Implementation of Sign Language Recognition Using Image Processing System

Abstract
Sign Language Recognition has emerged as one of the important area of research in Computer Vision. The difficulty faced by the researchers is that the instances of signs vary with both motion and appearance. Thus, in this paper a novel approach for recognizing various alphabets of Indian Sign Language is proposed where continuous video sequences of the signs have been considered. The proposed system comprises of three stages: Pre-processing stage, Feature Extraction and Classification. Pre-processing stage includes skin filtering, histogram matching. Eigen values and Eigen Vectors were considered for feature extraction stage and finally Eigen value weighted Euclidean distance is used to recognize the sign. It deals with bare hands, thus allowing the user to interact with the system in natural way. We have considered 24 different alphabets in the video sequences and attained a success rate of 96.25%.
1. Introduction

The aim of this article is to build a system which is useful for physically challenged people to communicate with society and useful for education purpose. A Sign Language is a language in which communication between people are made by visually transmitting the sign patterns to express the meaning. Thus, because of which it has attracted many researchers in this field from long. Many researchers have been working in different sign languages like American Sign Language, British Sign Language, Taiwanese Sign Language, etc. but few works has been made progress on Indian Sign Language. The hearing impaired people becomes neglected from the society because the normal people never try to learn ISL nor try to interact with the hearing impaired people. This becomes a curse for them and so they mostly remain uneducated and isolated. Thus recognition of sign language was introduced which has not only been important from engineering point of view but also for the impact on society.

Thus we propose a special purpose image processing algorithm based on Eigen vector to recognize various signs of Indian Sign Language for live video sequences with high accuracy. Various difficulties faced by different researchers have been tried to minimize with our approach. Recognition rate of 96.25% was achieved. The experiment was carried out with bare hands, thus removing the difficulty faced using the gloves. We have extended our work for video sequences in this project. There have been many previous works which extracted certain features of the hand for finger detection. Some common features extracted include hand silhouettes, contours, key points distributed along hand (fingertips, joints) and distance-transformed images. There have also been works where finger detection has been accomplished via color segmentation and contour extraction. But this technique requires fine-tuning every time the system switches to a new user as the color complexion varies from person to person.

2. System Overview

ISL Alphabet:

Indian Sign Language was developed so that the deaf people in the society can interact with the normal people without any difficulties. Here in this project, we have considered the alphabets of ISL which involves considered which is shown in Figure.

![Figure 2.1: Various alphabets of Indian sign language](image-url)
3. Proposed Methodology

In this project we present a robust and efficient technique for finger detection. Our method has three main phases of processing viz., Edge Detection, Clipping and Boundary Tracing. The first phase employs canny edge operator and produces an edge detected image which reduces the number of pixels to be processed at runtime. The next phase clips the undesirable portion of the edge detected image for further processing. The final phase traces the boundary of the image and in the process detects finger tips which aid in finger detection.

![Phases Of Methodology](image)

Figure 3.1: Phases Of Methodology

4. Canny Edge Detection

Edge detection is a phenomenon of identifying points in a digital image at which the image brightness changes sharply or more formally, has discontinuities. The Canny algorithm uses an optimal edge detector based on a set of criteria which include finding the most edges by minimizing the error rate, marking edges as closely as Possible to the actual edges to maximize localization, and marking edges only once when a single edge exists for minimal response. According to Canny, the optimal filter that meets all three criteria above can be efficiently approximated using the first derivative of a Gaussian function.

\[
G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \ldots \quad (1)
\]

\[
\frac{\partial G(x,y)}{\partial x} = a. x. e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \ldots \quad (2)
\]

\[
\frac{\partial G(x,y)}{\partial y} = a. y. e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \ldots \quad (3)
\]

The first stage involves smoothing the image by convolving with a Gaussian filter. This is followed by finding the gradient of the image by feeding the smoothed image through a convolution operation with the derivative of the Gaussian in both the vertical and horizontal directions.

\[
G = \sqrt{Gx^2 + Gy^2} \quad \ldots \quad (4)
\]
\[ \theta = \arctan \left( \frac{\text{op}}{\Delta x} \right) \]  \hspace{1cm} \ldots \ldots (4)

The non-maximal suppression stage finds the local maxima in the direction of the gradient, and suppresses all others, minimizing false edges. The local maxima are found by comparing the pixel with its neighbour’s along the direction of the gradient. This helps to maintain the single pixel thin edges before the final thresholding stage. Instead of using a single static threshold value for the entire image, the Canny algorithm introduced hysteresis thresholding, which has some adaptivity to the local content of the image. There are two threshold levels, \( th, \) high and \( tl, \) low where \( th > tl. \) Pixel values above the \( th \) value are immediately classified as edges. By tracing the edge contour, neighbouring pixels with gradient magnitude values less than \( th \) can still be marked as edges as long as they are above \( tl. \) This process alleviates problems associated with edge discontinuities by identifying strong edges, and preserving the relevant weak edges, in addition to maintaining some level of noise suppression. While the results are desirable, the hysteresis stage slows the overall algorithm down considerably. The performance of the Canny algorithm depends heavily on the adjustable parameters, \( \sigma, \) which is the standard deviation for the Gaussian filter, and the threshold values, \( th \) and \( tl. \) \( \sigma \) also controls the size of the Gaussian filter.

