Hand-Gesture Recognition Using Kinect

Youn Soo Park and Jenna Warren

Boston University
Department of Electrical and Computer Engineering
8 Saint Mary's Street
Boston, MA 02215
www.bu.edu/ece

May 2, 2019

Technical Report No. ECE-2019-02

Contents

List of Figures

List of Tables

1 Introduction

Communication through hand gestures, whether it be done explicitly or implicitly, is a fundamental component of communication. Different hand gestures translate to information used to communicate with each other. For those that are hearing impaired or mute, hand gestures are used heavily in sign language to transmit information. In this project, we developed a method to correctly classify the nine hand gestures of American Sign Language representing the digits one through nine, as shown in the following figure. Using the depth detection feature of the Kinect Camera v2, we use image processing techniques to distinguish between each of the nine digits. This capability can be used in numerous applications, including laboratories, malls, and airports. The system is able to recognize hand gestures indicating numbers which could help a person enter numbers into a computer without physically touching the computer.

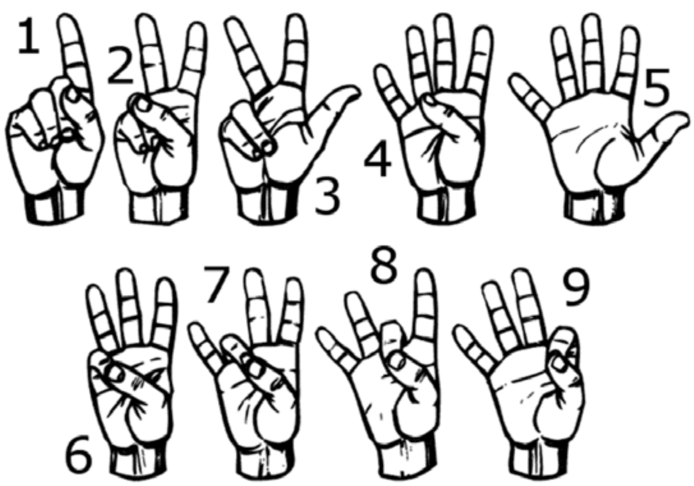

Figure 1 American Sign Language of numbers 1 through 9

In this project, we implemented two algorithms for classifying the nine hand gestures and compared the results.

2 Literature Review

Object recognition and classification is typically accomplished using one of the approaches described in the following sections. These approaches can generally be divided into two categories: end-to-end methods, and feature extractors. End-to-end methods include a method for classification as part of the algorithm. Feature extractors require a classifier that uses the extracted features to determine the class of a test input. Many classifiers require a dictionary of reference features to compare against.

2.1 Feature Extractors

2.1.1 Convexity Approach

The convexity detection consists of several steps: filtering the depth image, extracting hand contours, approximating the hand contour with a polygon, detecting concave and convex points of the approximation polygon, and finally, filtering convex and concave points to distinguish different gestures. The final goal of this approach is to find the convex points from the contour. The convexity approach extracts features of an image, which contains only the hand contour, one at a time.

To begin, we preprocess the depth image using a filter to get rid of noise near the hand. The hand contour is defined by the connection of edges. Next, we compute a convex hull for the hand using all the convex vertices in the hand contour. Then we compute the difference between gesture convex hull and contour which is the convex defect. Simply, it is the task of obtaining the point farthest away from each convex vertex. Different gestures are distinguished from one another by the relative position of fingertips.

2.1.2 Covariance Matrix

Covariance matrices describe the relationship between different components of a random variable. Covariance matrices can be used in applications to measure similarity between different aspects of an image.

To use covariance matrices to classify hand gestures, we first must generate a database of covariance matrices for each gesture that we hope to identify. These covariance matrices are computed from feature vectors describing each of the hand gestures. [3]

After assembling the database of covariance matrices, to classify a test image, we first must extract the same information from the test image to generate a feature vector containing the same data used in the database. Some features included in the feature vector may require additional preprocessing of the image.

The final step is to classify an input image. This can be done in several ways. One way to accomplish this is to use a nearest neighbor classifier which computes a distance between the covariance matrix of the input (test) image, and the covariance matrices of the database. There are numerous distance metrics that can be used for the classification calculation. The covariance matrix in the database that is closest to the input image based on the chosen distance metric is then used as the label for the hand gesture in the test input image when using nearest neighbor classification. Other classifiers, such as k-Nearest Neighbors, SVM, or decision trees could also be used in conjunction with the covariance matrix feature extractor.

2.2 Dynamic Time Warping

Dynamic time warping is an algorithm that finds an optimal alignment between two time-varying sequences under certain restrictions. The warping path finds the minimum distance to align the sequences. To find the best alignment between the two signals, the algorithm finds the path through a grid which minimizes the total distance between them. To create a mapping between the two signals, a path is created. The path starts at (0,0) and ends at (M,N) where M and N are the lengths of the two signals, as shown below.

