Multimodal Caricatural Mirror

Martin O.[(1)]*, Adell J.*[(2)]*, Huerta A.*[(3)]*, Kotsia I.*[(4)]*, Savran A.*[(5)]*, Sebbe R.*[(6)]

(1) : Université catholique de Louvain, Belgium
(2) Universitat Polytecnica de Barcelona, Spain
(3) Universidad Polytècnica de Madrid, Spain
(4) Aristotle University of Thessaloniki, Greece
(5) Bogazici University, Turkey
(6) Faculté Polytechnique de Mons, Belgium

***Abstract*—This project aims at creating a multimodal 'caricatural' mirror, where users see and hear their own emotions amplified by an avatar, mimicking the user's facial expressions and prosody using a wide screen and loudspeakers. The goal of the project is also to bring together researchers from various fields so as to build a whole system using everyone's expertise. The main technical challenges include facial animation, automatic face tracking, automatic vocal and facial features extraction and multimodal emotion recognition and synthesis.**

***Index Terms*—Emotion Recognition, Face analysis, Prosody analysis, Facial Animation, Prosody synthesis, Multimodal Interfaces**

I. Introduction

Researches on automatic emotion recognition and synthesis is currently focusing the attention of an ever-growing community of researchers from various fields (signal processing, artificial intelligence, psychologists, human-computer interactions, …). Many different prototypes of emotion recognition systems have already been developed but it remains very difficult to compare the results of such systems, due to the lack of common databases and experimenting protocols.

In this project, we will then focus on the development of a system that would involve the user both interactively and emotionally, giving the opportunity to users to assess the usability of such a system, while at the same time generating real emotional experiences on the user's side. This affective response from the user will be used to further train the system, the emotions generated being expressed in a more natural way than when expressed 'on demand', as it is the case for most of the existing databases. We could then view the system as a way of building a multimodal database, which could later be used by the researchers of the SIMILAR network of excellence to compare the performances of their emotion recognition algorithms.

This paper will be divided in three major sections. The first section describes the visual modality, which involves both the analysis and synthesis of facial expressions. The second section will deal with the automatic analysis of user's prosody and the exaggeration of the extracted prosodic features, in order to 'caricaturate' the user's voice. Eventually, the last section concerns the integration of both modalities for synchronized and realistic multimodal facial animation.

II. Visual modality

In this section, we will describe the technical challenges related to automatic analysis and synthesis of facial expressions. In the first time, we will describe the steps that have to be achieved to capture the user's emotional state. We will finish our description of the visual modality by presenting the techniques involved in the generation of realistic facial animation.

When tackling the problem of automatic analysis of facial expressions, one generally decomposes it in three sub-problems:

- Automatic Face Detection and Tracking
- Automatic Facial Features Detection and Tracking
- Automatic Expression Recognition/Classification

We will then successively detail each of these challenges, along with the solutions we implemented.

A. Face Detection

The first thing to do when one wants to design a facial expression recognition system is to select the experimental conditions under which experiments will have to be run. In our case, as we want the system to be fully automatic, we have to start by detecting the user's face inside the scene.

Although it seems like an easy problem at first glance, we quickly realized that the high variability in the types of faces encountered makes the automatic detection of the face a tricky problem. After discussing within the team the techniques that could be used for face detection and tracking, we searched in the literature for existing prototypes.

Many different techniques have been tried in order to solve the problem of detecting a face in a scene. After a deep inspection of the state-of-the-art, it appears that there isn't a unique solution to the problem. Rather, the best face trackers have been obtained by using a combination of the available techniques. A technique formally known as *boosting* seems to suit particularly well our needs: the joint use of several weak detectors (classifiers which are not able to precisely detect a face in a scene) may lead to a robust face detector. The robustness is achieved by exploiting the independence between the criteria used by the different individual detectors.
To understand how the *boosting* is achieved, we need to detail a bit the construction of our face detector. First, each of the individual detectors D are trained over the entire training set, producing a recognition rate R_D . Then, we assign a score S_D to each of the D classifier, according to its recognition rate R_D. The face detector then balance the influence of each of the individual detectors on the final decision, according to their relative individual performances.

In the scope of this project, we chose to use an open-source implementation of such a boosted face tracker. After comparing existing systems, we decided to use the OpenCV [1] face tracker. Already trained on a large database of face/non-face images, it produces efficient face detection in all kinds of settings, thus completely fitting our needs. The result obtained with this face tracker is depicted in the figure below.

