VHDL IMPLEMENTATION OF AMERICAN HAND SIGN RECOGNITION SYSTEM USING RADIAL BASIS FUNCTION NEURAL NETWORK

Bagyalakshmi.1, (Mrs.) Dr.R.Brindha2

1(PG Scholar, VLSI Design, Dept of ECE, Avinashilingam Institute for Home Science and Higher Education For Women-University, Coimbatore, India, bagyayayaraj@gmail.com)
2(Prof and Head, Dept of ECE, Avinashilingam Institute for Home Science and Higher Education For Women-University, Coimbatore, India, brin1kalai@yahoo.co.in)

Abstract- This paper presents hand sign recognition using radial basis function algorithm implemented in VHDL. Radial Basis Function Neural Network has a simple structure and a fast training process. The hand sign recognition system has 3 steps: 1) Image pre-processing, 2) Feature extraction and 3) Classification. An image is pre-processed and converted to feature vector which will be compared with the training set image feature vectors. The system was designed to recognize 24 American sign languages hand signs. The proposed ANN leads to 100% recognitionrate for the training set. This hand posture recognition system is coded using VHDL and Matlab, simulated using ModelSim 10.0b and Matlab R2012.

Keywords— Hand posture recognition, Radial Basis Function Neural Network (RBFNN), Self organizing Map (SOM, American Sign Language (ASL)).

1. INTRODUCTION

The quality of our life is improved by the smart environments. Hand posture recognition provides an attractive method for human to computer interaction (HCI) in many applications. The hand gestures will enable human to interface with the computer without any mechanical devices. Hand gestures are classified as either hand postures or dynamic hand gestures. Hand postures are static hand poses without any movements. Hand gestures are dynamic movements, which are a sequence of hand postures. Hand gesture recognition system can be applied to different domains, they are sign language translation, virtual reality, smart environments, robot control, medical systems etc.,

The primary goal of hand posture recognition is to create a system which can identify specific hand postures. This project will focus on recognizing 24 hand signs of American Sign Language. ASL is the language used by auditory handicapped people in the United States of America. ASL consists of nearly 6000 gestures of common words and it must be recognized by the finger spelling used to communicate with others. One hand is used for finger spelling and 26 gestures are used to communicate the 26 letters of the alphabet. Among the 26 letters 2 of them are dynamic gestures and this project is aimed to recognize static hand gesture of 24 letters

The existing system is based on a hybrid network proposed in [1] and feature vectors proposed in [2]. Hand posture model in [2] uses feature vectors that are the frequency spectra of horizontal and vertical projection histograms of hand sign images. The proposed system is based on Radial Basis Neural Network proposed in [3]. One of the most effective vector classifiers is based on self-organizing maps (SOMs) [4]. The sensor devices are used glove-based methods to detect hand and finger motions into useful data [5].

The paper by V. Bonato [9] proposed a real time gesture recognition system for mobile robots. In a mobile robot vision system is embedded and used for recognition. A popular feature extraction algorithm is scale-invariant feature transform (SIFT). F. C. Huang has proposed [10] proposed an all-hardware SIFT accelerator-the fastest of its kind knowledge. It has two interactive hardware components, the first one is the key point identification, and the second is the feature descriptor generation. Gamage.N [6] presented a static hand sign recognition in Malaysian sign language using linear projection methods.

A number of approaches to the video-based recognition of hand gestures have been introduced in [7] and [8]. Methods of vision-based recognition of hand gestures fall into two categories, i.e., 3-D-model based and appearance-model based. Vision Based Hand Gesture Recognition is proposed by P. Garg [8] the current approach of Human to computer interaction with keyboard, mouse and pen are not enough for the development of ubiquitous computing. Many approaches have used 3-D-hand-model-based techniques [11]–[13] that depend on 3-D kinematic hand models.

The remaining chapters of this paper are organized as follows. Section 2 reviews existing systems of hand sign recognition. In Section 3 proposed methodology is discussed, the simulation results are discussed in Section 4 and followed by the conclusion in Section 5.

2. EXISTING METHODOLOGY

The process flow of hand posture recognition is shown in Fig.1.

A. Input Image

In the existing method, the hand gestures of American Sign Language alphabets are chosen for recognition and Fig. 2 shows input images representing hand
signs of alphabets in English. The input image is pre-processed to generate a feature vector which consists of \( P \times P \) pixels in RGB color format.

