Accurate and Quasi-Automatic Lip Tracking

Nicolas Eveno, Alice Caplier, and Pierre-Yves Coulon

***Abstract*—Lip segmentation is an essential stage in many multimedia systems such as videoconferencing, lip reading, or low-bit-rate coding communication systems. In this paper, we propose an accurate and robust quasi-automatic lip segmentation algorithm. First, the upper mouth boundary and several characteristic points are detected in the first frame by using a new kind of active contour: the "jumping snake." Unlike classic snakes, it can be initialized far from the final edge and the adjustment of its parameters is easy and intuitive. Then, to achieve the segmentation, we propose a parametric model composed of several cubic curves. Its high flexibility enables accurate lip contour extraction even in the challenging case of a very asymmetric mouth. Compared to existing models, it brings a significant improvement in accuracy and realism. The segmentation in the following frames is achieved by using an interframe tracking of the keypoints and the model parameters. However, we show that, with a usual tracking algorithm, the keypoints' positions become unreliable after a few frames. We therefore propose an adjustment process that enables an accurate tracking even after hundreds of frames. Finally, we show that the mean keypoints' tracking errors of our algorithm are comparable to manual points' selection errors.**

***Index Terms*—Active contour, deformable model, points tracking, segmentation.**

I. Introduction

VISUAL information provide a precious help to the listener under degraded acoustical conditions. As soon as they are seen with enough accuracy, lip contours are effectively used by human beings to improve speech intelligibility. The motivation of the present study is to extract visual information for automatic speech recognition, videoconferencing, and the speaker's face synthesis under natural lighting conditions.

During the last few years, many techniques have been proposed to achieve lip segmentation. Some of them use only low-level spatial cues such as color and edges. Zhang [1] uses hue and edge information to achieve mouth localization and segmentation. There is no shape or smoothness constraint, so the segmentation is often very rough, which makes this method unsuitable for applications that require a high level of accuracy, such as lip reading or clone synthesis. In [2], a linear discriminant analysis (LDA) is used to separate the lip pixels from the skin pixels and thus to extract the lip contour. Even if the LDA is followed by a smoothing operation, the resulting segmentation is often noisy.

Because of their ability to take smoothing and elasticity constraints into account, the "snakes" [3] have been widely applied to lip segmentation [4]–[6]. They can give quite good results, but most of the time the tuning of parameters is very difficult to achieve, and the snakes often converge to wrong results when the initial position is far from the lip edges. Moreover, the mouth corners' detection is generally difficult because they are located in low gradient areas, which leads to rough final contours. Some authors propose to detect the mouth corners by a specific algorithm [5] and to keep them still during the snake convergence. This improves accuracy, but it does not address the problem of parameters adjustment.

To make segmentation more robust and realistic, *a priori* shape knowledge has to be used. By designing a global shape model, boundary gaps are easily bridged and overall consistency is more likely to be achieved. This supplementary constraint ensures that the detected boundary belongs to possible lip shape space. For example, active shapes models (ASMs) can be used [7], but they need a large training set to cover a high variability range of lip shapes. Moreover, the images of this training set have to be cautiously calibrated. The face orientation and the lighting conditions have to be constant, otherwise the ASM method leads to unreliable results.

To avoid a restricting training step, a parametric description can be used to design models. As introduced by Yuille [8], a parametric deformable template is a parameterized mathematical model used to track the movement of a given object. In our case, the lip shape is approximated by a set of curves which is uniquely described by some parameters. Several parametric models have already been proposed. Tian [9] uses a simple three-states geometric model made of parabola. The color and shape information is used to know which model to use: mouth tightly closed, closed, or open. Then, four keypoints are used to draw the model. The position of the model is generally good, but it does not fit the boundary with accuracy because only symmetrical parabolic shapes can be generated. To make the model more flexible, other authors propose to use two parabola instead of one for the upper boundary [10] or to use quartics instead of parabola [11]. This improves accuracy, but the models are still limited by their rigidity, particularly in the case of an asymmetric mouth (see Fig. 6).

