An evaluation of real-time requirements for automatic sign language recognition using ANNs and HMMs - The LIBRAS use case

Mauro dos Santos Anjo, Ednaldo Brigante Pizzolato, Sebastian Feuerstack
Computer Science and Engineering Department
Universidade Federal de São Carlos - UFSCar
São Carlos - SP (Brazil)
Emails: maurosanjo@gmail.com, ednaldo@dc.usfscar.br, sebastian@feuerstack.org

Abstract—**Sign languages are the natural way Deafs use to communicate with other people. They have their own formal semantic definitions and syntactic rules and are composed by a large set of gestures involving hands and head. Automatic recognition of sign languages (ARSL) tries to recognize the signs and translate them into a written language. ARSL is a challenging task as it involves background segmentation, hands and head posture modeling, recognition and tracking, temporal analysis and syntactic and semantic interpretation. Moreover, when real-time requirements are considered, this task becomes even more challenging. In this paper, we present a study of real time requirements of automatic sign language recognition of small sets of static and dynamic gestures of the Brazilian Sign Language (LIBRAS). For the task of static gesture recognition, we implemented a system that is able to work on small sub-sets of the alphabet - like A,E,I,O,U and B,C,F,L,V - reaching very high recognition rates. For the task of dynamic gesture recognition, we tested our system over a small set of LIBRAS words and collected the execution times. The aim was to gather knowledge regarding execution time of all the recognition processes (like segmentation, analysis and recognition itself) to evaluate the feasibility of building a real-time system to recognize small sets of both static and dynamic gestures. Our findings indicate that the bottleneck of our current architecture is the recognition phase.**

Keywords: LIBRAS, Sign Language, Computer Vision, Gesture recognition.

I. Introduction

A wave, a jump, a contortion, a smile, a desperation expression or any other body motion are people's reactions to some happenings and are also means of communication. These gestures may represent more than 50% of a conversation's content as pointed out by Birdwhistell [1]. Besides, sign languages (which use gestures) are the natural way Deafs use to communicate. There are many sign languages all over the world and LIBRAS is the Brazilian version with its own lexical as well as syntactic and semantic rules. It is a visual demanding language and the meaning of the signs is strongly attached to the context of the communication or situation. Even inside a deaf person's family there are people who do not understand it. The signs may be performed with both hands in front of the body with or without movements and with or without touching other body's parts. Besides all the hand and face movements, it is also important to notice that face expressions contribute to the meaning of the communication. They may represent, for instance, an exclamation or interrogation mark. As sign languages are very different from spoken languages and it is hard to learn them, it is natural, therefore, to expect computer systems to capture gestures to understand what someone is feeling or trying to communicate with a specific sign language in order to promote social inclusion. This can be accomplished by automatic gesture recognition (AGR) [2].

AGR is also important to Human-Computer Interaction as multimodal interfaces (MMI) can be significantly enhanced by using gestures to complement some already common inputs like touch, keyboard, and mouse, since it is a natural way of communication - like speech.

According to Hong [3], learning and recognizing gestures are difficult as the gestures vary from instance to instance (position, trajectory, pose, background,...). Mitra's survey [2] points out that gesture recognition is a complex task that can be divided into two categories: static and dynamic gestures. The former recognizes postures without movement whilst the latter includes movements (and they have to track the user hands or any other significant limbs over time to output a recognized gesture).

On the one hand, the major concerns of static gestures are the segmentation of the hands and head postures from the scene and the classification of the hand configuration and face expressions. The dynamic gesture recognition, on the other hand, has further challenges, such as: (1) recognizing the hands trajectory; (2) recognizing the sequence of hand postures; and (3) recognizing the face expression.

The task, therefore, could be divided into three categories regarding challenges: segmentation of body's parts, pattern classification and gesture recognition. The major challenges of segmentation are:

- High degree of freedom: hands and arms can have a high degree of mobility which increases the search space for movement patterns;
- Human parts overlapping: due to the high degree of freedom, arms can overlap each other and also may be in front of the chest or head;
- Face expressions: face detection is a trivial task, but the task of recognizing a face expression is not that simple as it needs to take eyes, eye brown, nose, mouth and forehead variability into account;
- 3D trajectory: Signs of any sign language are 3D; A 2D projection may make things a bit harder;

From the pattern classification viewpoint, some important challenges are:

- Large vocabulary of signs: there are 46 different hand postures [4], but different posture angles, different relative position to the body or even different face expressions for the same posture produce different meanings;
- Variations of hand postures perspectives: as the camera is put in a fixed place, the relative position of people may produce different perspectives;

And from the gesture recognition viewpoint, there are also other important challenges:

- Variations of sign trajectories: it is almost impossible to do the same movements in the same manner all the time. So, although the trajectory plays an important role for the gestures classification, for the same word or sentence, the trajectory varies for the same person and from person to person;
- Variations of speed at performing a sign: the communication is a sequence of signs which could be just compared to a sequence of frames. Some people may perform the same sign in a sequence of - let's say - 20 frames whilst others may do the same in 30, 35, 40 or 50 frames. This could be related to the context of the information or just the way people perform the sign.

Considering that Human-Computer Interaction through signs requires real-time processing and that for multimodal interaction real-time processing is a basic requirement to sync gesture recognition with other modes like voice or body movements, researchers started to address this topic [5], [6], [7].

With the recent development of Microsoft's Kinect for XBox gaming console and its personal computer version, segmenting and tracking parts of the human body have become easier and a variety of applications has been investigated: Healthcare [8]; Augmented Reality Environments [9] and Gesture Recognition [10] among others.

In this paper we focus on static and dynamic gesture recognition using a software architecture based on Artificial Neural Networks (for pattern classification) and Hidden Markov Models (for time alignment). We also report some information related to real-time recognition. In order to evaluate the software architecture, we chose the Brazilian Sign Language (LIBRAS) as use case.

The paper is structured as follows: In section 2 we present the related work; in section 3, the issues related to segmenting hands and heads from the scene; in section 4, the problem of classifying the signs; in section 5, the time alignment problem of dynamic gestures; in section 6, the methodology aspects of our investigation, considering our system's architecture, the datasets, the scenarios of the experiments and the results; and, in section 7, we present the conclusions and future work.

