Cheok et al,. Int J Comput Softw Eng 2020, 5: 158
https://doi.org/10.15344/2456-4451/2020/158

International Journal of
Computer & Software Engineering

Research Article Open Access

Sign Language Recognition Using Convexity Defects and Hough Line

Ming Jin Cheok[1,*], Zaid Omar[1] and Mohamed H. Jaward[2]

[1]*Universiti Teknologi Malaysia, Skudai, 81300, Malaysia*

[2]*School of Engineering, Monash University Subang Jaya, 47500, Malaysia*

Abstract

This paper presents a novel feature extraction framework for sign language recognition application to assist the deaf and speech-impaired in daily communication. Our method extracts visual features from the hand gesture using convexity defects, K-curvature and Hough line which can differentiate visually similar signs. In segmentation stage, Canny edge detection and skin color segmentation with histogram backprojection method are used to extract the hand region from the background. The features to be extracted are categorized into shape, orientation and motion. The shape features of the sign language will then be extracted using convexity defect, K-curvature and Hough line techniques. The orientation feature will be extracted by calculating the palm center to wrist angle. The trajectory motion of the dynamic gesture is extracted using the chain code method. Lastly, Decision Tree classification is employed in classification of both static and dynamic gestures. The proposed framework is carried out in smartphone platform to recognize 26 alphabets, 10 numbers and eight dynamic American Sign Language (ASL). The average accuracy of 29 static ASL achieved is 85.72% and average accuracy of 10 dynamic ASL achieved is 77%. Through this research, it is found that the proposed framework can recognize larger sign languages database as compared with previous convexity defect-based sign language recognition research.

Publication History:

Received: August 08, 2020
Accepted: September 26, 2020
Published: September 28, 2020

Keywords:

Computer Vision, Gesture Recognition, Image Processing, Machine Learning, Sign Language

Introduction

Sign language is a form of communication used by the deaf and verbally-challenged community. It involves the use of hand, body parts and facial expression to express emotion and to deliver message. However, it is seldom used by the normal hearing community and as with minority languages, not many are able to understand. This poses communication barriers between the deaf community and the rest of the society, which creates a need for sign language recognition research to be conducted to bridge the communication gap.

Hand gesture and sign language recognition has been an active field of research. It can be achieved through vision-based and sensor-based approaches. Vision-based approaches collects images or video frames captured using camera as input of the system. It can be achieved using either appearance-based or model-based approaches. Appearance-based approaches uses visual cues as the recognition features. Model-based approaches attempts to infer the posture of finger, joint angle and the palm in 2D or 3D space [1]. Sensor-based approaches involve the use of sensing devices usually to be worn by the user when performing gesture recognition. Some common sensors include the using of inertial measurement, electromyography, flex sensors and radar.

In hand gesture and sign language recognition research, some of the more commonly used techniques in segmentation of the hand region from the background is skin color segmentation [2-4]. It can easily determine the global position of the hand, and it is commonly conducted in YCbCr or HSV color space as chrominance channel are easily separated from luminance channel which allows the lighting variation factor to be diminished [5]. However, sensitivity of skin color in an image towards variation in skin color, background illumination, and other factor poses challenges [6]. More adaptive skin color segmentation and hybrid of other features can improve this problem such as training a Gaussian Mixture Model in [7,8], or dynamic skin color modelling method such as in research [9]. Some other segmentation methods include hand tracking using Continuously Adaptive Mean-Shift (CAMShift) in [10,11] and using Picture Information Measure (PIM) to quantify entropy of image in [12].

Some notable appearance-based feature extraction method includes the extraction of Shift-Invariant Feature Transform (SIFT) features from the gesture. For example, in paper [13], SIFT features are extracted from six signs and the features are rotational invariant. Research in [14,15] also extracted SIFT features from the hand gestures, and the features are simplified by first quantizing with K-means clustering and then mapped into Bag-of-Features (BoF). Representing SIFT features using BoF reduces and uniforms the dimensionality of each SIFT features extracted.

Speeded Up Robust Feature (SURF) is another notable feature extraction method proposed in [16]. The authors in paper [17] extracted SURF features from a moving hand gestures of consecutive frames, and by analyzing correlation between SURF points, classification of 26 gestures achieves 84.6% accuracy. Computational performance of SIFT and SURF has been compared in paper [15], as the extraction of features is usually a computationally heavy process and hence the efficiency of the technique plays an important role. It is showed that SURF has a faster processing speed.

Principle Component Analysis (PCA) is another commonly extracted features in hand gesture recognition research. In paper [18], PCA is extracted together with kurtosis position and chain code, where kurtosis position is used in finding edges and chain code is used in tracking the hand trajectory. The hybrid of these three features has shown to outperform any features alone with an error rate of 10.91%.

Model-based sign language recognition based on convexity defect features has been widely applied in research in this field [19-30]. The advantage of using this approach as compared to other appearance-

***Corresponding Author:** Ming Jin Cheok, Universiti Teknologi Malaysia, Skudai, 81300, Malaysia; E-mail: mingjin_91@hotmail.com

Citation: Cheok MJ, Omar Z, Jaward MH (2020) Sign Language Recognition Using Convexity Defects and Hough Line. Int J Comput Softw Eng 5: 158. doi: https://doi.org/10.15344/2456-4451/2020/158

based approaches is that the former usually does not involve computationally heavy feature extraction and training of features [23]. From these researches, it is observed that the selection of complementary features and rules that define convexity defects greatly affects the scalability and performance. In paper [19], the numbers of fingers raised in the hand gestures are extracted using convexity defects method. The author calculates the mean, standard deviation and contour area of each gesture, and Naïve Bayes classifier is used to classify five gestures. Author in [27] extracts just convexity defects features from hand gestures and multiclass SVM is used to classify 10 gestures and these results in an accuracy of 93%. In paper [25], the author utilizes convexity defects features to count the number of fingers raised and is able to identify four classes of gestures in a robot controlling application. Research in [29] also able to identify the numbers of fingers raised using convexity defects method and able to recognize six gestures with average accuracy of 95%. However, the authors in these researches did not address the identification of which finger of the hand was raised.

Some researches identify exactly which fingers are raised in a sign language, and define the palm area, fingers, and forearm from the hand gesture. In paper [22], the authors first identify palm and wrist location and by using angle information, the exact fingers that are raised can be identified. The system is also able to identify the fingers when both hands are present. However, finger identification using angle threshold is prone to misclassification when fingers are not raised vertically but with a skewed angle. In paper [20], the authors first identify the hand palm by finding minimum inscribed circle. K-curvature features is then utilized together with convexity defects to find the finger location from the hand gestures, which is able to identify nine hand gestures in real-time.

The challenges in identifying visually similar sign languages however are not addressed in previous research, such as the ASL "K" and "V". Therefore, we further proposed a framework to use Hough Line features with features extracted using convexity defects and K-curvature to improve the separation criteria between the signs. We also proposed the utilization of finger webs feature to improve the separation criteria. Our proposed method has shown to be able to distinguish visually similar signs with high accuracy.

Materials and Method

System overview

The main camera of the iPhone 5s is utilized to capture video stream of American Sign Language (ASL) performed in real-time. The video camera collects frames of images in 2D RGB color space, and it does not contain depth information.

In appearance-based approaches, exhaustive number of training images are required to train the database. In model-based approach, this is not required. This proposed model-based framework focus on correctly modelling the different posture, orientation and trajectory of the ASL in 2D space. Firstly, signer's hand's information is obtained for calibration purpose to improve the robustness of segmentation process. The input image is then downsized for better computational efficiency. Next, skin color, edges and area information are extracted to segment the hand region from the background.

In feature extraction stage, convexity defects, k-curvature, Hough lines and several other features are extracted to determine the hand posture. The information of hand orientation is determined by calculating the palm center to wrist angle. For static finger-spelled sign, hand posture and orientation information are sufficient to classify the signs. For dynamic gestures however, trajectory motion of the gesture is extracted and quantized using chain code method. Lastly, Decision Tree classification is used to classify the spatial, orientation, and trajectory features to its respective signs. The output of the recognized sign language will then be displayed as a text and

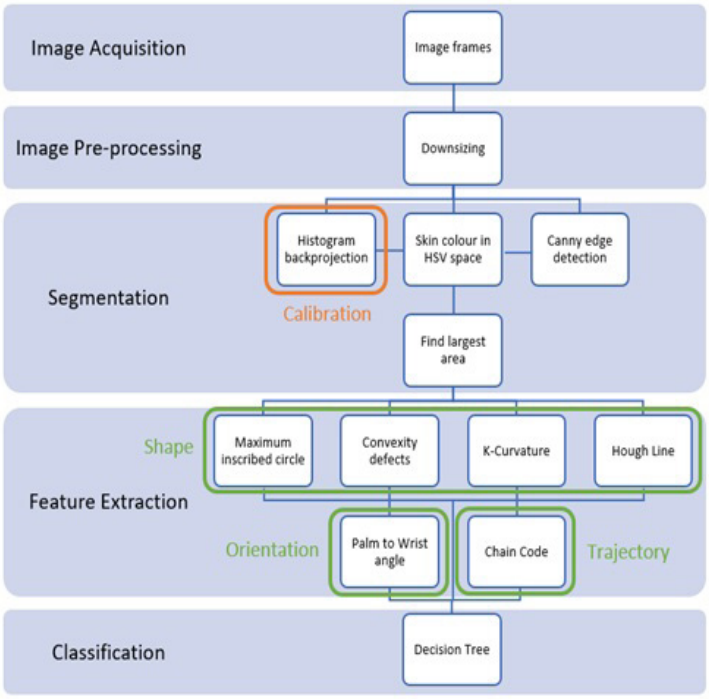

Figure 1: Flowchart of Sign Language Recognition Framework.

played as an audio speech. Figure 1 shows an overall algorithm flow and techniques used.

Parameter used

In this paper, the anatomies of hand involved in the hand recognition framework are the palm, arm, wrist, fingertips, finger webs and the finger joints as labelled in Figure 2.

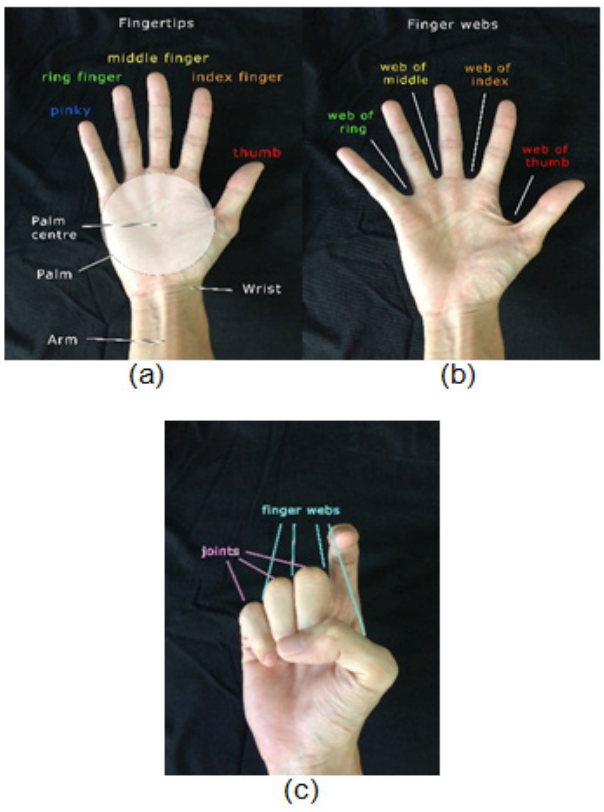

Figure 2: Anatomy of the hand.

Each finger is given a color identifier as follows, thumb labelled as red color, index finger labelled as orange color, middle finger labelled as yellow color, ring finger labelled as green color and lastly pinky labelled as blue color. These color identifiers will be used throughout this research. The webbed area between fingers is referred to as the finger webs. The identification of each finger web follows the finger that precedes it in a counter-clockwise direction. For instance, in Figure 2(b), the finger web between the thumb and the index finger will be referred to as the web of the thumb and the same applies to the rest of the finger webs.

In Figure 2(c), it can be observed that the finger webs position remains the same, despite different posture of fingers. When two adjacent fingers are not raised, the small 'valley' of a finger web remains which serves as a useful feature which can be utilized for the recognition of the hand posture. The joint of the fingers closest to the palm is called as Proximal interphalangeal joint, however, it will be simply referred to as 'joint' in this paper as shown in Figure 2c.