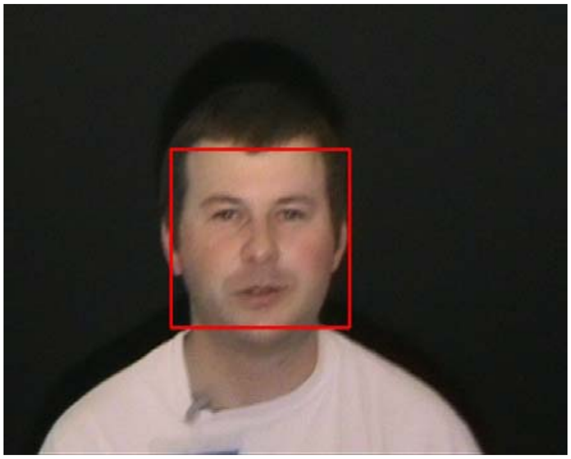

Figure 1 : Output of the OpenCV face tracker

In practice, the face tracker finds the face localization on the first image of the video sequence and surrounds it with a bounding box. The resulting face image is then passed to the facial features detection module whose goal is to initialize the facial features tracker.

B. Facial Features Detection

The main goal of the facial feature detection algorithm is to specify the exact position for a set of desired points on the image. These points will be later used for the extraction of the facial features position. As crucial points are considered the points that correspond to the inner and outer side of the eyebrows and eyes, as well as the corners of the mouth, thus forming a set of 10 points in total.

The method proposed by Sobottka et al. [2], was followed to extract the position of the desired points in the image. Facial features consist of a facial region of particular interest, as they differ from the rest of the face due to their low brightness. Therefore, an analysis of the local minima of the brightness values can specify the correct position for the facial features that are included in the region under examination.

The image is converted in black and white format, to make the whole procedure more robust. Let $\{x,y\}$ be the coordinates of each facial feature under examination. The image is divided in columns, forming in that way three equal regions. The interval in which the y coordinate of the eyes region belongs, can be obtained by examining the pixels' brightness at the columns that correspond to the one third and two thirds of the image's width (left and right eye region respectively). Following the same method, the interval in which the y coordinate of the mouth region belongs, can be obtained by examining the column that corresponds to the middle of the image's width.

The graphic plot of the pixels' brightness at those specific columns indicates the position of local minima in the image. Those local minima correspond to the eyebrows and eyes region for the case of the column at the 1/3 and 2/3 of the image's width, and to the mouth area for the case of the 1/2 of the image width. The local minima actually appear in a sequential way, just like the way the facial features appear in a facial image if that is analyzed from the top to the bottom.

At the images below, someone can actually see the way the eyebrows, eyes and mouth regions are defined by the lower values of their brightness.

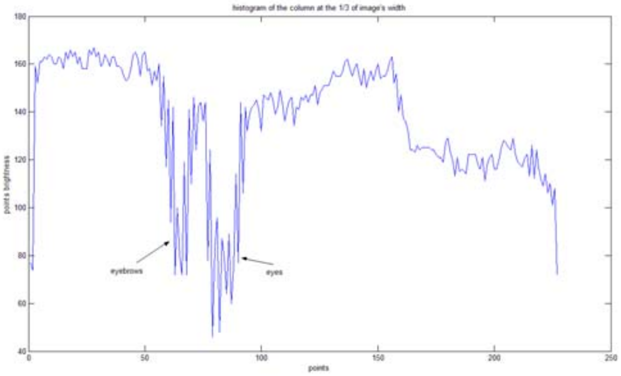

Figure 2: Pixels brightness at the column corresponding to the 1/3 of the image width

The same procedure is followed to define the *x* coordinates of the facial features region in the image, with pixel brightness values taken now for specific rows (1/3 of the image's height for the eyes region and 2/3 of the image's height fot the mouth region).

In that way, three big regions are defined, one for each eye as well as the mouth (see Figure 5 below).