In this paper, we propose a much more flexible model made of cubic curves. It is positioned by several keypoints located on the mouth boundary, and it is fitted by using edge information. As in [9], we use an interframe tracking to enhance the speed and the robustness of the segmentation. However, we show that, without any adjustment of the keypoints' positions, the tracking errors accumulate and the segmentation becomes unreliable after a few frames.

The novelties of our study lie first in the initial frame keypoints' detection. We introduce a quasi-automatic method that only requires the manual selection of a single point located above the mouth. This point is used as a seed for a new kind of active contour: the "*jumping snake*." Unlike classic snakes, its parameters are easy to choose, and its convergence is ensured

Manuscript received April 30, 2003; revised September 3, 2003.

The authors are with the Laboratoire des Images et Signaux, 38031 Grenoble Cedex, France (e-mail: eveno@lis.inpg.fr; alice.caplier@lis.inpg.fr; Pierre-Yves.Coulon@lis.inpg.fr).

Digital Object Identifier 10.1109/TCSVT.2004.826754

1051-8215/04$20.00 © 2004 IEEE

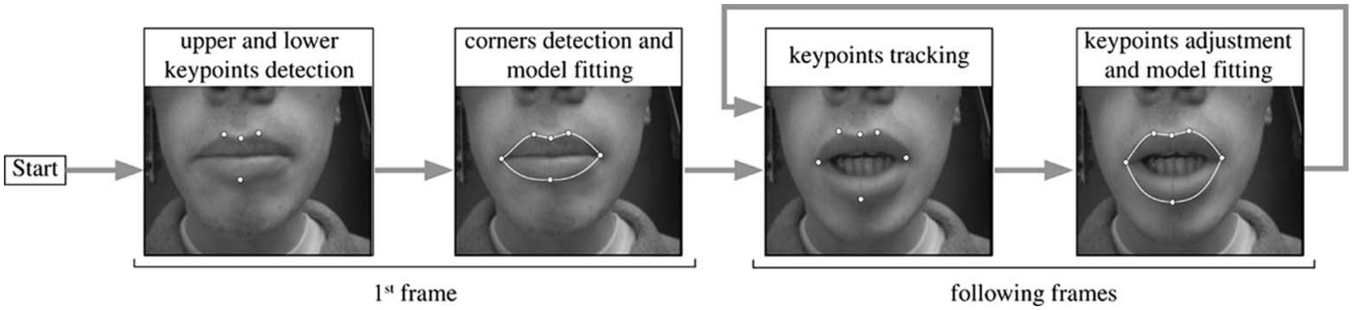

upper and lower keypoints detection
corners detection and model fitting
keypoints tracking
keypoints adjustment and model fitting
Start
1st frame
following frames

Fig. 1. Overview of the proposed algorithm.

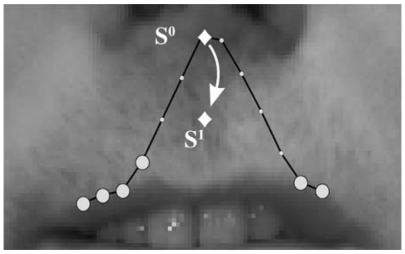


Fig. 2. Position of the seed S^1 is computed by using S^0 and the points associated with the highest mean flows (big dots).

even if the initial seed is far away from the mouth. Second, we show that the model we introduce is flexible enough to reproduce the specificities of very different lip shapes. This enables accurate segmentations, even in the challenging case of an asymmetric mouth. Lastly, the proposed adjustment process of keypoints positions ensures a good stability of the algorithm. Even after hundreds of frames, the tracking errors are correctly compensated. This finally leads to an accurate keypoints detection and a realistic lip segmentation. The whole algorithm is described in the flowchart of Fig. 1.

The outline of this paper is as follows. In Section II, we explain how the "jumping snake" algorithm can be used to detect the upper and lower keypoints in the initial frame. Section III presents the polynomial model and its fitting. In Section IV, we detail the tracking/adjustment process and the contour extraction in the following frames. Experimental results and their analysis are presented in Section V. Section VI concludes this paper.