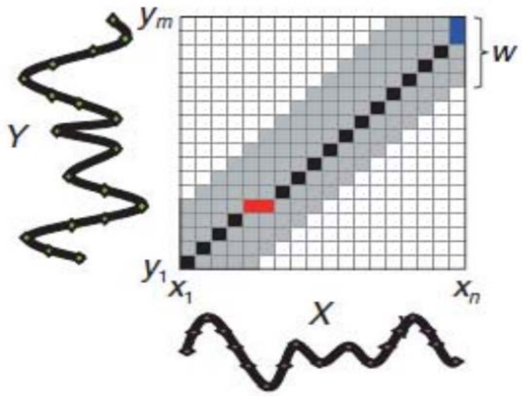

Figure 2 Dynamic Time Warping from Kadethankar and Joshi (2017)

In most cases, dynamic time warping is used to align time-varying signals. In particular, dynamic time warping is often used in speech recognition systems. However, in this project, which deals with static images, we must use a different feature vector to represent the images.

2.3 End-to-End Methods

2.3.1 Implicit Shape Model

The implicit shape model is a probabilistic framework that combines both image segmentation and recognition [4].

The first step using this model, is to generate a database of small segments of interest in images representative of the possible classifications of the test images. Input test, images are also decomposed into small segments of interest and compared against the database.

The image segments of the input test image are compared against image segments in the database corresponding to the same area of the image. The algorithm determines which area of the original image a segment came from based on a probabilistic model generated from the original database.

The final classification of an image is determined through a vote. Each segment of the test image is classified as belonging to one of the hand gestures, and the class with the most votes is determined to be the gesture of the original image.

Because this method works on smaller portions of an image rather than the whole image, it can be more accurate using fewer samples than other probabilistic methods.

2.3.2 Circle Method

To use the circle method for hand gesture recognition, as described in [2], one first must detect the hand in the image. Using thresholding, or another form of image segmentation, the image is transformed into a binary image such that the hand is displayed as white (pixel value equal to 1), and everything else is displayed as black (pixel value equal to zero).

The next step is to detect the center of the hand, and draw a circle centered at this point. The edge of this circle intersects the fingers and wrist of the hand.
By detecting the values of the image at all pixels in the image along the circle, “spikes” can be observed at locations where the circle is intersecting a finger or wrist of the hand.

In the experiments conducted by Malima [2], the algorithm detected the number of fingers that were being held up, but therefore was limited to distinguishing between hand gestures displaying between one and five fingers. This method is summarized in the following figure.

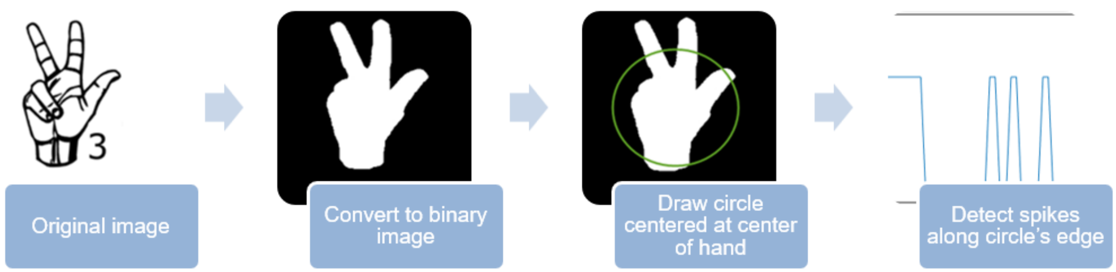

Figure 3 Summary of circle method

2.3.3 Deep Learning

Deep learning, neural networks, and convolutional neural networks remain some of the most successful methods used in image recognition tasks today. These methods attempt to mimic how the human brain works and detect patterns present in labeled training images. The network applies these learned patterns to test images.

While these methods can be extremely effective, they require a large amount of labeled data. For this reason, we will not be studying deep learning or neural network methods in this project.

3 Implementation and Results

3.1 Data Collection

Using a support package for Kinect, both color image, and depth image data were loaded directly into MATLAB. The data was collected from several different people to ensure that the implemented methods are robust to small changes in hand configuration, hand size, and distance from camera. The data contained 10 images for each of the nine gestures and they were collected from 4 different people. In total, 40 images were collected for each of the nine gestures that will be distinguished between.

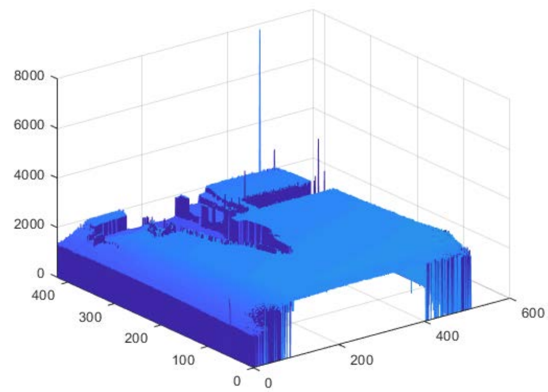

Figure 4 Example of original image and depth image

3.2 Preparing Data

Both methods implemented for classifying the hand gestures required a binary image as input data. To create a binary image of the hand, the depth of the hand was detected from the depth image. This process assumes that the hand is always the closest object in the depth image. By sorting all depths in the image, the mean of the first 20 values is taken as an initial threshold, T_i. All depths in the image larger than $T_i + 10$ are set to zero. The value 10 (millimeters) represents an approximate depth of the hand. All remaining depths are assumed to be part of the hand and pixels at these locations are set to 1, creating a binary image.