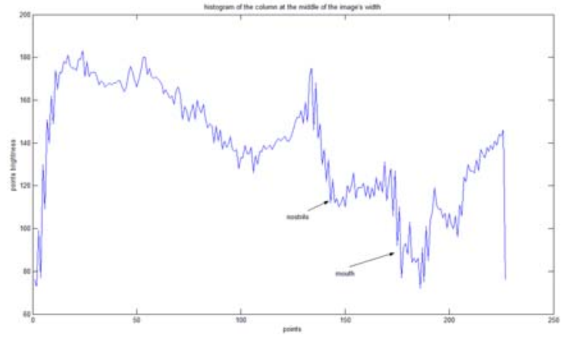

Figure 3: Pixels brightness at the column corresponding to the 1/2 of the image width

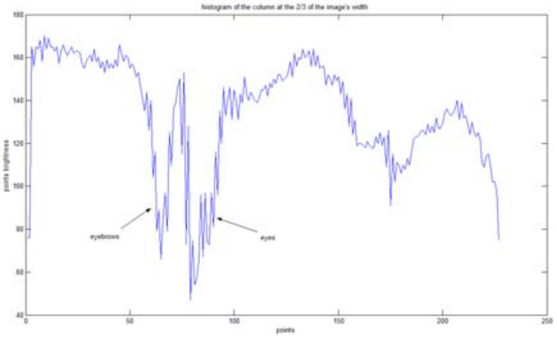

Figure 4: Pixels brightness at the column corresponding to the 2/3 of the image width

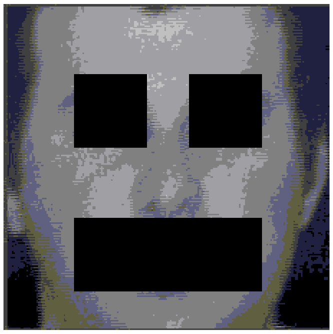

Figure 5: Facial features regions detected in the first frame of the image sequence

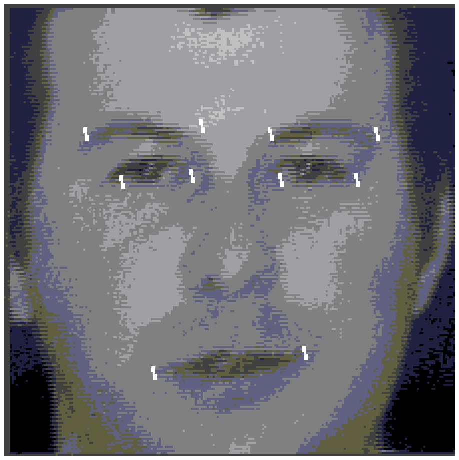

Figure 6: Facial features detected in the first frame of the image sequence

In the above-mentioned regions, every desired facial feature is uniquely characterized. For example, the outer corner of the left eyebrow is the first dark pixel at the top up and left corner, while the inner corner of the eyebrow is the first dark pixel at the top up and right corner. The eyes are more difficult to detect, but if the eyebrow region is defined, the follow similar rules apply for the region left unused. That way all of the desired facial features are detected, so as to be used for the initialization of the Candide grid to the first frame of the image sequence, as described in the following section (Figure 6).

C. *Facial Features Real-Time Tracking*

The facial features information extraction is performed by a grid adaptation system, based on deformable grid models.

The algorithm is based on tracking a large number of previously selected feature points in the facial region as depicted at the first frame of the video sequence. The video sequence progresses in time, depicting the facial expression evolving, to reach its highest intensity at the last frame of the video sequence. The nodes of the fitted Candide grid (output of initialization process) are tracked using a pyramidal implementation of the well-known Kanade-Lucas-Tomasi (KLT) algorithm [3]. As soon as the tracking algorithm computes the displacement of all the tracked features, the resulting configuration (containing the new positions of the model nodes) is deformed. The displacements of model nodes, are assumed to be the driving forces of the model deformation, thereby providing an accurate and robust model based facial feature tracking method.

The system can be seen in Figure 7.

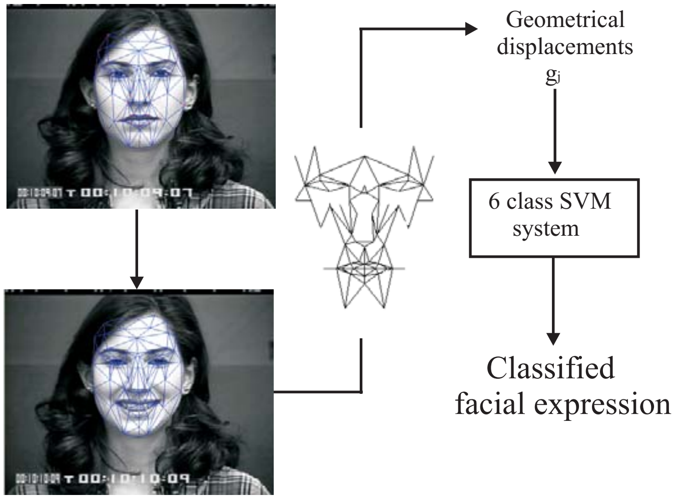
Geometrical displacements g_j
6 class SVM system
Classified facial expression