II. Related Work

Static gesture recognition has been investigated by many researchers: Bretzner [11], Chen [12], Fang [13], Wysoski [14] and Phu [15], among others. Bretzner [11], for instance, presented a scale-space blob model for the hands to recognize four static postures and with them control a TV device. Chen [12] and Fang [13] used Haar-Like features and AdaBoost training to recognize static gestures. Chen proposed a two-layered system to recognize four static hand postures. Fang used color information and scale-space blobs in a twenty five level pyramidal structure in order to detect the fingers and the palm of the hand. With homogeneous background the system recognize six hand postures with 98% recognition rate; with complex backgrounds, the recognition rate drops down to 88%. Phu [15] and Wysosky [14] investigated the use of Multi-Layer Perceptron [16] as a classifier for the problem of hand postures recognition. The former obtained 92.97% recognition rate when working on a set of 10 postures; the latter achieved 96.7% when applied on a set of 26 postures of the American Sign Language.

Hong [3], Modler [17], Vadafar[18], Elmezain [19] Pizzolato [20], Bauer [21], Wang [22], Brasher [23] and Pahlevanzadeh [24] (just to mention a few), tackled the problem of dynamic gesture recognition. Hong proposed a system based on finite-state machines (FSMs). The system first segments head and hands using skin color detectors and then uses FSMs to model the temporal behavior of each gesture. Modler proposed a system to recognize cycled gestures (those that the initial and the final postures are the same) using Time-Delay Neural Networks [25]. The gestures are performed with only one hand in a controlled background. The overall recognition rates were 99%. Vadafar proposed an approach called Spatio-Temporal Volumes. The idea is to create a feature vector based on temporal 3D shapes created through sequences of segmented hand shapes over time. Three different classifiers were tested: k-nearest neighbor, Learning Vector Quantization (LVQ) and Artificial Neural Networks (ANNs). For the task of recognizing six gestures, the k-nearest neighbor presented the best performance: 99.98%. Elmezain proposed a stereo vision system (in order to capture depth information) based on Hidden Markov Models (HMM) [26] in order to model the temporal behavior of the dynamic gestures. The system presented a recognition rate of 98,33% for all the alphabet letters (A-Z) and digits (0-9) drawn with the fingers on the space. Pizzolato proposed a two layer architecture based on ANNs and HMMs to recognize 15 animal names spelled with LIBRAS gestures. The first layer uses an ANN to recognize static hand postures of LIBRAS and the second level uses HMMs to model the temporal behavior of each word. Working on batch mode, the system presented a recognition rate of 91.1%. Bauer proposed a HMM based system to recognize 97 different gestures of the German Sign Language (GSL) and the system presented a recognition rate of 91.7%. Wang's system focused on modeling small units of the gestures (similar to the phonemes). This approach tackled 5.119 gestures of the Chinese Sign Language. The system was able to perform phrase recognition and showed an overall recognition rate of 90%. Brasher proposed a system to be used on the CopyCat game to teach the American Sign Language to deaf children. The system used colored gloves to recognize 541 phrases with 91.75% recognition rate. Pahlevanzadeh proposed a system based on Fourier Descriptors to model the trajectory gestures and Generic Cosine Descriptors for the hand shape modeling. Tested for 15 dynamic gestures it showed a recognition rate of 100%.

III. The Segmentation Problem

As stated, interpretation of human actions in computer systems is a complex task due to problems of segmenting human figures in a scene and tracking body parts with a high degree of freedom. The difficulty of the segmentation is the attempt to handle all issues that rise in vision systems like: illumination; cluttered backgrounds; clothing; occlusions; velocity of movement; quality and configuration of cameras and lens; among many others. In order to make the systems simpler, the most common approach used by researchers is to eliminate some of the variables by stating system constraints or using extra devices like in the following situations:

- Controlled environments: to eliminate some environment variables, for instance avoiding cluttered backgrounds by using a plain wall or Chroma Key and constraining user clothing permits segmentation by simple thresholding [20].
- Colored markers: to track a specific body part a common approach is the use of a colored marker. For instance a colored glove to track a user hand and segmenting it by thesholding in a color space model [27], [28], [29], [30].
- 3D Gloves: sometimes a great precision of tracking is needed, like finger movements. In this case an extra device is needed, like a 3D Glove that embeds gyroscopes and accelerometers to track users movements [31], [32].

As the objective for many researchers is to interact naturally with the computers, it is important to avoid extra wearable devices or specific restrictions of environment. To do so, researchers tried to abstract how humans perceive objects in space. They found out that what eases human perception of objects is the Stereopsis or impression of depth that we can perceive due to human binocular vision. Researches tried to replicate that in computer systems with an approach called Stereo-Vision. The Computer Stereo-Vision is obtained by a pair of visual cameras trying to estimate depth information using stereo correspondence algorithms [33].

The major problem of this approach is that it relies on visual properties (corners, edges, color), and all of them are prone to problems due to illumination variation, camera and lens quality, environment, clustered backgrounds, among others. Further on, this reduces the accuracy of the estimation and also requires hardware level code, like GPU or even Field-Programmable Gate Array (FPGA), to achieve real-time data. Another solution is to use depth cameras, that instead of visual spectrum they use the infra-red spectrum to avoid illumination and other environment constraints. These cameras are a recent development, and have achieved good results in depth estimation in a range of 80cm to 4m. Microsoft Kinect is one example of such devices. It is a product initially developed for the XBox 360 gaming console with the purpose of enabling the user to interact with the games using nothing but body movements. Kinect is a device with an Infra-Red (IR) projector, a VGA camera and an IR camera. Kinect estimates the depth of the objects in the scene doing a stereo-vision approach, like aforementioned, in the IR spectrum. This technology was developed by the Israeli company PrimeSense. The IR Projector projects a known dotted pattern in the environment and the IR camera (along with the embedded algorithm patented by PrimeSense) can estimate depth with a resolution of 640x480 pixels.

IV. The Pattern Classification Problem

Pattern classification is, from the theoretical viewpoint, an automatic transformation of observed features into a set of symbols or classes. The classes are built based on some training data through some learning process. The learning process can be supervised or unsupervised. There are many methods to perform pattern classification, such as:

- Bayes Classifier;
- Decision Trees;
- Gaussian Mixture Models;
- Support Vector Machines (SVMs);
- Artificial Neural Networks (ANN); and
- Conditional Random Fields (CRFs).