In some images, this process leaves some noise that affects the boundary identification, and therefore the classification process. To address this noise, white segments consisting of fewer than 50 pixels are set to 0 using the `bwareaopen` function in MATLAB.

Figure 5 Summary of getting the boundary

In most cases, this process produced a binary image of the hand that clearly showed the hand gesture. In other cases, the resulting binary image contained issues that would affect the classification. Some of these issues included:

- Sleeve captured as part of the binary image
- Fingers too close together resulting in one large finger in the binary image
- Due to hand placement, the wrist was at the same depth as the rest of the hand, making the overall boundary of the hand significantly different than other samples
-

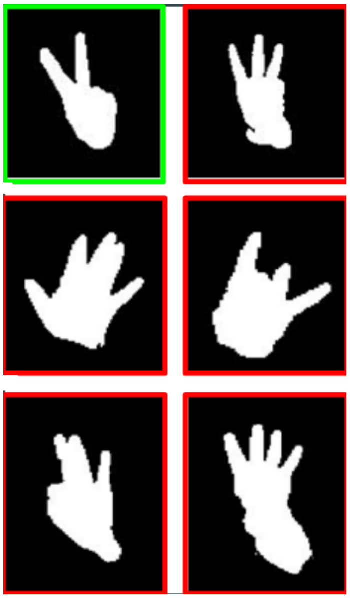

Figure 6 Examples of good (green) and bad (red) samples

3.3 Circle Method

Using functions from the Image Processing Toolbox in MATLAB we find the center point of the hand, and the points along the boundary of the hand. This information is used to draw a circle centered at the center point of the hand. Through experimentation we determined that using a circle with a radius that was 75% of the distance from the center of the hand to the farthest point in the hand provided the optimal results.

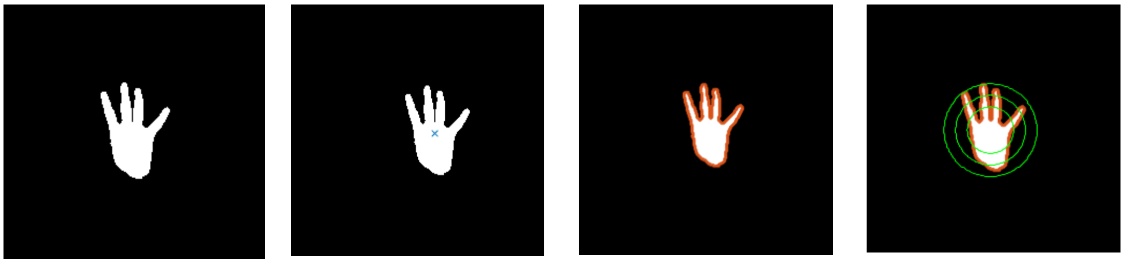

Figure 7 Steps to implement circle method

By retrieving the pixel values of the image at points along the circle, we get a signal like the following plot. Using the findpeaks function, we count the number of peaks in the signal, which represents the number of times the circle crossed the hand. To account for the wrist, we subtract one from this number to get the final classification of the hand gesture.

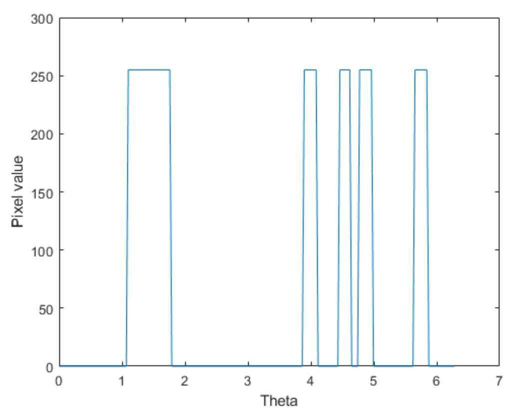

Figure 8 Plot of pixel values as a function of angle values. The plot shows a wide spike indicating where the circle crosses the wrist and four spikes indicating where the circle crossed the fingers.

3.3.1 Circle Method Challenges

The circle method as described handles cases where the number of peaks in the final signal directly corresponds to the number of fingers, but it does not account for situations where the number of fingers held up in the image do not correspond to the hand gesture, such as the case of hand gestures 6-9, which all contain three fingers held up in different patterns.

To account for this, two methods were introduced.

3.3.1.1 Analyze the spacing of the peaks

To distinguish between the hand gestures 3, 6, 7, 8, and 9, we implemented a method for analyzing the spacing between the peaks in the signal. There are subtle differences between these signals the data was not consistent enough for this approach to be successful. In addition, simply stretching out the fingers on the hand would result in a misclassification.

