

# Low Resource Lip Finding and Tracking Algorithm for Embedded Devices

Jesús F. Guitarte Pérez<sup>1</sup>, Klaus Lukas<sup>1</sup>, and Alejandro F. Frangi<sup>2</sup>

<sup>1</sup> Siemens AG, Corporate Technology, Otto-Hahn-Ring 6, Munich, Germany.

<sup>2</sup> Computer Vision Group, Aragon Institute of Engineering Research, University of Zaragoza, María de Luna 1, Zaragoza, Spain.

jguitarte@yahoo.es, lukas@siemens.com, afrangi@unizar.es

## Abstract

*One of the best challenges in Lip Reading is to apply this technology to an embedded device. In the current solutions the high use of resources, especially in reference to visual processing, makes the implementation in a small device very difficult.*

*In this article a new, efficient and straightforward algorithm for detection and tracking of lips is presented. Lip Finding and Tracking is the first step in visual processing for Lip Reading. In our approach the Lip Finding is performed between a small amount of blobs, which should fulfill a geometric restriction. In terms of computational power and memory the proposed algorithm meets the requirements of an embedded device; on average less than 4 MHz<sup>1</sup> of CPU is required. This algorithm shows promising results in a realistic environment accomplishing successful lip finding and tracking in 94.2% of more than 4900 image frames.*

## 1. Introduction

Automatic Speech Recognition (ASR) will play an important role in future embedded devices, such as mobile phones, PDAs and also in car environments.

One of the most important problems of ASR is the degradation of the acoustic signal, which usually induces a significant recognition rate decrease. Several techniques, such as, for example, Noise Compensation based on acoustic signal processing [1,2] have been developed to improve the robustness of ASR when the acoustic signal is corrupted. Other solutions use visual information, as it will be available in future devices, e.g. by using the camera of 3G mobile devices. This technique is called “Lip Reading” and it exploits the additional information that can be found in the lips movement during speech [3]. This visual information can be combined with the acoustic in-

formation in order to improve the recognition rate [4]. Compared to conventional acoustic recognition, the combined system can achieve a 55% error rate reduction for various signal/noise conditions [5]. This technique can also be used in combination with Noise Compensation Algorithms [6].

Lip Reading systems have been developed in several laboratories and research projects but without taking special care on embedment restrictions. In this article we describe an embeddable algorithm for the first stage of most Lip Reading systems: Lip Finding and Tracking.

We propose a system that finds the position of the lips: Detection and Tracking of the Region of Interest (ROI). Several approaches can be found in the literature to this end; some of them are very robust and can work under complex lightning conditions. However, they usually require many computing and storage resources like, e.g., those based on large Neural Networks [5]. Other solutions work in Real-Time but only in desktop workstations where resource availability is less restrictive than in a small device. Yang et al. [7] proposed a top-down approach, which works by first finding the face using color information. Subsequently, for every frame, a set of facial features is extracted (eyes, nostrils and lip corners). In contrast, our approach extracts a smaller set of features that do not require a top-down search. The eyebrows and lips are searched only when there is no information about the position of the lip in the last frame. Otherwise, only a lip tracking is performed.

The aim of this paper is to describe a system that can automatically perform the Lip Finding and Tracking. The system must be able to work without special light conditions as well as without any kind of reflected markers or special make up placed on speaker’s lips.

This paper is organized as follows. In Section 2, the Lip Finding and Tracking System is described. In Section 3, the requirements for implementing the system in a commercial embedded system are introduced. Section 4 presents the experimental results of the evaluation of the system. Finally, Section 5 provides some conclusions.

<sup>1</sup> ARM920T: 150 MHz, 16Kbyte bi-directional cache. External Memory Access Speed: 150 nsec. for non sequential and 10 nsec. for sequential access.

## 2. Lip Finding and Tracking System

A comment regarding the *Lip Finding* and the *Lip Tracking* algorithms should be made. *Lip Finding* is applied when no previous information of the lip position is available. This happens in the first frame of a sequence or whenever the lips cannot be correctly located in the previous frame. *Lip Tracking* proceeds when knowledge on the position of the lips in the previous frame is available. This information can be used to update the lips' coordinates by inspecting a region close to the last position rather than in the whole image. Finally, *Feature Extraction* gives a criterion to verify whether the lips have been found or not.

