Available online at www.sciencedirect.com

ScienceDirect

Expert Systems with Applications 32 (2007) 24–37

Expert Systems with Applications

www.elsevier.com/locate/eswa

American sign language (ASL) recognition based on Hough transform and neural networks

Qutaishat Munib *, Moussa Habeeb, Bayan Takruri, Hiba Abed Al-Malik

Department of Computer Information System, King Abdullah II School for Information Technology, University of Jordan, University Street, Amman 11942, Jordan

Abstract

The work presented in this paper aims to develop a system for automatic translation of static gestures of alphabets and signs in American sign language. In doing so, we have used Hough transform and neural networks which is trained to recognize signs. Our system does not rely on using any gloves or visual markings to achieve the recognition task. Instead, it deals with images of bare hands, which allows the user to interact with the system in a natural way. An image is processed and converted to a feature vector that will be compared with the feature vectors of a training set of signs. The extracted features are not affected by the rotation, scaling or translation of the gesture within the image, which makes the system more flexible.

The system was implemented and tested using a data set of 300 samples of hand sign images; 15 images for each sign. Experiments revealed that our system was able to recognize selected ASL signs with an accuracy of 92.3%.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: American sign language; Neural network; Hough transform; Canny edge detection; Sobel edge detection; Feature extraction

1. Introduction

The sign language is the fundamental communication method between people who suffer from hearing defects. In order for an ordinary person to communicate with deaf people, a translator is usually needed to translate sign language into natural language and vice versa (International Bibliography of Sign Language, 2005; International Journal of Language & Communication Disorders, 2005).

As a primary component of many sign languages and in particular the American Sign Language (ASL), hand gestures and finger-spelling language plays an important role in deaf learning and their communication. Therefore, sign language can be considered as a collection of gestures, movements, postures, and facial expressions corresponding to letters and words in natural languages.

A gesture is defined as a dynamic movement, such as waving hi, hello or good-bye. Simple gestures are made in two ways (Sturman & Zeltzer, 1994; Watson, 1993). The first way involves a simple or complex posture and change in the position or orientation of the hand, such as making a pinching posture and changing the hand's position. The second way entails moving the fingers in some way with no change in the position and orientation of the hand, for example, moving the index and middle finger back and forth to urge someone to move closer. A complex gesture is one that includes finger; wrist or hand movement (i.e. changes in the position and orientation). There are two types of gesture interaction: communicative gestures work as a symbolic language (which is our focus in this research) and manipulative gestures provide multi-dimensional control. Moreover, we can divide gestures into *static* gestures (hand postures) and *dynamic* gestures (Cutler & Turk, 1998; Hong et al., 2000). Indeed the hand motion conveys as much meaning as their posture does.

* Corresponding author.
E-mail address: maq@ju.edu.jo (Q. Munib).

0957-4174/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2005.11.018

A static sign is determined by a certain configuration of the hand, while a dynamic gesture is a moving gesture determined by a sequence of hand movements and configurations. Dynamic gestures are sometimes accompanied with body and facial expressions.

The aim of sign language recognition is to provide an easy, efficient and accurate mechanism to transform sign language into text or speech. With the help of computerized digital image processing (Gonzalez, Woods, & Eddins, 2004) and neural networks techniques (Haykin, 1999), the system that can recognize the alphabet flow can recognize and interpret ASL words and phrases. For a gesture recognition system, there are four main components: gesture modeling, gesture analysis, gesture recognition and gesture-based application systems.

1.1. American sign language

American Sign Language (ASL) (International Bibliography of Sign Language, 2005; National Institute on Deafness & Other Communication Disorders, 2005) is a complete language that employs signs made with the hands and other gestures, including facial expressions and postures of the body. ASL also has its own grammar that is different from other sign languages such as English and Swedish. ASL consists of approximately 6000 gestures of common words with finger spelling used to communicate unclear words or proper nouns. Finger spelling uses one hand and 26 gestures to communicate the 26 letters of the alphabet. The 26 alphabets of ASL are shown in Fig. 1.

1.2. Related work

Attempts to automatically recognize sign language began to appear in the literature in the 90s. Research on hand gestures can be classified into two categories first category, relies on electromechanical devices that are used to measure the different gesture parameters such as the hand's position, angle, and the location of the fingertips. Systems that use such devices are usually called glove-based systems (e.g. the work of (Grimes, 1983) at AT&T Bell Labs developed the "Digital Data Entry Glove"). Major problems with such systems, that they force the singer to wear cumbersome and inconvenient devices. As a result, the way by which the user interacts with the system will be complicated and less natural.

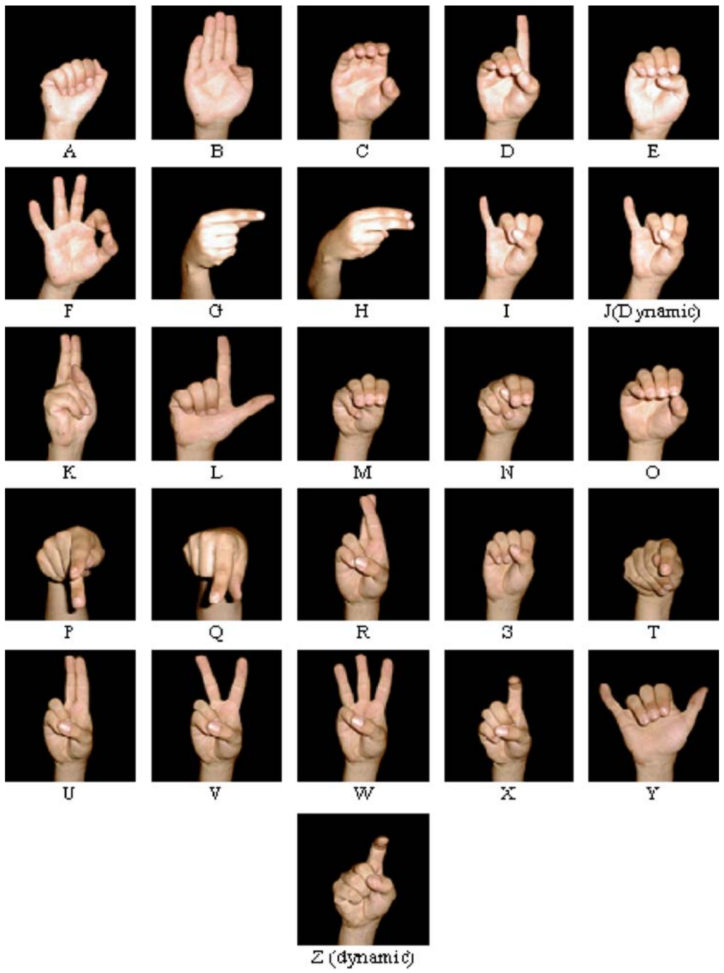


Fig. 1. The American sign language finger spelling alphabet.

The second category exploits machine vision and image processing techniques to create visual based hand gesture recognition systems. Visual-based gesture recognition systems are further divided into two categories. The first one relies on using specially designed gloves with visual markers called "visual-based gesture with glove–markers (VBGwGM)" that help in determining hand postures (Dorner & Hagen, 1994; Fels & Hinton, 1993; Starner, 1995). A summary of selected research efforts listed in Table 1.

But using gloves and markers do not provide the naturalness required in human–computer interaction systems. Besides, if colored gloves are used, the processing complexity is increased.

As an alternative, the second kind of visual based hand gesture recognition systems can be called "pure visual-based gesture (PVBG)" (i.e. visual-based gesture without glove–markers). This type tries to achieve the ultimate convenience naturalness by using images of bare hands to recognize gestures.