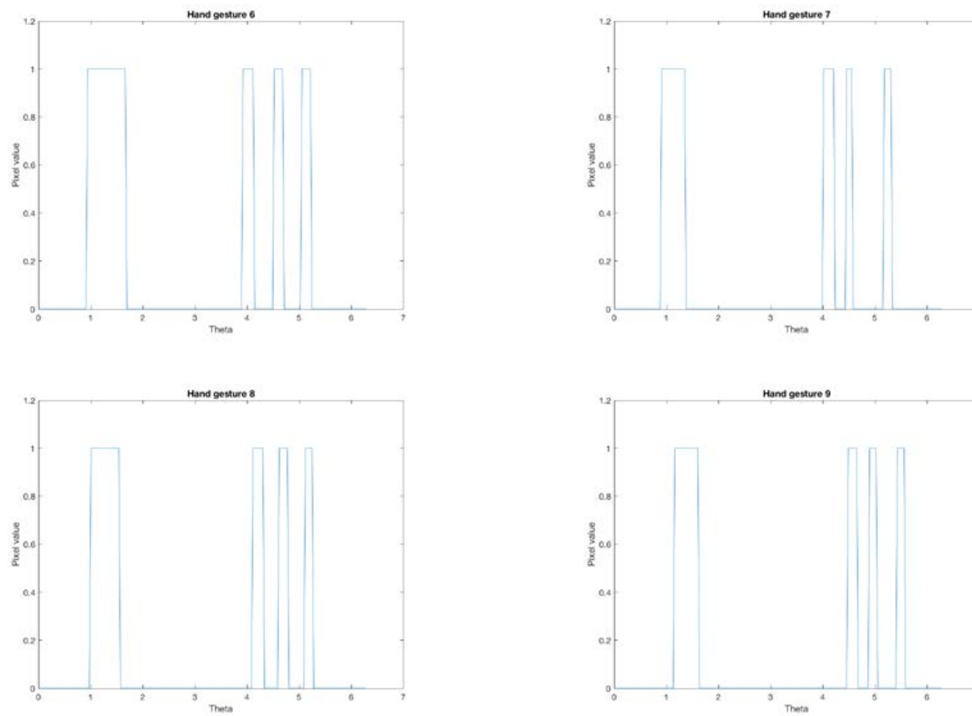

Figure 9 Comparison of plots from hand gestures 6-9.

3.3.1.2 Analyze the image using concentric circles

Because the thumb and pinky fingers are shorter than other fingers, we attempted to use a series of concentric circles to distinguish between gestures that contained these fingers.

This method was marginally successful at identifying gesture 3 (thumb and first two fingers) and 6 (three fingers) for a small set of data all from a single person. However, it was not at all effective once the hands of other people were introduced into the data. This was due to differences in proportion between the sizes of different fingers on their hands.

In addition, this method was never able to distinguish between gestures 7, 8, and 9, which all contain the pinky and two fingers.

3.3.2 Circle Method Results

The overall performance of the circle method was best when simply using the number of peaks. The accuracy of the circle method for each of the hand gestures is shown in the following table. We tested forty depth images for each of the nine gestures, for a total of 360 trials. As shown in the table, the algorithm was generally successful for gestures 1-5. As expected, gestures 6-9 were always classified as gesture 3, and therefore never correct.

Cases where the algorithm misclassified gestures 1-5 were all a result of the thresholding issues discussed in the previous section.

Hand gesture	Accuracy rate
1	✓ - 100%
2	✓ - 85%
3	✓ - 82.5%
4	✓ - 95%
5	✓ - 95%
6	✗ - 0%
7	✗ - 0%
8	✗ - 0%
9	✗ - 0%

Table 1 Results of circle method based on forty trials

3.4 Dynamic Time Warping

The dynamic time warping algorithm takes two time-varying signals as inputs. It attempts to align the two input signals by warping them and finally returns a distance between the warped signals.

```
[distance, warpedSignal_1, warpedSignal_2] = dtw(signal_1, signal_2)
```

To apply the dynamic time warping algorithm to the binary image data, a 1-dimensional signal had to be extracted from the image. We began by using the x and y coordinates of the boundary of the hand as an input signal. As in the circle method, the boundary coordinates were found using built-in functions from the Image Processing Toolbox in MATLAB. We then construct a complex number from each pair of coordinates.

Sample signals of hand gestures are shown in the following figures.