### 2.1 Lip Finding Algorithm

This algorithm is based on a geometric model of the face. Structures of pixels are evaluated in order to know if their relative positions match a simplified prior model of the face. In particular, this model accounts only for the relationships between location of the eyebrow(s) and the mouth. This algorithm is composed by *directional filtering*, *segmentation* and a *searching and matching process*.

#### 2.1.1 Directional Filtering and segmentation

In order to save memory no color information is used; only the luminance (Y) component from the YUV color space will be taken into account (see Figure 1.a.) This grayscale image is filtered by a horizontal filter. After filtering, image thresholding takes place, where the threshold could be fixed or adapted to the variance of the filtered image. A binary image is obtained where all pixels which belongs to a contour with a horizontal component are set to "one" (white in Figure 1.b.)

A segmented structure is obtained from the run-length coding. A blob is defined as a group of pixels connected according to a specific neighborhood relationship and sharing a common characteristic [8]. In this case the shared characteristic is to belong to the same horizontal contour.

Each blob is described only by the area and the coordinates of its centre. Blobs are filtered according to their area; therefore very small or very big blobs will be disregarded. This will only imply that our algorithm will work for a limited range of distances between the camera and speaker. In our applications either the speaker holds the device in his hands or the camera is located at a fixed distance on the dashboard (car environment). The remainder blobs after filtering are showed in Figure 1.c.

#### 2.1.2 Search and Matching Process

The search process is performed by only taking into account those blobs whose positions make them likely to belong to parts of the face. Only approximately 10-25 blobs per frame are left to take part in the search. The algorithm takes the set of blobs that best matches the prior model of a face. The center of the ideal mouth  $\underline{C}_{id} = f\{\underline{C}(e_r), \underline{C}(e_l)\}$  is computed from the centers of mass of the detected eyebrows. This location is subsequently compared to the position of the nearest blob  $\underline{C}(m)$ , and the distance between both centers is considered as a measure of the resemblance with the face model. The set of three blobs that minimizes this distance is considered as the detected face. When this measure exceeds a certain value the algorithm assumes that no face has been found. This search process is shown in Figure 1.d.

Let  $\underline{C}(e_r)$ ,  $\underline{C}(e_l)$ , and  $\underline{C}(m)$  be the coordinates of the blob centers representing the right eyebrow, the left eyebrow, and the mouth, respectively. The objective is to find the three blobs  $e_r, e_l, m$  that minimize the distance:

$$dist\{\underline{C}_{id}, \underline{C}(m)\}_{e_r, e_l, m}$$

where:

$$\underline{C}_{id} = \begin{cases} C_{id\_x} = \max\{C_x(e_r), C_x(e_l)\} - 0.5 * abs\{C_x(e_r) - C_x(e_l)\} \\ \quad + K * \{C_y(e_r) - C_y(e_l)\} \\ C_{id\_y} = \max\{C_y(e_r), C_y(e_l)\} - 0.5 * abs\{C_y(e_r) - C_y(e_l)\} \\ \quad + K * \{C_x(e_r) - C_x(e_l)\} \end{cases}$$

and where  $K$  is a geometrical aspect ratio obtained empirically from a large face database. It is defined, as shown in Figure 1.d, as  $K = b/a = 1.2$ .

Problems could arise with people with very bushy eyebrows or with glasses. In these cases only one single segment would be recognized as an eyebrow. This situation was taken into account and if there are no segments that satisfy the condition of the face according to the first search algorithm, a second model with a single large eyebrow is assumed and the process is repeated.



Fig. 1.a



Fig. 1.b

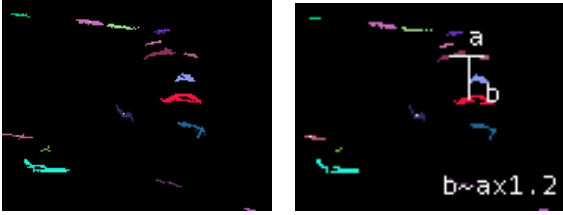


Fig. 1.c

Fig. 1.d

Figure 1: Stages of the Lip Finding algorithm. a) Grayscale image, b) horizontal filtering and thresholding, c) Segments, and d) Search process.