Several parameters are used throughout this paper, and these parameters and its description are summarized in Table 1. These parameters are kept constant throughout this experiment. These parameters differ across different experiment setup, such as when different signers or data acquisition camera is used. The parameter row_{max} and col_{max} are the height and width of the input frame respectively. The value p_{min} is introduced to avoid redundant points detected for a single hand feature. For instance, a fingertip could have several maximum points detected, introducing p_{min} will eliminate points too close with each other. The optimal constant value is determined through visual observation and it is set to be relative to the radius of palm circle, r_{palm} so it will be scale invariant. The parameters k_{th}, k_{in}, k_{mi}, and k_{ri} are introduced to define a range of horizontal distance for each different finger web. This range is used to assign each finger web point extracted to the respective finger web of the hand. k_{th} is represented by red vertical line; k_{in} is represented by yellow vertical line; k_{mi} is represented by green vertical line; kri is represented by green vertical line. Lastly the white vertical line is the vertical line passing through palm center as shown in Figure 3.

The parameters k_{raised}, k_{bent}, $k_{th\text{-}raised}$, $k_{pi\text{-}raised}$ are introduced to quantize each finger to different finger posture namely not raised, half-raised and fully raised. The parameter k_{raised} are represented by orange circular lines and the parameters k_{bent}, $k_{th\text{-}raised,}$ $k_{pi\text{-}raised}$ are represented by green circular lines in Figure 4. The parameter $k_{def\text{-}dist}$ is introduced specifically to identify the finger posture of a fully raised finger to be close-by with an adjacent finger or not. All the constant values used in these parameters are obtained through visual observation of optimal parameters which fit the hands of all five signers. The flags will be used in the algorithm later in this chapter to assign each sign or observation to a certain category which will be used as a decision factor in the Decision Tree classification. Parameter stop Frame Count is introduced to track the frames count of dynamic gestures.

Hand segmentation

In segmentation stage, skin color, edges and area information are used to extract the hand region from the background. An adaptive segmentation method by utilizing a manual calibration process whereby skin pixel input of the signer are used to facilitate the segmentation process. A one-time calibration process is carried out to help increase the invariance towards signers' skin tone and different background illumination. Calibration is performed to collect skin pixels values of the signer's hand in HSV color space. The image captured is first converted into HSV color space. The pixel value obtained will be used for auto detection and segmentation of hand from the background. A red rectangular box will first be displayed on the screen as shown in Figure 5(a), and signer are prompted to place their hand covering the box area for skin pixel value extraction. The size of rectangular box used is obtained through visual observation of sizes that fits the hand in this experiment. Calibration operation can be performed again whenever the signer or the environment has been changed.

HSV color space is used as the chrominance and luminance channels are separated per sec. By ignoring the luminance channel, the variation in background illumination factor can be reduced, and thus making it suitable to be used for skin color segmentation [31]. Histogram backprojection method [32] is used to incorporate an environment adaptive segmentation. Histogram backprojection can localize a known object in a scene based on its color appearance. It uses a histogram of a known object of interest to find the ROI in the new input image that matches the color value of the histogram. It calculates and replaces all the pixels value in the new input image with the probability of the pixel belonging to the object in the histogram.

Parameters	Description	Initialized value
row_{max}	The maximum number of row in the image or the image height.	480
col_{max}	The maximum number of column in the image or the image width.	360
p_{min}	The minimum possible pixels deviation from each hand feature such as finger webs, minima points, etc.	$r_{palm}/6$
k_{th}	The ratio of thumb distance to palm radius.	$1.1r_{palm}$
k_{in}	The ratio of index finger distance to palm radius.	$0.45\ r_{palm}$
k_{mi}	The ratio of middle finger distance to palm radius.	$0.45\ r_{palm}$
k_{ri}	The ratio of ring finger distance to palm radius.	$0.9\ r_{palm}$
k_{raised}	The minimum distance between fingertip to palm center for the finger to be considered raised	$2.4\ r_{palm}$
k_{bent}	The minimum distance between fingertip to palm center for the finger to be considered half-raised	$1.6\ r_{palm}$
$k_{th\text{-}raised}$	The minimum distance between thumb fingertip to palm center for the thumb to be considered raised	$1.6\ r_{palm}$
$k_{pi\text{-}raised}$	The minimum distance between pinky fingertip to palm center for the pinky to be considered raised	$1.6\ r_{palm}$
$k_{def\text{-}dist}$	The minimum distance between defect point to palm center for the defect to be considered belonging to a pair of fingers in adjacent to each other.	$1.8\ r_{palm}$
$k_{tip\text{-}dist}$	The maximum distance between two close by hand feature, i.e. fingertip with fingertip or fingertip with finger web.	$0.8\ r_{palm}$
$k_{th\text{-}fist}$	The minimum distance of thumb to be considered raised in fist posture.	$1.5\ r_{palm}$
a_{cross}	The maximum angle of deviation before it is considered as finger crossed	55°
is_signK	The flag to differentiate between sign 'K' and 'V'.	true / false
is_signR	The flag to differentiate between sign 'R' and 'U'.	true / false
is_signS	The flag to indicate sign 'S'.	true / false
is_vertical	The flag when true indicates the yaw orientation of hand is in upright position.	true / false
is_roll	The flag when true indicates the hand is in roll and horizontal yaw orientation.	true / false
is_motion	The flag when true indicates motion is detected. Trajectory points will be tracked.	true / false
stop Frame Count	The counter of Frames stop motion.	0

Table 1: Finger posture and the respective distance criteria.

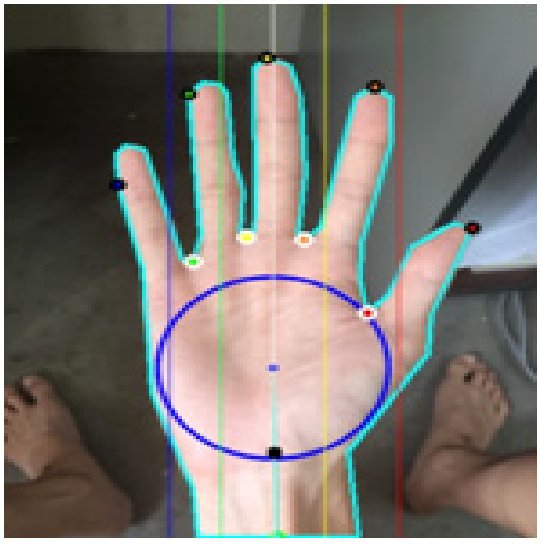

Figure 3: Vertical lines representing different distance ratio to palm radius.

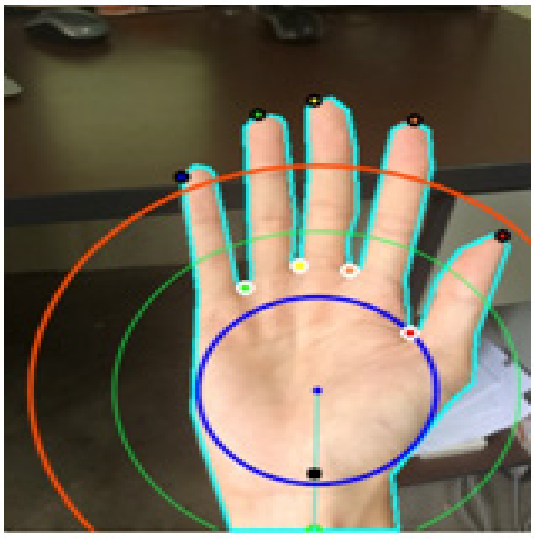

Figure 4: Circular lines representing different distance ratio to palm radius.

Hence, histogram backprojection is suitable to be utilized to find the skin pixels in the new input images.

The Hue and Saturation value of the skin pixels collected within the red rectangle in Figure 5(a) will be used to construct a 2D histogram. The histogram is then back-projected to the new input image to find the pixels which falls under the skin pixels range in the histogram, the output image will be stored as D(x, y), and lastly thresholded into a binary image B(x, y). The result of histogram backprojection is as shown in Figure 5(d). Histogram backprojection is useful in determining the global location of the skin colored pixels. However, it is not able to accurately determine every skin color pixels of the

hand especially when the illumination is not uniform, such as when the light sources are from the side instead of on top of the hand. Edge information has shown to be a suitable complementary feature for skin color in research [33], where it is used to locate the palm and arm region accurately. In order to improve the result, HSV image is converted into grayscale image, Canny edge detection is then performed on the grayscale image to extract all the edges found on the image, both foreground and background, as shown in Figure 5(e). The edges information is more invariant to shading and can be used to feed into the skin color segmentation to help identify skin pixels with color range falls outside the normal skin pixels range.

From the edge detected image, the outermost layers of contours are extracted, while the inner layers are ignored. The outermost contours are then used as mask on the Histogram backprojected image D(x,y). The edges detected serves as a limiting boundary. For contours which contain skin pixels, each pixel value is then checked if it falls under the range of a lower threshold value. If the value of the pixels falls under the range of skin value, the pixel is considered as skin pixels. The process is iterated for all the pixels within the contours.

Next, opening morphological operations erosion followed by dilation are performed to remove the small noise present in skin pixel region. Background with clusters of pixels matching skin color will still appear in the image, as in Figure 5(f). In order to remove the background pixels, an assumption is made to facilitate the process. It is assumed that the hand region is always the cluster with largest skin area. This statement is always true given the scope and setting of this experiment where the hand gesture is close to the camera. In this paper, n is defined as the number of contour among the edge detected contours. The outermost contours in the image are detected, and the largest area of cluster, $contour_i(n)$ is then selected by finding the area with maximum number of enclosed pixels and is represented by $p_{hand}(n)$. The area information helps removes the background object which are misclassified as skin pixels. The output is a binary image where the hand area extracted as foreground, while other individual smaller clusters of skin pixels removed, as shown in Figure 5(g). Finally, the binary image is used as a mask on the original input image in Figure 5(b) to extract the hand region from the background, the final output of the segmentation process is as shown in Figure 5(h).

Feature extraction

The general features of a hand gestures can be categorized into shape, orientation and trajectory motion. Each category of features is extracted using different techniques. The shape properties are extracted using convexity defects, K-curvature and Hough lines techniques. The orientation properties are determined using palm and wrist angle information as well as other features derived from shape properties. The trajectory motion properties are extracted using chain code method. The following subsections describe the procedures taken in the feature extraction stage.

Finding palm center and radius

From the output image of segmentation process, the palm center is first identified by finding the center of the maximum inscribed circle inside the hand contour. Maximum inscribed circle inside hand contour is obtained by iterating across the points inside the contour to find the point with the largest distance from the contour perimeter. The point furthest away from the hand contour is the center of the palm. This method utilizes the anatomy of the hand whereby finger webs location falls almost exactly on the perimeter of the circle that covers the palm of the hand. The palm center is defined as center point of palm, c_{palm} and radius of palm, r_{palm}. The outputs are center point of palm, c_{palm} and radius of palm, r_{palm} identified.

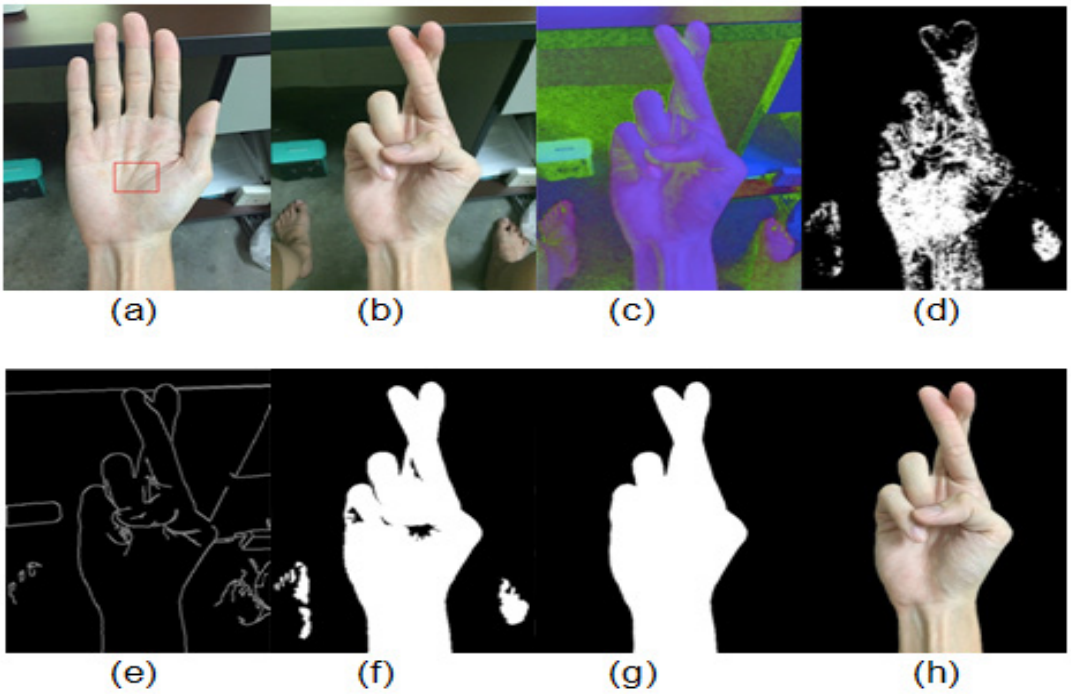

Figure 5: Segmentation result (a) Calibration rectangular box, (b) Input image, (c) HSV color space image (d) Histogram backprojection (e) Canny edge detection (f) Skin pixel selection (g) Area selection and (h) Output image.