Figure 7: Diagram flow of the system used for facial expression classification

The Candide grid is automatically initialized on the first frame of the image sequence, depicting a face in its neutral state. To achieve that, the points detected following the procedure described at the previous section are used. The points chosen for the initialization were those of the greatest importance, being the ones responsible for the formation of facial movement according to *Facial Action Coding System (FACS)*[4]. The software automatically adjusts the grid to the face and then tracks it through the image sequence, following the facial expression landmarks evolving through time [5]. At the end, the grid adaptation software produces the deformed Candide grid that corresponds to the full facial expression appearing at the image sequence.

The deformed Candide grid produced by the grid adaptation algorithm [5],that corresponds to the greatest intensity of the facial expression shown, contains 104 nodes. Only some of these nodes are important for the recognition of the facial expressions. For example, nodes on the outer contour of the face do not contribute much to facial expressions. Thus, a subset of 62 nodes is chosen, that control the movement according to *FACS* used for describing the facial expressions, so as to perform facial expression recognition [6].

D. Facial Expression Recognition using Support Vector Machines

The classification is performed based only in geometrical information, not taking into consideration any luminance or color information. The geometrical information used is the displacement of one point d_i , defined as the difference between the last and the first frame's coordinates

$$d_i^c = \begin{bmatrix} \Delta x_i^j(c) \\ \Delta y_i^j(c) \end{bmatrix} , \quad i = 1,...,62 \text{ and } j = 1,..n \qquad (1)$$

where c is the number of classes of facial expressions to be recognized, here equal to 6, i is the number of points taken under consideration, here equal to 62 and j is the number of image sequences to be examined.

In that way, for every image sequence to be examined, a feature vector F_j^c is constructed, containing the geometrical displacement of every point taken into consideration, thus having the following form

$$F_j^c = \begin{bmatrix} d_1^j(c) \\ d_2^j(c) \\ ... \\ d_{62}^j(c) \end{bmatrix} \qquad (2)$$

That feature vector is used as an input to a multi class Support Vector Machine system (SVM), with six classes implemented for the experiments, that classifies each set of grid's geometrical displacements to one of the 6 basic facial expressions (anger, disgust, fear, happinnes, sadness and surprise).

Let $\mathrm{S} = \{\{\mathrm{F}_\mathrm{j}^\mathrm{c}\}_{\mathrm{j}=1}^{\mathrm{n}}, l_j\}$ the training data set of labeled training patterns, $F_j \in R^d$, where n is equal to the number of grids to be examined, *d* denotes the dimensionality of the training patterns, here equal to 62*2=124, and $l_j \in \{1,..,6\}$.

The decision function $f(F,a)$, which classifies a vector F, is chosen from a set of functions defined by the parameter a . The parameter a should be chosen in such a way that for any F the function should be able to provide a classification l as close to the estimation as possible.

The main idea of an SVM system is to construct a hyperplane that will separate the desired classes, in such a way that the margin (defined as the distance between the hyperplane and the nearest point) is maximal. Therefore, to generalize, the following equation should be minimized

$$\phi(w,\xi) = \frac{1}{2}\sum_{m=1}^{6}(w_m \cdot w_m) + C\sum_{i=1}^{n}\sum_{m \neq y_i}\xi_i^m \qquad (3)$$

with constraints

$$(w_{l_i} \cdot F_i) + b_{l_i} \geq (w_m \cdot F_i) + b_m + 2 - \xi_i^m \qquad (4)$$

$$\xi_i^m \geq 0, i = 1,...n \;\; m \in \{1,...6\} \setminus l_i\} \qquad (5)$$

The decision function that is derived from equations 3 and 4 is the following

$$f(F) = \arg\max_{k}[(w_i \cdot F) + b_i], i = 1,...,6 \qquad (6)$$

The solution to this optimization problem in dual variables can be found by the saddle point of the Lagrangian

$$L(w,b,\xi,\alpha,\beta) = \frac{1}{2}\sum_{m=1}^{6}(w_m \cdot w_m) + C\sum_{i=1}^{6}\sum_{m=1}^{6}\xi_i^m$$
$$-\sum_{i=1}^{6}\sum_{m=1}^{6}\alpha_i^m[((w_{l_i} - w_m)\cdot F_i) + b_{l_i} - b_m - 2 + \xi_i^m] - \sum_{i=1}^{6}\sum_{m=1}^{6}\beta_i^m\xi_i^m$$
$$(7)$$

