Dynamic System Design for American Sign Language Recognition

Joe Naoum-Sawaya, Mazen Slim, Sami Khawam and Mohamad Adnan Al-Alaoui
Electrical and Computer Engineering Department.
American University of Beirut
Beirut, Lebanon
Email: {joe.sawaya@gmail.com, mazenslim@gmail.com, sami.khawam@gmail.com, adnan@aub.edu.lb}

Abstract – In this paper, we present a real time system for recognizing isolated and sentence American Sign Language (ASL) gestures. We mainly focus on image preprocessing to isolate hand pose and motion which allow fast and accurate recognition. Preprocessing is accomplished through a series of adaptive modules forming a very adaptive system. Such a system achieved high accuracy rate even with a noncomplex gesture recognition technique. This evaluation was conducted using a set of 5 gestures and translation in 4 different directions.

Keywords
Sign Language, Static and Dynamic Gesture Recognition, Motion Analysis, Pattern Analysis, Template Matching, Background Rejection, Tracking.

1. INTRODUCTION

The concept of “Smart Homes” has become the ultimate goal of Machine Intelligence Engineering. The trend of drifting away from traditional to modern architecture design is building more and more momentum, thus requiring smoother Human-Machine Interaction (HMI). American Sign Language (ASL) presents an ordered set with pre-assigned meanings and context structures. Most gestures or signs in ASL are mapped to a single word (command) in the English language. Further emotions may be expressed in ASL through Head and Eyes motion. These features of ASL make it an ultimate solution for controlling home appliances through body language. In the presented system we will focus on hand motion rather than facial feature tracking however the adaptive modules presented in this paper may be well integrated in facial tracking systems [1]-[3] to enhance its accuracy.

1.1 Overview

Earlier researches in sign language recognition employed template matching techniques to identify the different gestures. This technique relied on comparing the executed gesture to a set of stored gestures forming the gestures database. However these systems lacked precision due to the limited degree of freedom between the different hand signs. This drawback resulted in common incorrect decisions since most hand signs are very similar thus not allowing the system to differentiate between them. More recent researches shifted towards the use of Hidden Markov Models (HMM). In HMM, each hand gesture is a sequence of static hand poses with occurrence probabilities. Even though the degree of freedom drawback previously discussed is reduced however it is not eliminated. Anticipating for the different hand poses would create an infinite number of sequences. Creating an adaptive system that adapts and learns from environment conditions as well as degrees of freedom of the motion will highly increase the performance of these recognition systems. We demonstrate this performance increase by implementing a template matching recognition module. Results are very promising.

1.2 System setup

This paper presents the various techniques used to implement the system on a Pentium-based machine. Gestures are inputted to the system via a USB webcam. Histogram equalization is applied to the grabbed frames to emphasize the contrast. Background pixels are filtered from the grabbed frames in order to extract an optimal skin color range. To further enhance the processed frames, a morphological filter introduces a series of Erosion and Dilation [4], [5] hence fills small background blobs. Larger blobs are filled through a flood filling technique. The resultant image is smoothed by passing it through a series of down-sampling and up-sampling processes. At this stage, the resulting image will only contain the gesture. All other pixels are set to black. To ease processing at further modules, the image size is reduced to limit the region of interest to the gesture area. These methods are discussed in Section 2.

Section 3 discusses the recognition modules. The gesture is recognized through a template matching method that compares the grabbed gesture to a set of predefined gestures previously stored in a database. At this stage static hand gestures are recognized [5]. The window gesture is tracked using CAMSHIFT algorithm [6], [7]. The window tracking combined with the previously detected gesture are used in the detection of Dynamic Gestures. The detected gesture is also used to enhance the stored templates through weighing the detected gesture and the stored templates.

Section 4 of this paper reveals the results achieved through our implementation and describes possible applications. Section 5 contains the conclusion of our work and the suggestion of possible future enhancements.
2. GESTURE EXTRACTION

2.1 Histogram Equalization

Histogram Equalization is applied to grabbed frames in order to emphasize the gesture shape. Pixel colors are computed according Equation 1.

\[ j \rightarrow k = \sum_{i=0}^{N} \frac{N_i}{N_{total}} \times C_{max} \]  

(1)

Images grabbed by the described system are 3 channels images\(^1\). Hence equalization is performed on the 3 channels, each on its own (Figure 1).