Finding finger webs and fingertips

After obtaining the palm center, the location of potential finger webs is determined using K-curvature and convexity defects features. K-curvature measures the extent of deviation of a point in a curve. For each point in the set of points describing a curve, it determines the angle between the two lines, α that starts at the point in question and ends k points away in either direction. In Figure 6, A is the point in query, while B and C are points lying k points away in both direction. The lower the value of K, the more iterations are required, however the results are more accurate. For more accurate results, the value of K used is k=1. If K-curvature, curv (n) represented by α where n is the number of contours, is smaller than a certain threshold, it is a potential turning point.

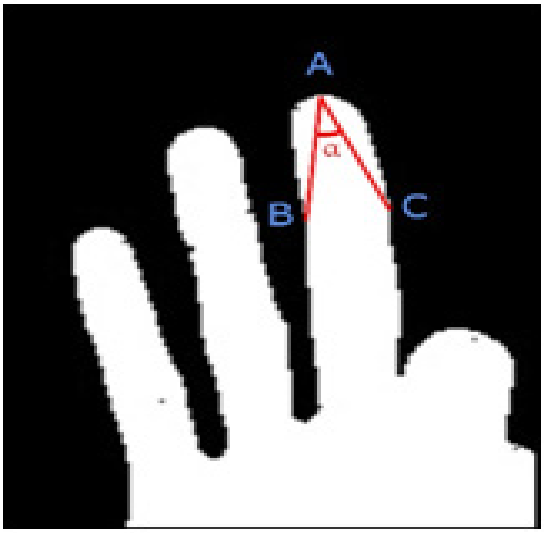

Figure 6: K-curvature diagram.

K-curvature is identified by traversing across points on hand contour until a turning point is found. Then, the point in query is then checked if it is a minimum point or maximum point. Each minimum point will be stored in an array of vector defined as $p_{minima}(n) = [(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)]$ and maximum point will be stored in array of vector defined as $p_{maxima}(n) = [(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)]$. In an upright position, a minimum point is potentially a finger web, and a maximum point is potentially a fingertip. The entire closed contour or edges of the hand will be examined to find all potential minimum and maximum points. The actual points of interest are only the points representing fingertips and finger web, however, there will be many other points detected due to the noise presence in hand contour. The points are then filtered using the following criteria:

1. The turning points must be vertically above palm center.
2. The turning points must alternate between minimum point and maximum point.
3. The horizontal distance between adjacent turning points must exceed p_{min}.

The K-curvature is the alternative feature in determining the turning points. The k K-curvature at the turning points are comparatively lower, hence an angle threshold can be set to filter the K-curvatures. However, this threshold method is scale dependent, the K-curvature value at fingertip is larger when the hand is closer to camera, and vice versa. Hence, a better method of finding the turning points representing fingertips and finger webs is to determine the local minima and local maxima of K-curvature alternatively. The value of K-curvature is negative when it is a maximum point, where curv(n)<0. The value is positive when it is a minimum point, where positive curv(n)>0. In other words, find the local minimum K-curvature at curv(n)< 0 which represents the maximum point, and after the point is determined, proceed to find the local minimum K-curvature at curv(n)>0, and repeat the process until the hand contour loop is completed. For calculation purpose, let the Euclidean distance between two points a and b defined by Equation 1 below:

$$dist(a,b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2} \quad (1)$$

The outcome from the above process is a set of points potentially representing the fingertips and finger webs. Further process is required to accurately identify the fingertips and finger webs. Algorithm 1 iterate through every point in $p_{hand}(n)$ which meet the criteria of a minimum or maximum point. It then categories the points into minimum point and maximum point and stores them into different arrays. The curvature curv(n)<0 signifies a maximum point, while curv>0 signifies a minimum point. A variable $curv_{min}$ and several flags namely, is_min and is_turned will be used. The variable $curv_{min}$ with initial value set to the maximum possible angle $curv_{min}$= 180 is used to store the local minimum value. The flag is_min=false is initialized to indicate previous point is a potential minimum point. The flag is_turned= false is initialized to indicate the changes in curv(n) sign. The maximum points, $p_{maxima}(m)$ and minimum points, $p_{minima}(m)$ extracted are depicted in Figure 7, where maximum points are labelled with red circles and minimum points are labelled with blue circles.

```
For n pairs of points in p_hand(n)
    If y_phand(n) < yc_palm
        x_dist = dist(p_hand(n), c_palm)
        Calculate curv(n) < 0
    If curv(n) < 0
        If is_min = true
            is_turned = true
        End if
        If curv(n) < curv_min
            curv_min = curv(n)
        End if
            is_min = false
    Else
        If is_min = false
            is_turned = true
        End if
        If curv(n) < curv_min
            curv_min = curv(n)
        End if
            is_min = true
    End if
        If is_turned = true and is_min = true
            p_maxima(m) ← curv(n)
            is_turned = false
            curv_min = 180
    Else if is_turned = true and is_min = false
            p_minima(m) ← curv(n)
                is_turned = false
            curv_min = 180
    End if
  End if
End for
```

Algorithm 1: Finding and filtering the maximum and minimum point.

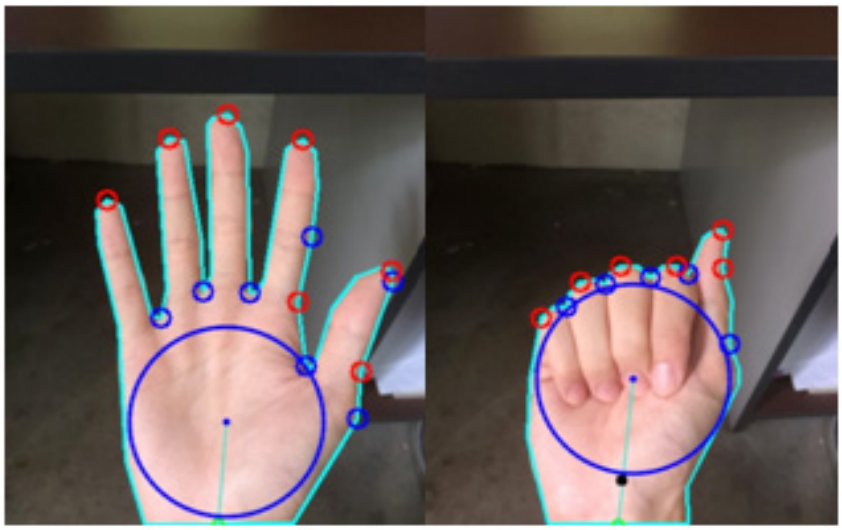

Figure 7: Maximum points (red) and minimum points (blue).

Convexity defects method are introduced next to extract the finger webs and fingertips location. Convexity defects is a cavity in the outer boundary of an object which the area is formed by its convex hull. A convex hull is the smallest convex polygon which contains all the points given a set of points on the plane. The convex hull of the hand is first determined in order to find the convexity defects. Convexity defects are the furthest point which deviates from the convex hulls. Essentially, the finger tips form the convex hull and the finger webs are the convexity defects. The intrinsic characteristic of convexity defects makes it very useful to identify fingers apart from the hand. Defect is the point which are curved inwards and is the furthest point from the convex hull, which is suitable in determining the location of finger webs. Each convexity defects starts with a convex, p_{start} and ends with another convex point, p_{end}. The point furthest away from the convex points which are called the defects are represented by p_{far}. For instance, the thumb indicates the starting of first convex points, and the index finger indicates the end of the first convex points in anti-clockwise direction as shown in Figure 8.

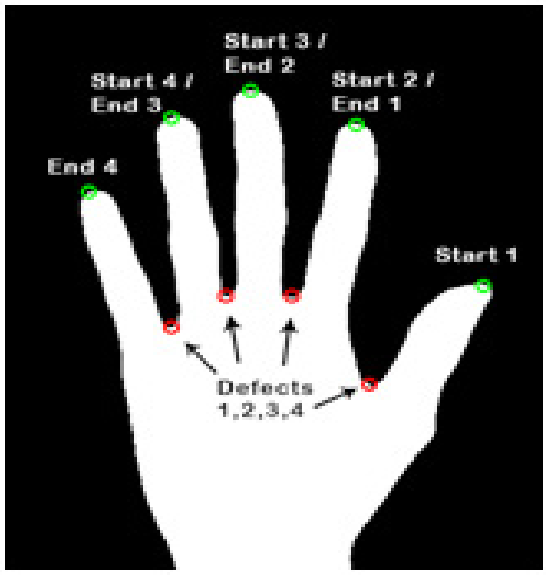

Figure 8: Convex points (green) and defect points (red).

After the convexity defects are extracted, the points extracted are filtered using the criteria as follows:

1. Either the start or end points of a convexity defects must be vertically higher than the palm center.
2. The vertical distance of a convex points and palm center must exceed 4pmin as this is the optimal value obtained through observation when tested with hands of different sizes.
3. The distance between each convex point and the convex point before it must exceed pmin. This applies also to defect points.

Rule 1 states that fingertips and web vertical position is always higher than the palm center. Rule 2 is not always true, but it is necessary to avoid misclassified fingertips. Rule 2 is set so that only upright fingers can be determined by convex points, the other position of fingers will be determined using maximum points obtained from K-curvature. Rule 3 is set to filter out potentially duplicated convex points and defect points existing on single fingertip and finger web respectively.

From the output from both K-curvature and convexity defects, the fingertips and webs location can be identified. The nature of convexity defects has higher rate of false negative rate of identifying an actual hand feature, as for every convexity defect pairs to exist, each convex point must be accompanied by another convex and defects point. The nature of K-curvature on the other hand, do not assume underlying relation between points, and hence it has higher false positive rate of identifying an actual hand feature. Hence, the convex points and defect points determined through convexity defects method will always be accurate.

Meanwhile, the maximum and minimum points determined from K-curvature is then used for exhaustive potential hand feature point search to make up for any missing fingertips and web not identified by convexity defect. The opposite does not work as well due to the different intrinsic characteristic of convexity defects and K-curvature. By using this framework, the points identified from K-curvature will be used to fill the missing convex and defects points. The output is an array of $p_{convex}(n)$ and $p_{defect}(n)$ which potentially represents the fingertips and finger webs respectively as shown in Figure 9.

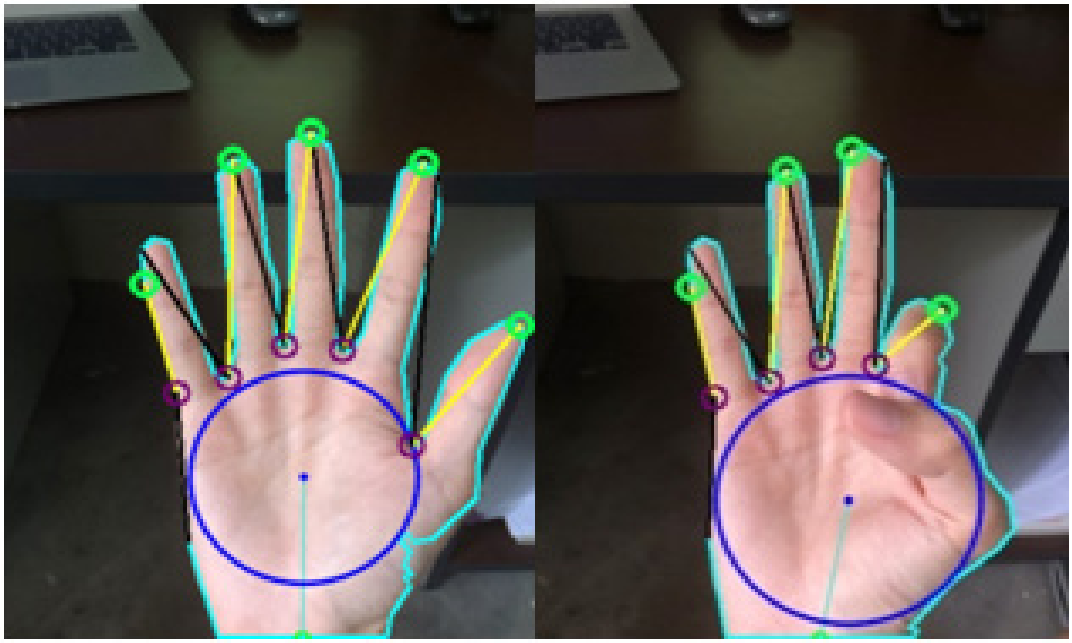

Figure 9: Convex points (green), defect points (indigo), convex start line (black) and convex end line (yellow).