Among many factors, five important factors must be considered for the successful development of a vision-based solution to collecting data for hand posture and gesture recognition (Ong & Ranganath, 2005; Starner, 1995; Sturman & Zeltzer, 1994; Watson, 1993).

- The placement and number of cameras used.
- The visibility of the object (hand) to the camera for simpler extraction of hand data/features.
- The extraction of features from the stream or streams of raw image data.
- The ability of recognition algorithms to extracted features.
- The efficiency and effectiveness of the selected algorithms to provide maximize accuracy and robustness.

A number of recognition techniques are available and in some cases they can be applied for the two types of vision-based solutions (i.e. VBGwGM and PVBG). In general these recognition techniques can be categorized into three broad categories:

A. Feature extraction, statistics, and models.
This technique can be classified into six sub-categories:
1. Template matching (e.g. research work of Darrell & Pentland, 1993; Newby, 1993; Sturman, 1992; Watson, 1993; Zimmerman, Lanier, Blanchard, Bryson, & Harvill, 1987).
2. Feature extraction and analysis, (e.g. research work of Rubine, 1991; Sturman, 1992; Wexelblat, 1994, 1995).
3. Active shape models "smart snakes" (e.g. research work of Heap & Samaria, 1995).
4. Principal component analysis (e.g. research work of Birk, Moeslund, & Madsen, 1997; Martin & James, 1997; Takahashi & Kishino, 1991).
5. Linear fingertip models (e.g. research work of Davis & Shah, 1993; Rangarajan & Shah, 1991).
6. Causal analysis (e.g. research work of Brand & Irfan, 1995).

B. Learning algorithms.
This technique can be classified into three sub-categories:
1. Neural network (e.g. research work of Banarse, 1993; Fels, 1994; Fukushima, 1989; Murakami & Taguchi, 1991).
2. Hidden Markov Models (e.g. research work of Charniak, 1993; Liang & Ouhyoung, 1998; Nam & Wohn, 1996; Starner, 1995).
3. Instance-based learning (research work of Kadous, 1995; also see Aha, Dennis, & Marc, 1991).

C. Miscellaneous techniques.
This technique can be classified into three sub-categories:
1. The linguistic approach (e.g. research work of Hand, Sexton, & Mullan, 1994).
2. Appearance-based motion analysis (e.g. research work of Davis & Shah, 1993).
3. Spatio-temporal vector analysis (e.g. research work of Quek, 1994).

Regardless of the approach used (i.e. VBGwGM or PVBG etc.), many researchers have been trying to introduce hand gestures to Human–Computer Interaction field.

Table 1
A summary of gloves used

Research	Gloves used
Dorner and Hagen (1994)	They use a cotton glove, with various areas of it painted different colors to enable tracking (i.e. gloves with rings of colors around each joint)
Starner (1995)	Two colored gloves: an orange glove on the left hand and a yellow glove on the right hand
Fels and Hinton (1993) and Fels (1994)	VPL DataGlove Mark II with a Polhemus tracker as input devices; wearing a glove that the user moves in certain ways, users would learn to generate vocal sounds (Glove-Talk)
Takahashi and Kishino (1991)	VPL DataGlove
Murakami and Taguchi (1991)	VPL DataGlove
Kramer and Leifer (1990)	CyberGlove
Vamplew (1996)	A single CyberGlove with position tracking
Rung-Huei and Ouhyoung (1996)	DataGlove as input devices.
Kadous (1996)	Power gloves
Grobel and Assan (1996)	Colored gloves
Wexelblat (1994, 1995)	CyberGlove on each hand

Charayaphan and Marble (1992) investigated a way using image processing to understand American Sign Language (ASL). Their system can correctly recognize 27 out of the 31 ASL symbols. Fels and Hinton (1993) developed a system using a VPL DataGlove Mark II with a Polhemus tracker as input devices. In their system, the neural network method was employed for classifying hand gestures. Another system using neural networks developed by Banarse (1993) was vision-based and recognized hand postures using a neocognitron network, a neural network based on the spatial recognition system of the visual cortex of the brain. Heap and Samaria (1995) extend active shape models, or "smart snakes" technique to recognize hand postures and gestures using computer vision. In their system, they apply an active shape model and a point distribution model for tracking a human hand. Starner and Pentland (1995) used a view-based approach with a single camera to extract two-dimensional features as input to HMMs. The correct rate was 91% in recognizing the sentences comprised 40 signs. Kadous (1996) demonstrated a system based on power gloves to recognize a set of 95 isolated Auslan signs with 80% accuracy, with an emphasis on computationally inexpensive methods. Grobel and Assan (1996) used HMMs to recognize isolated signs with 91.3% accuracy out of a 262-sign vocabulary. They extracted the features from video recordings of signers wearing colored gloves. Vogler and Metaxas (1997) used computer vision methods to extract the three-dimensional parameters of a signer's arm motions, coupled the computer vision methods and HMMs to recognize continuous American sign language sentences with a vocabulary of 53 signs. They modeled context-dependent HMMs to alleviate the effects of movement epenthesis. An accuracy of 89.9% was observed. Yoshinori, Kang-Hyun, Nobutaka, and Yoshiaki (1998) used colored gloves and have shown that using solid colored gloves allows faster hand features extraction than simply wearing no gloves at all. Liang and Ouhyoung (1998), used HMMs for continuous recognition of Taiwan sign language with a vocabulary between 71 and 250 signs with DataGlove as input devices. However, their system required that gestures performed by the signer be slow to detect the word boundary. Yang and Ahuja (1999) investigated dynamic gestures recognition as they utilized skin colour detection and affine transforms of the skin regions in motion to detect the motion trajectory of ASL signs. Using a time delayed neural network, they recognised 40 ASL gestures with a success rate around 96%. But their technique potentially has a high computational cost when false skin regions are detected. A local feature extraction technique is employed to detect hand shapes in sign language recognition by Imagawa, Matsuo, Taniguchi, Arita, and Igi (2000). They used an appearance-based eigen method to detect hand shapes. Using a clustering technique, they generate clusters of hand shapes on an eigenspace with accuracy achieved a round 93% recognition of 160 words. Bowden and Sarhadi (2002) developed a non-linear model of shape and motion for tracking finger spelt American sign language. Their approach based on one-state transitions of the English Language which are projected into shape space for tracking and model prediction using an HMM like approach. Symeonidis (2000) used orientation histograms to recognize static hand gestures, specifically, a subset of American Sign Language (ASL). A pattern recognition system used a transform that converts an image into a feature vector, which will then be compared with the feature vectors of a training set of gestures. The system was implemented with a perceptron network. The main problem with this technique is how good differentiation one can achieve. This of course is dependent upon the images but it comes down to the algorithm as well. It may be enhanced using other image processing techniques like edge detection. For farther information and hot topics on this issue a modern and an excellent survey can be found in (Ong & Ranganath, 2005).

2. System design and implementation

2.1. System overview

Our system is designed to visually recognize all static signs of the American Sign Language (ASL), all signs of the ALS alphabets, single digit numbers used in ASL (e.g. 3, 5, 7) and a sample of words (e.g. love, meet, more) using bare hands. The users/signers are not required to wear any gloves or to use any devices to interact with the system. However, different signers vary their hand shape size, body size, operation habit and so on, which bring about more difficulties in recognition. Therefore, we realized the necessity to investigate the signer-independent sign language recognition to improve the system robustness and practicability in the future by using Hidden Markov Model (HMM) (Seymore, McCallum, & Rosenfeld, 1999). The combination of powerful Hough transformation with excellent image processing and neural networks capabilities has led to the successful development of ASL recognition system using MATLAB (Gonzalez et al., 2004). Our method relies on presenting the gesture as a feature vector that is translation, scale and rotation invariant.