Binary images | *X-coordinates of boundary of hand* | *Y-coordinates of boundary of hand*

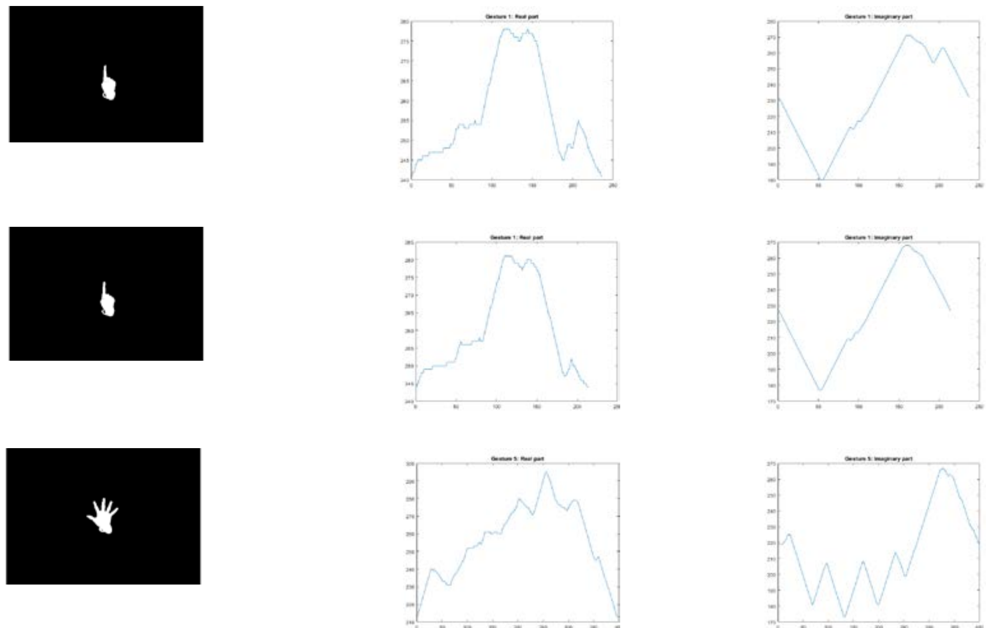

Figure 10 Comparison of X and Y signals of gestures 1, 1, and 5

The second column represents the x-coordinates of the boundary of the hand. The third column shows the y-coordinates. You can see in these plots that the shape of the signals from the two images depicting gesture 1 are very similar. There is a clear difference in the signals from gesture 5.

The following figures show sample output of the dynamic time warping algorithm. The plots on the top row show the original signals. The second row shows the signals after applying dynamic time warping. The plots on the left compare the two images depicting hand gesture 1. The plots on the right compare gestures 1 and 5. The squared distance between gesture 1 and 1 is approximately 500 after warping, while the squared distance between gesture 1 and 5 is approximately 9000. These distances can be used to indicate which of the signals come from the same hand gesture.

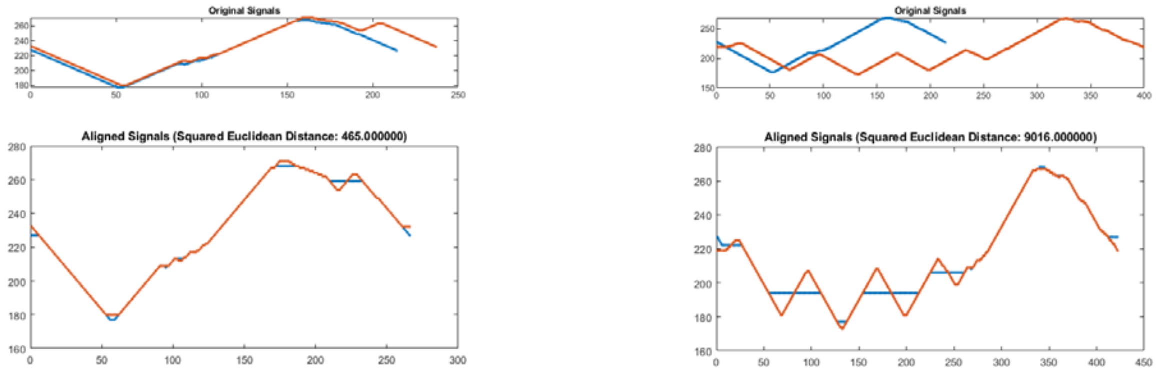

Figure 11 DTW of sample signals: Original signals shown on top and the aligned signals at the bottom. Left plots show gestures 1 (blue) and 1 (red). Right plots show gestures 1 (blue) and 5 (red).

The following table illustrates sample distances between two signals using one sample for each gesture. With this sample data, distances between warped signals of the same gesture are always smaller than the distances between warped signals of two different gestures.

MSE	1	2	3	4	5
1	137.875	1.0978e03	1.9986e03	8.0738e03	1.1695e04
2	1.0978e03	448.034	1.4146e03	3.9783e03	8.2403e03
3	1.9986e03	1.4146e03	1.3663e03	3.5072e03	4.0906e03
4	8.0738e03	3.9783e03	3.5071e03	582.8073	1.6694e03
5	1.1694e04	8.2403e03	4.0906e03	1.6695e03	135.9483

Table 2 MSE of two signals showing that the same gestures have the minimum distance

We allocated 28 reference samples for each hand gesture (7/10 from each of the four people we collected data from) to generate a dictionary against which we would compare test images.