Identifying finger webs

Finger webs are robust features with relative to the palm center, as the finger web position remain the same for any posture of hand gesture. Finger webs of the hand are always equally distanced from each other on the around the perimeter of palm circle. Finger webs can be identified from $p_{minima}(n)$ and $p_{far}(m)$ obtained earlier. A variable $ia_{defect}(n)$, n=0,1,2,3 is introduced to store potential defects points, where 0 represents web for thumb, 1 for index finger, 2 for middle finger and 3 for ring finger. In Algorithm 2, all the p_{defect} and p_{minima} which matches the finger webs distance criteria are stored in respective ia_{defect}. In Algorithm 3, all potential p_{minima} are stored in ib_{defect} which will be used later when not all finger webs points are captured in Algorithm 2. The final output is identification of all four finger webs as shown in Figure 10.

For n pairs of points in $p_{minima}(n)$
  For m points in $p_{far}(m)$
    If dist($p_{minima}(n)$, $p_{far}(m)$) < p_{min}
      Calculate $x_{dist} = | xp_{minima} - xc_{palm} |$
      If $xp_{minima} > xc_{palm}$
        If $x_{dist} < k_{in}$
          $ia_{defect}(1) \leftarrow p_{minima}(n)$
        Else if $x_{dist} < k_{th}$
          $ia_{defect}(0) \leftarrow p_{minima}(n)$
          End if
      Else
        If $x_{dist} < k_{mi}$
          $ia_{defect}(2) \leftarrow p_{minima}(n)$
        Else if $x_{dist} < k_{ri}$
          $ia_{defect}(3) \leftarrow p_{minima}(n)$
          End if
      End if
      End if
  End for
End for

Algorithm 2: Finding the finger webs.

For n points in $ia_{defect}(n)$
  For m points in $p_{defect}(m)$
    Calculate $x_{dist} = | xp_{defect} - xc_{palm} |$
    If $xp_{defect} > xc_{palm}$
      If $x_{dist} < k_{in}$ **and** n = 1
        $ib_{defect} \leftarrow p_{defect}(m)$
      Else if $x_{dist} < k_{th}$ and n = 0
        $ib_{defect} \leftarrow p_{defect}(m)$
        End if
    Else if $xp_{defect} < xc_{palm}$
      If $x_{dist} < k_{mi}$ **and** n = 2
        $ib_{defect} \leftarrow p_{defect}(m)$
      Else if $x_{dist} < k_{ri}$ **and** n = 3
        $ib_{defect} \leftarrow p_{defect}(m)$
        End if
      End if
  End for
  If $ia_{defect}(n) \neq 0$
    $f_{web}(n) = ia_{defect}(n)$
  Else
    $f_{web}(n) = ib_{defect}$
  End if
End for

Algorithm 3: Identification of all four finger webs.

Identifying fingertips

This section labels the fingertips according to convex points $p_{convex}(n)$ collected in previous stage. The recognition of fingertips can be performed by using the relative position of the fingertip to the known finger webs. For instance, the point representing the fingertip which sits in between of the web of index finger and web of middle finger on the hand contour, is certainly the fingertip of the middle finger. The identity of each fingertip can be easily determined once the identity of each finger web is identified. The output of Algorithm 4 is the labelled fingertips $f_{tip}(n)$, n=0,1,2,3,4 where 0 represent thumb, 1 represents index finger, 2 represents middle finger, 3 represents ring finger and 4 represents pinky. The fingers are labelled from thumb to pinky in red, orange, yellow, green and blue respectively as shown in Figure 11.

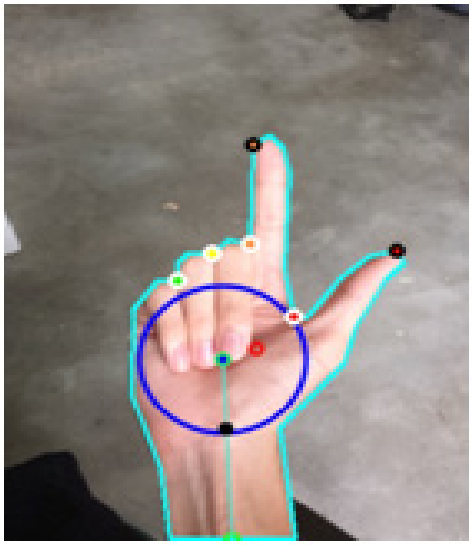

Figure 10: Labelled finger webs.

For n points in $p_{convex}(n)$
  For m points in $f_{web}(m)$
    If $xp_{convex} > xc_{palm}$
      If $xp_{convex}(n) - xf_{web}(n) > k_{th}$
        $f_{tip}(0) = p_{convex}(n)$
      Else if $xp_{convex}(n) - xf_{web}(n) > k_{in}$
        $f_{tip}(1) = p_{convex}(n)$
      Else
        $f_{tip}(2) = p_{convex}(n)$
      End if
    Else
      If $xp_{convex}(n) - xf_{web}(n) > k_{pi}$
        $f_{tip}(4) = p_{convex}(n)$
      Else
        $f_{tip}(3) = p_{convex}(n)$
      End if
    End if
  End for
End for

Algorithm 4: Identification of Fingertips.

Figure 11: Labelled FIngertips.

Hough line features

Lines can be represented by polar coordinate system, r = xcosθ + ysinθ. Any line can be represented in the form (r, θ) where θ ∈ [0,360] and r≥0. The equation of a Hough line can be written as in Equation 2, where given n pairs of points (x, y), r_θ represents the perpendicular distance from the line to the origin at a constant θ.

$$r\theta = x_n \cos\theta + y_n \sin\theta \quad (2)$$

First, an array of (r_θ, θ) is created as the accumulator variable. The size of the array is determined by the desired resolution or accuracy. For instance, for angle resolution, θ_{min} = 1°, the size of θ array, θ_N =180. Then, find the value of r_θ for every pair of (x_n, y_n) for all values of θ. Then increment the value of the respective (r_θ, θ) in accumulator cell by one. At the end, the pair of (r_θ, θ) in the accumulator variable which has higher value signifies potential presence of a line. A threshold, pcount is then set to determine how many iterations is required for the pair (r_θ, θ) be classified as a line. In this research, the parameters θ_{min}= 1° and p_{count} = 5 are used.

Hough lines detected are used to study the posture of the fingers when fingers are overlapped or when it is hidden inside the palm. In this research, Hough line are extracted from Canny edge detected images as shown in Figure 5(e) and is used specifically for three situations. The first situation is to detect the lines formed by the thumb in fist posture. The mask is for Hough line detection are as shown in the pink circle in Figure 12. The pink circle is slightly vertically higher than the palm circle so that the thumb position will be the center of the mask. It can be observed that both sign 'E' and sign 'S' both have horizontal Hough line detected as the thumb is above the fingers. Meanwhile, sign 'T', 'N' and 'M' does not have the Hough line detected.

The second situation where Hough line is used is to detect the crossing of fingers by finding the angles of lines of index fingers and middle fingers to differentiate between sign 'U' and 'R'. Sign 'R' is the only sign in the scope of this research which involves the fingers crossing with each other. Signs 'R' and 'U' share similarity in convexity defects and K-curvature properties. Hence, additional features will be required to differentiate these two signs. The crossing of fingers can be detected by finding the Hough lines of the fingers and calculate the angle of deviation of the line representing the fingers. Sign 'R' will have Hough lines which are slanted due to finger being crossed as shown in Figure 13, this feature can be used to distinguish the two signs apart.

In the third situation, Hough line is used is to identify the roll orientation of hand by detecting presence of fingers inside the palm area when hand is in horizontal orientation. When hand is in default horizontal orientation, fingers are present inside the palm area. When hand is in roll horizontal orientation, there will be no fingers present inside the palm area. This is used to help categorize 'C', 'O', and 'Q' apart from 'G', 'H', and 'P', where the former has no fingers present inside the palm area. However, there is an exception to the case where Sign 'O' has Hough line detected despite it is in roll orientation as in Figure 14(b). The size of the square mask is defined by $mask_{sqr}(x_1, y_1, x_2, y_2)$, and size of mask for circular mask is defined by $mask_{cir}(x_1, y_1, x_2, y_2)$. In Algorithm 5, the size of circular and square masks used are determined by defining the area where Hough lines information is required. The Hough lines detected are stored in $l_{Hough}(n)$. The angle of the lines in $l_{Hough}(n)$, $al_{Hough}(n)$ are then calculated.

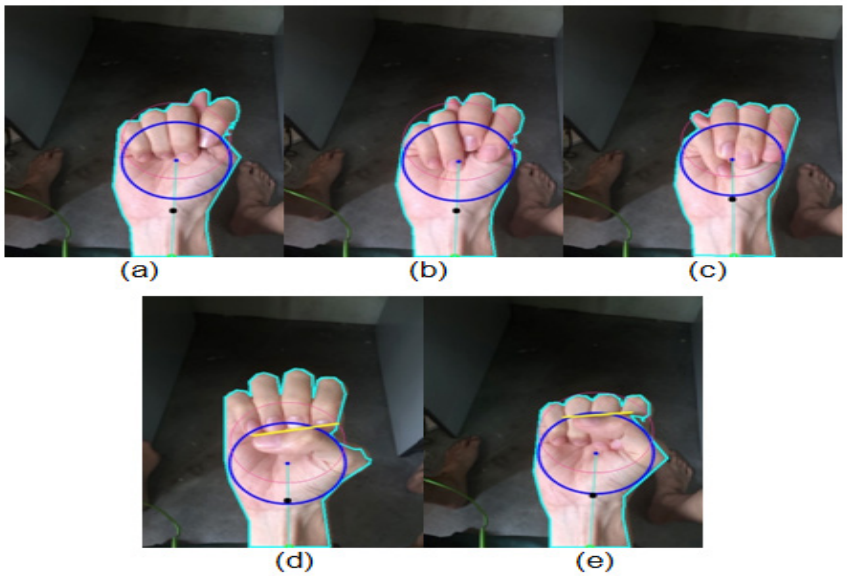

Figure 12: (a) Sign 'T' (b) Sign 'N' (c) Sign 'M' (d) Sign 'E' with Hough line (e) Sign 'S' with Hough line.

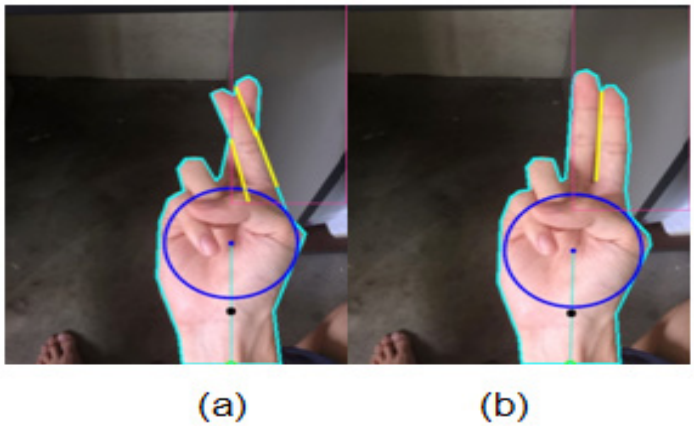

Figure 13: (a) Sign 'R' with slanted Hough line (b) Sign 'U' with straight Hough line.