![Figure 1: (a) Original Image; (b) Equalized Image](image1)

2.2 Background Rejection

Initially, the hand is detected by performing motion detection over a static background. The background characteristics for each RGB channel of each pixel are calculated over a number N of frames. The mean and standard deviation are respectively calculated as shown in Equation 2.

\[ m(x,y) = \frac{S(x,y)}{N} \]

\[ \sigma(x,y) = \sqrt{\frac{S_{q}(x,y)}{N} - \left( \frac{S(x,y)}{N} \right)^2} \]  

(2)

where \( S(x,y) \) is the sum of the values of the pixel over N frames for each channel, and \( S_q(x,y) \) is the sum of squares. Subsequently, motion is detected in any pixel \( p(x,y) \) that satisfies, in any of the three channels (Figure 2) the condition \( |m(x,y) - p(x,y)| > c \sigma(x,y) \). The value of c was taken to be 3\(^2\).

![Figure 2: (a) Original Image; (b) Image after Background Rejection](image2)

2.3 Skin Color Extraction and Thresholding

After background rejection, the system waits for a major change in pixel values within a rectangle (80x60) in the vicinity of the center. Once that happens, the pixels of that rectangular area are surveyed. The hue angle domain, from 0 to 180 degrees, is divided into 18 intervals of 10 degrees span each. The number of pixels with hue values belonging to each interval are counted and ranked. If the three intervals with the highest scores are adjacent, the skin color hue value is set to the center of the middle interval with a tolerance of 15 degrees. If only two intervals with the highest scores are adjacent, the skin color hue value is set to the boundary value between the two with a tolerance of 10 degrees. Otherwise, the skin color hue value is set to the center of the interval with the highest score with a tolerance of 15 degrees. Thresholding (or color segmentation) then takes care of transforming the 3-channel to a binary image: pixels with values belonging to the skin color range are changed to white while pixels outside this range are set to black [4, 5] (Figure 3).

![Figure 3: Thresholding](image3)

2.4 Morphological Filter

Two morphological filters are used in order to enhance the image obtained after thresholding: erosion and dilation [5]. A sequence of erosions and dilations is performed to get the best possible image (Figure 4).

![Figure 4: Template after Morphological Filters](image4)

2.5 Flood Filling

The flood filling technique finds contours of connected threshold pixels and then fills the area encircled by the contours (Figure 5).

![Figure 5: Template after Flood Filling](image5)

2.6 Smoothing Filter

At this stage the system analyzes a binary image with only black and white colors. Since the shape rather than color matters in our system, hand contours should ideally be continuous in order to enhance the recognition process. The image is passed through a smoothing filter that better shapes the contour. This filter is a 2D Gaussian filter of the form shown in Equation (3). The resulting image is shown in Figure 8.

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

(3)

![Figure 6: Continuous Gaussian Distribution](image6)

![Figure 7: Discrete Gaussian Distribution](image7)

---

\(^1\) RGB color space consists of three components R, G, and B that reflect the intensity of Red, Green, and Blue in a pixel.

\(^2\) The well-known “three sigmas” rule
3. GESTURE RECOGNITION

3.1 CAMSHIFT Algorithm

The CAMSHIFT (Continuously Adaptive MeanSHIFT) [6] algorithm is used in our system to draw a window around the hand to simplify template matching. It is also used for tracking in dynamic gesture recognition [7]. An advantage in the CAMSHIFT algorithm is that a window fitting the gesture is drawn around it. The window size is adjusted so as to fit the gesture area hence it adapts itself to the size of the gesture reflected by any variation in the distance between the camera and the hand. With this feature our system will be transparent to the distance separating the camera from the hand.

In our system we are applying the CAMSHIFT algorithm in a special way. The tracking is carried out on the white pixels only (i.e. after thresholding), leading to a more accurate computation of the dimensions of the window.

In our system we are applying the CAMSHIFT algorithm in a special way. The tracking is carried out on the white pixels only (i.e. after thresholding), leading to a more accurate computation of the dimensions of the window.

3.2 Template Matching

At this stage the image is a 1 channel 8 bit image. Each pixel has a color value in the range [0,255]. Two levels thresholding is applied on the image. Pixels in the range [0, 85] are pulled to 0, those in the range [85, 170] are pulled to 1, [170, 255] are pulled to 2. After the 2 levels thresholding each pixel will have a value of 0, 1, or 2. The image is compared to a set of templates previously stored templates. A similarity metric between the image and the template is calculated according to Equation 4.

\[ S(u, v, \theta) = \frac{\sum_{x=0}^{W-1} \sum_{y=0}^{H-1} \left[ 2 - f(x, y) - T(\chi \cos \alpha, \gamma \sin \alpha) \right]}{2XY} \]

\[ \chi = u + x \]

\[ \gamma = v + y \]

\[ \alpha = \theta + \tan^{-1} \frac{y}{x} \]

The image and the template should be of the same dimensions; hence the image in the window estimated by the CAMSHIFT algorithm is resized so as to fit the sizes of the templates.