![Flow diagram of hand posture recognition](image)

**B. Pre-processing**

As per [1] process of hand sign recognition is shown in Fig.3. The input image is first resized into 256 x 256 pixels. The images are prepared and were saved in .bmp format. The pre-processing of an image has following steps.

a) Binary Quantization

b) Histogram calculation for horizontal and vertical projection

c) Discrete Fourier Transform

**C. Binary Quantization**

The hand region is segmented by detecting the red region, and each input color pixel is quantized to a binary value. The threshold value for the binary quantization is zero. The binary quantization of the image is done with

\[
I(x, y) = g(\text{Red}(x, y), \text{Green}(x, y) + \text{Blue}(x, y)) \cdot g(\text{Red}(x, y), \rho)
\]

where \( I(x, y) \) is the binary pixel value, and \( \text{Red}(x, y), \text{Green}(x, y), \) and \( \text{Blue}(x, y) \) are the color component values of a pixel at the \( (x, y) \) coordinates. The \( \rho \) is a threshold parameter and \( g(.) \) is a threshold function

\[
g(x, \rho) = \begin{cases} 
1 & \text{if } x \geq \rho \\
0 & \text{otherwise}.
\end{cases}
\]

**D. Histogram Calculation for Horizontal and Vertical Projection**

Binary values are used to calculate the two histograms, \( PH(y) \) and \( PV(x) \). The horizontal and vertical projection histograms of \( I(x, y) \) are calculated in the next preprocessing sub module. The histogram value is the sum of pixel values along a particular direction. Horizontal projection histogram \( PH(y) \) and vertical projection histogram \( PV(x) \) are defined by

\[
PH(y) = \sum_{x=0}^{P-1} I(x, y)
\]

\[
PV(x) = \sum_{y=0}^{P-1} I(x, y).
\]

**E. Discrete Fourier Transform**

\( PH(y) \) and \( PV(x) \) are sequentially fed to the DFT circuits and two DFTs are activated to compute the magnitude spectra \( FH(n) \) and \( FV(n) \) of \( PH(y) \) and \( PV(x) \), at the final stage of the pre-processing. \( FH(n) \) and \( FV(n) \) are calculated sequentially by two DFT circuits given by

\[
A(k) = \sum_{n=0}^{P-1} x(n) . \cos(2\pi nk/p)
\]

\[
B(k) = \sum_{n=0}^{P-1} x(n) . \sin(2\pi nk/p).
\]

The magnitude spectrum is given by

\[
X(k) = \sqrt{A(k)^2 + B(k)^2}.
\]
F. Feature Extraction

The DFT result is used as the feature vector and fed to the classifier network. Each input vector element $\xi_i$ to the classifier network is defined as

$$\xi_i = \begin{cases} \mathcal{F}(i) & 0 \leq i < D/2 \\ \mathcal{F}(i - D/2) & D/2 \leq i < D \end{cases}$$

As shown in Fig. 4, the magnitude spectra $F_H(n)$ and $F_V(n)$ of the same hand posture images placed in different locations are identical because they lack phase information for the hand posture's position. Therefore, the use of a magnitude spectrum as a feature vector makes the recognition robust to the change in position of input hand postures.

![Image](image1.png)
![Image](image2.png)

![Image](image3.png)
![Image](image4.png)

![Image](image5.png)

G. Classification

The existing system classification is implemented using ANN algorithm. The hybrid network consists of Self organizing map (SOM) and a single layer feed forward neural network that is trained with the Hebbian learning algorithm and it is shown in Fig 5.

1. Self Organizing Map (SOM)
2. Hebbian Learning Network

1. Self Organizing Map (SOM)

SOM is an unsupervised learning algorithm to form a nonlinear mapping from a given high-dimensional input space to a lower-dimensional map of neurons. The hybrid model consists of an unsupervised, SOM for stimulus calculation and a supervised network for category acquisition and naming and it is shown in Fig 5. The classifier reads the D-dimensional vectors from pre-processing, and it classifies them into H classes.

The SOM consists of $K = M \times M$ neurons, each of which contains a D-dimensional vector $\mathbf{m}_i$ called the weight vector

$$\mathbf{m}_i = [\mu_0, \mu_1, \ldots, \mu_{D-1}] \in \mathbb{R}^D$$

The SOM operation is divided into two phases. In the initial learning phase weight map is trained with a set of input vectors. That is in the training phase dataset consist of 24 static hand postures and feature vectors of each image are extracted using the DFT and the result is stored in a file. This file can be used during the classification step. After that, weights of the map retain unchanged and the map is used in the recall phase. During the learning phase, input vectors $\xi_0 \ldots \xi_{D-1}$ are given to the SOM in multiple iterations.

![Image](image6.png)

The distances to all weight vectors are calculated for each input vector, and then a winner neuron is determined. The winner neuron weight vector has the shortest distance to the input vector. The Manhattan distance $d_i$ is given by

$$d_i = \sum_{j=0}^{D-1} |\xi_j - \mu_{ij}|$$

After the winner neuron is determined, vectors of the neuron and its neighborhood neurons are updated with the following equation so that they are closer to the input vector

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \mathbf{h}_{ij} \cdot d_i.$$  

Neighborhood function $h_{ij}$ is defined as

$$h_{ij} = \sigma(t) \exp\left(-\frac{||r_i - r_j||^2}{2\sigma^2(t)}\right)$$
Where \( r_c \) and \( r_i \) corresponds to the location vectors of the winner neuron-\( c \) and neuron-\( i \). The neighborhood function provides a topology-preserving nature, i.e., two vectors that are neighbours in the input space will also be represented close to each other on the map. The topology-preserving nature is one of the most important features of SOM.