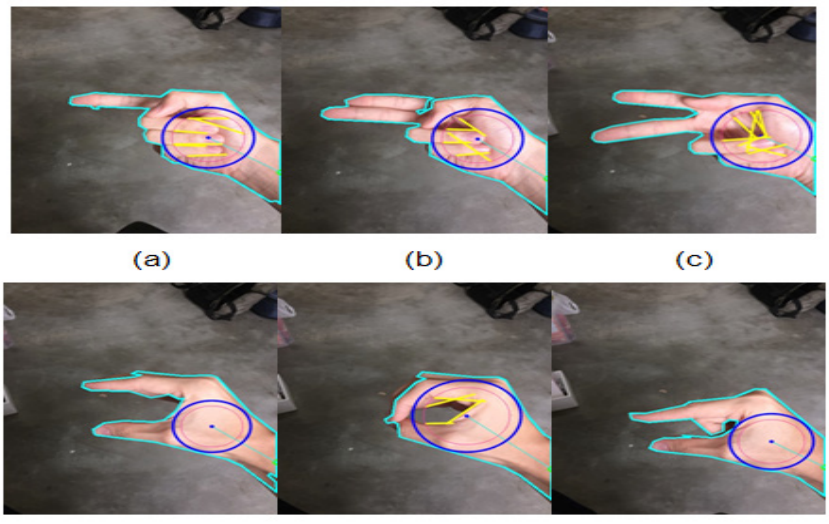

Figure 14: Sign in horizontal orientation (a) Sign 'G' (b) Sign 'H' (c) Sign 'P' (d) Sign 'C' (e) Sign 'O' (f) Sign 'Q'.

```
If is_vertical = true
    If f_post(n) = {0,1,1,0,0}
        Create mask_sqr(xc_palm, 0, col_max, 0.9yc_palm)
        Detect l_Hough(n) in mask_sqr
        For n points in l_Hough(n)
          Calculate a_lHough(n)
            If -a_cross < a_lHough(m) < a_cross
                is_signR = false
            Else
                is_signR = true
            End if
        End for
    Else
        Create mask_cir(x_cpalm, y_cpalm, r_palm)
        Detect l_Hough(m) in mask_cir
        For n points in l_Hough(m)
          Calculate a_lHough(n)
            If -a_cross < a_lHough(m) < a_cross
                is_signS = false
            Else
                is_signS = true
            End if
        End for
    End if
Else
    Create mask_cir(x_cpalm, y_cpalm, 0.8r_palm)
    Detect l_Hough(n) in mask_sqr
    For n points in l_Hough(q)
        If l_Hough(q) ≠ 0
            is_roll = false
        Else
            is_roll = true
        End if
    End if
End if
```

Algorithm 5: Hough Line feature identifcation.

Identifying posture for fingers

Each individual finger can have three different posture, namely raised, half-raised, and not raised. When two fingers are involved, the variation of position can be extended to fingers crossed or fingers side-by-side. Hence, for each finger, the posture can be quantized into five postures. With the furthest fingertip distance from palm center being a raised finger, second being the half-raised and not raised being the nearest to palm center. After the position of all four fingers webs are identified, the position is utilized to determine which finger do the fingertip detected belong to. This is performed by using the edge of the hand gesture detected.

The edges of hand gestures are a closed contour, hence all neighboring points on the contour can be traced. The status of each individual finger is determined based on the following rule:

1. The convex points detected are matched with the respective finger based on the relative position of the convex point on the hand contour to the known finger webs.
2. The status of each finger either raised, half-raised or not raised is identified by the distance of the fingertip from the palm center.
3. The horizontal distance between two fingers, if it is lower than a threshold, then it is in side-by-side position.
4. When criteria 3 is met, check for the edges of the fingers, if the edges are not parallel, then the fingers are in crossed posture.

This section labels each fingertip $f_{tip}(n)$, n=0,1,2,3,4 where 0 represent thumb, 1 represents index finger, 2 represents middle finger, 3 represents ring finger and 4 represents pinky. The finger posture for vertical orientation have several shapes which are quantized into five possible state as shown in Table 2. Figure 15 depicts five different quantized level of fingers. Thumb and pinky are exception and only have two postures namely finger posture 0 and 1, which are not raised and fully raised respectively due to the shorter length and location of the fingers on the hand. Table 3 shows the distance criteria used to identify the posture of each finger.

Finger posture	Description
0	Finger is not raised
1	Finger is fully raised
2	Finger is half-raised or bent inwards
3	Fingers are fully raised and are side-by-side
4	Fingers are crossed

Table 2: All finger posture and description.

Symbol	$f_{post}(n)=0$ (not raised)	$f_{post}(n)=1$ (fully raised)	$f_{post}(n)=2$ (half-raised)
n=0 (thumb)	$d_{tc} \le k_{th\text{-}raised}$	$d_{tc} > k_{th\text{-}raised}$	-
n=1 (index)	$d_{tc} \le k_{bent}$	$d_{tc} > k_{raised}$	$k_{bent} < dtc \le k_{raised}$
n=2 (middle)	$d_{tc} \le k_{bent}$	$d_{tc} > k_{raised}$	$k_{bent} < dtc \le k_{raised}$
n=3 (ring)	$d_{tc} \le k_{bent}$	$d_{tc} > k_{raised}$	$k_{bent} < dtc \le k_{raised}$
n=4 (pinky)	$d_{tc} \le k_{pi\text{-}raised}$	$d_{tc} > k_{pi\text{-}raised}$	-

Table 3: Finger posture and the respective distance criteria.

The posture of the finger can be identified by the distance between fingertip and palm center, dtc. Hence, the distance dtc is quantized into three levels. The thumb and pinky are shorter of all fingers and are only quantized into two levels. The distance of quantized level can be seen in Figure 16, where the orange circle represents k_{raised}, while the green circle represents k_{bent}, $k_{th\text{-}raised}$, and $k_{pi\text{-}raised}$.

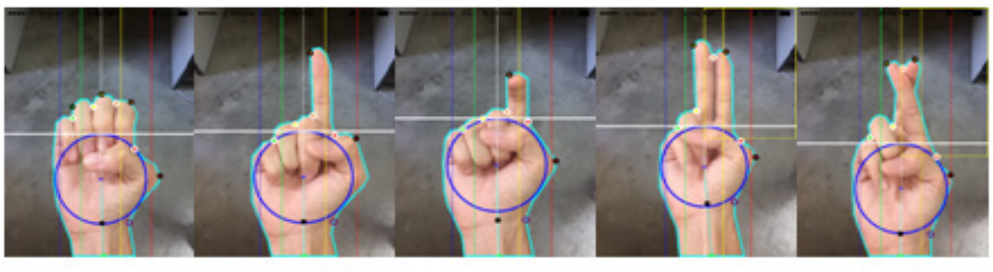

Figure 15: (a) Finger posture '0' (b) Finger posture '1' (c) Finger posture '2' (d) Finger posture '3' (e) Finger posture '4'.

The posture of the finger can be identified by the distance between fingertip and palm center, d_{tc}. Hence, the distance d_{tc} is quantized into three levels. The thumb and pinky are shorter of all fingers and are only quantized into two levels. The distance of quantized level can be seen in Figure 16, where the orange circle represents k_{raised}, while the green circle represents k_{bent}, $k_{th\text{-}raised}$, and $k_{pi\text{-}raised}$.

The posture 3 applies to all fingers other than thumb. The fingers can only be categorized as touching adjacent fingers only when it is fully raised, i.e. f_{post} = 1. Next, when fingers are side-by-side, the defects are further away from the palm center, as shown in Figure 17. Using these two information, the adjacent fingers can be recognized by applying distance threshold of the defects away from the palm center. For instance, the sign 'V' and 'U', as well as '4' and 'B', both sign 'V' and sign '4' have fingers fully raised but not touching each other, hence the defects point between index and middle finger will be further away from palm center.

Sign 'K' has similar finger posture as sign 'V' but the former has the thumb raised in between of index and middle finger as shown in Figure 18. The presence of thumb can be identified by detecting the presence of maximum points, $p_{maxima}(n)$ in the region between index, middle finger and palm center. If maximum point is detected within the region, it is sign 'K', else it is sign 'V'.

In ASL, there are several signs which involves no fingers being raised explicitly, namely 'M', 'N', 'S' and 'T'. The identification of these signs involves detection of detailed features which have lesser margin of differences. Convexity defect methods are unable to detect the contours of finger joints as the defect depth are too low. K-curvature method can be used to detect the curves formed by finger joints. This

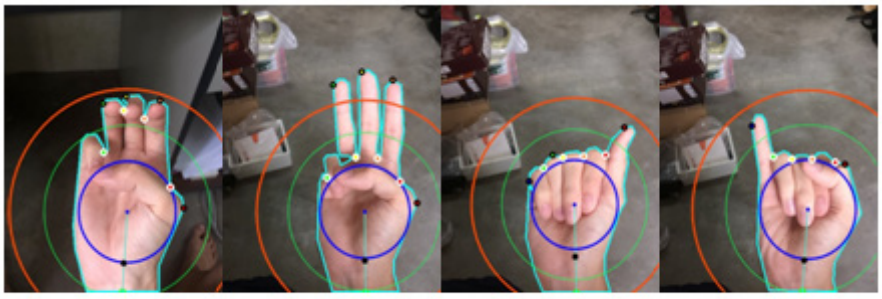

Figure 16: (a) Fingers half-raised (b) Fingers fully raised (c) Thumb fully raised (d) Pinky fully raised.

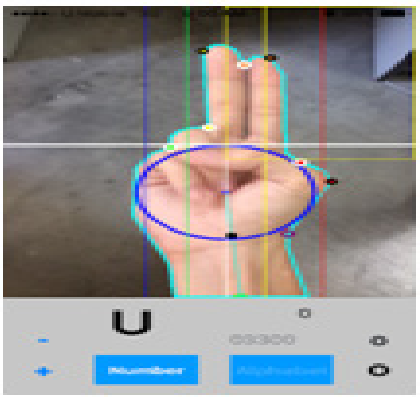

Figure 17: (a) Fingers half-raised (b) Fingers fully raised (c) Thumb fully raised (d) Pinky fully raised.

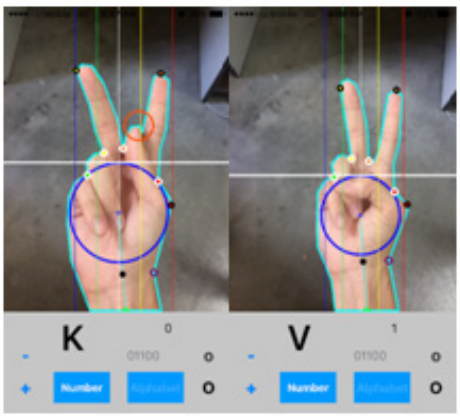

Figure 18: Thumb present in sign 'K' but not in sign 'V'.

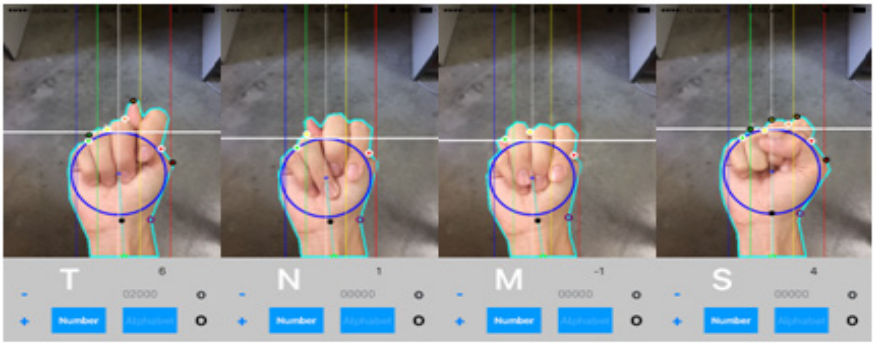

Figure 19: Sign with fist posture.

section finds the maximum points which represent the finger joints and the raised thumb, and then identify the position of thumb. First a distance threshold which can clearly separates the thumb from the finger joints is to be set, which is $k_{th\text{-}fist}$. Maximum points which are lower than $k_{th\text{-}fist}$ will be classified as middle phalanx, while points higher will be classified as thumb. The feature will be used to classify sign 'T', 'N', and 'M' which has fingers raised and result is as displayed in Figure 19. The position of thumb, $k_{th\text{-}post}$ for sign 'T' is $k_{th\text{-}post}$= 1; for sign 'N' is $k_{th\text{-}post}$= 2; and for sign 'M' is $k_{th\text{-}post}$= 3.

