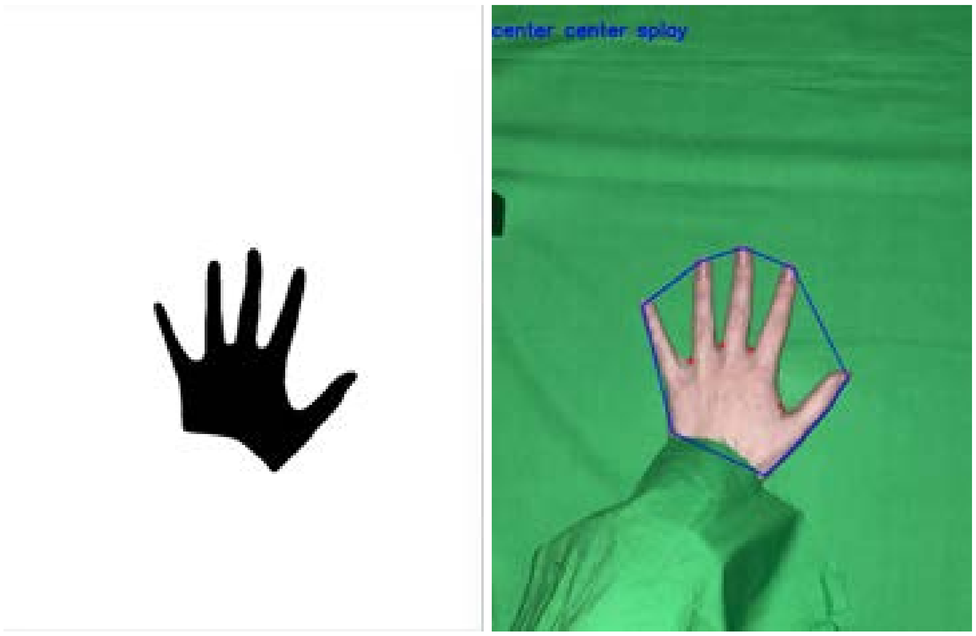


Figure 4.1.1: Designer First Image - Correctly Recognized

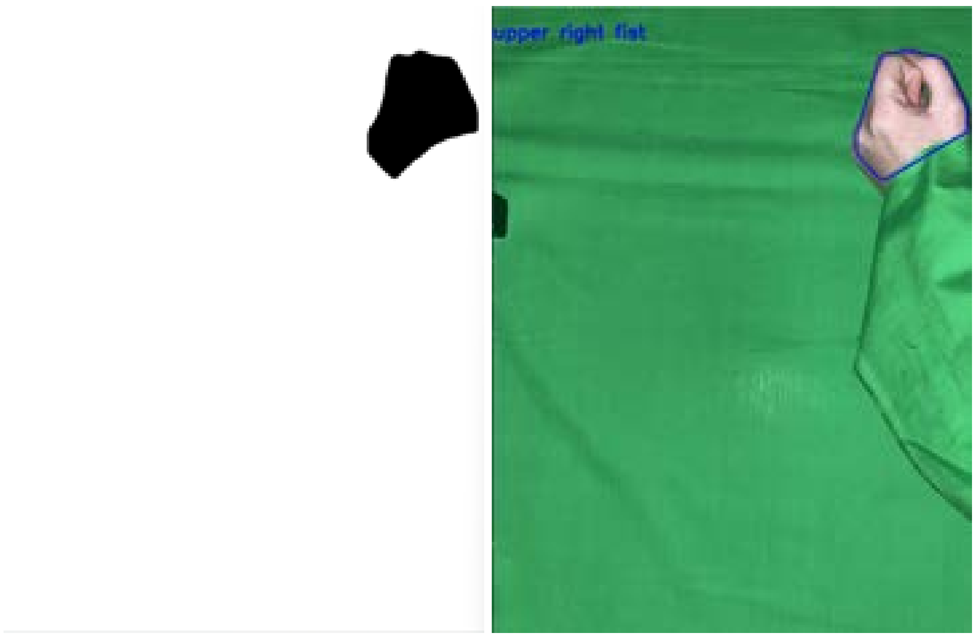


Figure 4.1.2: Designer Second Image - Correctly Recognized

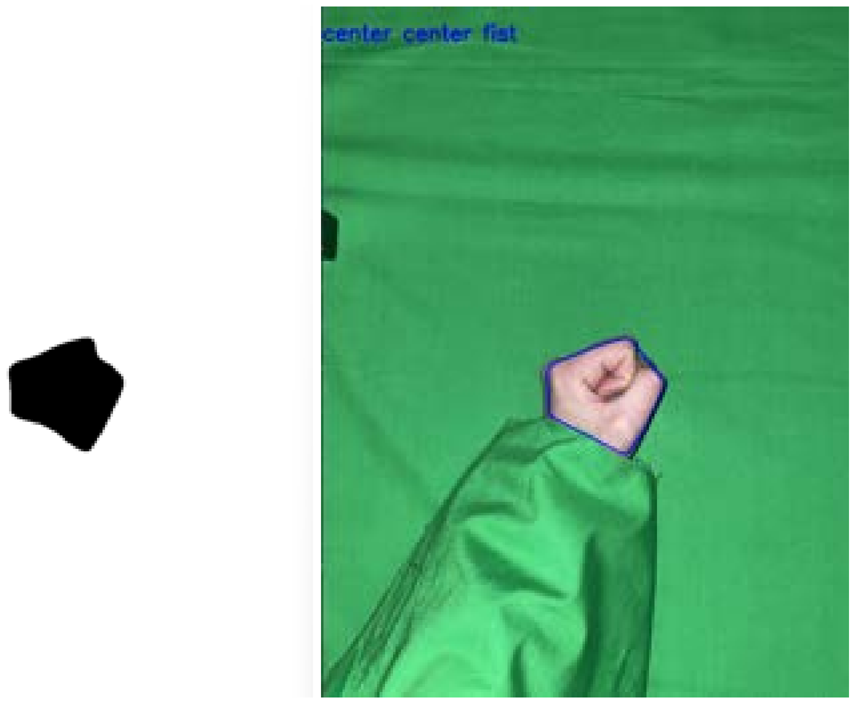


Figure 4.1.3: Designer Third Image - Correctly Recognized

Key Sequence 1: Outside User Results

The next three images were captured with the help of an outside user. The user was explained the goal of the program and told the words splay, palm, and one example image of each was provided to the user. Then, the user was asked to recreate this sequence of images. The system failed identifying the upper right fist. This was due to the fact the system assumes no defects to either be a fist or a palm but uses the area size to differentiate. Since some of the user's wrist was visible, the system counted this in the area and identified the image as a palm.