Artificial Neural Networks (ANNs) have been widely used to solve pattern recognition problems in a great variety of applications like time forecasting [34], medical

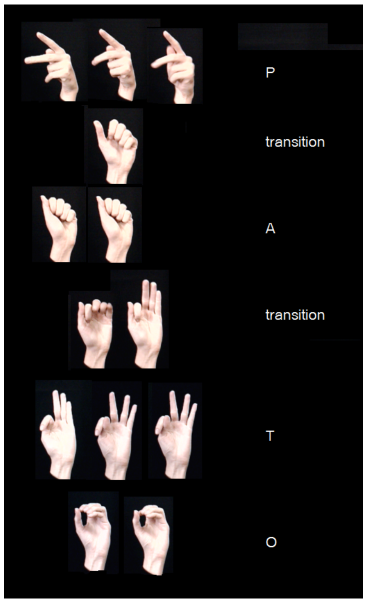


Fig. 1: A sequence of hand postures to spell the word pato (duck in portuguese)

diagnosis [35], astronomy [36], gesture recognition [2], among others. The Multi-Layer Perceptron (MLP) [16] is a feed-forward ANN that is an evolution of the Standard Perceptron [37]. The purpose of this statistical model is to learn how to distinguish between nonlinear separable data. MLP is composed by neurons disposed in input, hidden and output layers; its structure is of a directed graph with all neurons of one layer fully connected with the next layer. The connections between neurons are called weights (as they are values to weight the inputs). Each neuron has an activation function that processes all the weighted inputs. The MLP learns to solve the pattern classification problem by using a learning algorithm that tries to model the mapping of inputs, with a predefined dimension, in a subset of possible outputs and the knowledge of the network is stored in the weights. Each of them has an initial value and through the learning process, all of them are updated until convergence is reached.

V. The Time Alignment Problem

Similar to the spoken language, signs may be performed in different ways according to the speed or ability of the person (or also due to the context of the communication). A schematic representation of the sequence of static signs to spell 'pato' (duck in Portuguese) is presented in figure 1. Although it is not an actual sequence, one can see that the letters can span several frames. Further on, there are transition frames that may be assigned to a symbol not related to the letters of the word.

In order to handle the exposure of signs with different durations, one could use Dynamic Time Warping algorithm (DTW). With it, it is possible to measure similarity between two sequences of data which may vary in time or speed. Through a DTW a computer program can find an optimal match between two given sequences.

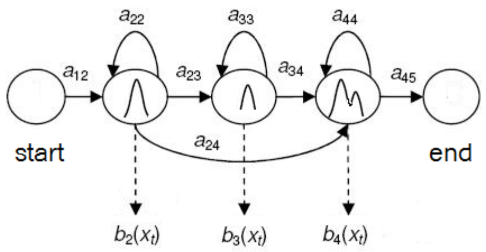


Fig. 2: HMM with 3 states

However, sign duration is not the only problem regarding gesture recognition. As can be seen at figure 1, there are hand posture transitions between every two signs. And they may have different durations as well.

One method to tackle this problem is Hidden Markov Models (see figure 2).

Hidden Markov Models attempt to model the joint probability distribution of a sequence observations x and their relationship with time through a sequence of hidden states y. A HMM is described by a tuple:

$$\lambda = (A, B, \pi) \tag{1}$$

in which A denotes a matrix of possible state transition probabilities, B is a vector of probability distributions governing the observations and π is a vector of initial states probabilities. The probability of given sequence of observations x being emitted alongside the state path described by y is given by

$$p(x, y) = \prod_{t=1}^{T} p(y_t | y_{t-1} * p(x_t | y_t) \tag{2}$$

where the observations $x_t \in$ x can be either continuous or discrete, univariate or multivariate. It is possible to extract the likelihood of a sequence x by marginalizing y out of the equation:

$$p(x) = \sum_y p(x, y) = \sum_y \prod_{t=1}^{T} p(y_t | y_{t-1} * p(x_t | y_t) \tag{3}$$

in which the summation over all possible state sequences can be computed efficiently using the Forward algorithm. A very comprehensive explanation of HMMs and their canonical problems can be found in [26].

A. Hidden Markov Models for Classification

Exploring the fact that an HMM is able to provide the likelihood for a given sequence x, it is possible to create a classifier by creating a model λ_i for each sequence label $\omega_i \in \Omega$ (see figure 3). Treating each model λ_i as a density model conditioned to an associated class label ω_i, one can apply the Bayes' rule to obtain the a posteriori probability

$$P(\omega_i|x) = \frac{p(\omega_i) * p(x|\omega_i)}{\sum_j^k p(x|\omega_j)} \tag{4}$$

and then decide for the class with maximum a posteriori.

Of course, creating models for each label means that the system needs to evaluate all of them in order to reach a decision. This implies that as the number of models increases so does the time to compute it (and therefore, has in implicit real-time performance). Later on we will produce some results on this respect.

VI. Methodology: tools, experiments and Results

In this paper we focus on the real-time requirements to perform static and dynamic gesture recognition. For static gesture recognition we used ANNs and for dynamic gesture recognition a combination of ANNs and HMMs. The input device chosen for the experiments was the Kinect.

We have carried out two types of experiments: static and dynamic gestures recognition. A tool called Gesture User Interface (GestureUI) was developed in order to encapsulate all the work that has been done and also to ease the implementation of additional applications. GestureUI was implemented with multi-platform frameworks and tools; for image processing and visual spectrum cameras frame-grabbing, OpenCV was used while the interface was developed in C++ using wxWidgets in the Code::Blocks IDE. For integration with Kinect (or any other device) we used the middleware OpenNI .

We based our tool on previous work [20] where 27 signs of the LIBRAS alphabet were classified by a MLP and produced 90,7% recognition rate for the static gestures. We still used binary images of 25x25 pixels as input although it is not the best statistical option due to the high dimensionality of the inputs. We found out that this is not a problem for small sets of gestures, as we describe later on.

The basic structure of our classifier (figure 4) for static gesture recognition has an input layer of 625 inputs (25x25 binary image), followed by one hidden-layer with 100 neurons and an output layer with 5 possible classes.

It was very interesting to get good recognition results over small sets of static gestures using binary images as feature vector. We believe, however, that:

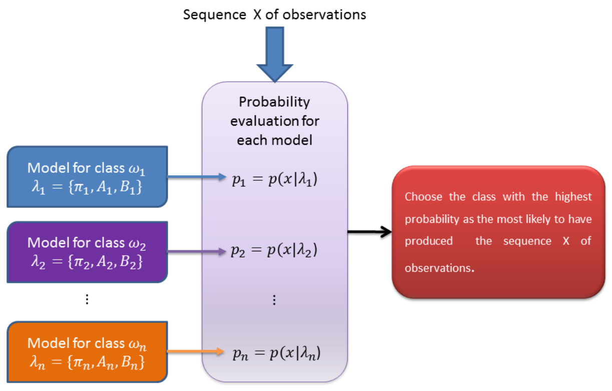
- we could get the same performance (or even better) by using a small feature vector based on shape descriptors; and
- shape descriptors would capture slight differences among similar gestures.