For n points in $p_{maxima}(n)$
  If dist($p_{maxima}(n)$, $p_{maxima}(n+1)$) > p_{min}
    $p_{fist}(m) = p_{maxima}(n)$
    Increment m by 1
  End if
End for
Sort points in $p_{fist}(m)$ from rightmost to leftmost
For m points in $p_{fist}(m)$
  If dist($p_{fist}(m)$, c_{palm}) > $k_{th\text{-}fist}$
    pth = pfist(m)
  End if
End for
Calculate number of points on pfist(m) the right side of pth and save the number into $k_{th\text{-}pos}$

Algorithm 6: Identification of fist posture.

Finding hand orientation

There are three axes of orientations of the hand, namely yaw, pitch and roll. Pitch and roll angle will result in partial occlusion of fingers. Hence the usage of 3D depth data or a second camera is required to obtain the accurate pitch and roll angle information. In this research, only the yaw angle and roll angle is taken into consideration. The hand orientation is categorized into vertical orientation as shown in Figure 20 and horizontal orientation as shown in Figure 21 based on the hand yaw angle. The hand orientation is categorized as vertical orientation or in upright position when angle of yaw deviation, a_{yaw} < 60°. The flag is_vertical is then set to true which will be used in Decision Tree classification in the later stage. The hand is categorized into horizontal orientation when the angle of rotation is 60° < a_{yaw} <90°. In this research, the only signs which are in horizontal orientation are the sign 'C', 'G', 'H', 'O' and 'Q'. When ayaw <60°, the signs are in vertical orientation and therefore, a_{yaw} information is used to offset the orientation, and only the signs other than the aforementioned five signs will be considered as candidate signs. Meanwhile the roll angle deviation only takes two possibilities, which is when hand palm is facing directly at the camera or when hand palm direction is parallel to the camera. The roll orientation of hand is determined by using Hough line detection as discussed in Section 3.7.2.7. The flag is_roll is set to true when the hand is facing the side instead of facing the camera, as shown in Figure 22.

Figure 20: Signs with vertical orientation.

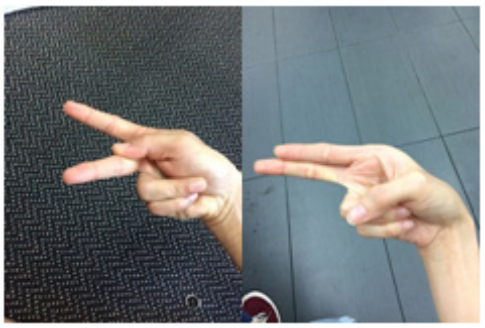

Figure 21: Signs with horizontal orientation.

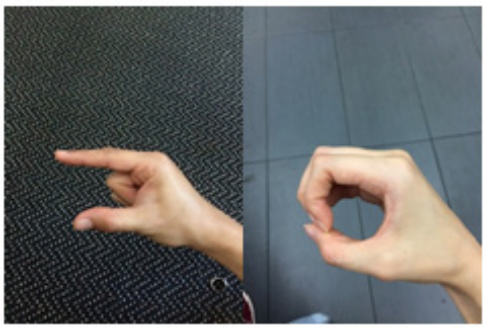

Figure 22: Signs with horizontal and roll orientation.

The yaw orientation, a_{yaw} is obtained by finding the angle between palm center and wrist point. Finding point representing the wrist requires utilization of defect points and arm middle line. The arm middle line can be found by first identifying the leftmost and rightmost points on the bottom border. These points are then used to find the middle point of the arm on the bottom border of image. The line connecting this point to the palm center is the line signifying the angle of arm as shown in Figure 23. The wrist point can then be identified by finding the perpendicular intersection point of the last defect points to the line signifying the angle of arm. Lastly, the angle of yaw orientation, a_{yaw} can be determined. The orientation of the hand is set to be horizontal position instead of the default vertical orientation when a_{yaw} >60°.

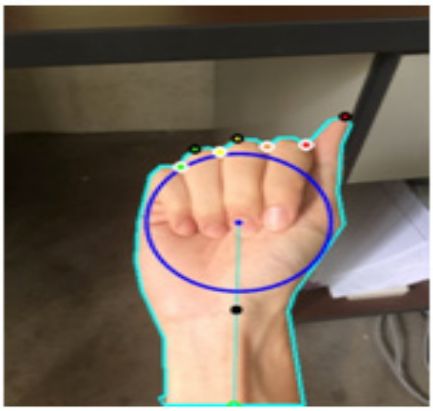

Figure 23: Wrist points (black dot), Arm border points (green dot), Palm center points (blue dot).

Finding hand motion trajectory

The trajectory motion information of dynamic gesture is extracted using Chain code method. The trajectory motion of hand gesture

is quantized into 12 directions, with 30° in each angle, as shown in Figure 24 below. A trajectory information can only be extracted when there are two frames and above. Hence, the trajectory information is only extracted starting the second frame, z >1. In this research, the frames are considered to consist of trajectory motion only when the x or y position displacement of subsequent frames exceeded a certain threshold, in this paper is 10 pixels.

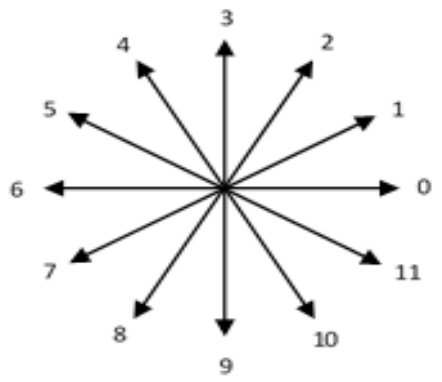

Figure 24: Chain code with 12 directions.

The motion information is obtained by calculating the difference between palm centers in subsequent frames. This can be used to determine the directions in angle of the subsequent movement as well as the speed of travel. The chain code, $c_{code}(n)$ is then identified. The chain code obtained is then filtered by removing the duplicated codes to simplify the classification process, $sc_{code}(m)$, and the result are as shown in Figure 25.

Finding classifying the sign languages

Decision Tree consists of split nodes, which are the internal nodes used to test the input; and leaf nodes, which are the terminal nodes used to infer a set of posterior probabilities for the input, based on statistics collected from training data. Each split node sends the incoming input to one of its children, according to the test result. The proposed framework is able to obtain the posture, orientation as well as the trajectory motion information of the ASL performed in binary representation. Hence the features extracted can be easily classified using simple Decision Tree instead of classification method which requires training. The respective static ASL are able to be classified by identifying the posture and orientation of each individual fingers as shown in Table 4. The dynamic ASL are classified using additional motion trajectory information as shown in Table 5. Simple decision tree can be used to classify the ASL using the following hierarchy:

Sign	Finger representation	Flag is_vertical	Flag is_roll	Other Flags
A	10000	true	-	-
B	03333	true	-	-
D & 1	01000	true	-	-
E	02222	true	-	-
F & 9	00111	true	-	-
I	00001	true	-	-
L	11000	true	-	-
W & 6	01110	true	-	-
X	02000	true	-	-
Y	10001	true	-	-
3	11100	true	-	-
4	01111	true	-	-
5	11111	true	-	-
7	01101	true	-	-
8	01011	true	-	-
V & 2	01100	true	-	is_signK = false
K	01100	true	-	is_signK = true
U	03300	true	-	is_signR = false
R	03300	true	-	is_signR = true
S	00000	true	-	thpost =0
T	00000	true	-	thpost =1
N	00000	true	-	thpost =2
M	00000	true	-	thpost =3
C	00000	false	true	defect =0
O & 0	00000	false	false	-
Q	00000	false	true	defect =1
G	01000	false	false	-
H	03300	false	false	-
P	01100	false	false	-

Table 4: Static ASL and decision criteria.

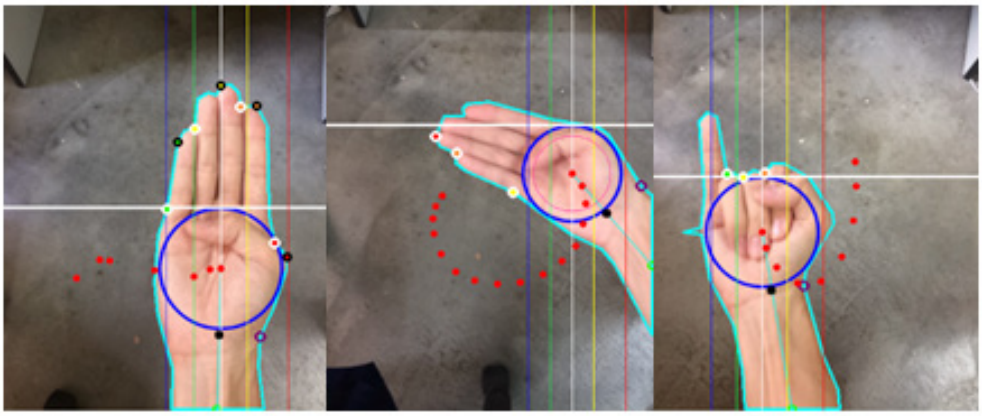

Figure 25: Trajectory motion (red dots).

1. Is static or dynamic? Determined by finding the position of hand moment center of all four subsequent frames. If the hand center differs by less than a threshold, it is static.
2. Orientation of hand is vertical or horizontal?
   Determined by the palm center to wrist angle, is horizontal if a_{yaw} >60°
3. Shape representation of finger posture. Determined by calculating the convexity defects, K-curvature and Hough lines.
4. Chain code for dynamic gestures. The trajectory motion of gestures are recognized after every four frames. The chain code divides the possible movement of the subsequent frame into 12 sections with 30° each.

Sign	Finger representation	Flag is_vertical	Chain Code
J	00001	true	975 / 864
Z	01000	true	070
Yes	00000	true	999
No	00000	false	999
Thank you	03333	true	333
Hi	03333	true	000
Bye	03333	true	666
Please	03333	false	Clockwise
Sorry	00000	false	Anti-clockwise
Welcome	03333	false	975 / 864

Table 5: Dynamic ASL and decision criteria.

Results and Discussion

The size of still images sampled are 3264×2448 pixels, while the size of video frames are 720×540 pixels at 30 frames per second. The proposed framework recognizes 29 unique static posture and 10 unique dynamic signs. As some signs are represented by the same static hand posture, the system can translate a total of 44 ASL including 26 alphabets, 10 numerals and 8 ASL words. The alphabetic ASL recognized are as in Figure 26, numeral ASL recognized are as in Figure 27 and lastly dynamic ASL are in Figure 28 below.

The experiment is carried out separately for static ASL and dynamic ASL. For static ASL, 50 images are collected from 5 different signers for each of the 29 static ASL. A total of 1450 images of static ASL are collected. For dynamic ASL, 10 videos are collected from each of the 10 dynamic ASL, which consist of a total of 100 videos. The proposed framework uses calibration of hand threshold instead of training of classification database, hence all images can be used as test images. The recognition rate for the experimental results obtained for static ASL are as shown in Table 6.

The recognition rate for experimental results obtained for dynamic ASL are as shown in Table 7. For some test images, the features extracted do not match any of the predefined rules of the ASL groups. For these images which could not be recognized, they are not classified into any classes of sign language, hence some test images will not be have classification result. Based on the proposed framework, the average accuracy achieved for 29 static ASL is 85.72%, and average accuracy achieved for 10 dynamic ASL is 77%.