The system has two phases: the feature extraction phase and the classification, as shown in Fig. 2. Images were prepared using Portable Document Format (PDF) form so the system will deal with images that have a uniform background (PDF Reference, 2004). The feature extraction applied an image processing technique which involves using algorithms to detect and isolate various desired portions of the digitized sign. During this phase, each colored image is resized and then converted from RGB to grayscale one. This is followed by an edge detection technique the so-called Canny edge detection (Canny, 1986).

The goal of edge detection is to mark the points in an image (sign image) at which the intensity changes sharply. Sharp changes in image properties usually reflect important events and changes in world properties. The Canny (Canny, 1986) operator was originally designed to be an

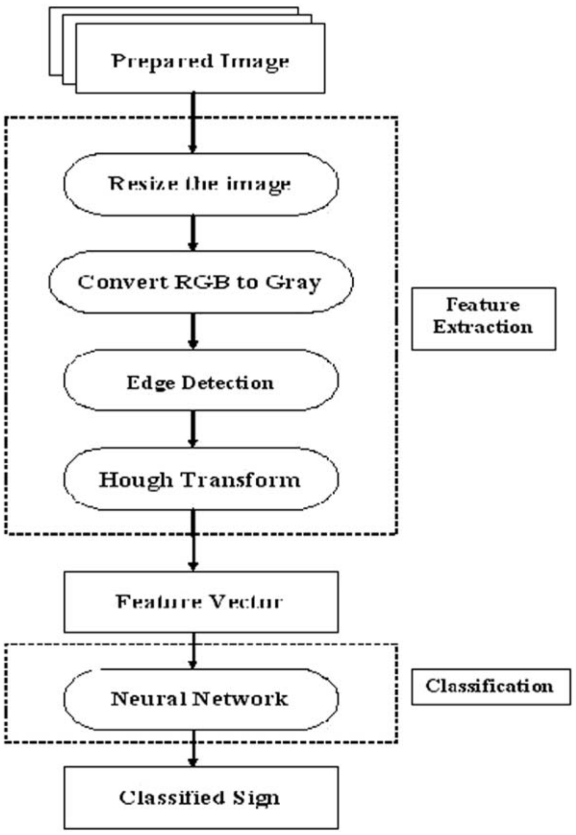


Fig. 2. System overview.

optimal edge detector (according to particular criteria—there are other detectors around such as Sobel and Roberts cross operators that also claim to be optimal with respect to slightly different criteria). In general most of these operators take as input a gray scale image, and produce as output an image showing the positions of tracked intensity discontinuities.

The next important step is the application of Hough transform (Hough, 1962). The Hough transform is a feature extraction technique used in digital image processing. The classical transform identifies lines in the image, but it has been extended to identifying positions of arbitrary shapes. The transform universally used today was invented by Richard Duda and Peter Hart in 1972 (Duda & Hart, 1972), who called it a "generalized Hough transform" after the related 1962 patent of Paul Hough.

In the classification stage, a 3-layer, feed-forward back propagation neural network (Haykin, 1999) is constructed. It consists of (160) inputs, (214 ∗ 3) neurons for the first hidden layer and (214 ∗ 2) neurons for the second hidden layer, and (214 ∗ 1) output neurons in this classification network.

2.2. Feature extraction phase

2.2.1. Resize the image

Images of signs were resized to *80 by 64*, by default, "imresize" uses nearest neighbor interpolation to determine the values of pixels in the output image but other interpolation methods can be specified. We use '*bicubic*' method because if the specified output size is smaller than the size of the input image, "imresize" applies a lowpass filter before interpolation to reduce aliasing. Therefore, we get the default filter size is 11-by-11.

2.2.2. Convert from RGB to Grayscale

To alleviate the problem of different lighting conditions of signs taken and the HSV color space "(Hue, Saturation, Value) also called HSB (Hue, Saturation, Brightness)" non-linearity by eliminating the hue and saturation information while retaining the luminance. The RGB color space (Red, Green and Blue which considered the primary colors of the visible light spectrum) is converted through gray scale image to a binary image.

Binary images are images whose pixels have only two possible intensity values. They are normally displayed as black and white. Numerically, the two values are often 0 for black, and either 1 or 255 for white. Binary images are often produced by thresholding a grayscale or color image, in order to separate an object in the image from the background. The color of the object (usually white) is referred to as the *foreground color*. The rest (usually black) is referred to as the *background color*. However, depending on the image which is to be thresholded, this *polarity* might be inverted, in which case the object is displayed with 0 and the background is with a non-zero value. This conversion resulted in sharp and clear details for the image (as shown in Fig. 3).

It is obvious, as shown in Fig. 3 that the RGB color space conversion to HSV color space then to a binary image produced images that lack many features of the sign.

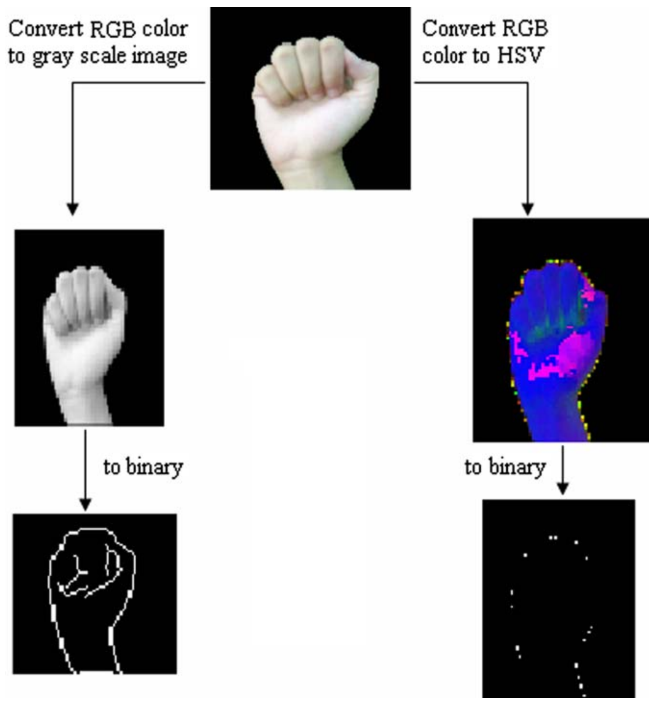


Fig. 3. Transformation of RGB color image into a binary image.

2.2.3. Edge detection

Hough transform is used to identify the parameters of a curve that best fits a set of given edge points. Edge detection is obtained from feature detector using Canny edge detector. We used Canny edge detection technique because it provides the optimal edge detection solution. Here in the proposed system an ‘optimal’ edge detector means:

- good detection—the algorithm should mark as many real edges in the image as possible,
- good localisation—edges marked should be as close as possible to the edge in the real image,
- minimal response—a given edge in the image should only be marked once, and where possible, image noise should not create false edges.

Canny edge detector results in a better edge detection compared to Sobel edge detector as shown in Table 2. The output of the edge detector defines ‘where’ features are in an image, whereas, Hough transform determines both ‘what’ the features are and ‘how many’ of them exists in the image.

As shown in Table 2 Sobel method was not appropriate method to use because it hides many important details needed to make a representative feature vector while Canny method is better, but in some cases it produces extra details more than needed. To solve this problem we decide to use a threshold of (0.25) after testing different threshold values and observing its results on the overall recognition system. Table 3 shows some samples of signs with threshold and other without.