with the dummy variables

$$\alpha_i^{l_i} = 0, \xi_i^{l_i} = 2, \beta_i^{l_i} = 0, i = 1,...6 \qquad (8)$$

and constraints

$$\alpha_i^m \geq 0, \beta_i^m \geq 0, \xi_i^m \geq 0, i = 1,...6 \quad m \in \{1,..6\} \setminus l_i$$
$$(9)$$

which has to be maximized with respect to α and β and be minimized with respect to w and ξ.

By further processing [7] equation 6 is finally expressed as

$$f(F,a) = \arg\max_{m}[\sum_{i=1}^{m}(c_i^m A_i - a_i^m)(F_i \cdot F) + b_m] \qquad (10)$$

or equivalently

$$f(F,a) = \arg\max_{m}[\sum_{i:l_i=m} A_i(F_i \cdot F) - \sum_{i:l_i \neq m} a_i^m(F_i \cdot F) + b_m]$$
$$(11)$$

where a is a parameter that defines which function is suitable for correctly classifying a vector F, c_i^n is the following notation

$$c_i^n = \left\{ \begin{array}{l} 1\ ifl_i = n \\ 0\ ifl_i \neq n \end{array} \right\} \qquad (12)$$

and A_i is defined as

$$A_i = \sum_{m=1}^{6}\alpha_i^m \qquad (13)$$

The SVM system created constructs a maximal linear classifier in a high dimensional feature space, Z(x), defined by a positive kernel function, $k(F,F')$, specifying an inner product in the feature space,

$$Z(F)Z(F') = k(F,F'). \qquad (14)$$

The kernel used was a 3rd degree polynomial function, defined in general as

$$k(F,F') = (F \cdot F' + 1)^d \qquad (15)$$

where d was equal to 3.

E. Facial Animation

To give the caricatural mirror effect, we decided to generate the visual features of the user on a face model. First of all, the existing open source facial animation softwares were searched, but we could not find a proper one for our task. Therefore, we decided to develop our own facial animation engine with available face models. Candide3 [A2] was the best choice among the other face models, since it includes action units (AUs) and MPEG-4 FAPs. It is a simple 3D mesh model including 184 polygons as shown in Figure 8.

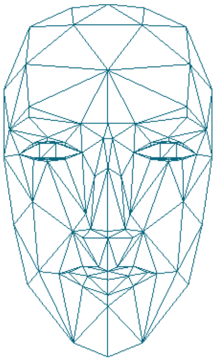

Figure 8 Wireframe model of Candide3

The software for the animation was written using OpenGL, and SDL libraries, since they are open source and run on many platforms. OpenGL is a well-known API for 3D rendering, and SDL provides low-level interfaces to reach media devices like video and sound card. Using these APIs, our software is capable of animating the face and playing the sound synchronously.

There are two different animation procedures in this project: from facial feature tracking and emotion recognition.

1) Animation From Facial Feature Tracking

In this procedure, the tracked facial features are directly used to animate the face model. Since in the tracking algorithm the same face model, Candide3, is employed, the features, which are the coordinates of the vertices of this polygonal model, are used without any operation to deform the polygonal face mesh. However, due to the prosody amplification in the speech, the speech rate is usually altered and therefore a synchronization problem between speech and lip movements occurs. To overcome this, the animation frames are simply scaled in time, and the final animation is obtained.

2) Animation From Emotion Recognition: In this part, for each detected emotion an animation sequence is generated. However, since our task is to generate the emotion not the speech animation, we decided to ignore the visual speech and made our face model just say "mamama...". By this way, our job for lip synchronization is simplified and we just consider the mouth opening and closure for synchronization.

To generate the emotions, first of all, static expressions were prepared. In our case, there are six emotions as happiness, surprise, sadness, anger, fear and disgust, and a neutral mood. Also, for each emotion we have three states, mouth closure for silence, pressed lips during the sound /m/ and mouth openning for /a/. Modifying the parameters of the Candide3 model created these expressions. There are 65 animation units that control the local movements on the face, like movements for lips and eyebrows, and are realized by simple vertex translations. The prepared expressions are depicted in Figure 9.