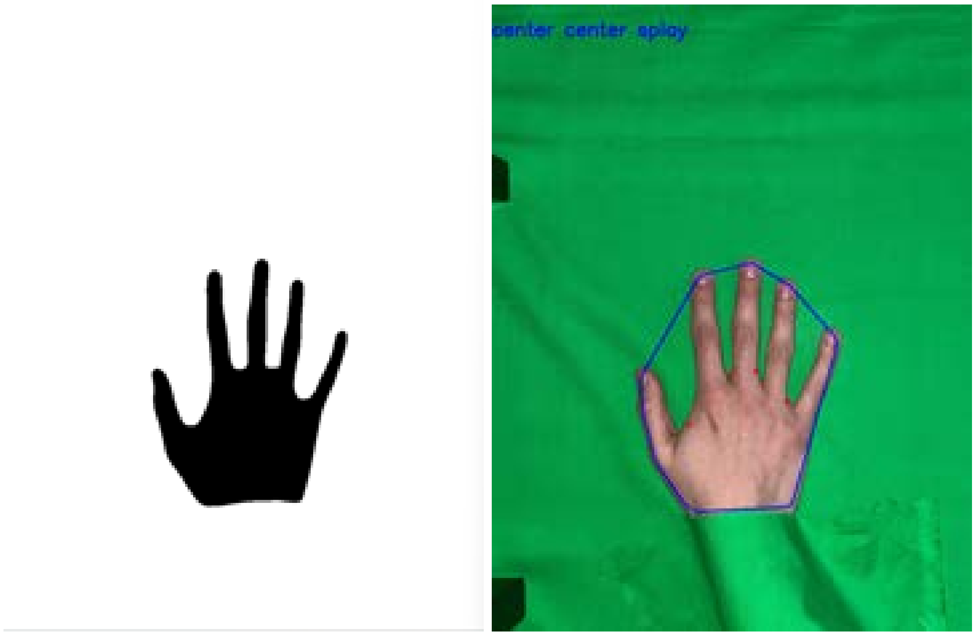


Figure 4.1.4: Outside User First Image - Correctly Recognized

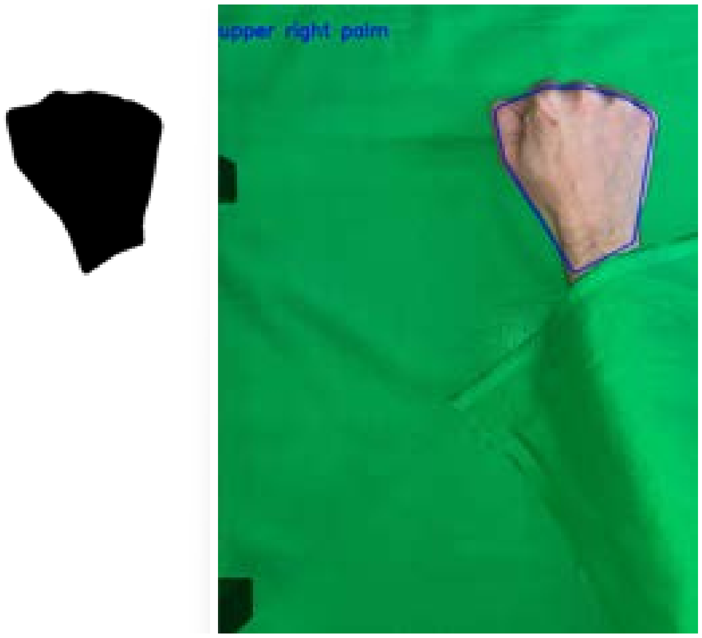


Figure 4.1.5: Outside User Second Image - Incorrectly Recognized

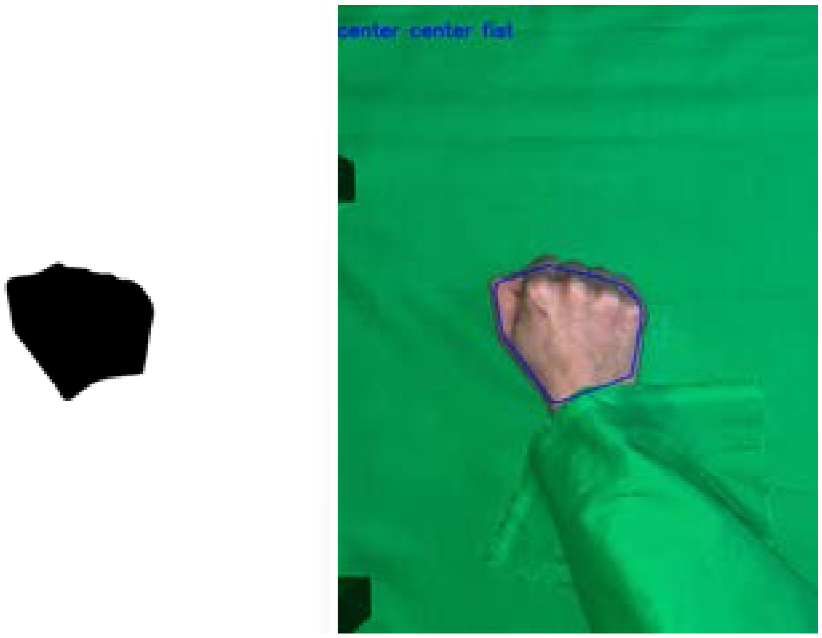


Figure 4.1.6: Outside User Third Image - Correctly Recognized

Key Sequence 2: Difficult< center left palm, center right splay, upper center palm>

The goal of this sequence is to purposefully test the weaker aspects of the visual combination lock. A user has to assume where the center of sides of the image would be which increases the difficulty of choosing the correct position. Furthermore, palms could be incorrectly identified as a fist if the user has a very small hand or holds their hand in a specific way that decreases the convex hull area. Also, depending on the user they may not splay their hand far enough apart for the system to register it as a splay.

Key Sequence 2: Designer Results

The following images are the results of the designer attempting to unlock the system with the combination detailed above. The designer was able to successfully unlock the system, most likely due to familiarity with the regions of the image and the requirements for palm versus splay.