When creating a new interface in GestureUI the user has to choose the segmentation approach between two categories:

- Visual Camera: In this mode, the software uses a monocular system that tracks colored gloves. The user can define different colors for each hand. The system is able to detect the markers by thresholding the HSV color-space model and has been presented in [20].
- Kinect: This mode provides automatic segmentation of the user, and uses the Virtual Wall algorithm (detailed in next subsection) to segment and track user hands.

The user can also set the type of recognition:

- Static Gestures: hand postures are supposed to be presented to the tool for training and recognition.
- Dynamic Gestures: gestures that comply hand and arms movements are fed into the system. The results of the MLP are fed to HMMs which can be trained to learn the poses and trajectories of the dynamic gestures or can recognize the gesture that has been performed.



Sequence X of observations
Probability evaluation for each model
Model for class ω_1 $\lambda_1 = \{\pi_1, A_1, B_1\}$
Model for class ω_2 $\lambda_2 = \{\pi_2, A_2, B_2\}$
Model for class ω_n $\lambda_n = \{\pi_n, A_n, B_n\}$
$p_1 = p(x|\lambda_1)$
$p_2 = p(x|\lambda_2)$
$p_n = p(x|\lambda_n)$
Choose the class with the highest probability as the most likely to have produced the sequence X of observations.

Fig. 3: Maximum likelihood decision using multiple HMMs

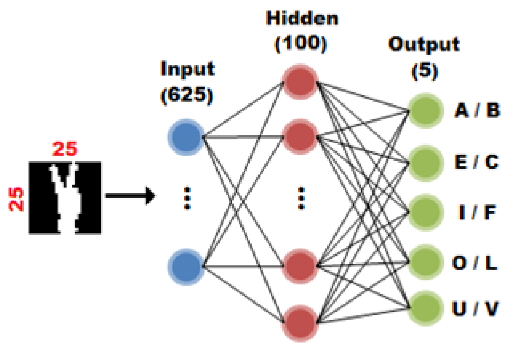


Fig. 4: Architecture of the MLP Classifier for static gesture recognition

After configuring the system, the user can train new models or use previously trained ones for recognition.

To train new models the training and testing set must be representative, including possible variations of user input. To ease this task the software is able to collect samples in real-time using a sequential trigger that snaps the images of the postures and organize them in folders by gesture name. These samples can be used to train a new MLP model, letting the user configure all the properties of the ANN.

When turned into the execution mode, the system captures the hands postures and feeds the MLP with them. The output of the recognized gestures are shown in the screen.

As an example, in figure 5, a user presents two signs (one for each hand) and the system outputs the corresponding letters (the leftmost sign is L and the rightmost is V).

There are two pre-conditions to identify dynamic gestures:

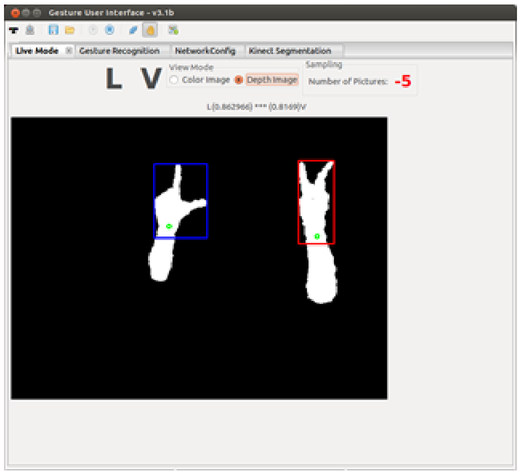
- At least one of the hands must be visible;
- At least one of the hands must be moving;

Therefore, the system understands that a dynamic gesture has finished if:

- There is no visible hand;
- There is no movement.

A. *Virtual Wall and Hand Tracking algorithm*

We have used the Kinect controller to enable user interaction in indoor non-controlled environments. PrimeSense provides a middleware called NITE that enables the system to segment the users figures in the scene and to obtain: a) a mask with all user pixels and b) the user Center of Mass (CoM). Before we explain our algorithm for hand tracking and segmentation we need to state some domain information about LIBRAS. When the person is performing the gesture, the hands are always in front of the torso and sometimes there is contact



View Mode: Color Image [✓ Depth Image]

L(0.862966) *** (0.8169)V

Fig. 5: GestureUI recognizing static gestures L and V

with the users face or even the torso itself. Furthermore, in LIBRAS the person can perform a gesture with the dominant hand or both hands. The dominant hand is the right one for right-handed people and the left one for left-handed people. In this context we developed a simple algorithm to segment user hands called Virtual Wall that sets a depth threshold that works like an invisible wall in front of the user. The position of this wall is calculated using the users CoM in the following way:

$$VW_{depth} = CoM_{depth} - \alpha \tag{5}$$

where α is an offset to avoid that the Virtual Wall is transposed by the face or torso. It can be set with an empiric value depending on how the application will be used. With this approach the user can move around and the Virtual Wall will follow him/her. After obtaining only the blobs that are in front of the Virtual Wall by thresholding the depth map, we obtain a binary image. We then extract the blobs that correspond to the hands of the user from the binary image and eliminate some undesired noise blobs. First one has to find the blobs, or connected components, positions and calculate their area. To do so a linear-time component labeling algorithm [3] is applied. After obtaining the list of blobs, we created a simple algorithm to track the hands based on the aforementioned domain restriction of LIBRAS:

Algorithm 1 Algorithm for finding a list of blobs

```
Sort blobs detected in descending order of area
Select one or two major blobs as hands
if (there is only one blob) then
    dominant-hand = blob
    Define the Region Of Interest (ROI) of the present hand as the minimum rectangle that encloses the blob
    Calculate the CoM of the hand
end if
if (two blobs are visible) then
    left-hand = leftmost-blob
    right-hand = rightmost-blob
    Define the Region Of Interest (ROI) of the present hands as the minimum rectangle that encloses the blobs
    Calculate the CoM of each hand.
end if
```