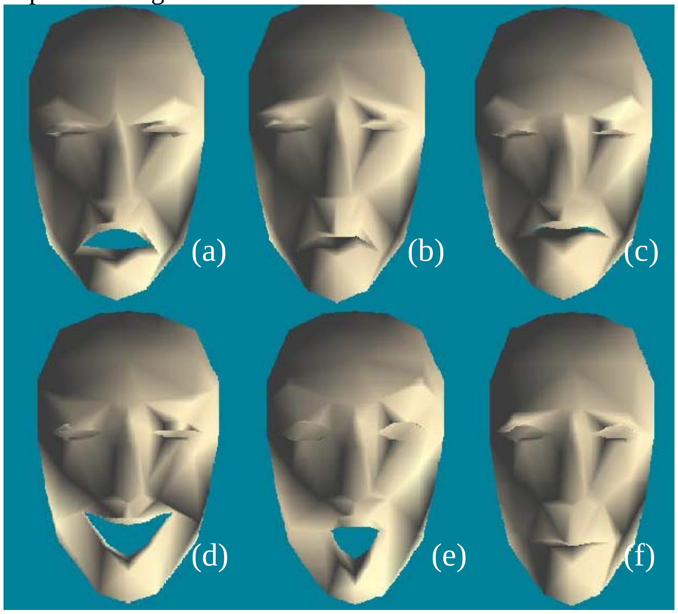

Figure 9 The six basic emotions (anger (a), sadness (b), disgust (c), happiness (d), surprise (e), fear (f)) that were prepared with Candide3

However, there is an important issue in animating a face model. There should be smooth transitions between the expressions otherwise animations seem jerky. For this purpose, sigmoid function (Eq 16), which is a monotonously increasing function, is employed.

$$y = \frac{1}{1 + \exp(-x)} \qquad (16)$$

Using the sigmoid function, first, the transitions for the vowel /a/ were modeled. The duration for the sound /a/ is divided into three regions: the entrance phase, steady-state phase and the exit phase. During the steady state, just the target expression is displayed, but for the transitions phase interpolations with the sigmoid function are performed. For the interpolation, a sigmoid function for the target state (/a/) and an inverse of it (1-sigmoid(x)) for the other state were employed to determine the weights for each expression. To model the transitions with sigmoids we need two parameters: one for the position, and one for the time scaling. The sigmoids are positioned at the center of the transition regions, which are chosen as 45% of the duration for the both entrance and exit phases by observation. By this way, the smoothness of the transition is also automatically adapted to the speaking rate, for example we have sharper transitions with higher speaking rates. On the other hand, with the scaling factor, we have a control on the smoothness. Again empirically the scaling factor is chosen as three. Moreover, we need also smooth transitions for emotions. For this task, again, sigmoid interpolation is performed, by the same procedure as for the /a/ sound.

After obtaining weights for the expressions, the next step is to calculate the weighted sum of the model coordinates. However, instead of directly calculating the vertex coordinates, first, weighted sum of animation unit parameters are computed and then they are applied to the model.

III. VOCAL MODALITY : PROSODY PROCESSING

A. Introduction

The main goal of the application is to be able to emphasize the expression generated by the user in both image and vocal features. In speech it is well known that prosodic parameters carry most of the expressive and emotional information (citation). Due to this, in this project we have focused on the amplification of expressive variation of such parameters. There are three main parameters that are considered to constitute prosody: Fundamental frequency, rythm and energy. We have discarded the energy parameter due to its weakness (citation) and only rythm and pitch have been taken into account. Although there are other parameters of prosody such as voice quality that could support the amplification of expressive events, they have been discarded after preliminar experiments.

When trying to amplify emotional events it is very important to keep the linguistic structure of speech. Speech is a process that carries information in all their aspects and speech parameters are interrelated within each other. The modification of one parameter may influence another one and the linguistic structure contained in speech might be lost. Therefore, since we do not want to distort speech but to emphasize expressive events, it is needed to be done in such a way that language is not disturbed. This idea has been present throughout all the work presented here.

In the system presented here the speech is recorded from the user, some characteristics are extracted from the voice and used to generated models that will lead the speech modification. Afterwards the modification of the original voice is performed and finally played back together with the animation generated. In figure X there is a global overview of the whole process.

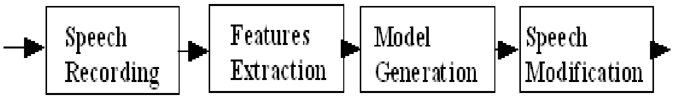

Figure 10:
General overview of the speech transfomation module.