Table 3
Canny edge detector with/without threshold for some signs

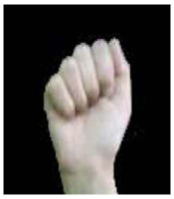
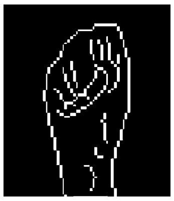
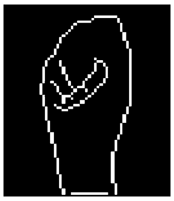
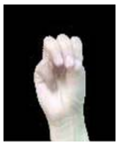
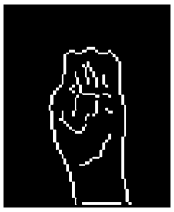
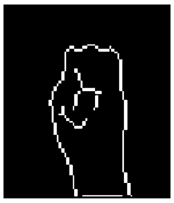
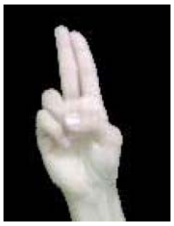
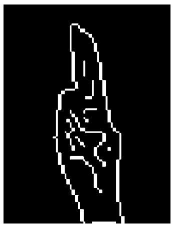
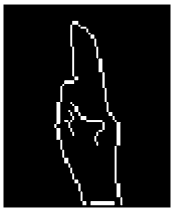
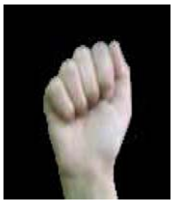
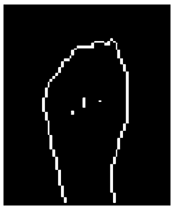
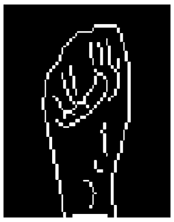
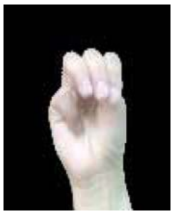
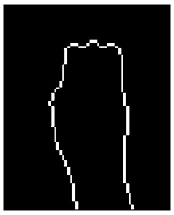
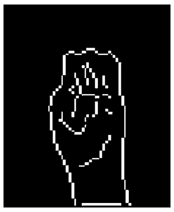
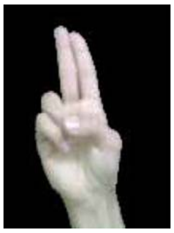
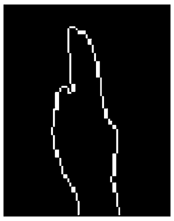
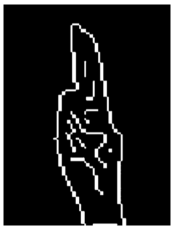
Sign	Original image	Canny without threshold	Canny with threshold (.25)
Sign ‘A’			
Sign ‘M’			
Sign ‘U’			

Table 2
Choosing appropriate edge detection method

Sign	Original image	Sobel	Canny
Sign ‘A’			
Sign ‘M’			
Sign ‘U’			

2.2.4. Hough transform

We have used radon function which represents an image as a collection of projections along various directions. We have applied radon on two different ranges of theta (θ), (0–360) and (-180 to 180) degrees. The range (0–360) led to misclassification in the classification phase. While the range (-180 to 180) was better, therefore, it has been selected.

Our algorithm is based on studying the distribution of variation for each line layers by computing the mean (average) and standard deviation along the range (-180 to 180) for each radius. The feature vector that results by applying this algorithm is called ‘Theta Radius Distribution Matrix’. This feature vector is the input for the classification phase.

The main advantage of using Hough transform technique is that it is tolerant to gaps in feature boundary descriptions and it is relatively unaffected by image noise.

For simplicity let us introduce *Hough transform* for straight lines which is in fact a special case of *Radon transform*. Consider a function $f(x,y)$ on a 2-D Euclidean plane is defined as

$$R(\rho,\theta)=\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}f(x,y)\delta(\rho-x\cos\theta-y\sin\theta)\,\mathrm{d}x\,\mathrm{d}y$$

where δ is the Dirac delta function. The δ term forces integration of f along the line specified by ρ and θ and, and is equivalent to the *Hough transform* for binary images.

For shapes other than straight lines the Radon transform is expressed by replacing the argument of δ by a function which forces integration of the image along the shape (that is, its boundary).

For the proposed system to consider sign language recognition, the following parameters for a generalized signs can be defined: $a = \{y, s, \theta\}$ where $y = (x_r, y_r)$ is a *reference origin* for a shape in the image space, θ is shape *orientation*, and $s = (s_x, s_y)$ describes two orthogonal *scale factors*, along x- and y-axis respectively. For the sake of simplicity we shall discuss the specific (but significant) case where $\boldsymbol{s}$ is a scalar, and the parameter space has four dimensions.

For simplicity, consider the sign "O" taken from a side angle (shown in Fig. 4) which represents a circle with fixed radius r_0; since the radius is fixed, we do not need the scale parameter, and due to the symmetry the orientation parameter is redundant as well, so the accumulator is congruent to the image.

But this is not the case if to consider an arbitrary shape recognition as shown in Fig. 5. Let the tangent line at the point x be directed by angle ϕ (note that this angle is measured in respect to some fixed direction in the image space); we agree on taking the angle in the range $(0, \pi]$ and to some precision due to our computational constraints. Let y be a point chosen arbitrarily as the reference point for the shape—it is convenient, though not necessary, to choose a point close to the centroid of the shape—and the vector $r = y - x$ displacement vector.

Note that the shape boundary may contain other points at which the gradient (tangent) has the same direction—e.g., x' as shown in Fig. 5, and the corresponding displacement vector is different $r' \neq r$. We shall store the displacement vectors as function mapping each possible ϕ to a set of displacement vectors. The resulting data structure is called the "*Theta Radius Distribution Matrix*". Its format is illustrated by the table below, where y stands for the reference point chosen for the shape, B for shape boundary, $\phi(x)$ denotes the gradient (tangent) direction at x.

i	ϕ_i	$R_{\phi i}$
1	$\Delta\phi$	$\{r \mid y - r = x,\ x \in B,\ \phi(x) = \Delta\phi\}$
2	$2\Delta\phi$	$\{r \mid y - r = x,\ x \in B,\ \phi(x) = 2\Delta\phi\}$
$\vdots$	$\vdots$	$\vdots$
$\pi/\Delta\phi$	π	$\{r \mid y - r = x,\ x \in B,\ \phi(x) = \pi\}$

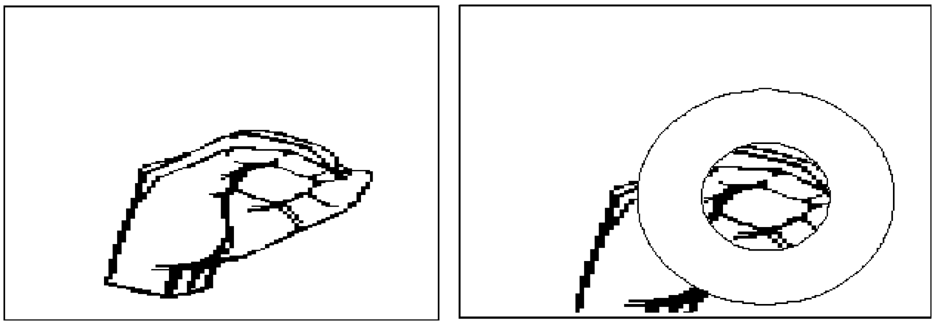


Fig. 4. Simplification of "O" sign.