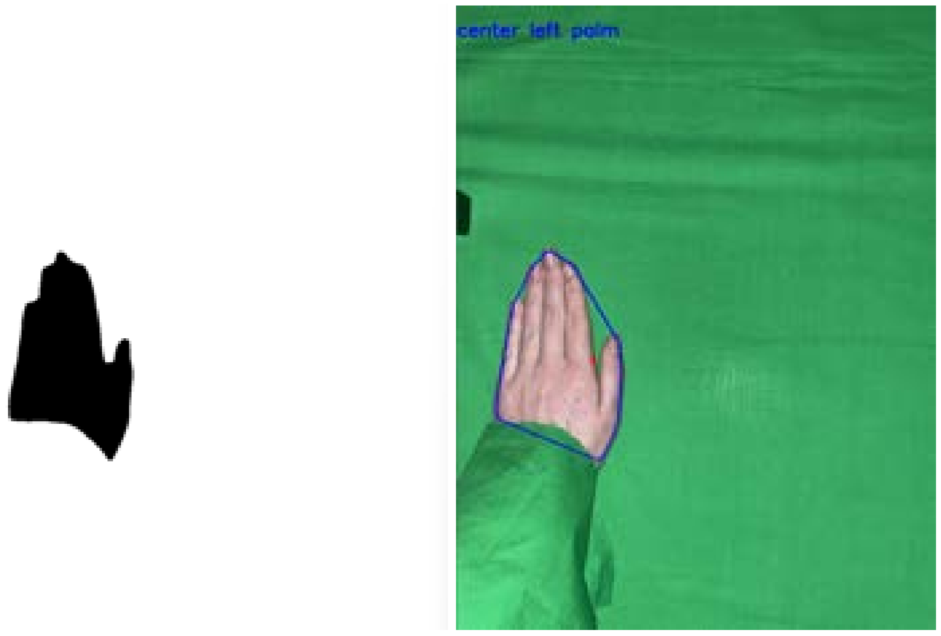


Figure 4.2.1: Designer First Image - Correctly Recognized

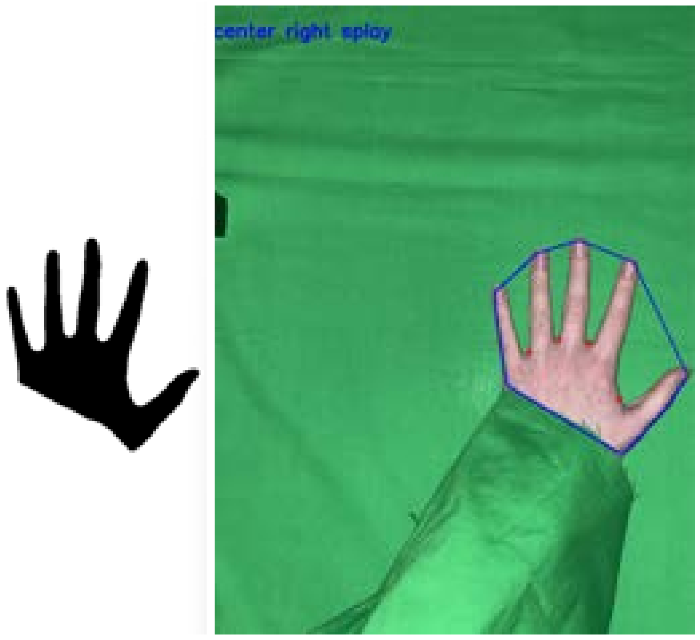


Figure 4.2.2: Designer Second Image - Correctly Recognized

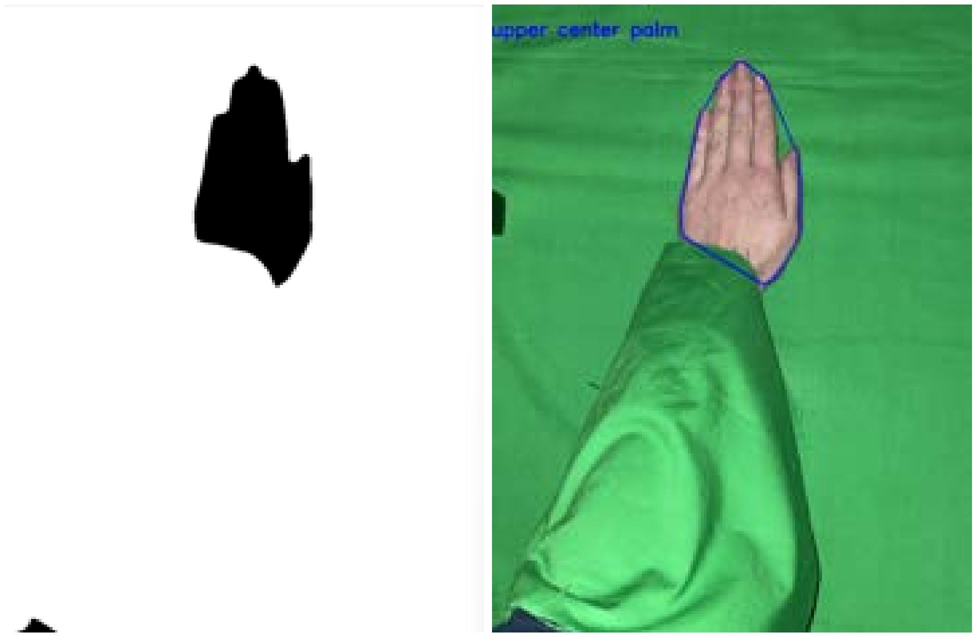


Figure 4.2.3: Designer Third Image - Correctly Recognized

Key Sequence 2: Outside User Results

The next three images were captured with the help of an outside user. The user was given the same instructions as before. The user failed to unlock the system due to their mispositioning of their gesture of the center left palm, which the system recognized as an upper left palm. Furthermore, when the user attempted a palm they slightly separated their ring and pinky fingers which would have created an issue with earlier versions of the system. Fortunately, the system accurately recognized the gesture as its intention.