## 2.2 Lip Tracking Algorithm

*Lip Tracking* is applied only when in the last image the lips were found. In this case, we rely on the hypothesis that the position of the lips will not be very different between one frame and the next one. Also, a movement vector can be used to give a better approximation between frames. Lips will be searched in an area that is 10% larger than the region where the lips were located in the previous frame.

## 2.3 Feature Extraction

This part of the algorithm receives the position where the lips are supposed to be located from *Lip Finding* or *Lip Tracking*. It finds the features that describe the lips. If the typical features of the lips are found, the verification is completed and the lip coordinates are updated.

In the *Feature Extraction* implemented for this work, the upper and lower lip contours are sought. First, the upper lip contour is obtained with a gradient filter that highlights bright-to-dark intensity changes (from top to bottom in the image). Then another dark-to-bright filter is applied to obtain the lower contour of the lips. If both segments have the properties of a lip, the verification will be completed, and the lip position is updated. In this implementation, we know that the lips are correctly located using the relative position between the two lips and their geometrical properties.

## 3. Requirements

Since the algorithm is intended for integration in an embedded device, it is important to restrict the resources that can be consumed. The *Lip Finding* algorithm saves an important amount of resources by performing the search only between a small amount of blobs as indicated in Section 2.1.2. Extra savings have been accomplished since the *Lip Tracking* process can be applied most of the time in comparison to *Lip Finding*. The former process searches in a small region of approximately 5% of the area of an image.

The fulfillment of the requirements is tested with an emulator for the ARM920T<sup>1</sup>  $\mu$ C, an exemplary

microprocessor suitable for Third Generation of Mobile Devices (UMTS). Table 1 lists the algorithm demands

|              | CPU (MHz) |
|--------------|-----------|
| Lip Finding  | 40 MHz    |
| Lip Tracking | 1.8 MHz   |

Table 1: Demands of Lip Finding and Tracking.

In the requirements tests a frame rate of 15 f.p.s. has been used. It must be taken into account that, as long as the lips are being found, only *Lip Tracking* is applied, so *Lip Finding* is used only when the lips were not found in the previous image and they must be searched within the whole image.

In the sequences used to test the system the percentage of images where *Lip Finding* was applied was only 5%, so we would obtain a mean CPU use of 3.71 MHz. The Code Memory consumption is about 9 Kbytes without consideration of the linked C standard libraries that may be shared by several software modules and therefore do not increase the memory consumption.

These results were obtained without any kind of platform-specific or assembler optimization of the algorithm. Even in this situation the current software module is compliant with the available resources in an embedded device.

## 4. Results

The algorithm was evaluated verifying results by the visual inspection. A set of 33 speakers of both sexes, with ages between 20 and 60 years, with different skin colors and with different grades of facial hair have tested the system in sessions of 10 s each (150 frames each speaker). No special light conditions have been used and no special make up or reflected markers were placed on the speaker's lips. The distance between the camera and the speaker was between 10 and 70 cm, and people were asked to look at the camera. From the total of 4950 images using the Lip Finding and Tracking algorithm the error rate was 5.8%. We have also obtained the results using only Lip Finding without Tracking, so without using the information about the previous frame; in this situation the *error rate* raises up to 27.4%, see Figure 5. In general, people where the algorithm failed have difficult lip contrast due to very dark skin color or a very bushy beard.

We have classified the error frames on one hand as *false alarm*, when our system has found a structure that does not match the lips, and on the other hand as *not found*, when the system knows that it cannot find the lips in the image and gives therefore no output. The percentage of errors classified like *false alarm*

for the Lip Finding algorithm is 39.2 % (of the total number of errors) and this value is increased to 47.2% when Tracking is applied, since some erroneous frames are propagated with the tracking system.