B. *Aspect Ratio Hand Cropping Heuristic*

One problem of classification that arises when applying the Virtual Wall method is the appearance of the arm when a gesture is performed. When performing static gestures from the Libras alphabet, the person's arm is, most of the time, positioned orthogonally in relation to the horizontal axis. When we frame the person's sign, it contains a significant part of the arm. This reduces the

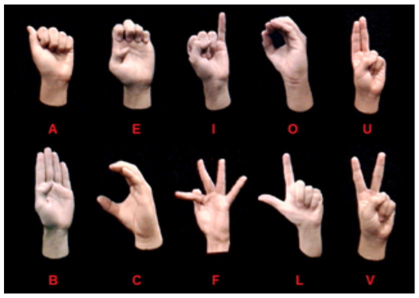


Fig. 6: The static gestures of Libras alphabet used for training the ANNs

recognition rate as many signs have a similar pattern (the arm). In order to tackle this problem, we introduced a heuristic to eliminate or at least reduce the visible arm: Aspect Ratio Hand Cropping Algorithm (ARHCA). It tries to eliminate the arm by a simple aspect ratio checking. We defined a fixed aspect ratio to reduce the Region of Interest (ROI) in the following way:

$$AspectRatio = \frac{ROI_{height}}{ROI_{width}} \tag{6}$$

$$\begin{cases} CoM_y + \alpha & if AspectRatio > \beta \\ ROI_{heigth}, & otherwise \end{cases} \tag{7}$$

where β is the aspect ratio threshold desired and α is an offset in relation to the y coordinate of the Center of Mass of the ROI (CoM_y). These parameters can be changed in real-time but empirically we defined β= 1.34 and α= 1.05*ROI_{height}. After recalculating ROI_{height}, the ROI_{width} needs to be adjusted so the resulting ROI keeps being the smallest rectangle enclosing the blob of the hand.

C. The Experiments with Static Gestures

We trained two MLPs to recognize the following groups: 1) A, E, I, O, U and 2) B, C, F, L, V. The former is a set containing all the vowels of the alphabet. The latter we chose only 5 consonants, so the set has the same size as the one with vowels.

We gathered a training set of 150 images and a testing set of 250 images of each static gesture in figure 6. The testing set was never presented to the MLP during training process.

The datasets were created in order to represent possible small rotations and also scale variance that users may present at real world situations. This eases the generalization capacity of the MLP classifiers. Furthermore, as we obtained images with variable scales using our segmentation algorithms, we had to prepare all images to be used by the classifier doing the following procedure:

1) Calculate the aspect ratio of the ROI image and resize proportionally trying to achieve 25 pixels in at least one dimension;

TABLE I: Recognition rates for A, E, I, O, U

Static Gesture	Without ARHCA	With ARHCA
A	65%	100%
E	60%	100%
I	74%	100%
O	70%	100%
U	68%	100%
Mean	67.4%	100%

TABLE II: Recognition rates for B, C, F, L, V

Static Gesture	Without ARHCA	With ARHCA
B	73%	100%
C	77%	100%
F	80%	100%
L	73%	100%
V	74%	100%
Mean	75.4%	100%

2) The resized image will not be binary anymore due to approximations of the resizing algorithm, so it must be thresholded;
3) If the resized image has one of its dimensions different from 25, center it in a 25x25 image

Due to 25x25 resizing, we lose resolution for the hand representation, which is the most important part for the MLP to distinguish between gesture classes. Therefore, we had to collect an additional testing and training set using the Aspect Ratio Hand Cropping algorithm in order to improve the level of detail in the images. With the two datasets we could compare performances of recognition using both datasets (see tables I and II).

It is important to notice that the training and testing were performed with 25x25 binary images which do not deliver a good 2D resolution (as stated before). Slight differences among some gestures could be better captured by some shape descriptor which should be evaluated in the future work. From a real-time viewpoint, working with the raw image (25x25) is a worst scenario than with a feature vector of size 80, for instance (like the one explained latter on for the dynamic gesture recognition).

One can see from the results, a major improvement when the ARHCA algorithm was used to collect sample data. Also we can see that the second MLP performed better, due to the visually perceptible discrepancy between patterns of B, C, F, L, and V that eased the discrimination of the gestures even with resolution loss (table II).

In order to evaluate the execution time of the whole static gesture recognition system, we subdivided the steps of execution in three categories:

- Segmenting: involves capturing the depth map, detecting the user position and using the Virtual Wall algorithm;
- Analyzing: time to detect blobs; extract hands Region of Interest; and prepare images to feed the ANNs (resizing and centering);
- Recognizing: time for the trained MLP to process the

TABLE III: Mean Times of execution along with the standard deviation of each category

Task	Mean	Standard deviation
Segmenting	7,295.17 μsec	3,169.06 μsec
Analysing	2,180.62 μsec	724.56 μsec
Recognition	651.26 μsec	131.27 μsec

input gesture and to output a classified category.

We gathered five streams of videos with 25 seconds each, in which a user was performing random gestures with both hands while moving towards and backwards in front of the Kinect. The computer used for testing was an Intel i7 processor with 6GB of RAM.

The measurement of these categories times was performed using the POSIX Time Boost library with a microsecond (μsec) precision clock.

With the streamed videos we obtained 3,750 measurements of frame processing. The measured mean times and its standard deviations are presented in table III.

Summing the means of execution time we obtained 10,127.05 μsec which is the time necessary for the system to analyze one single frame. This means that the system can process frames with a frequency of 98.81Hz. However, there is an overhead of about 6 milliseconds to update the interface and give feedback to the user. So the system is actually able to process frames at 62.5Hz. Unfortunately, the Kinect version we used is able to output only 30 frames per second which limits the system to 30Hz.

D. The Experiments with Dynamic Gestures

The LIBRAS's signs are performed with hands. Face expressions or head movements may be part of the sign. In order to understand the technical details of the recognition it is important to know how the system works: the user stays in front of the computer and starts performing the sign of a specific word; the signed word is a sequence of gestures (like the spelled word in figure 1) which is analogous to a sequence of frames in a video (words like wardrobe or shoes in LIBRAS are performed in a dynamic fashion). For instance, figure 7 shows a small sequence of gestures (from A to F) related to the sign for shoes whilst figure 8 presents the sequence for wardrobe; as indicated before, the user performs the sequence of gestures to compose the sign and the system identifies the most likely Hidden Markov Model that would produce such sequence (following the schema presented in figure 2). If one imagines a model with 6 states (like the one in figure 9) each of them responsible to recognize the poses of figure 7, then, as the user performs the poses, these states produce better results than the states of the other words (notice that the self transitions are responsible to match the same pose for that state).