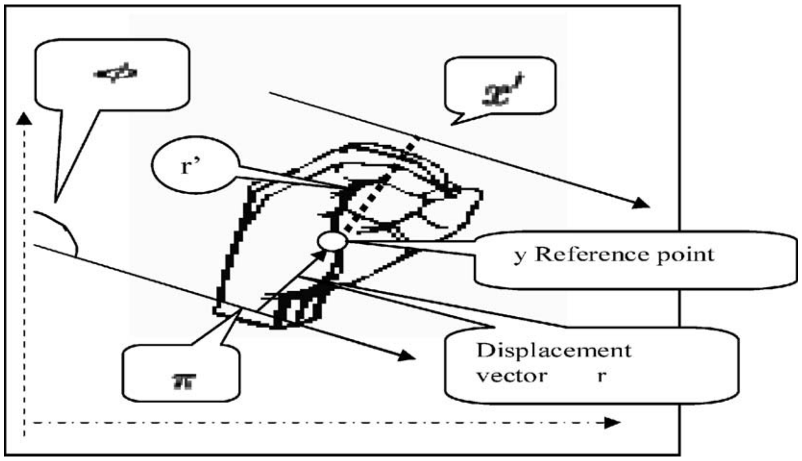


Fig. 5. Coordination in image space.

One may choose to measure directions rather from $-\frac{\pi}{2}$ to $\frac{\pi}{2}$ to, or in any other range of the same magnitude—the table rows will be just cyclically shifted.

The reference point inside the image sign can be chosen according to the following heuristics:

$$y = (y_1, y_2) = \frac{1}{|B|}\left(\sum_B x_1, \sum_B x_2\right)$$

that is, the reference is made to the mean of the pattern points. This will keep r relatively small (not much larger than necessary), consequently reducing the error.

2.3. Classification phase

The classification neural network is shown in Fig. 6, the neural network has (200) instances as its input vector, and 214 output neurons in the output layer.

2.3.1. Network architecture

When working with neural networks it is hard to predict what the result will be. Sometimes practice represents the best solution. Decision in this field is very difficult; you have to examine different architectures and decide according to their results.

Therefore, after several experiments, it has been decided that the proposed system should based on supervised learning in which the learning rule is provided with the set of examples (the training set).

2.3.2. Target architecture

The size of our target matrix is 214 by 200. Fig. 7 depicted a part of the target where each row specifies the class to which an instance belongs to.

2.3.3. Creating the network

Creating a network object is accomplished by training a feed-forward back propagation network, using the MATLAB built-in function (newff). It requires four input and returns the network object, the first input is $(R, 2)$ matrix of minimum and maximum values for each of the R elements of the input vector. The second input is an array containing the sizes of each layer; the third input is a cell array containing the names of the transfer functions used in each layer. The final input contains the names of the training functions used. The following command was used to create a three-layer network:

$$\text{net1} = \text{newff}(\text{input\_range}, [214 * 3\ 214 * 2\ 214], \{\text{'logsig' 'logsig' 'logsig'}\}, \text{'traingdx'})$$

There are $(214 * 3)$ neurons in the first hidden layer, $(214 * 2)$ neurons in the second hidden layer, and $(214 * 1)$ neurons in the output layer. For the three-layer Log-Sigmoid ('logsig') transfer function was used.

After executing the above command, a network object is created, weights and biases of the network are initialized and the network is ready for training.

2.3.4. Training the network

The training process starts by a set of examples of proper network behavior-network inputs and target outputs. During training, the weight and biases of the network are iteratively adjusted to minimize the network performance (net.performfcn). The default performance function, for feed-forward networks, is the mean square error (MSE), which is defined as the average squared error between the networks outputs and target outputs.

The training function that was used is *traingdx*. It works accurately with noise patterns in training, and they increase the network accuracy on unseen samples. That is why we

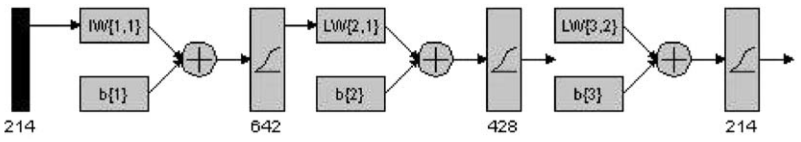
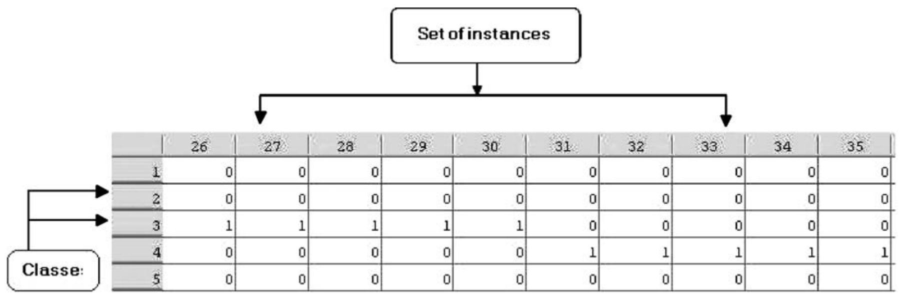


Fig. 6. Classification network.


Set of instances
Classes

	26	27	28	29	30	31	32	33	34	35
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	0	0	0	0	0
4	0	0	0	0	0	1	1	1	1	1
5	0	0	0	0	0	0	0	0	0	0

Fig. 7. Part of target matrix for the training set.

choose it in the final implementation. The number of epochs was 10,000 and the goal was 0.0001.

2.3.5. Testing and simulating the network

The system was tested with (100) images, (five images for each sign) untrained images; previously unseen for the testing phase. The MATLAB built-in function (sim) simulates a network. The behavior of (sim) takes the network input, and the network object, then returning the network outputs. More than one network was trained and simulated.

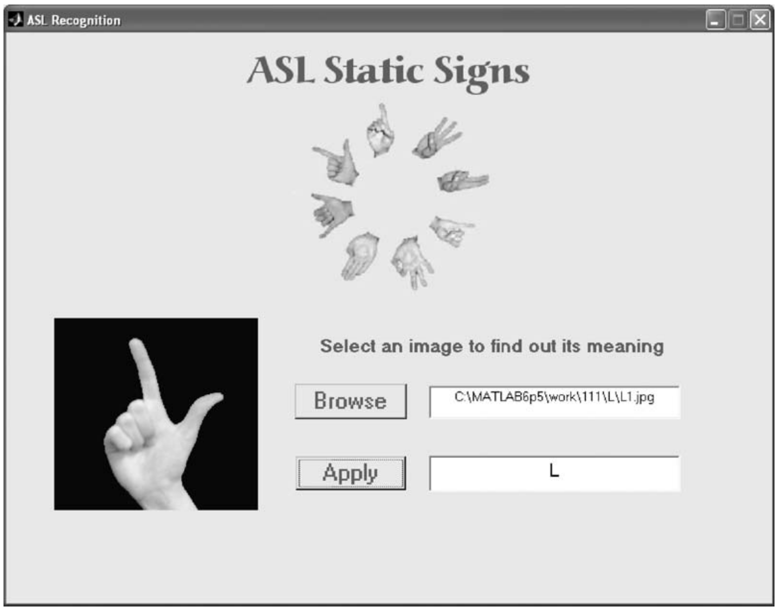
For user convenience and simplicity, we have created a GUI that helps in finding out the results. An example is shown in Figs. 8 and 9 respectively.