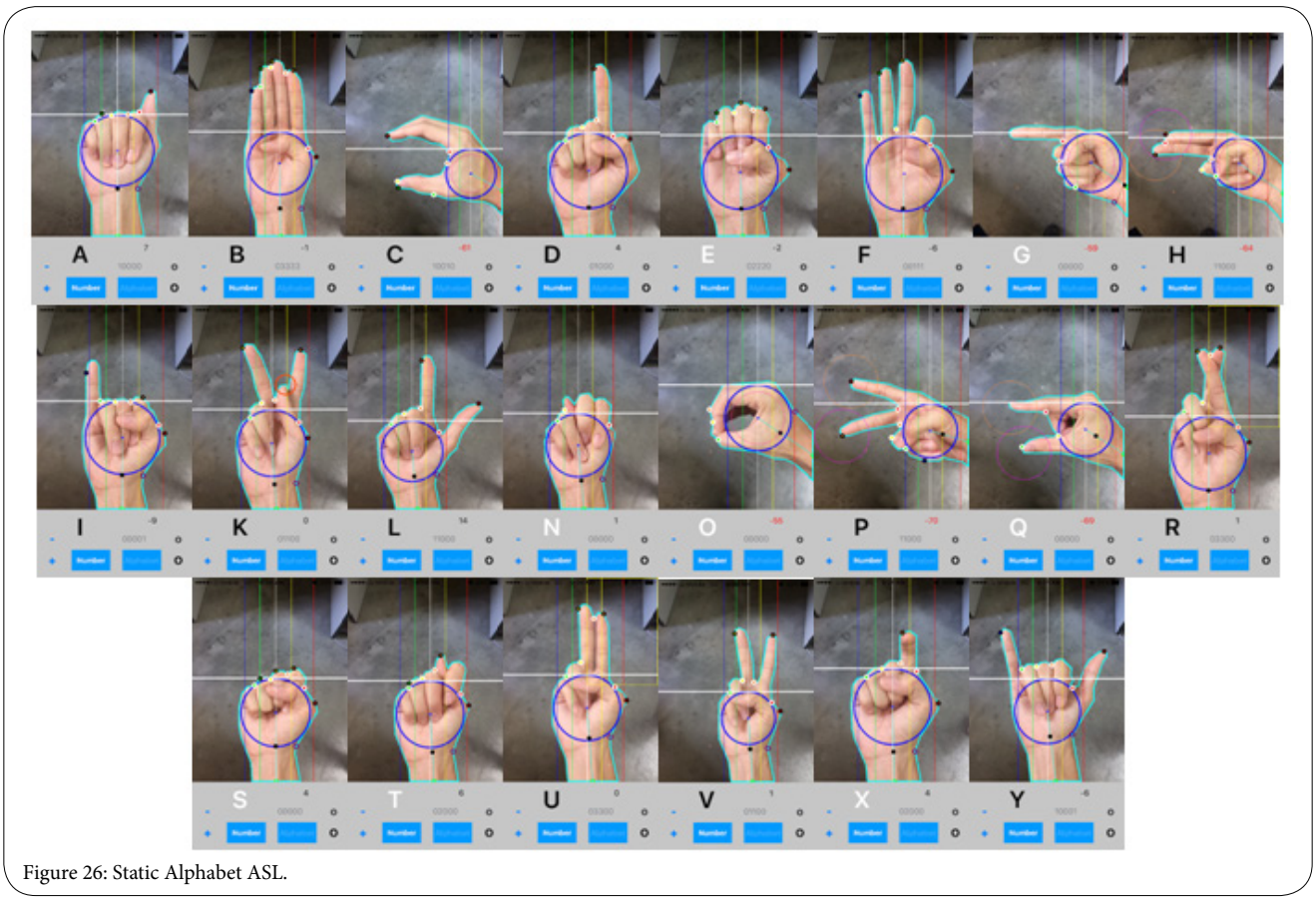

Figure 26: Static Alphabet ASL.

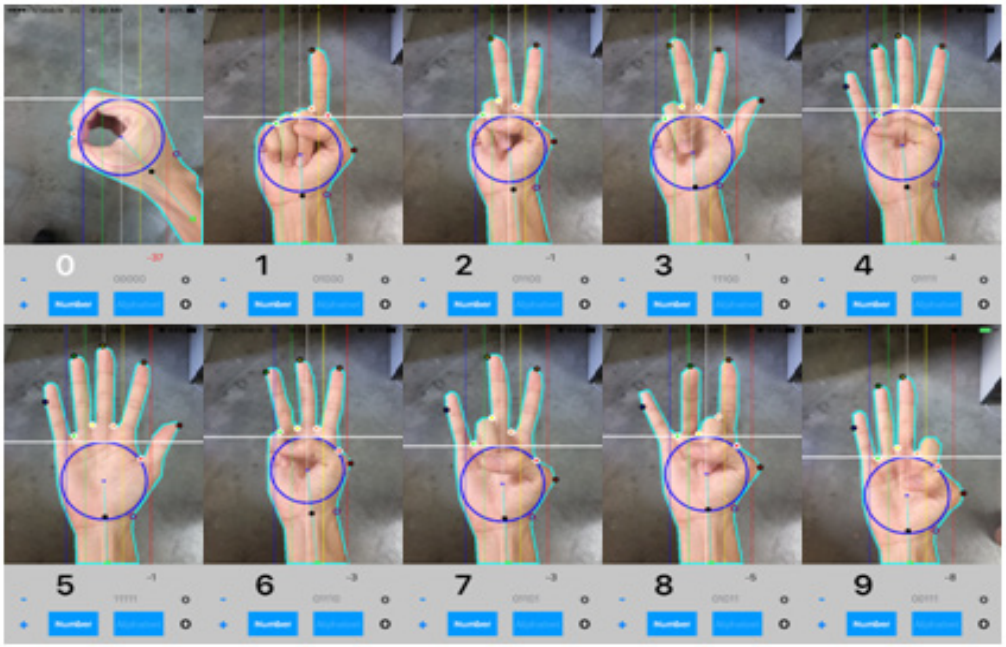

Figure 27: Static Numerical ASL.

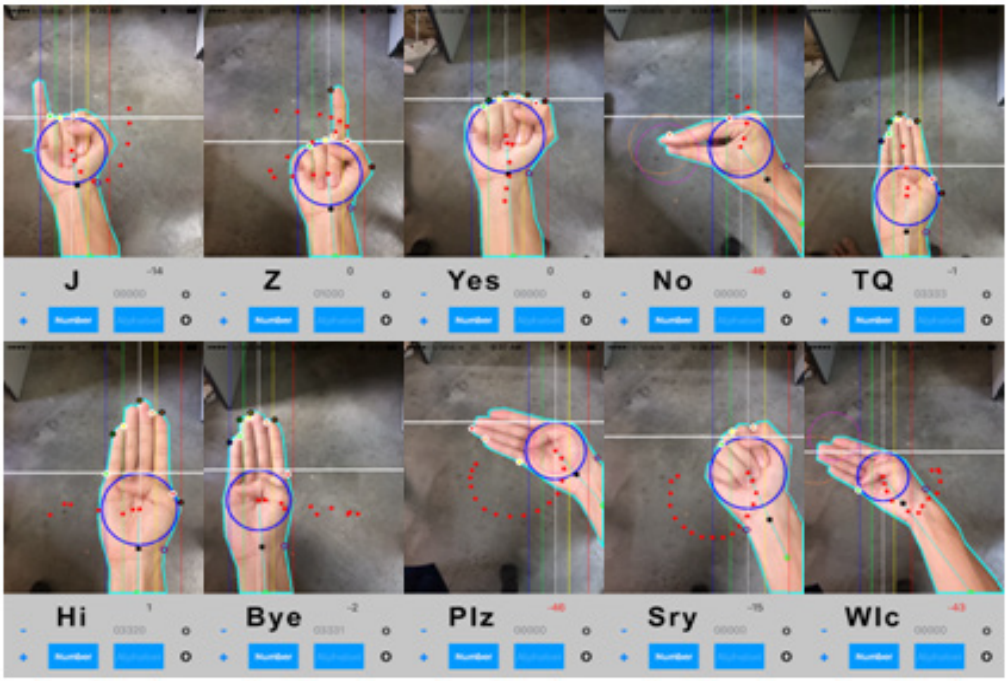

Figure 28: Dynamic ASL.

In this research, the experimental results of two other sign language recognition work is carried out to verify the performance of the proposed method as compared to the benchmark researches. First, a state-of-the-art appearance-based sign language recognition framework experiment based on SURF and SVM are carried out. The experiment is based on research [13] which extracts SURF features from the hand gestures. The SURF features extracted are clustered into 29 classes using *K*-means clustering method. Next, BoF is applied to represent the SURF features with probability features in order to to simplify the dimensionality. Lastly, a one-vs-all SVM is then applied classify the test images. Using the dataset collected in this experiment, with 50 test images from each 29 classes, a 10-fold cross validation is performed. The recognition of six classes of gestures obtained by the author in [13] are as shown in Table 8. Whereas the average accuracy obtained using the dataset collected in this experiment, the accuracy of recognizing 29 classes of ASL is 48.48%, as shown in Figure 29. Similar recognition approaches are also used in research [14,15].

Sign	Accuracy (%)	False Negative		
A	90	S(6)	X(4)	
B	94	4(6)		
C	96	Q(4)		
D & 1	90	X(8)		
E	72	M(10)	N(12)	S(4)
F & 9	96			
G	92	H(6)		
H	96	G(4)		
I	84	S(6)		
K	80	V(12)		
L	88	D(4)		
M	66	N(4)	T(12)	S(10)
N	72	M(8)	T(14)	S(4)
O & 0	100			
P	80	Q(12)		
Q	78	P(18)		
R	78	U(12)		
S	76	E(6)	M(10)	N(8)
T	72	M(6)	N(12)	S(8)
U	88	R(6)	V(4)	
V & 2	90	K(10)		
W	84	V(8)		
X	82	D(12)	T(6)	
Y	88	A(8)	I(4)	
3	96			
4	86	B(6)		
5	92			
7	88	8(6)		
8	92	7(4)		
Average	85.72			

Table 6: Recognition Rate of 26 Static ASL.

Sign	Accuracy (%)
J	80
Z	90
Yes	70
No	70
Thank you	80
Hi	90
Bye	90
Please	70
Sorry	70
Welcome	60
Average	77

Table 7: Recognition Rate of 10 Dynamic ASL.

Posture Name	Number of Images	Correct	Incorrect	Recognition Accuracy	Recognition Time (Second/ frame)
Fist	97	92	5	94.85%	0.017
Palm	102	100	2	98.04%	0.017
C	112	108	4	96.43%	0.017
V(Two)	95	91	4	95.79%	0.017
Five	134	127	7	94.78%	0.017

Table 8: Recognition rate from research [13].

The second benchmark experiment done is a model-based approach. The experiment is based on method in research [29]. In the research, distance transform is used to obtain the palm center and radius information. Convexity defects is then used to extract all the fingertip position. Decision Tree method is used in classification stage. The database used for this experiment is only 15 static ASL, this is because it is not able to recognize horizontal orientated signs namely 'C', 'G', 'H' 'O' and 'Q' as well as signs in fist posture namely 'M', 'N'. 'S' and 'T'. Also, gesture which are similar such as 'V' and 'U' are unable to be determined with the limited features. The average accuracy achieved for second experiment is 65.73%. Table 9 shows the accuracy obtained from the recognition of five gestures in research [29]. The accuracy of recognizing 15 ASL using the benchmark method are as shown in Figure 30.

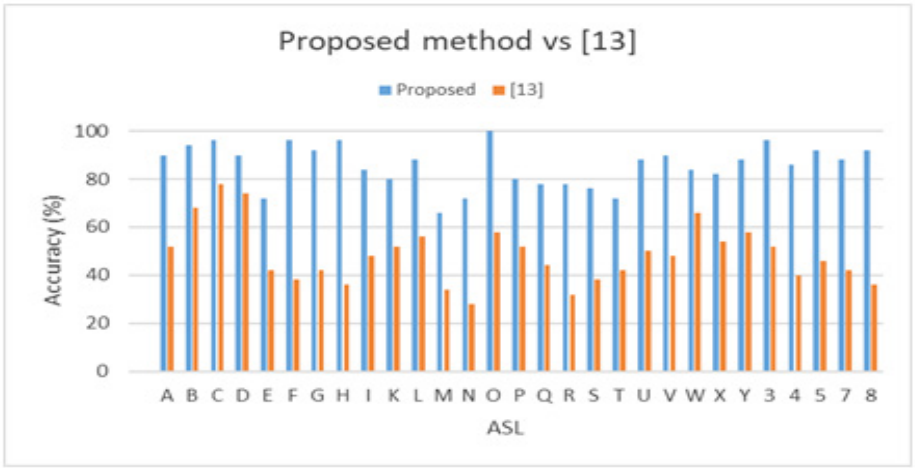

Figure 29: Accuracy of proposed method and [13].

Posture Name	Recognition rates for the number of fingers						Average rate
	0	1	2	3	4	5	
Method in research [29]	92%	100%	100%	90%	100%	86%	95%

Table 9: Recognition rate from research [29].