B. Prosodic Features Extraction

We planned to modify the pitch and the rythm of the user speech. These parameters had to be extracted from the original voice. In order to extract them different techniques have been used for each of them. In order to extract the pitch the autocorrelation method that is implemented on PRAAT has been used. This program gives a set of times stamps with a pith value for each. Then, linear interpolation is performed to fill the unvoiced regions. Finally, we get a contour that has a value for every time stamp. In order to eliminate micro prosody and some effects related with the errors of the algorithm, the contour is smoothed by means of a low pas filter with a cut off frequency of 10Hz.

On the other hand, it is needed to extract the speech rate. In the application presented here the system need to be fast so we could not perform a recognition task to count phonemes it had to be done from the raw signal itself. To tackle this problem some speech exclusive characteristics have been taken into account. As can been seen in the spectrogram in Figure X the speech segments with higher energy are fricatives and vowels, and the difference between them is that fricatives contain their energy mainly in high frequencies.

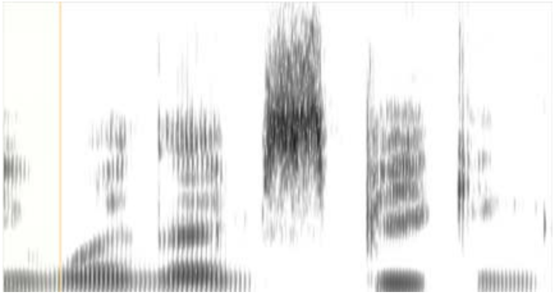

Figure 11:
Spectrogram of "To see" we can see how fricative contain more energy in high frequencies. Above 2KHz.

We can profit this effect by filtering low pass the signal with a cut off frequency of 2KHz. Then energy peak detection is performed. Since major part of the energy is contained in vowels after filtering, these peaks can be considered as an estimate of the speaking rate. It is measures in *vowels/second*.

In summary, two mean features are extracted: Pitch and Speaking rate. The first one using a classical algorithm in speech processing and the second one is only estimated by a simple algorithm that tries to find vowels position.

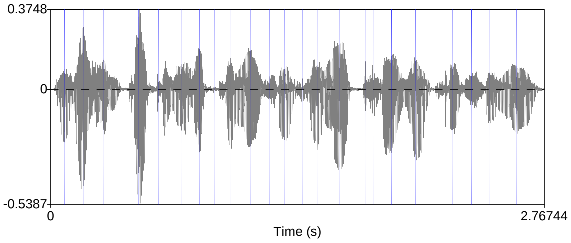

Figure 12:
Results from detecting energy maxima, mainly related to vowels. Then the speech rate is roughly measured as vowels/second.

C. Prosody Amplification

1) Pitch Variations Amplification

Pitch has a drift in a sentence that makes its mean go down. It is due to the fact human lungs are limited and the pressure of the air flowing through the vocal strings diminishes over time. Then, the pitch decreases when this pressure decreases. We have considered that this drift is a base pitch around which variations occur. Then we decided to amplify variations around this drift line. In order to calculate this drift a regression line can be performed when working only with a single sentence. But since in the present project information about beginning and ending sentences is not known, phrase breaks needed to be detected. However, this is a still left to solve problem. Then, in order to avoid it a highly smoothed base pitch line has been used. These base line is a rough approach to calculate the pitch drift but taking into account pitch jumps in phrase brakes.

This base line pitch contour is extracted using the same algorithm as the one described in previous section but with a cut off frequency of 0.5 Hz instead of 10Hz.

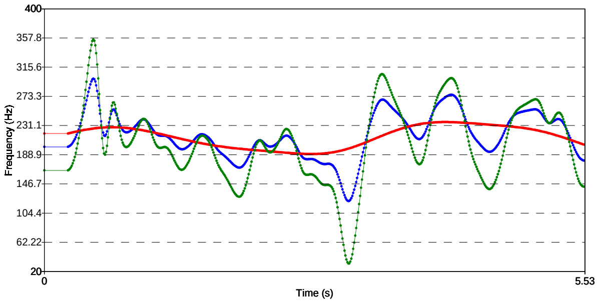

Figure 13:
In a blue line the original contour of the voice is shown. The red line is the base pitch line obtained by smoothing the contour. Green line shows the new generated contour that has been amplified with respect to the base line.