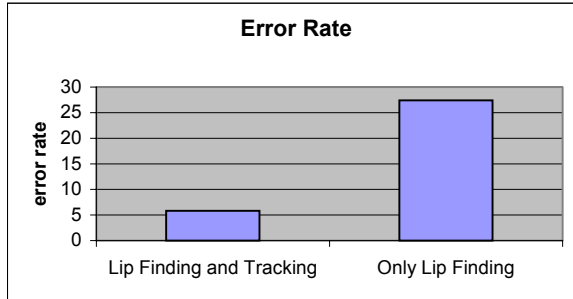


Figure 5: Error rate

In addition, to give a better criterion of the generalization ability of the results when this algorithm is applied to different users, it is important to know how the erroneous frames are distributed over the different speakers. Figure 6 shows that the 78.8% of the speakers have an error rate smaller than 5%.

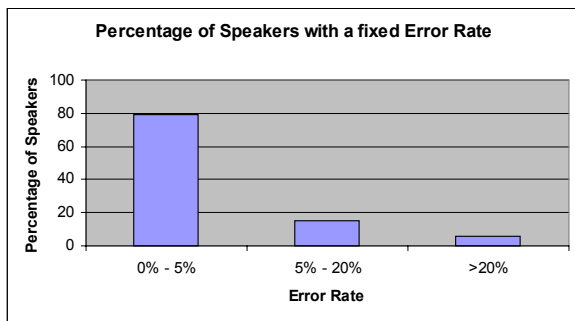


Figure 6: Percentage of speakers vs. error rate

### 5. Conclusion

An efficient algorithm to find the lips has been presented. It has good performance in a realistic environment, with a 5.8% error rate. Since it consumes very few resources it can be implemented in an embedded device. The error rate is lower when tracking is used, but if we want to reduce the false alarm and the burst of errors length a better Feature Extraction must be used in order to guarantee a good tracking. It is also necessary for Lip Reading to provide a set of characteristics useful for recognition; ASM (Active Shape Models) is one of the best chances. False alarm and the error rate could also be reduced by using first a color classification algorithm. The idea is to apply the algorithm explained in this paper only in the regions where the color is similar to that of the skin. Some pilot tests were already performed along this

lines and important improvements were found especially in backgrounds full of horizontal structures like plants' leaves.

### 6. Acknowledgement

The authors wish to express special thanks to Markus Simon, Ewald Frensch, Nikos Paragios, Visvanathan Ramesh and Eduardo Lleida for their good advises and interesting discussions. Siemens AG has supported the work of J. F. Guitarte under a PhD. contract. The work of A. F. Frangi was supported by Grants TIC2002-04495-C02 and FIT-070000-2002-935 of Spanish Ministry of Science and Technology under a Ramón y Cajal Research Fellowship.

### 7. References

- [1] R. Singh, R. M. Stern, and B. Raj, "Signal and Feature Compensation Methods for Robust Speech Recognition," *CRC Press LLC*, pp. 219-243, 2002.
- [2] ETSI ES 202 050 V.1.1.1, "Speech Processing Transmission and Quality aspects (STQ); Distributed Speech Recognition; Advanced front-end feature extraction algorithm; Compression algorithms," *European Telecommunications Standards Institute*, October 2002.
- [3] McGurk H., and MacDonald J., "Hearing lips and seeing voices," *Nature*, 264, pp. 746-748, December 1976.
- [4] Petajan E. D., "Automatic lipreading to enhance speech recognition," *Proc. IEEE Computer Vision and Pattern Recognition*, pp. 44-47, 1985.
- [5] U. Meier, R. Stiefelhagen, J. Yang, and A. Waibel, "Toward Unrestricted Lip Reading," *International Journal of pattern Recognition and Artificial Intelligence*, Vol. 14, No. 5, pp. 571-785, 2000.
- [6] S. J. Cox, I. A. Matthews, and J. A. Bangham, "Combining noise compensation with visual information in speech recognition," *Proc. ESCA/ESCOP Audio-Visual Speech Processing*, pp. 53-56, 1997.
- [7] J. Yang, R. Stiefelhagen, U. Meier and A. Waibel, "Real-time Face and Facial Feature Tracking and Applications," *Proc. Audio-Visual Speech Processing*, pp. 79-84, 1998.
- [8] R. C. Gonzalez, and R. E. Woods, "Digital image processing," *Prentice Hall*, 2001.