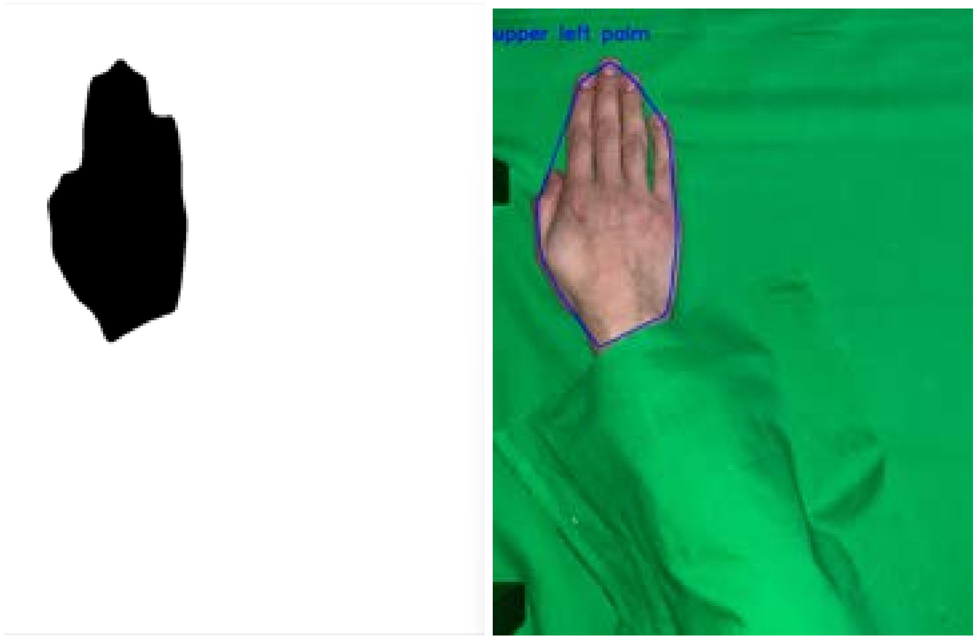


Figure 4.2.4: Outside User First Image - Incorrectly Recognized

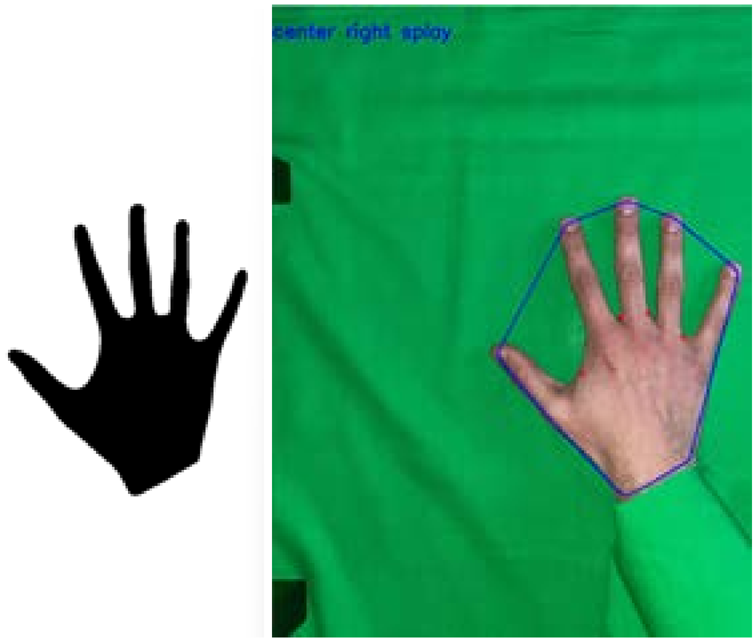


Figure 4.2.5: Outside User Second Image - Correctly Recognized

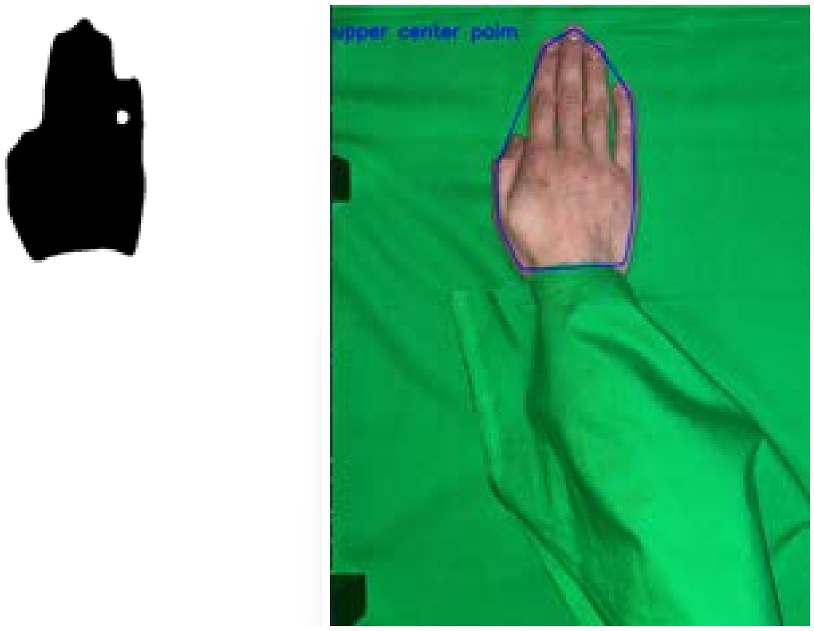


Figure 4.2.6: Outside User Third Image - Correctly Recognized

Key Sequence 3: Medium<lower left fist, center center splay, lower right palm>

This sequence intends to be a happy medium for the system. It uses more clear positioning but also uses the entire range of gestures. From the previous iterations it is clear the center positions can become difficult for a user to determine and corners and true center are easier for the user to understand.

Key Sequence 3: Designer Results

The following images are the results of the designer attempting to unlock the system with the combination detailed above. The designer was able to successfully unlock the system with the following images.