2. Hebbian Learning Network

Winner neuron information is taken from the SOM; the Hebbian learning network performs the category acquisition. Neurons representing a particular class are selected by the Hebbian learning rule and they are grouped together. The vector’s class is indicated by the teaching signals \( t_0, t_1, \ldots, t_{H-1} \) and it is fed to the network. Then appropriate winning neurons and the teaching signals are selected by the Hebbian learning. The selected connections are enabled to feed the selected input signal to the corresponding output node (OR gate). Finally the class of the image is identified.

3. PROPOSED METHODOLOGY

This method proposes an efficient classifier which is a Radial Basis Function Neural Network (RBFNN) for the hand sign recognition system. RBFNN is popular for pattern classification and signal processing. It is a two step algorithm and training the network is much faster.

A. NETWORK ARCHITECTURE

A radial basis function network (RBF networks) is single layer network and it is associated with radial functions. RBFNN consists of three layers as shown in Fig.6.

1. Input layer

The input layer has number of source nodes whose number is equal to the dimension \( D \) of the input vector \( X \). The entire input vector is connected to each of the hidden layer.

![Fig.6 RBFNN Architecture](image)

2. Hidden layer

The second layer of RBFNN is the hidden layer in which all input nodes are directly connected to nonlinear units. The hidden unit contains a basis function and typical radial function is the Gaussian and is given by

\[
h(x) = \exp \left( -\frac{(x - c)^2}{2r^2} \right)
\]

Its parameters are its centre \( c \) and its radius \( r \).

3. Output layer

From the input space to the hidden unit transformation is nonlinear, whereas the hidden unit space to the output space transformation is linear. The output is computed by taking a weighted sum of the activation values from every hidden unit and it is given by

\[
y(x) = \sum_{j=1}^{n} w_j h_j(x).
\]

Its parameters are \( w_j \) is weight vector and \( h_j(x) \) is gaussian function.

B. IMPLEMENTATION OF RBFNN

RBFNN has a hybrid learning algorithm: unsupervised learning sets the RBF centers; supervised learning trains the hidden to output weights. The proposed RBFNN for hand sign recognition is shown in Fig.7. The input image is pre-processed by binary quantization, vertical and horizontal projection histograms are taken followed by discrete Fourier transforms (DFTs) that will calculate the magnitude spectrum. The magnitude spectrum is used as the feature vector to be fed to the network.

![Fig. 7 Flow diagram of hand posture recognition using RBNN](image)

The system is very robust to location change of the image by the use of magnitude spectrum. For the hand posture recognition the RBFNN has to be trained. Training algorithm and application algorithm are applied based on the paper [3]. In the training phase of RBFNN, a set of training instances is given. In each training instances feature vector is described. Then it is linked with the desired outcome, which is further represented by a feature vector called output vector. By setting random weight, the neural net is trained according to learning algorithm. When the training phase is complete the weights are fixed. After training the network, it is tested with a new set of test images to check whether the
classifier has learnt the knowledge to classify the input images.

4. SIMULATION RESULTS AND ANALYSIS

The simulation result for implementing existing methodology is determined using Matlab and Modelsim software is shown in the Fig.8. The simulation result for implementing SOM-Hebb classifier determined using Matlab and Modelsim software is shown in the Fig.8. Total time taken for running existing method is 4.221nS and power is 12.4mW. The simulation result for implementing RBFNN classifier is shown in the Fig.9. Total time taken for running RBFNN is 4.040 nS and power is 9.2mW.

Fig.8 Simulation Results for hand sign recognition using SOM-Hebb classifier (a) Input Image (b) Extraction of Red part (c) Binary Quantization (d) Training of SOM-Hebb Classifier (e) Testing of SOM-Hebb Classifier (f) Output Image and its class

Fig.9 Simulation Results for hand sign recognition using RBFNN classifier (a) Input Image (b) Extracted Image (c) Output Image and its class
5. CONCLUSION

The main advantage of the proposed system has an ability to train the classifier network faster. The RBFNN achieves a recognition rate of up to 100%. It has been found that the existing method used for SOM-Hebb classifier have delay and power complexities. This drawback has been overcome by using the RBFNN classifier in the proposed approach. However, the proposed method has yielded satisfactory results compared to the existing approaches. The various advantages of RBFNN are short learning time and two step algorithm. This RBFNN classifier for hand sign recognition is coded using VHDL and simulated using ModelSim 10.0b and Matlab. The delay and power consumption of this RBFNN is analyzed using Xilinx ISE 9.1 software.

6. REFERENCES