3. Experiments results and analysis

In this section, we evaluate the performance of our recognition system by testing its ability to classify signs for both training and testing set of data. The effect of the number of inputs to the neural network is considered. In addition we discuss some problems in the performance of some signs due to the similarities between them.

3.1. Data set

The data set used for training and testing the recognition system consists of grayscale images for all of the 20 signs

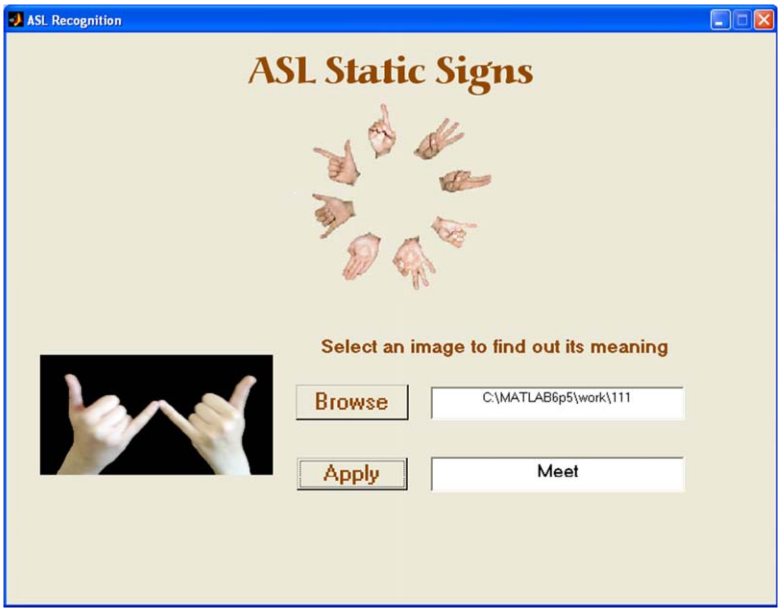


Select an image to find out its meaning

C:\MATLAB6p5\work\111\L\L1.jpg

L

Fig. 8. Example on GUI simulating sign ‘L’.



Select an image to find out its meaning

C:\MATLAB6p5\work\111

Meet

Fig. 9. Example on GUI simulating sign ‘meet’.

used in the experiment, these 20 signs are shown in Fig. 10. Also 15 samples for each sign were taken from 15 different volunteers. For each sign, 10 out of 15 samples were used for training purpose, while the remaining five signs were used for testing. The samples were taken from different distances by digital camera, and with different orientations.

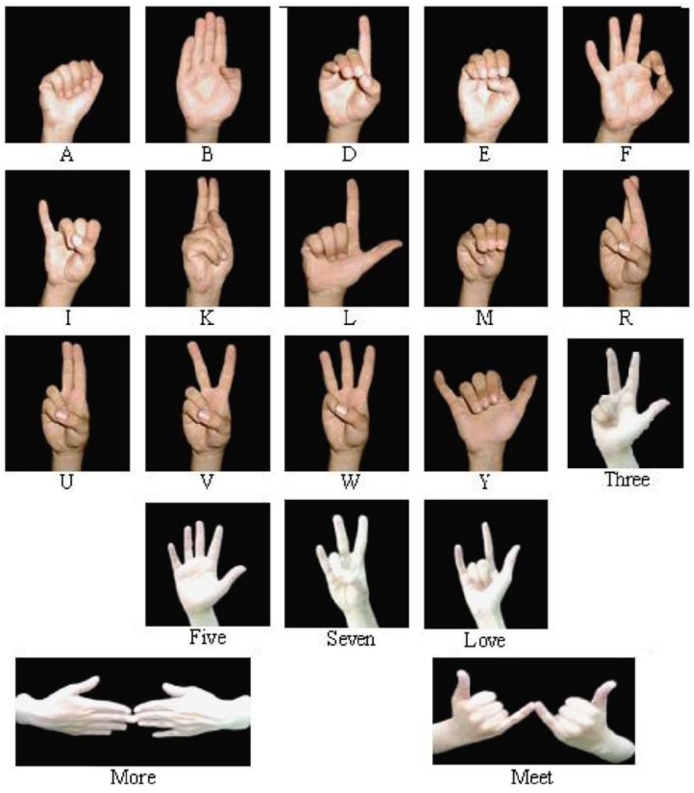


Fig. 10. ASL signs used in the proposed system.

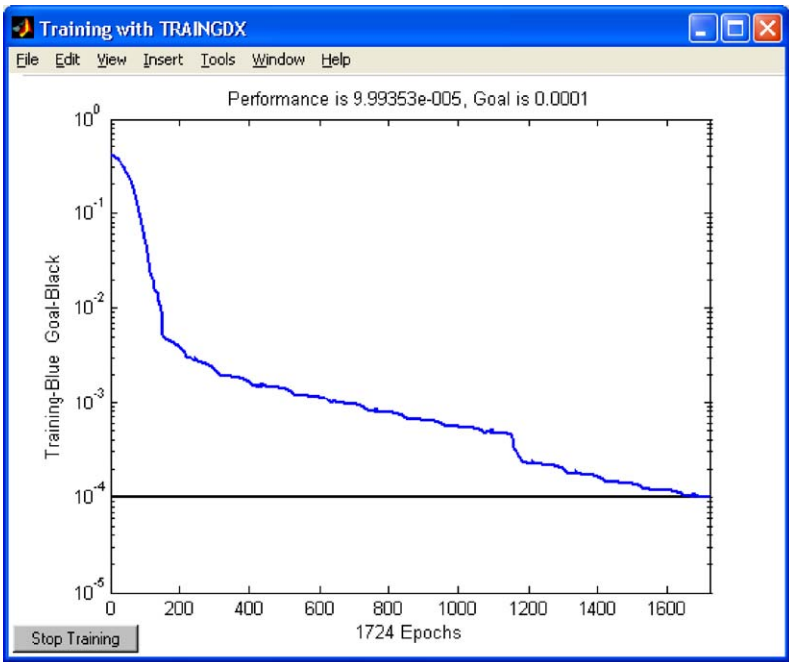


Fig. 11. Training chart for six samples for each sign.

This way, we were able to obtain a data set with cases that have different sizes and orientations, so we can examine the capabilities of our feature extraction scheme.

3.2. Recognition rate

We evaluate the performance of the system based on its ability to correctly classify samples to their corresponding classes. The recognition rate is defined as the ratio of the number of correctly classified signs to the total number of samples, i.e.

$$\text{Recognition rate} = \frac{\text{number of correctly classified signs}}{\text{Total number of signs}} * 100\%$$

Through the experiment of the proposed system, firstly we trained our system on six samples for each sign and with no threshold for Canny edge detector. The training chart for this network is shown in Fig. 11. The testing results, for both training and testing data are shown in Table 4.

Although the overall system has a good performance it cannot be guaranteed because the performance (recognition rate) of the tested data is very low (51%) and the number of training data is too small that does not have different reasonable orientations.

In order to obtain a more satisfactory result, we trained the network on eight samples for each sign and with (0.15) threshold for Canny edge detector. The training chart for this network is shown in Fig. 12. The testing results for both training and testing data are shown in Table 5.

As shown, the results above were much better than the previous. Another experiment which was the most satisfactory, was the trained network on 10 samples for each sign and with (0.25) threshold for Canny edge detector. The training chart for this network is shown in Fig. 13. The testing results for both training and testing data are shown in Table 6. However, the details results for the last network are shown in Table 7.