Of course, a more complex system would allow recognition of a sequence of words (signs) in order to produce

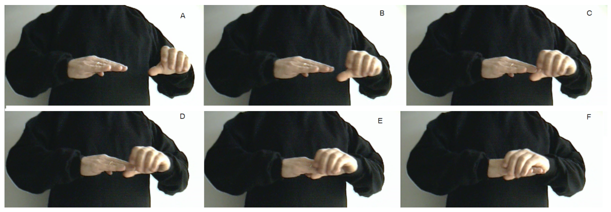


Fig. 7: Small sequence of gestures for shoes (LIBRAS)

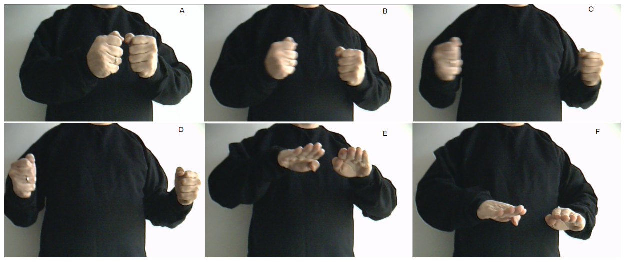


Fig. 8: Small sequence of gestures for wardrobe (LIBRAS)

a phrase. Such system would have to allow the connection of the end of one model to the beginning of all the models (or at least some of them - according to some grammar) in order to recognize several words (signs) in a phrase. Figure 10 shows an example of 4 HMMs (for 4 signs) in which the system is hypothetically trying to make a transition from the first sign (on top) to the other three (excluding itself). The blue circles stand for the beginning of the models whilst the red ones to the end. The other circles at the figure represent the HMM states for each sign (the self transitions are omitted). The signs in this example have the same number of HMM states, but this is not the case for real modelling. And, it is important to notice that, according to the Viterbi algorithm [38], only after computing the last gesture it is possible to identify the best sequence of states and, therefore, the best sequence of signs. From the explanation it is possible to conclude that such systems must have techniques (like prunning [39]) to optimize the search space.

Our system, so far, works only with single words. And, to be more precise, the experiments, so far, do not take face expressions and head movements into account. Hands and arms are the most important information. Therefore, the first step is to extract them from the scene using the Virtual Wall algorithm. The result: blobs that need to be labeled. For this task the blob labeling algorithm [40] - which uses the internal and external contours to identify the blobs in linear time - is used. After labeling, three kinds of objects may be found:

1) Noise: blobs that may appear due to Kinect's depth estimation;
2) User's legs: when the user is sitting, it is likely that the legs will trespass the virtual wall;
3) Hands and arms: the objects we really want.

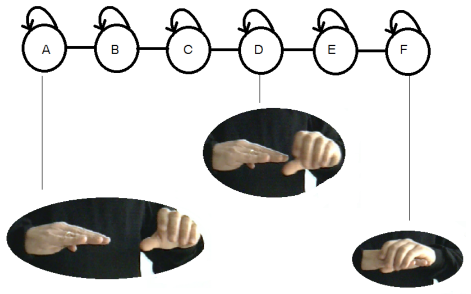


Fig. 9: HMM states associated to the sign for shoes

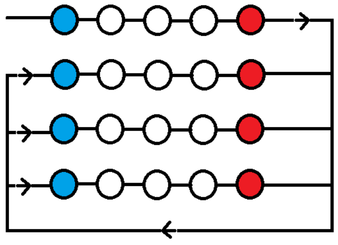


Fig. 10: The connection of one Hidden Markov Model to the others

The first one can be easily discarded by a threshold area. All the blobs with an area (in pixels) smaller than the threshold are discarded. For the experiment, the minimum area was 400 pixels. The remaining blobs are sorted and the two with biggest areas are labeled as the person's hands. It is important to notice it is still possible to have one or two big blobs associated to the legs, but a simple heuristic of eliminating blobs that overlap the bottom border of the image is enough to get rid of this problem.

In order to describe the hands actions, a feature vector is created in order to gather information of both hands (postures and movements). Our feature vector is based on the work of Thiago Trigo e Sergio [41] and contains:

1) Geometric descriptors: aspect ratio, circularity, spreadness, roundness, solidity and finger tips;
2) Trajectory descriptor : 3D position relative to the Center of Mass of the person; and
3) Fourier descriptors: invariable to translation, rotation and scale.

We created two 7-word gesture groups in order to evaluate the recognition performance:

1) Adorar (to like), armário (wardrobe), carro (car), comprar (to buy), eu (I), querer (to want), sapato (shoes); and
2) Banheiro (toilet), borboleta (butterfly), casa (house), cesta (basket), complicado (complicated), nós (we), simples (simple).

They were chosen due to their complexities which include partial face occlusion and different contact positions between the hands and the body parts.

In order to evaluate the feasibility of our feature vector, we created three testing cases:

- Scenario 1: Trajectory, Fourier and Geometric descriptors compose the feature vector (80 dimensions);
- Scenario 2: Only 3D position and Fourier descriptors were used to compose the feature vector (68 dimensions); and
- Scenario 3: Only 3D position and Geometric descriptors (aspect ratio, circularity, roundness, solidity and finger tips) were used to compose the feature vector (18 dimensions).

For each of the 14 words we have collected 40 samples for the training set and another 40 for the testing set. The samples were collected through the GestureUI and we cared about:

- Changing the position of the capturing device to capture different angles of the gestures;
- Changing the perspective of the people performing the signs in order to make the system more robust to perspective; and
- Instruct them to perform the signs at different speeds.

All the training samples were used to create 16 or 32 clusters (using K-means algorithm [42]). Each word had its own HMM with different numbers of states. The recognition was performed via Viterbi algorithm [26].

In the first scenario the system presented a recognition rate of 98.39% with 16 clusters and 99.82% with 32 clusters.

In the second scenario (without geometric descriptors) the system presented a poor recognition rates: 12.50% with 16 clusters and 14.82% with 32 clusters.

In the third scenario (without Fourier descriptors) the system presented a recognition rate of 99.64% with 16 clusters. We could not evaluate the performance with 32 clusters as 8 of them were empty (probably due to a low dimensionality). We believe, however, that if we had more words in our vocabulary, we would have had more different gestures and, as consequence, more clusters.

Again, the tasks were divided into 3 categories: segmentation, analysis and recognition and the measurement of these categories times was performed using the POSIX Time Boost library with a microsecond (μsec) precision clock.