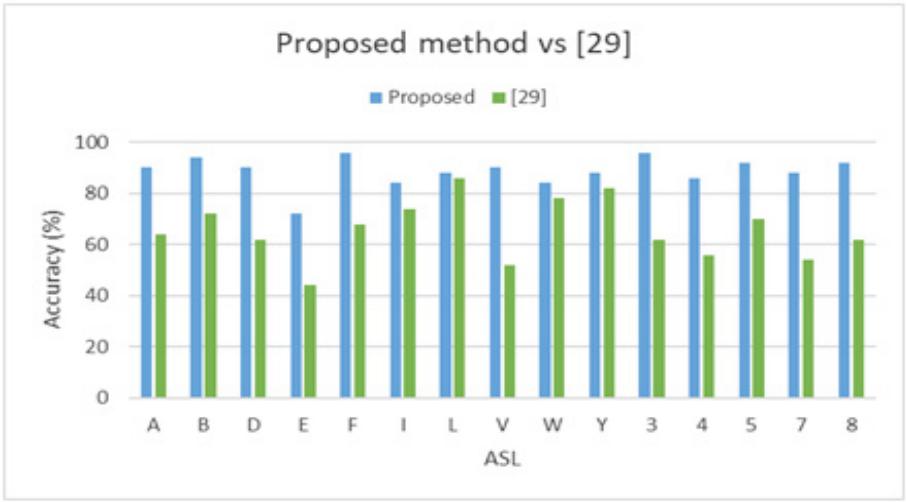

Figure 30: Accuracy of proposed method and [29].

From the research conducted in paper [13], the number of gestures used are only six, and the average accuracy obtained is 96.23%. However, when the number of gestures is increased to 29 gestures instead of six in our experiment, the accuracy greatly deteriorates to 48.48%. It is because as the number of classes increases, the separability between the SURF features of different gestures decreases, hence, the misclassification rate increases. Furthermore, in appearance-based recognition framework, expensive feature extraction and computational time are required to extract and to train all the visual features from the images, which are also stated in research [23].

In 2D model-based approach, convexity defects are a notable method for fingertip recognition. In order to demonstrate the effect of selection of features in affecting the scalability of the gesture, the second experiment is done using method as described in [29]. The feature extraction phase is replaced using the method in [29] while all the other stages uses the same as the proposed method as only the feature extraction algorithm are to be compared. However, only 15 ASL are able to be recognized, as there are not enough features and rules to define the other ASL. The average recognition rate achieved is 65.73%. This first experiment shows that model-based convexity defects can perform better than the commonly used appearance-based method such as the SURF and SVM techniques. The use of convexity defects and its intrinsic nature of describing the fingers are proven to outperform the SURF and SVM appearance-based feature extraction methods. The second experiment shows that the proposed convexity defect method out-performed the method used in [29] due to a more well-defined set of features and rules. The similar convexity defect rules are also observed in research [19,25,30].

In this paper, the classification criteria of each ASL are written explicitly, and as opposed to appearance-based method, it does not require computational heavy algorithm. The classification of ASL can be achieved by implementing a low computational cost Decision Tree classifier. The average accuracies achieved is in our paper is 85.72% for 29 static ASL and 77% for 10 dynamic ASL. The use of finger webs as an intermediate feature in determining the fingertips is proposed in this paper. The finger web has the advantage of being always present despite different posture of hand gestures. This allows the potential fingertips candidate from convexity defects and K-curvature features to be correctly identified to each finger of the hand despite various hand posture. The usage of Hough line feature in finding the edges of the fingers allow ASL which are visually similar to be differentiated. Besides, algorithm designed to specifically differentiate two different visually similar ASL has shown to be able to differentiate the ASL apart.

With robust rules and features being implemented based on convexity defects, K-curvature and Hough line, the framework can be extended to recognize different posture of hand gestures. This includes other Sign Languages and also different varieties of hand gestures. The recognition of new signs can be achieved by defining the finger representation and other features representing the new signs, and these features can be appended to the signs already defined in Table 4. Decision Tree classification can then be used to classify the new signs when the feature criteria of the new signs added is met.

The proposed method uses the 11 scale-invariant k parameters introduced in Table 1. The coefficients in the parameters are the measurement of the ratio of different hand landmarks with respect to the palm circle. This allows the framework to be scale-invariant and hence is also adaptive to different sizes of hand.

The coefficients are optimally obtained through measurement of ratio that best represents the hand anatomy of signers. The coefficients can be represented by lines and circles as shown in Figure 3 and Figure 4 respectively. The accuracy of the algorithm does not rely directly on the values of the coefficients. Its function is to set a rough estimation or reference on the location of each hand landmark. For instance, the finger web of the thumb should fall between the vertical lines

representing kth and kin. They can be verified visually and can be easily recalibrated if required. The optimum coefficients are then tested and verified through different hand posture of 5 different signers.

The same set of coefficients has been shown to be able to recognize ASL by five different signers with different palm size. It can be extended to recognize other signers with similar hand anatomy. When ASL of new signers with relatively different hand ratio such as a larger palm with shorter fingers are introduced, the framework has the flexibility of calibration of the coefficients to recognize ASLs of new signers. As with other sign language algorithms, when outliers are introduced to the system, calibration or training of classifiers will be required to obtain better accuracy.

In this research, the experiment is performed in indoor locations, all under well-lit lighting conditions. The segmentation accuracy might be affected when subjected to a different or extreme background lighting condition. The proposed feature extraction framework is however independent of the segmentation techniques used in this research. Hence, the framework can be implemented with other segmentation methods that can output two information of hand gestures, namely edges information as shown in Figure 5(e) and binary information as in shown Figure 5(g). Thus, different segmentation methods can be used to improve the segmentation accuracy of the proposed framework in real-world applications such as with different skin color or background conditions.

This research is implemented on a smartphone to facilitate the image data collection process and can be implemented on other platforms and devices. Lastly, this paper proposes a sign language recognition framework with the aims to contribute to the bridging of communication gap between the deaf and the rest of the society.

Conclusion

In appearance-based recognition framework, expensive feature extraction and computational time are required to extract and to train all the visual features from the images, which are also stated in research [23]. This is not required in the proposed method. In 2D model-based approach, convexity defects is a notable method for fingertip recognition as its intrinsic nature of describing the fingers are suitable to be used to describe the fingers' posture. In this paper, the classification criteria of each ASL are written explicitly, and as opposed to appearance-based method, it does not require computational heavy algorithm. The classification of ASL can be achieved by implementing a low computational cost Decision Tree classifier. The average accuracies achieved is in our paper is 85.72% for 29 static ASL and 77% for 10 dynamic ASL. The use of finger webs as an intermediate feature in determining the fingertips is proposed in this paper. The finger web has the advantage of being always present despite different posture of hand gestures. This allows the potential fingertips candidate from convexity defects and K-curvature features to be correctly identified to each finger of the hand despite various hand posture. The usage of Hough line feature in finding the edges of the fingers allow visually similar ASL to be differentiated. Besides, algorithm designed to specifically differentiate two different visually similar ASL has shown to be able to differentiate the ASL apart. With robust rules and features being implemented, the framework can be extended to recognize hand gestures with different posture including other Sign Languages. The recognition of new signs can be achieved by defining the finger representation and other features representing the new signs, and these features can be appended to the system. Decision Tree classification can then be used to classify the new signs when the feature criteria of the new signs added is met. Lastly, this paper proposes a sign language recognition framework with the aims to contribute to the bridging of communication gap between the deaf and the rest of the society.

Competing Interest

The authors declare that they have no conflicts of interest.

References

1. Murthy GR, Jadon RS (2009) A review of vision based hand gestures recognition. International Journal of Information Technology and Knowledge Management 2: 405-410.
2. Shaik KB, Ganesan P, Kalist V, Sathish BS, Jenitha JM, et al. (2015) Comparative study of skin color detection and segmentation in HSV and YCbCr color space. Procedia Computer Science 57: 41-48.
3. Tsagaris A, Manitsaris S (2013) Colour space comparison for skin detection in finger gesture recognition. International Journal of Advances in Engineering & Technology 6: 1431.
4. Qiu-yu Z, Jun-chi L, Mo-yi Z, Hong-xiang D, Lu L, et al. (2015) Hand Gesture Segmentation Method Based on YCbCr Color Space and K-Means Clustering. Interaction 8: 106-116.
5. Yun L, Lifeng Z, Shujun Z (2012) A hand gesture recognition method based on multi-feature fusion and template matching. Procedia Engineering 29: 1678-1684.
6. Kakumanu P, Makrogiannis S, Bourbakis N (2007) A survey of skin-color modeling and detection methods. Pattern recognition 40: 1106-1122.
7. Nanivadekar PA, Kulkarni V (2014) Indian Sign Language Recognition: Database Creation, Hand Tracking and Segmentation. In: Circuits, Systems, Communication and Information Technology Applications. International Conference, IEEE.
8. Rekha J, Bhattacharya J, Majumder S (2011) Shape, texture and local movement hand gesture features for indian sign language recognition. International Conference on Trendz in Information Sciences & Computing.
9. Sun HM (2010) Skin detection for single images using dynamic skin color modeling. Pattern recognition 43: 1413-1420.
10. Ghotkar AS, Kharate GK (2012) Hand segmentation techniques to hand gesture recognition for natural human computer interaction. International Journal of Human Computer Interaction 3: 15.
11. Akmeliawati R, Dadgostar F, Demidenko S, Gamage N, Kuang YC, et al. (2009) Towards real-time sign language analysis via markerless gesture tracking. Instrumentation and Measurement Technology Conference.
12. Shin JH, Lee JS, Kil SK, Shen DF, Ryu JG, et al. (2006) Hand region extraction and gesture recognition using entropy analysis. International Journal of Computer Science and Network Security 6: 216-222.
13. Dardas NH, Georganas ND (2011) Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. IEEE 11: 3592-3607.
14. Kim D, Dahyot R (2008) Face components detection using SURF descriptors and SVMs. Machine Vision and Image Processing Conference. IEEE.
15. Dardas N, Chen Q, Georganas ND, Petriu EM (2010) Hand gesture recognition using bag-of-features and multi-class support vector machine. Haptic Audio-Visual Environments and Games. IEEE.
16. Bay H, Tuytelaars T, Van Gool L (2006) Surf: Speeded up robust features. In: European conference on computer vision. Springer.
17. Bao J, Song A, Guo Y, Tang H (2011) Dynamic hand gesture recognition based on SURF tracking. Electric Information and Control Engineering. IEEE.
18. Zaki MM, Shaheen SI (2011) Sign language recognition using a combination of new vision based features. Pattern Recognition Letters 32: 572-577.
19. Shukla J, Dwivedi A (2014) A method for hand gesture recognition. Communication Systems and Network Technologies. IEEE.

20. Yeo HS, Lee BG, Lim H (2015) Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware. Multimedia Tools and Applications 74: 2687-2715.
21. Tofighi G, Monadjemi SA, Ghasem-Aghaee N (2010) Rapid hand posture recognition using adaptive histogram template of skin and hand edge contour. Machine Vision and Image Processing. IEEE.
22. Han G, Choi H (2014) MPEG-U based advanced user interaction interface system using hand posture recognition. Advanced Communication Technology. IEEE.
23. Billiet L, Mogrovejo O, Antonio J, Hoffmann M, Meert W, et al. (2013) Rule-based hand posture recognition using qualitative finger configurations acquired with the Kinect. Proceedings of the 2nd International Conference on Pattern Recognition Applications and Methods.
24. Van□o M, Minárik I, Rozinaj G (2012) Gesture identification for system navigation in 3D scene. ELMAR. IEEE.
25. Ganapathyraju S (2013) Hand gesture recognition using convexity hull defects to control an industrial robot. Instrumentation Control and Automation. IEEE.
26. Manresa C, Varona J, Mas R, Perales FJ (2005) Hand tracking and gesture recognition for human-computer interaction. ELCVIA Electronic Letters on Computer Vision and Image Analysis 5: 96-104.
27. Lahiani H, Elleuch M, Kherallah M (2015) Real time hand gesture recognition system for android devices. Intelligent Systems Design and Applications. IEEE.
28. Tariq M, Iqbal A, Zahid A, Iqbal Z, Akhtar J, et al. (2012) Sign language localization: Learning to eliminate language dialects. Multitopic Conference. IEEE.
29. Choi J, Seo BK, Lee D, Park H, Park JI (2013) RGB-D camera-based hand shape recognition for human-robot interaction. Robotics (ISR). IEEE.
30. Hussain I, Talukdar AK, Sarma KK (2014) Hand gesture recognition system with real-time palm tracking. India Conference. IEEE.
31. Yun L, Lifeng Z, Shujun Z (2012) A hand gesture recognition method based on multi-feature fusion and template matching. Procedia Engineering 29: 1678-1684.
32. Swain MJ, Ballard DH (1992) Indexing via color histograms. Active Perception and Robot Vision: Springer.
33. Chen FS, Fu CM, Huang CL (2003) Hand gesture recognition using a real-time tracking method and hidden Markov models. Image and vision computing 21: 745-758.