For each test image we find its feature vector, and then apply the dynamic time warping function (dtw from the Signal Processing Toolbox in MATLAB) on this feature vector and each feature vector contained in the dictionary, resulting in a distance between the two warped signals. To classify the test image, we take the mode of the classes of the 8 closest feature vectors after warping.

Three different distance metrics were tested: Euclidean, absolute, and squared distance. These distance metrics are described by the following equations. In each of the following equations, let the vector V represent a signal obtained from a test image, and let the vector W represent a signal from the dictionary. V and W are both K-dimensional signals composed of a number of samples. For example, $V_k[m]$ is the value of k-th dimension ($k =$

$1, \ldots, K$) of sample number m of signal V. Then, $d_{m,n}(V, W)$ represents the distance between the m^{th} sample of V and the n^{th} sample of W. The vectors represented by V and W are described in sections 3.4.1.1 through 3.4.1.5.

$$d_{m,n}(V, W) = \sqrt{\sum_{k=1}^{K} (V_k[m] - W_k[n])^2}$$

Equation 1. Euclidean distance metric

$$d_{m,n}(V, W) = \sum_{k=1}^{K} |V_k[m] - W_k[n]| = \sum_{k=1}^{K} \sqrt{(V_k[m] - W_k[n])^2}$$

Equation 2. Absolute distance metric

$$d_{m,n}(V, W) = \sum_{k=1}^{K} (V_k[m] - W_k[n])^2$$

Equation 3. Squared distance metric

3.4.1 Dynamic Time Warping Results

Four different types of feature vectors were used with the dynamic time warping algorithm. The method for collecting these feature vectors and their performance are described in the following sections.

3.4.1.1 Complex signal made of boundary coordinates

Taking the coordinates of the boundary of the hand in the binary image, we create a series of complex numbers with the real part containing the x-coordinate, and the imaginary part containing the y-coordinate. The following confusion matrices show each distance metric, Euclidean distance (top), absolute distance (middle), and squared distance (bottom). Overall there was not a large difference in the performance of the different distance metrics. The accuracy of each of the distance metrics was 72.22%.

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	1	-	-	-	1
3	-	-	8	-	1	-	-	-	3
4	-	-	-	11	-	-	1	-	-
5	-	-	-	-	12	-	-	-	-
6	-	-	-	2	-	5	5	-	-
7	-	1	-	1	-	1	8	1	-
8	-	-	3	-	-	1	2	6	-
9	-	-	3	-	-	-	-	3	6

Table 3 Confusion matrix for classification using Euclidean distance on complex signal made of boundary coordinates

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	1	-	-	-	1
3	-	-	8	-	1	-	-	-	3
4	-	-	-	12	-	-	-	-	-
5	-	-	-	1	11	-	-	-	-
6	-	-	-	2	-	5	5	-	-
7	-	1	-	1	-	1	8	1	-
8	-	-	3	-	-	1	2	6	-
9	-	-	3	-	-	-	-	3	6

Table 4 Confusion matrix for classification using absolute distance on complex signal made of boundary coordinates

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	1	-	-	-	1
3	-	-	8	-	1	-	-	-	3
4	-	-	-	11	-	-	1	-	-
5	-	-	-	-	12	-	-	-	-
6	-	-	-	2	-	5	5	-	-
7	-	-	-	1	-	1	8	1	1
8	-	-	-	-	-	-	2	6	4
9	-	-	3	-	-	-	-	3	6

Table 5 Confusion matrix for classification using squared distance on complex signal made of boundary coordinates

3.4.1.2 Fourier descriptor of boundary coordinates

The next feature vector we experimented with was obtained by taking the discrete Fourier transform of the complex signals used in the previous section, also known as Fourier descriptors. Using this feature vector, the accuracy of the Euclidean and absolute distance metrics was slightly improved. The accuracy of Euclidean distance metric increased to 76.85%. The accuracy of the absolute distance metric increased to 77.77%. The accuracy of the squared distance metric stayed the same as the previous result: 72.22%.

The improved accuracy may be due to small variations in the signals being suppressed from the transform.

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	-	2	-	-	-
3	-	-	12	-	-	-	-	-	-
4	-	-	-	12	-	-	-	-	-
5	-	-	1	3	8	-	-	-	-
6	-	1	-	3	-	6	2	-	-
7	-	-	-	-	-	3	9	-	-
8	-	-	3	-	-	-	-	8	1
9	-	-	1	-	-	4	1	-	6

Table 6 Confusion matrix for classification using Euclidean distance on Fourier descriptor of boundary coordinates

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	-	2	-	-	-
3	-	-	12	-	-	-	-	-	-
4	-	-	-	12	-	-	-	-	-
5	-	-	1	3	8	-	-	-	-
6	-	1	-	2	-	7	2	-	-
7	-	-	-	-	-	3	9	-	-
8	-	-	3	-	-	-	-	8	1
9	-	-	1	-	-	4	1	-	6

Table 7 Confusion matrix for classification using absolute distance on Fourier descriptor of boundary coordinates

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	7	-	-	-	3	2	-	-
3	-	-	12	-	-	-	-	-	-
4	-	-	-	12	-	-	-	-	-
5	-	-	1	3	8	-	-	-	-
6	-	-	-	3	-	4	5	-	-
7	-	-	-	-	-	3	9	-	-
8	-	-	-	1	-	2	-	8	1
9	-	-	1	-	-	4	1	-	6

Table 8 Confusion matrix for classification using squared distance on Fourier descriptor of boundary coordinates

3.4.1.3 Distance from center point of hand to boundary points

We then used the distance from the center of the hand to the boundary coordinates as a feature vector. The accuracy of each of the distance metrics significantly improved to 88.89%.