The non-maximal suppression stage finds the local maxima in the direction of the gradient, and suppresses all others, minimizing false edges. The local maxima are found by comparing the pixel with its neighbour’s along the direction of the gradient. This helps to maintain the single pixel thin edges before the final thresholding stage. Instead of using a single static threshold value for the entire image, the Canny algorithm introduced hysteresis thresholding, which has some adaptivity to the local content of the image. There are two threshold levels, \( th, \) high and \( tl, \) low where \( th > tl. \) Pixel values above the \( th \) value are immediately classified as edges. By tracing the edge contour, neighbouring pixels with gradient magnitude values less than \( th \) can still be marked as edges as long as they are above \( tl. \) This process alleviates problems associated with edge discontinuities by identifying strong edges, and preserving the relevant weak edges, in addition to maintaining some level of noise suppression. While the results are desirable, the hysteresis stage slows the overall algorithm down considerably. The performance of the Canny algorithm depends heavily on the adjustable parameters, \( \sigma, \) which is the standard deviation for the Gaussian filter, and the threshold values, \( th \) and \( tl. \) \( \sigma \) also controls the size of the Gaussian filter.

![Image of a hand gesture before and after edge detection.](image)

Figure 4.1 : Image of a hand gesture before and after edge detection.

The bigger the value for \( \sigma, \) the larger the size of the Gaussian filter becomes. This implies more blurring, necessary for noisy images, as well as detecting larger edges. As expected, however, the larger the scale of the Gaussian, the less accurate is the localization of the edge. Smaller values of \( \sigma \) imply a smaller Gaussian filter which limits the amount of blurring, maintaining finer edges in the image. The user can tailor the algorithm by adjusting these parameters to adapt to different environments with different noise levels.

4.1 Clipping

Clipping is used to cut a section through the data (image) currently being rendered. The image contents that pertain to the area of interest are retained after clipping. The edge detected image contains portions which are unnecessary for further analysis. Hence we eliminate them by adopting two techniques discussed below. The first technique examines pixels from the bottommost \( y \) level and at each level checks if there are three or more consecutive white pixels. If the above condition is satisfied we mark this \( y \)-level as “\( y1 \)”. The second technique exploits the fact that most of the edge detected images of hand gestures have the wrist portion which has a constant difference between the either ends on the same \( y \)-level. When it approaches the palm and region of the hand above it, this difference increases drastically. We make use of this fact and find the \( y \)-level where this event occurs and mark this \( y \)-level as “\( y2 \)”. 
Now we choose the maximum of \((y_1, y_2)\) as the clipping y-level. All the pixels below this y-level are now cleared by overwriting them with a black pixel.

### 4.2 Boundary Tracing

This phase of the algorithm is the heart of processing. The edge detected image which is clipped serves as the input to this phase. The output is the traced image where the trace-points are highlighted in blue and the points where the fingertip is detected are highlighted in red. This phase consists of the following steps: identifying the optimal y-level, identifying the initial trace direction, tracing with appropriate switch of direction, re-joining the trace on encountering breaks, fingertip detection. In the explanation of the above steps, the following are assumed.

#### Table 4.2.1: Boundary tracing variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minx</td>
<td>Minimum x-coordinate of the set of white pixels in the clipped image.</td>
</tr>
<tr>
<td>Maxx</td>
<td>Maximum x-coordinate of the set of white pixels in the clipped image.</td>
</tr>
<tr>
<td>Miny</td>
<td>Minimum y-coordinate of the set of white pixels in the clipped image.</td>
</tr>
<tr>
<td>Maxy</td>
<td>Maximum y-coordinate of the set of white pixels in the clipped image.</td>
</tr>
<tr>
<td>Dx</td>
<td>Difference between minx and maxx.</td>
</tr>
<tr>
<td>Dy</td>
<td>Difference between miny and maxy.</td>
</tr>
<tr>
<td>UD</td>
<td>Up/Down flag which indicates upward or downward direction trace.</td>
</tr>
</tbody>
</table>

The proposed system is shown in this Figure which comprises of 3 major stages-preprocessing stage which includes the skin filtering and histogram matching to find out the similarity between frames, Feature Extraction stage in which the Eigen values and Eigen vector are being considered as features and finally Eigen value weighted Euclidean distance based classification technique.
The proposed system is shown in this Figure which comprises of 3 major stages - preprocessing stage which includes the skin filtering and histogram matching to find out the similarity between frames, Feature Extraction stage in which the Eigen values and Eigen vector are being considered as features and finally Eigen value weighted Euclidean distance based classification technique.