The proposed template matching technique accounts for image translation as well as image rotation. Image translation is accounted through the (u, v) coordinates while the rotation is accounted through the translation angle (\( \Theta \)). Therefore, the grabbed image is compared to an array of rotated and translated templates and the maximum metric is taken as the matching metric. The gesture is recognized by the template with the highest matching metric.

3.3 Static Gesture Recognition

Static gestures are recognized by applying the previously discussed template matching technique. The content of the window tracked by the CAMSHIFT algorithm is passed to the template matching module. The output of this module is mapped to the corresponding gesture. This type of recognition does take into consideration neither the previously recognized gestures nor the position of the window.

3.4 Dynamic Gesture Recognition

This feature builds on the static gesture recognition feature and the CAMSHIFT algorithm discussed above. It mainly calculates the slope of the line linking the centers of two consecutive windows and stores these values in addition to the guessed static gesture at each of these moments. Each of the dynamic gestures is coded as a series of directions and static gestures, for instance, the code UpLeft/Thumb-UpLeft/Thumb-Still/Thumb could represent a certain task such as Raise Volume. Depending on the dynamic gestures code, each static “sub-gesture” detected gives a hint about the following gesture.

3.5 Template Adaptation

An adaptation is implemented algorithm to enhance the previously stored templates. These templates are modified taking into account the recognized gesture image as well as the corresponding gesture template as shown in Equation 5.

\[ T_{new}(x, y) = \lambda S(x, y) + (1 - \lambda)T_{old}(x, y) \]

Modifying the templates by the stated methods allows more adaptation of the system to environment conditions. This learning process also allows the system to enhance detection by increasing the dissimilarities between templates so as to better meet the recognized gestures (Figure 10).

4. IMPLEMENTATION AND APPLICATION

4.2. Performance Evaluation

The system was evaluated in different lighting and background conditions. The system revealed a remarkable adaptation to changing lighting conditions. This remarkable performance is due to the weight given to image enhancements before the gesture recognition process. The system was also evaluated in different noisy backgrounds. The system revealed remarkable performance in eliminating background objects form all grabbed pictures. The CAMSHIFT tracking system was
also evaluated in different conditions. The tracking system performed remarkably in different lighting and background conditions. Even for extremely fast motion the system correctly traced the trajectory and took correct decision. The proposed template matching technique was also evaluated in varying lighting and background conditions. Results revealed remarkable recognition abilities reaching 96% in daylight conditions and distinct backgrounds. This evaluation was conducted using a set of 5 gestures and translation in 4 directions and for a sample size of 300 random gestures. The proposed adaptation technique allows system adaptation, thus increasing decisions’ certainty (Table I).

Table I: Unknown Gesture Recognition

<table>
<thead>
<tr>
<th>Unknown Object</th>
<th>Template</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>![Thumb]</td>
<td>0.786247</td>
</tr>
<tr>
<td></td>
<td>![Hand]</td>
<td>0.877251</td>
</tr>
<tr>
<td></td>
<td>![Fist]</td>
<td>0.760113</td>
</tr>
</tbody>
</table>

Table II demonstrates how the proposed template adaptation enhances initial templates. This enhancement is revealed in the similarity metric that increases after each gesture-recognition. In the long run however, extensive incorrect decisions might lead the templates to diverge from the correct ones. This motivates varying λ in Equation 5, hence a lower weight is given to the original image in clear environments (system decisions are more trustful) and higher values in more vague environments.

Table II: The Effect of Template Adaptation on the Similarity Metric

<table>
<thead>
<tr>
<th>Unknown Object</th>
<th>Template</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>![Thumb]</td>
<td>0.887676</td>
</tr>
<tr>
<td></td>
<td>![Hand]</td>
<td>0.915506</td>
</tr>
<tr>
<td></td>
<td>![Fist]</td>
<td>0.930765</td>
</tr>
</tbody>
</table>

4.3 Application

As a part of this research, a TV control system was developed to demonstrate how such a system can be integrated into devices. Such an implementation can be a first step towards huge business applications which would constitute the foundation of modern home appliances and equipment.

5. CONCLUSION AND FUTURE WORK

The system discussed in this paper would have smoothed Human-Machine Interaction in as far as it does not involve any use for artificial tools. It is totally based on the user’s body organ (the hand) and can be further exploited in expressing one’s mood, which makes it suitable for use in future Smart-Homes.

Furthermore, such a dynamic system may be integrated as a part of other image processing systems to enhance decision making.

REFERENCES