We evaluated the system at the task of recognizing 7 words using 16 or 32 clusters. With 16 clusters, the evaluation is quicker than with 32 clusters as shown in table IV. As expected, a scenario with a high dimensionality and 32 clusters is more time consuming than the others. Besides recognition times (shown in table IV) we had also to consider: the segmentation time (7.29 ms), the analysis time (2.18 ms) and the interface overhead (6 ms). The whole processing time is 27.38 ms which

TABLE IV: Mean times and pattern deviation of 7-word dynamic gesture recogntion

Clusters	Scenario	Mean	Standard deviation
16	3	2.60 ms	0.68 ms
16	1	6.38 ms	1.60 ms
32	1	11.91 ms	3.00 ms

means that in the worst scenario, the highest frequency would be 36.52 Hz. The best scenario would perform at a frequency of 55.34 Hz.

We also measured the execution time for the task of recognizing 14 words. For the task we considered the best configuration of the experiments with seven words (55.34Hz) and obtained an average recognition time of 10.77 ms (with pattern deviation of 4.24 ms). The whole processing time was 26.24 ms (7.29 ms for segmentation; 2.18 ms for analysis; and 6 ms for interface overhead). With 14 words, the frequency is about 38,10 Hz.

In order to get good recognition rates, we had to double the number of clusters which increased the whole execution time about 45% (compared to the best results of seven word experiments). As the clusters represent the basic units of the gestures, it is expected that a restricted vocabulary (7 to 14 words) produces a small number of clusters (16 or 32 clusters) whilst a large vocabulary (more than 100 words) would produce a better distribution of the feature vectors among the clusters and the total of them would be 32 or 64. If we assumed that 64 clusters would be enough to represent a large number of words and that in a worst scenario with another increase of recognition time around 45% we would end up having around 38 ms of total execution time. This would produce a frequency around 26 Hz, which would still be considered real-time (above 24 Hz of the cinema).

VII. Conclusions and Future work

In this paper we detailed our experiments with segmentation and recognition of static and dynamic gestures in real-time using, as use case, the LIBRAS alphabet (for static recognition) and LIBRAS words (for dynamic recognition). For the static gesture recognition, we reported recognition rate of 100% for A, E, I, O, U and B, C, L, F, V using the Aspect Ratio Hand Cropping algorithm along with the Virtual Wall algorithm. The overall execution frequency of the system was 30 Hz limited by the Kinect device frame rate. However, the system is able to reach 62.5Hz.

For the dynamic gesture recognition, we performed different experiments regarding feature vectors and number of clusters with recognition rates as good as 99.82% for 7 words.

Regarding execution time, the system performed at 55.34Hz for 7 words in Libras and 38.10Hz for 14 words. This was due, mainly, to the number of the clusters and the dimensions of the feature vectors (as expected). For large vocabularies we expect the number of clusters to be below 64, which would still deliver real-time performance. It is important to notice that no optimization was performed in the whole recognition process.

As future work we intend to increase the number of static gestures that can be classified at the same time by using invariant shape descriptors, since binary images are not the best option for large datasets. Furthermore, improvements in the cropping hand algorithm are needed. The current version works only when the hands are pointing upwards. In order to solve this problem we intend to use depth information to improve the decision making for cropping. We also need to address the problem of recognizing head movements and face expressions. For the dynamic gesture recognition viewpoint, the evaluation of the HMMs could be optimized by the use of the pruning technique [43], [44]. We also believe that a better algorithm for searching the nearest cluster (such as [45]) would increase the overall performance.

Acknowledgment

The authors would like to thank the Brazilian funding agencies FAPESP and CNPq and the German Funding Agency (DFG) for sponsoring the work.

References

[1] R. Birdwhistell, *Kinesics and Context*, university of pennsylvania press ed., 1970.

[2] M. S. and T. Acharya, "Gesture recognition: A survey," *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 37, no. 3, May 2007.

[3] P. Hong, M. Turk, and T. S. Huang, "Gesture modeling and recognition using finite state machines," in *Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, March 2000, pp. 410–415.

[4] L. F. Brito, *Por uma gramatica de linguas de sinais*. Rio de Janeiro; UFRJ, Departamento de Linguistica e Filologia: Tempo Brasileiro, 1995.

[5] M. Correa, J. Ruiz-del Solar, R. Verschae, J. Lee-Ferng, and N. Castillo, "Real-time hand gesture recognition for human robot interaction," *RoboCup: Robot Soccer World Cup XIII*, pp. 46–57, 2009.

[6] L. Shi, Y. Wang, and J. Li, "A real time vision-based hand gestures recognition system," in *Proceedings of the 5th international conference on Advances in Computation and Intelligence*, 2010, pp. 349–358.

[7] T. Coogan, G. Awad, J. Han, and A. Sutherland, "Real time hand gesture recognition including hand segmentation and tracking," in *Proceedings of the Second international conference on Advances in Visual Computing*, 2006, pp. 495–504, volume I.

[8] Y. Chang, S. Chen, and J. Huang, "A kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities," *Elsevier, Research in Developmental Disabilities, Volume*, vol. 32, no. 6, 2011.

[9] E. S. Santos, E. A. Lamounier, and A. Cardoso, "Interaction in augmented reality environments using kinect," in *XIII Symposium on Virtual Reality (SVR)*. IEEE Computer Society, 2011, pp. 112–121.

[10] Z. Ren, J. Meng, J. Yuan, and Z. Zhang, "Robust hand gesture recognition with kinect sensor," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 759–760.

[11] L. Bretzner, I. Laptev, T. Lindeberg, S. Lenman, and Y. Sundblad, "A prototype system for computer vision based human computer interaction," 2001, technical Report, Department of Numerical Analysis and Computing Science KTH (Royal Institute of Technology), Stockholm, Sweden.

[12] Q. Chen, N. D. Georganas, and E. M. Petriu, "Hand gesture recognition using haar-like features and a stochastic context-free grammar," *IEEE Transactions On Instrumentation And Measurement*, vol. 57, no. 8, 2008.

[13] Y. Fang, K. Wang, J. Cheng, and H. Lu, "A real-time hand gesture recognition method," in *IEEE International Conference on Multimedia and Expo*, 2007.

[14] S. G. Wysoski, M. V. Lamar, S. Kuroyanagi, and A. Iwata, "A rotation invariant approach on static-gesture recognition using boundary histograms and neural networks," in *in Proceedings of the 9th International Conference on Neural Information Processing (ICONIP)*, 2002, pp. 2137–2141.