An example of this signal showing hand gesture three is shown in the following plot.

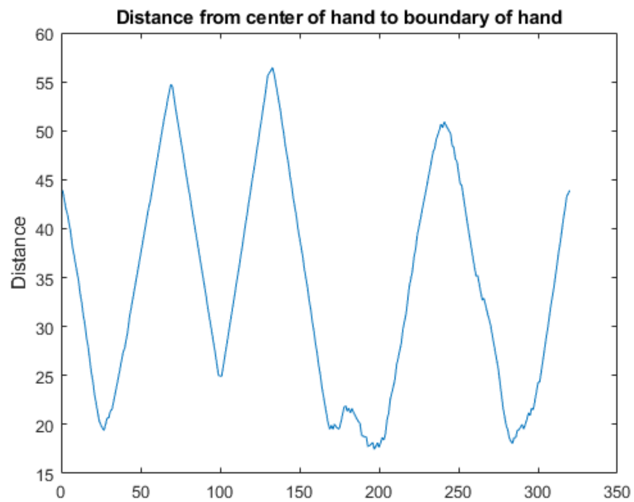

Figure 12 Plot showing relative distance from center point of hand to boundary points.

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	-	1	-	-	1
3	-	-	12	-	-	-	-	-	-
4	-	-	-	12	-	-	-	-	-
5	-	-	-	-	12	-	-	-	-
6	-	-	-	-	1	9	2	-	-
7	-	-	-	-	-	-	12	-	-
8	-	-	-	-	-	-	-	8	4
9	-	-	1	-	-	1	1	-	9

Table 9 Confusion matrix for classification using Euclidean distance on signal of distance from center point of hand to boundary points

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	-	1	-	-	1
3	-	-	12	-	-	-	-	-	-
4	-	-	-	12	-	-	-	-	-
5	-	-	-	-	12	-	-	-	-
6	-	-	-	-	1	9	2	-	-
7	-	-	-	-	-	-	12	-	-
8	-	-	-	-	-	-	-	8	4
9	-	-	1	-	-	1	1	-	9

Table 10 Confusion matrix for classification using absolute distance on signal of distance from center point of hand to boundary points

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	12	-	-	-	-	-	-	-	-
2	-	10	-	-	-	1	-	-	1
3	-	-	12	-	-	-	-	-	-
4	-	-	-	12	-	-	-	-	-
5	-	-	-	-	12	-	-	-	-
6	-	-	-	-	1	9	2	-	-
7	-	-	-	-	-	-	12	-	-
8	-	-	-	-	-	-	-	8	4
9	-	-	1	-	-	1	1	-	9

Table 11 Confusion matrix for classification using squared distance on signal of distance from center point of hand to boundary points

3.4.1.4 Signal from circle method

A fourth feature vector was obtained by taking the signal produced by the circle method. As you can see from the confusion matrix, this feature vector suffered a similar issue as the circle method, and often misclassified 6, 7, 8, and 9 as gesture 3. Only the absolute distance confusion matrix is provided since the performance of the Euclidean and squared distance were the same. The accuracy of this was 32.3%

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	6	3	3	-	-	-	-	-	-
2	-	10	1	1	-	-	-	-	-
3	-	7	5	-	-	-	-	-	-
4	-	-	1	8	3	-	-	-	-
5	-	-	-	4	8	-	-	-	-
6	-	-	12	-	-	0	-	-	-
7	-	3	7	-	2	-	0	-	-
8	-	2	8	2	-	-	-	0	-
9	-	1	11	-	-	-	-	-	0

Table 12 Confusion matrix for classification using absolute distance on signal obtained from Circle Method

3.4.1.5 Combination of signals obtained from circle method

Finally, we used a combination of signals obtained from circles of different sizes. We created a complex number, as in the first feature vector, but this time using the signals from a circle with a radius equal to 90% of the distance from the center of the circle to the farthest point in the hand, and the signal from a circle at 65% of this distance.

The results from this feature vector were much better than the single signal at 75%, as implemented in the previous section. The overall accuracy of this method using the absolute distance metric was 79.63%

A sample of this signal is shown in the following plot. The blue signal represents the signal from 90% distance, which was used as the real part of the feature vector signal. The orange signal represents the signal from 65% distance, which was used as the imaginary part of the feature vector.