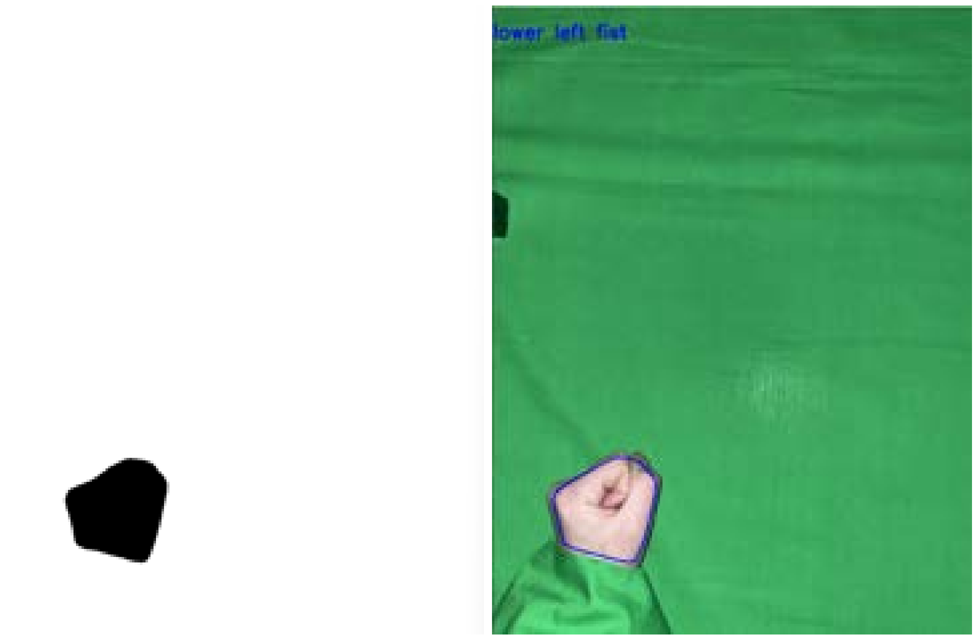


Figure 4.3.1: Designer First Image - Correctly Recognized

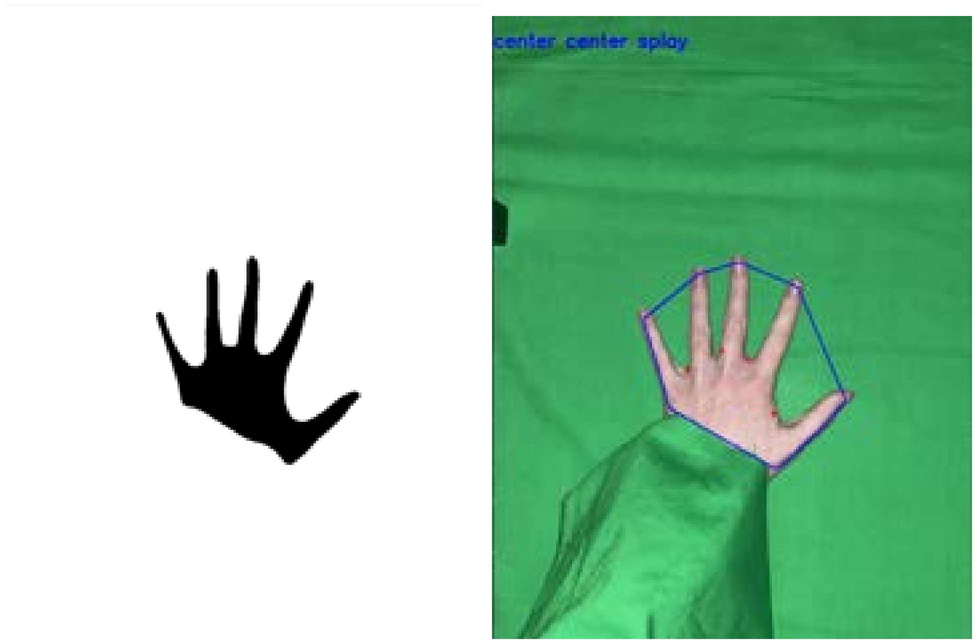


Figure 4.3.2: Designer Second Image - Correctly Recognized

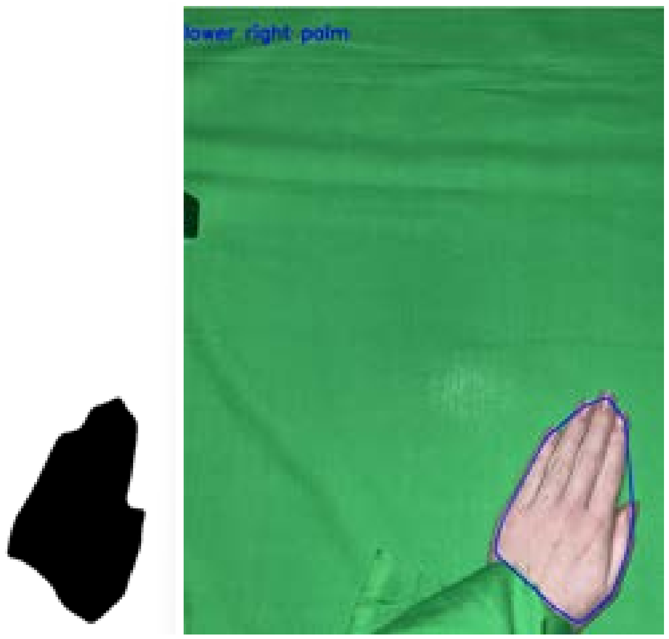


Figure 4.3.3: Designer Third Image - Correctly Recognized

Key Sequence 3: Outside User Results

The following images are the results of the outside user attempting to unlock the system with the combination detailed above. The outside user was able to successfully unlock the system with the following images.