5. Processing Of Sign Language Recognition

5.1 Data acquisition

The first step for our proposed system is the capturing of the video using webcam where different alphabets were taken into consideration. 10 different alphabets were considered for testing from 20 people. Some of the continuous video frames captured are given in Figure.
5.2 Detection of hand gesture

Skin Filtering was performed to the input video frames for detection of hand gestures. It was done so that the required hand could be extracted from the background. Skin Filtering is a technique used for separating the skin colored regions from the non-skin colored regions. The steps used in this skin filtering are shown in Figure.

![Block diagram of skin filter](image)

**Figure 5.2.1: Block diagram of skin filter**

At first, the input frame was converted to HSV color space. This step was taken because HSV color space was less sensitive to illumination changes compared to RGB. Then it was filtered, smoothed and finally the biggest binary linked object was being considered so as to avoid consideration of skin colored objects other than hand. The resultant image is a binary image with hand regions in white and background in black color. The filtered hand is then found out.

5.3 Histogram Matching

After extracting out the skin colored regions from the background, histogram matching is done in the next step. They following steps describe the process of histogram matching:

**Step 1:** The histograms of all the frames of the video are found out.

**Step 2:** The similarities of the consecutive frames are checked by finding out the difference of their histogram. Difference (n) = Hist(n) - Hist(n-1)

Where Hist represents Histogram and n represents current frame.

**Step 3:** If the difference is found to be above a certain threshold, they are considered as similar. This difference is found out for ‘n’ number of frames. We have chosen the threshold ‘n’ to be 17.

**Step 4:** If all the ‘n’ frames shows similarities, then it is considered to be an unidentified sign and further steps of feature extraction and classification is carried on.
6. Software & Components Used

**Matlab:** MATLAB (matrix laboratory) is a multiparalism numerical computing in environment and fourth generation programming language by math works. MATLAB allows matrix manipulations, plotting the functions and data implementation of algorithm creation of user interface with program written in other languages including c, c++, java and fortran.

**Components: Web camera**

The webcam is used to capture the image which is the input to the proposed system. The images have been captured using a 8 Mega Pixel Canon Power Shot S3 IS. The captured images are of resolution 640x480.

**PC**

A personal computer will be needed to implement the algorithm for the proposed system. The specification involves are, Intel core i5 running at 3.00 GHz processor machine, Windows 7 Home basic (64 bit), 4GB RAM.

7. Experimental Result

Different images were tested and found that the new technique of classification was found to show 97% accuracy. In this project we have successfully create the database for background image. With the help camera we had captured images from video and subtracted those images with background image. Resultant image will be required movement or input symbol. Now using skin filter, we have successfully differentiate between back ground image and body movement.

The data set used for training the recognition system consisted of 10 signs of ISL for 3 people. Thus a total of 30 images were stored in database. One parameter was considered in our system i.e. the threshold ‘n’ which is the number of frames it has to check for similarity to determine whether it was a sign or not.

![Sign language Recognition](image)

**Figure 7.1: Result Snapshot**
Advantages:
1. It bridges the gap between normal people and hearing impaired people.
2. Communication between physically challenged and normal people is improved.
3. Interaction of physically challenged people with the society or environment.
5. Dumb people will not be neglected from the society.

Disadvantages
1. Project can’t be implemented without camera and pc.

Application:
1. Industries.
2. Business Application.
3. Automatic guidance system.

8. Conclusion & Future Scope
A system was designed for Indian Sign Language Recognition. It was able to handle different static alphabets of Indian Sign Languages. We have tried to improve the recognition rate compared to the previous works. Moreover, we have considered both hands in our project. As we have performed the experiments with only the static images so out of the 26 alphabets only 10 were considered. We hope to deal with dynamic gestures in future. Moreover only 30 images were considered in this project so in future we hope to extend it further. In future work, sensor based contour analysis can be employed to detect which fingers in particular are open. This will give more flexibility to interpret the gestures. Furthermore, hand detection method using texture and shape information can be used to maximize the accuracy of detection in cluttered background. More importantly, we need to develop algorithms to cover other signs in the Indian Sign Language that have all the fingers closed. An even bigger challenge will be to recognize signs that involve motion (i.e., where various parts of the hand move in specific ways).
In our future work, we plan to complete other modules of the overall solution needed to enable communication between blind and deaf people. In particular, we will focus on translating the recognized sequences of signs to continuous text (i.e., words and sentences) and then to render the text in speech that can be heard by blind people.

References