3.3. Hardware and software

The system was implemented in MATLAB version 6.5. The recognition training and tests were run on a modern

Table 4
Results of training six samples for each sign and without Canny threshold

Data	No. of samples	Recognized samples	Recognition rate (%)
Training	120	120	100.0
Testing	100	51	51.00
Total	220	171	77.72

Table 5
Results of training eight samples for each sign and with (0.15) Canny threshold

Data	No. of samples	Recognized samples	Recognition rate (%)
Training	160	158	98.75
Testing	100	80	80.00
Total	260	238	91.53

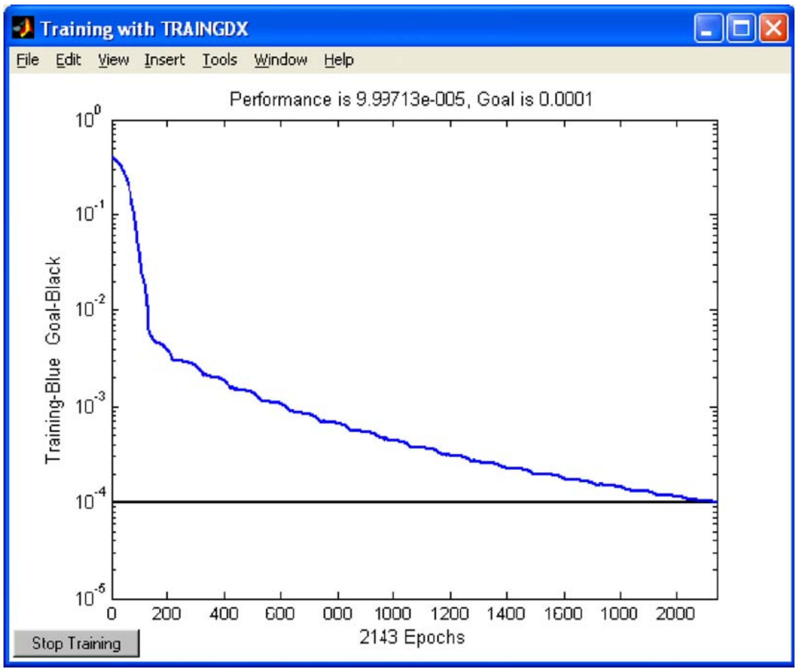


Fig. 12. Training chart for a network trained on eight samples for each sign and with (0.15) Canny threshold.

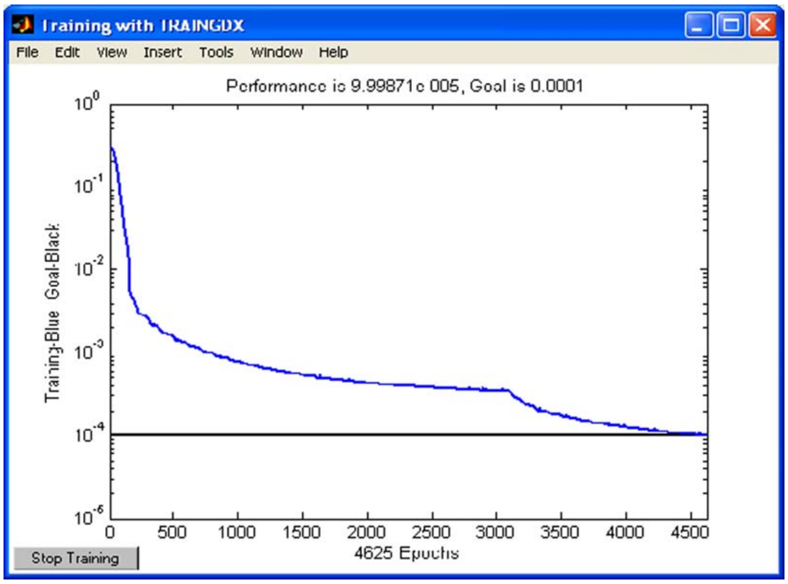


Fig. 13. Training chart for a network trained on 10 samples for each sign and with (0.25) Canny threshold.

Table 6
Results of training 10 samples for each sign and with (0.25) Canny threshold

Data	No. of samples	Recognized samples	Recognition rate (%)
Training	200	197	98.50
Testing	100	80	80.00
Total	300	277	92.33

Table 7
Results obtained when training 10 samples for each size with a Canny threshold of 0.25

Sign	Recognized samples	Misclassified samples	Recognition rate (%)
A	15	0	100
B	14	1	93.3
D	15	0	100
E	14	1	93.3
F	15	0	100
I	12	3	80.0
K	15	0	100
L	13	2	86.7
M	14	1	93.3
R	13	2	86.7
U	13	2	86.7
V	14	1	93.3
W	14	1	93.3
Y	11	4	73.3
3	15	0	100
5	15	0	100
7	13	2	86.7
Love	12	3	80.0
More	15	0	100
Meet	15	0	100
Total	277	23	92.33

standard PC (1.8 GHz AMD processor, 128 MB of RAM) running under Windows 2000.

4. Conclusions and future work

In this project, we developed a system for the purpose of the recognition of a subset of the American sign language. The system has two phases: the feature extraction phase and the classification phase. The work was accomplished by training a set of input data (feature vectors). Without the need of any gloves, an image for the sign is taken by a camera. After processing, feature extracting phase depends on Hough transformation which is tolerant to gaps in feature boundary descriptions and it is relatively unaffected by image noise. These vectors are fed to the neural network.

The proposed system proved to be robust against changes in gestures, position, size and direction. This is because the extracted features method used proved to be translation, scale, and rotation invariant. The proposed system was able to reach a recognition rate of about 98.5% for training data and 80% for testing data.

4.1. Future work

- The work presented in this project dealt with static signs of ASL only. Extending the system to be able to deal with dynamic signs is an attractive point for future work.
- The system deals with images that have a uniform background. Getting rid of this limitation will make the system more flexible and suitable for real life applications—as there is no control on the environment (i.e. background).
- It is important to consider increasing the data size, so we can have more accurate and highly performance system.
- Training the network to other types of images.

- Looking for possible changes in the environment by designing a new system that works in real-time environment.

References

Aha, D. W., Dennis, K., & Marc, A. (1991). Instance-based learning algorithms. *Machine Learning, 6*, 37–66.

Banarse, D. S. (1993). Hand posture recognition with the neocognitron network. Master's thesis, School of Electronic Engineering and Computer Systems, University College of North Wales, Bangor.

Birk, H., Moeslund, T. B., & Madsen, C. B. (1997). Real-time recognition of hand alphabet gestures using principal component analysis. In Proceedings of the 10th Scandinavian conference on image analysis.

Bowden, R., & Sarhadi, M. (2002). A non-linear model of shape and motion for tracking finger spelt American sign language. *Image and Vision Computing, IVC(20)*(9–10), 597–607.

Brand, M., & Irfan, E. (1995). Causal analysis for visual gesture understanding, MIT Media Laboratory Perceptual Computing Section Technical Report No. 327.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 8*(6), 1986, November.

Charayaphan, C., & Marble, A. (1992). Image processing system for interpreting motion in American sign language. *Journal of Biomedical Engineering, 14*, 419–425.

Charniak, E. (1993). *Statistical language learning*. Cambridge: MIT Press.

Cutler, R., & Turk, M. (1998). View-based interpretation of real-time optical flow for gesture recognition. IEEE International Conference on Automatic Face and Gesture Recognition.

Darrell, T., & Pentland, A. (1993). Recognition of space–time gestures using a distributed representation. MIT Media Laboratory Vision and Modeling Group Technical Report No. 197.