The objective of this base pitch line is to model the natural drift that pitch contour suffers. In order to keep this natural drift the smoothed contour is removed from the original contour. Therefore, the contour is amplified by multiplying it by a parameter. Then accented parts of the pitch are emphasized in a way that large movements a more amplified than small ones as can be seen in Figure X. Let *f* be the original contour, f_b the base line pitch contour and *a* a parameter, then the generated contour f_o is:

$$f_o(t) = a * (f(t) - f_b(t)) \tag{17}$$

The resulting speech is more emphasizing since pitch excursions are larger.

2) Speech Rhythm Amplification

When working with speech flow, there are two main problems. First of all, we need to decide which features will be used in order to modify the output speech flow. On the other hand, it is necessary to decide which kinds of modification suit for our task. The main task of our project is to emphasize the captured behavior of the user. Therefore, accelerated fast speech and slow down slow speech would be a good approach. The problem with this approach is that we need to have an absolute measure of what *fast* and *slow* mean. Due to this, it has been decided to use the mean of an utterance to decide what has to be slowed and which parts have to be speeded up.

After that then we needed a criteria to decide the amount of acceleration or deceleration. The mean speech rate value had to stay constant, and only variations from the mean are modified. Then a function that modifies the duration of speech is used. This function translates the speech rate calculated from the input into a function that indicates the stretch that will be applied to the sound.

This function is calculated by applying a transformation function to the speech flow contour. If *sr(t)* is the speech rate estimated, then the time transformation function *d(t) is:*

$$d(t) = f(sr(t)) \tag{18}$$

where *f(t)* can be expressed as:

$$f(t) = \frac{1}{(1 + e^{a(x-m)})} + 0.5 \tag{19}$$

where *a* is a parameter that is to be configured and *m* is the mean of the speaking rate contour. This function saturates at 0.5 and 1.5, therefore speech rate is only going to be reduced or increased in 50%. In Figure X there is an example of *f(t)* for a value of *a*=0.4 and *m*=2.

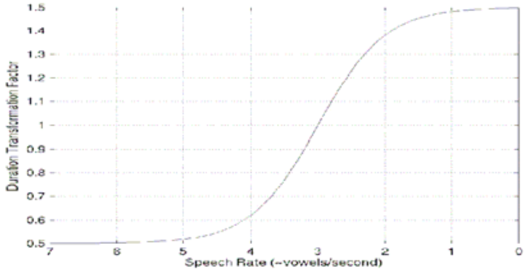

Figure 14:
In the horizontal axes the speech rate approximation is represented and on vertical axes the corresponding value for the duration modification.

Appendix

Implementation of prosody processing software

The main parts of the speech processing has been implemented using the PRAAT program which is distributed under GNU/GLP license. PRAAT could do the feature extraction job and the prosody modification. The prosody amplification and the time transformations have been implemented in C++ and inserted in the flow of the system by modifying Pitch and Duration files generated by PRAAT.

Acknowledgment

Olivier Martin is funded through a FIRST Europe fellowship from the Walloon Region (Belgium).

This work has been partially sponsored by the Spanish Government under grant TIC2002-04447-C02 (ALIADO project, http://gps-tsc.upc.es/veu/aliado) and the European Goverment under the SIMILAR Network of Excellence.

References

[1] Intel, “Open CV : Open source Computer Vision Library”, http://www.intel.com/research/mrl/research/opencv/.

[2] K.Sobottka and I.Pitas, "Segmentation and Tracking of Faces in Color Images", in Proc. of *2nd Int. Conf. on Automatic Face and Gesture Recognition 1996*, pp. 236-241, Killington, Vermont, USA, 14-16 October 1996

[3] J. Y. Bouguet, Pyramidal implementation of the Lucas-Kanade feature tracker, Intel Corporation, Microprocessor Research Labs, 1999

[4] T. Kanade, J. Cohn, and Y. Tian, Comprehensive Database for Facial Expression Analysis, Proceedings of IEEE International Conference on Face and Gesture Recognition, pages 46-53, March, 2000

[5] S. Krinidis, I. Pitas, "Statistical Analysis of Facial Expressions for Facial Expression Synthesis", submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004

[6] I. Kotsia, and I. Pitas, "Real time facial expression recognition from video sequences using Support Vector Machines", in Proc. of *Visual Communications and Image Processing (VCIP 2005)*, Beijing, China, 12-15 July, 2005

[7] J.Weston, C.Watkins, Multi-class Support Vector Machines, *Technical Report CSD-TR-98-04*, May 1998

[8] Paul Boersman and David Weenink. “Praat: doing phonetics by computers”, www.praat.org, July 2005.