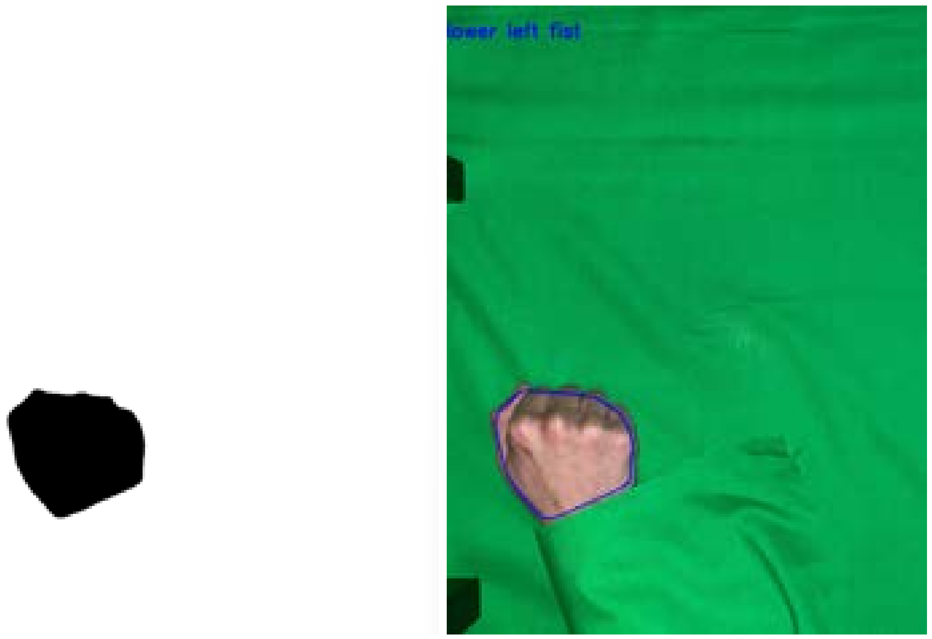


Figure 4.3.4: Outside User First Image - Correctly Recognized

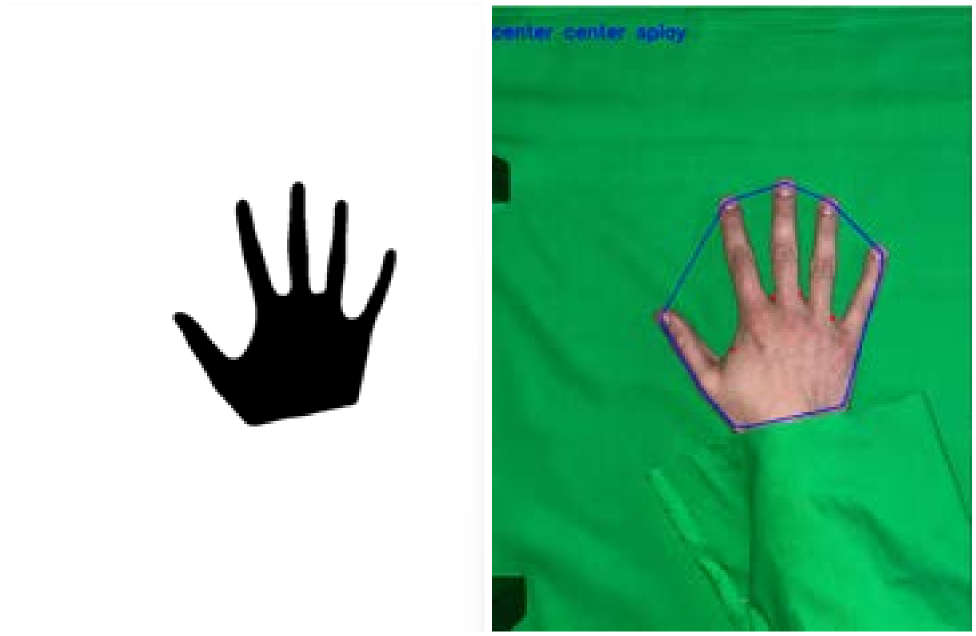


Figure 4.3.5: Outside User Second Image - Correctly Recognized

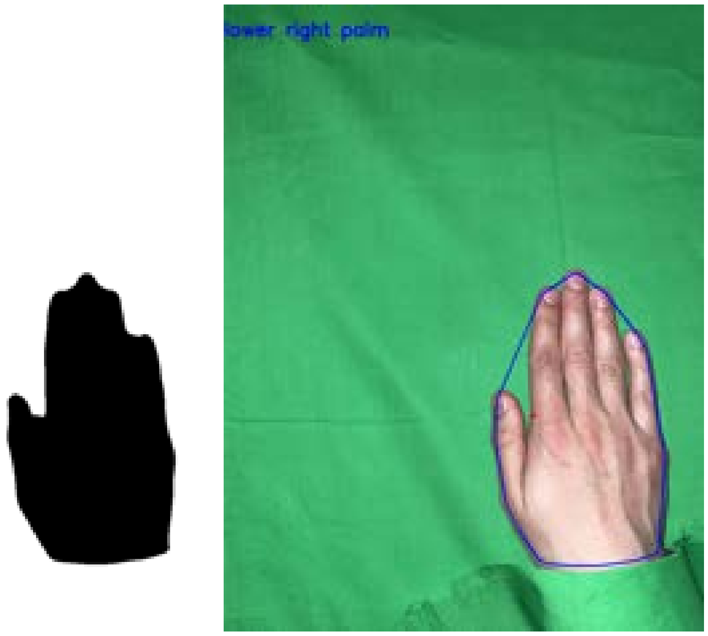


Figure 4.3.6: Outside User Third Image - Correctly Recognized

The overall success rate of the system for image recognition for the designer was 100 percent and for the outside user was 78 percent. Clearly, there is still room for improvement for the visual combination lock. A known issue is the differentiating between a fist and palm which relies on a value for the area of the convex hull of the gesture. If a palm is from a smaller hand or the fist is from a larger fist it may result in an inaccurate reading. Furthermore, any centering other than the true center could be an issue for a user in knowing where to place their hand.

The outside user recommended that the system would be more useful if there was not a necessity to use a sleeve to use the system. The user also reported some difficulty in understanding where to place their hand on the image.

In System 2.0 I would modify the system so that it could use the contour even if the hand came from outside the screen. This would eliminate the use for the green sleeve the user has to put over their arm. The user could then simply roll their sleeve up. Furthermore, I would gather more information from a range of hand sizes to increase the accuracy of a hand vs. fist. I would continue to use the method of counting the defects in order to detect fingers raised as this seemed to behave successfully for most images. Overall, the system was able to perform very well in set conditions but was vulnerable to some specific behaviors.