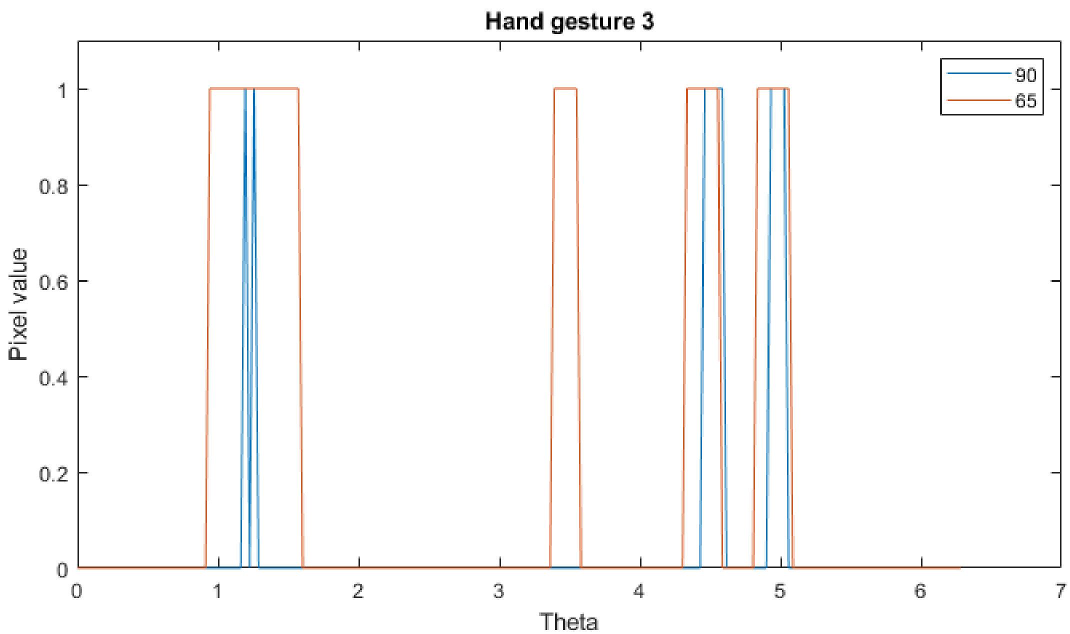

Figure 13 Real and imaginary parts of feature vector. Blue represents the signal at 90% distance (real), orange represents the signal at 65% distance (imaginary).

Predicted Label

True Label

	1	2	3	4	5	6	7	8	9
1	**11**	-	1	-	-	-	-	-	-
2	-	**11**	1	-	-	-	-	-	-
3	2	-	**10**	-	-	-	-	-	-
4	-	-	-	**8**	2	-	-	1	1
5	-	-	-	-	**10**	-	-	-	2
6	-	-	-	-	-	**11**	1	-	-
7	-	-	-	-	-	-	**8**	-	4
8	-	-	-	-	-	-	3	**8**	1
9	-	-	-	-	-	-	2	-	**10**

Table 13 Confusion matrix for classification using absolute distance on combination of signals obtained from Circle Method

4 Conclusions and Future Enhancements

We saw a significant improvement in the results of the dynamic time warping algorithm when we increased the size of the training data from 30% of the total data, to 70%. 70% of the data corresponds to 28 samples of each hand gesture, for a total of 252 images. We expect that continuing to increase the number of samples included in the dictionary would further improve our results.

Another issue we faced was that the binary images produced by the thresholding algorithm at times would contain issues that distort our results. We expect that improving the thresholding algorithm would improve the results of both methods of classification. One possible improvement to the thresholding algorithm would be to use a Markov model to identify the hand in the image.

A final improvement that could be made is to apply these methods to video data so the classification could occur in real-time.

References

[1] Wu, Jonathan, Janusz Konrad, and Prakash Ishwar. "Dynamic Time Warping for Gesture-based User Identification and Authentication with Kinect." *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (2013): 2371-375. Web.

[2] Malima, A., E. Ozgur, and M. Cetin. "A Fast Algorithm for Vision-Based Hand Gesture Recognition for Robot Control." *2006 IEEE 14th Signal Processing and Communications Applications* 2006 (2006): 1-4. Web.

[3] Kai Guo, P., Ishwar, and Konrad. "Action Recognition in Video by Covariance Matching of Silhouette Tunnels." *2009 XXII Brazilian Symposium on Computer Graphics and Image Processing* (2009): 299-306. Web.

[4] Leibe, Bastian et al. “An Implicit Shape Model for Combined Object Categorization and Segmentation.” *Toward Category-Level Object Recognition* (2006).

[5] Li, Bo, et al. "Gesture Recognition Based on Kinect v2 and Leap Motion Data Fusion." *International Journal of Pattern Recognition and Artificial Intelligence* (2018)

[6] Mesbahi, Soukaina Chraa, et al. "Hand gesture recognition based on convexity approach and background subtraction." *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*. IEEE, 2018.

[7] Xu, Yanan, Dong-Won Park, and G. Pok. "Hand Gesture Recognition Based on Convex Defect Detection." *International Journal of Applied Engineering Research* 12.18 (2017)