[15] J. J. Phu and Y. H. Tay, "Computer vision based hand gesture recognition using artificial neural network," in *Proc. Int'l Conf. on Artificial Intelligence in Engineering and Technology (ICAIET'06)*, Kota Kinabalu, November 2006.

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing*, vol. 1, pp. 318–362, 1986.

[17] P. Modler and T. Myatt, "Recognition of separate hand gestures by time-delay neural networks based on multi-state spectral image patterns from cyclic hand movements," in *IEEE International Conference on Systems Man and Cybernetics*, 2008, pp. 1539–1544.

[18] M. Vafadar and A. Behrad, "Human hand gesture recognition using spatio-temporal volumes for human-computer interaction," in *International Symposium on Telecommunications*, August 2008, pp. 713–718.

[19] M. Elmezain, A. Al-hamadi, and B. Michaelis, "Hand gesture recognition based on combined features extraction," *World Academy of Science, Engineering and Technology*, vol. 60, p. 395, December 2009.

[20] E. B. Pizzolato, M. S. Anjo, and G. C. Pedroso, "Automatic recognition of finger spelling for libras based on a two-layer architecture," in *Proceedings of the 2010 ACM Symposium on Applied Computing*. ACM, March 2010, pp. 969–973.

[21] B. Bauer and H. Hienz, "Relevant features for video-based continuous sign language recognition," in *Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, March 2000, pp. 440–445.

[22] Y. Wang and B. Yuan, "A novel approach for human face detection from color images under complex background," *Elsevier Journal of Pattern Recognition*, vol. 34, pp. 1983–1992, 2001.

[23] H. Brashear, V. Henderson, K. Park, H. Hamilton, S. Lee, and T. Starner, "American sign language recognition in game development for deaf children," in *8th International ACM SIGACCESS Conference on Computers and Accessibility*. New York: ACM, 2006, pp. 79–86.

[24] M. Pahlevanzadeh, M. Vafadoost, and M. Shahnazi, "Sign language recognition," in *9th International Symposium on Signal Processing and Its Applications*, 2007, pp. 1–4.

[25] J. C. Principe and B. Vries, "A theory for neural networks with time delay," in *Advances in Neural Information Processing Systems*, R. Lippmann, J. Moody, and D. Touretzky, Eds. San Mateo, CA,: Morgan Kaufmann, 1991.

[26] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Readings in speech recognition*, pp. 267–296, 1989.

[27] A. Bellarbi, S. Benbelkacem, N. Zenati-Henda, and M. Belhocine, "Hand gesture interaction using color-based method for tabletop interfaces," in *IEEE 7th International Symposium on Intelligent Signal Processing*, 2011.

[28] J. B. Cole, D. B. Grimes, and R. P. N. Rao, "Learning full-body motions from monocular vision: Dynamic imitation in a humanoid robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[29] S. Feuerstack, M. S. Anjo, J. Colnago, and E. Pizzolato, "Modeling of user interfaces with state-charts to accelerate test and evaluation of different gesture-based multimodal interactions," *Informatik*, pp. 4–7, October 2011.

[30] S. Feuerstack, M. S. Anjo, and E. Pizzolato, "Model-based design generation and evaluation of a gesture-based user interface navigation control," in *Proceedings of the 10th Brazilian Symposium on Human Factors in Computing Systems and the 5th Latin American Conference on Human-Computer Interaction*, B. C. Society, Ed., Porto de Galinhas, Brazil, October 2011, pp. 227–231.

[31] J. Weissmann and R. Salomon, "Gesture recognition for virtual reality applications using data gloves and neural networks," in *IEEE International Joint Conference on Neural Networks*, July 1999, pp. 2043–2046.

[32] A. Y. Yang, S. Iyengar, S. Sastry, R. Bajcsy, P. Kuryloski, and R. Jafari, *Distributed Segmentation and Classification of Human Actions Using a Wearable Motion Sensor Network*, ieee computer vision and pattern recognition workshops ed., 2008.

[33] R. Yang and M. Pollefeys, "Multi-resolution real-time stereo on commodity graphics hardware," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2003, pp. 211–217.

[34] M. W. Gardnera and S. R. Dorlinga, "Artificial neural networks (the multilayer perceptron) - a review of applications in the atmospheric sciences," *Atmospheric Environment*, vol. 32, no. 14-15, pp. 2627–2636, August 1998.

[35] Y. Hayashi and R. Setiono, "Combining neural network predictions for medical diagnosis," *Computers in Biology and Medicine*, vol. 32, no. 4, pp. 237–246, July 2002.

[36] A. Ciaramella, C. Donalek, A. Staiano, M. Ambrosio, C. Aramo, P. Benvenuti, G. Longo, L. Milano, G. Raiconi, R. Tagliaferri, and A. Volpicelli, *Applications of neural networks in astronomy and astroparticle physics*. Recent Research. Developments in Astronomy and Astrophysics: Research Signpost, 2005.

[37] F. Rosenblatt, "The perceptron - a perceiving and recognizing automaton," Report 85-460-1, Cornell Aeronautical Laboratory, Tech. Rep., 1957.

[38] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, April 1967.

[39] W. Liu and H. Weisheng, "Improved viterbi algorithm in continuous speech recognition," in *International Conference on Computer Application and System Modeling (ICCASM)*, China, October 2010, pp. 207–209.

[40] F. Chang, C. Chen, and C. Lu, "A linear-time component-labeling algorithm using contour tracing technique," *Elsevier Computer Vision and Image Understanding*, vol. 93, no. 2, pp. 206–220, 2004.

[41] T. R. Trigo and S. R. M. Pellegrino, "An analysis of features for hand-gesture classification." in *17th International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2010, pp. 412–415.

[42] S. P. Lloyd, "Least square quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, pp. 129–137, 1982.

[43] D. Willett, C. Neukirchen, and G. Rigoll, "Efficient search with posterior probability estimates in hmm-based speech recognition," in *in Proc. Int. Conf. Acoustics, Speech and Signal Processing*, 1998, pp. 821–824.

[44] A. X. L., "An overview of decoding techniques for large vocabulary continuous speech recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 89–114, 2002.

[45] H. B. Kekre and T. K. Sarode, "Centroid based fast search algorithm for vector quantization," *International Journal of Imaging (IJI)*, vol. 1, no. A08, pp. 73–83, 2008.