Davis J., & Shah M. (1993). Gesture recognition, Technical Report, Department of Computer Science, University of Central Florida, CS-TR-93-11.

Dorner, B., & Hagen, E. (1994). Towards an American sign language interface. *Artificial Intelligence Review, 8*(2–3), 235–253.

Duda, R. O., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM, 15*, 11–15.

Fels, S. (1994). Glove-TalkII: mapping hand gestures to speech using neural networks—an approach to building adaptive interfaces. PhD thesis, Computer Science Department, University of Toronto.

Fels, S., & Hinton, G. (1993). GloveTalk: a neural network interface between a DataGlove and a speech synthesizer. *IEEE Transactions on Neural Networks, 4*, 2–8.

Fukushima, K. (1989). Analysis of the process of visual pattern recognition by the neocognitron. *Neural Networks, 2*, 413–420.

Gonzalez, R. C., Woods, R. E., & Eddins, S. L. (2004). *Digital image processing using MATLAB*. Prentice Hall.

Grimes, G. (1983). Digital data entry glove interface device, Patent 4,414,537, AT & T Bell Labs.

Grobel, K., & Assan, M. (1996). Isolated sign language recognition using hidden Markov models. In Proceedings of the international conference of system, man and cybernetics, pp. 162–167.

Hand, C., Sexton, I., & Mullan, M. (1994). A linguistic approach to the recognition of hand gestures. In Proceedings of the designing future interaction conference, University of Warwick, UK.

Haykin, S. (1999). *Neural networks: a comprehensive foundation*. Prentice Hall.

Heap, A. J., & Samaria, F. (1995). Real-time hand tracking and gesture recognition using smart snakes. In Proceedings of interface to real and virtual worlds, Montpellier.

Hong, P. et al. (2000). Gesture modeling and recognition using finite state machines. IEEE international conference on automatic face and gesture recognition, pp. 410–415.

Hough, P. (1962). Method and means for recognizing complex patterns, US Patent (3,069,654).

Imagawa, K., Matsuo, H., Taniguchi, R., Arita, D., & Igi, S. (2000). Recognition of local features for camera-based sign language recognition system. In International conference on pattern recognition (ICPR), pp. 4849–4853.

International Bibliography of Sign Language, (2005). http://www.sign-lang.uni-hamburg.de/bibweb/F-Journals.html.

International Journal of Language & Communication Disorders, (2005). Available from http://www.newcastle.edu.au/renwick/ROL/Jnlcontents/000mgmgf.htm.

Kadous, W. (1995). GRASP: recognition of Australian sign language using instrumented gloves. Bachelor's thesis, University of New South Wales.

Kadous, W. (1996). Machine recognition of Auslan signs using PowerGlove: towards large-lexicon recognition of sign language. In Proceeding of the workshop on the integration of gesture in language and speech, Wilmington, DE, pp. 165–174.

Kramer, J., & Leifer, L. (1990). A "Talking Glove" for nonverbal deaf individuals. Technical Report CDR TR 1990 0312, Centre For Design Research, Stanford University.

Liang, R. H., & Ouhyoung, M. (1998). A real-time continuous gesture recognition system for sign language. In Proceeding of the third international conference on automatic face and gesture recognition, Nara, Japan, pp. 558–565.

Martin, J., & James L. C. (1997). An appearance-based approach to gesture recognition. In Proceedings of the ninth international conference on image analysis and processing, pp. 340–347.

Murakami, K., & Taguchi, H. (1991). Gesture recognition using recurrent neural networks. In Proceedings of CHI'91 human factors in computing systems, pp. 237–242.

Nam, Y., & Wohn, K. Y. (1996). Recognition of space–time hand-gestures using hidden Markov model. In *Proceedings of the ACM symposium on virtual reality software and technology '96* (pp. 51–58). ACM Press.

National Institute on Deafness and Other Communication Disorders, (2005). Available from: http://www.nidcd.nih.gov/health/hearing/asl.asp.

Newby, G. (1993). Gesture recognition using statistical similarity. In Proceedings of virtual reality and persons with disabilities.

Ong, S., & Ranganath, S. (2005). Automatic sign language analysis: a survey and the future beyond lexical meaning. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 27*(6).

PDF Reference, (2004). Addison-Wesley, 5th ed.

Quek, Francis (1994). Toward a vision-based gesture interface. In *Proceedings of the ACM symposium on virtual reality software and technology '94* (pp. 17–31). ACM Press.

Rangarajan, K., & Shah, M. (1991). Establishing motion correspondence. *CVGIP: Image Understanding, 54*, 56–73.

Rubine, D. (1991). Specifying gestures by example. In *Proceedings of SIGGRAPH'91* (pp. 329–337). ACM Press.

Rung-Huei, L., & Ouhyoung, M. (1996). A sign language recognition system using hidden Markov model and context sensitive search. In *Proceedings of the ACM symposium on virtual reality software and technology '96* (pp. 59–66). ACM Press.

Seymore, K., McCallum, A., & Rosenfeld, R. (1999). Learning hidden Markov model structure for information extraction. AAAI 99 workshop on machine learning for information extraction.

Starner, T. (1995). Visual recognition of American sign language using hidden Markov models. Master's thesis, Massachusetts Institute of Technology.

Starner, T., & Pentland, A. (1995). Visual recognition of American sign language using hidden Markov models. International Workshop on Automatic Face and Gesture Recognition, Zurich, Switzerland, pp. 189–194.

Sturman, D. (1992). Whole-hand Input. Ph.D dissertation, Massachusetts Institute of Technology.

Sturman, David, & Zeltzer, David (1994). A Survey of glove-based Input. *IEEE Computer Graphics and Applications, 14*(1), 30–39.

Symeonidis, K. (2000). Hand gesture recognition using neural networks. Master's thesis, University of Surrey.

Takahashi, T., & Kishino, F. (1991). Hand gesture coding based on experiments using a hand gesture interface device. *SIGCHI Bulletin, 23*(2), 67–73.

Vamplew, P. (1996). Recognition of sign language using neural networks. PhD Thesis, Department of Computer Science, University of Tasmania.

Vogler, C., & Metaxas, D. (1997). Adapting hidden Markov models for ASL recognition by using three-dimensional computer vision methods. In Proceedings of the IEEE international conference on systems, man and cybernetics, Orlando, pp. 156–161.

Watson, R. (1993). A survey of gesture recognition techniques. Technical Report TCD-CS-93-11, Department of Computer Science, Trinity College Dublin.

Wexelblat, A. (1994). A feature-based approach to continuous-gesture analysis. Master's thesis, Massachusetts Institute of Technology.

Wexelblat, A. (1995). An approach to natural gesture in virtual environments. *ACM Transactions on Computer–Human Interaction, 2*(3), 179–200.

Yang, M., & Ahuja, N. (1999). Recognizing hand gestures using motion trajectories. In IEEE international conference on computer vision and pattern recognition (CVPR), pp. 466–472.

Yoshinori, K., Tomoyuki I., Kang-Hyun, J., Nobutaka, S., & Yoshiaki, S. (1998). Vision-based human interface system: selectively recognizing intentional hand gestures. In Proceedings of the IASTED international conference on computer graphics and imaging, pp. 219–223.

Zimmerman, T., Lanier, J., Blanchard, C., Bryson, S., & Harvill, Y. (1987). A hand gesture interface device. In *Proceedings of CHI + GI'87 human factors in computing systems and graphics interface* (pp. 189–192). ACM Press.