II. Detection of Keypoints

A. The Jumping Snake Algorithm

Active contours, or snakes, have proved their efficiency in many segmentation problems. Since their introduction by Kaas *et al.* [3], many improvements have been proposed in the literature. But none of them has totally removed the two major weak points of the snakes: the choice of parameters and the high dependence on the initial position. The method presented here helps to address these problems.

To find the upper mouth boundary, we introduce a new kind of active contour that we call "jumping snake" because its convergence is a succession of jumps and growth phases [12]. It is initialized with a seed S^0 that can be located quite far away from the final edge (see Fig. 2). The seed is put manually above the mouth and near its vertical symmetry axis. The snake grows from this seed until it reaches a predetermined number of points. This growth phase is quite similar to the growing snake proposed by Berger and Mohr [13], in the sense that the snake is initialized with a single point and is progressively extended to its endpoints. Then, the seed "jumps" to a new position that is closer to the final edge. The process stops when the size of the jump is smaller than a threshold.

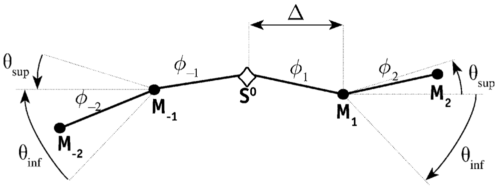


Fig. 3. From the seed S^0 ($\diamond$), the snake is extended by adding left and right endpoints ($\bullet$). The $\mathbf{R_{top}}$ mean flows ϕ_i through each segment have to be maximized.

In [14], we introduced the "hybrid edge" $\mathbf{R_{top}}$ that combines color and luminance information. It is computed as follows (in this paper vectors and matrixes are written in bold):

$$\mathbf{R_{top}}(x,y) = \nabla[h_N(x,y) - L_N(x,y)] \tag{1}$$

where $h_N(x,y)$ and $L_N(x,y)$ are respectively the pseudo hue and the luminance of pixel (x,y), normalized between 0 and 1. $\nabla[\cdot]$ is the gradient operator. The pseudo hue, introduced by Hulbert and Poggio [15], is less noisy than the usual hue and is higher for lips than for skin, as shown in [16]. It is computed as follows:

$$h(x,y) = \frac{R(x,y)}{G(x,y) + R(x,y)} \tag{2}$$

where $R(x,y)$ and $G(x,y)$ are the red and green components of the pixel, respectively (x,y). The hybrid edge $\mathbf{R_{top}}$ exhibits the top frontier of the mouth much better than the classic gradients of luminance or pseudo-hue do. It is used to guide the *jumping snake* toward the upper lip edge.

During the growth phase, left and right endpoints are added to the snake. They are located at a constant horizontal distance, denoted Δ, from the previous point. Moreover, the search area is restricted to the angular sector $[\theta_{\rm inf}, \theta_{\rm sup}]$ (see Fig. 3). The best left and right endpoints, denoted $M_{-(i+1)}$ and $\boldsymbol{M}_{i+1}$, are found in this area by maximizing the $\mathbf{R_{top}}$ mean flow through

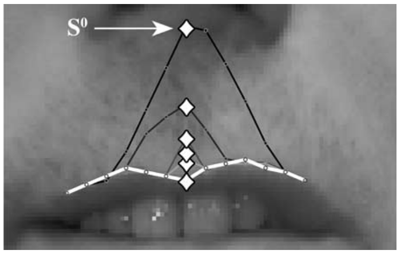


Fig. 4. Convergence of the jumping snake. After each jump, the position of the new seed (◇) is computed. At the end (white line), the snake lies on the upper lip boundary.

the end segments $\mathrm{M}_{-(\mathrm{i}+1)}\mathrm{M}_{-i}$ and $\mathrm{M_iM_{i+1}}$ (see Fig. 3). These two mean flows can be written as follows:

$$\phi_{i+1} = \frac{\int_{M_i}^{M_{i+1}} \mathrm{R}_{\mathbf{top}} \cdot \mathbf{dn}}{|\mathbf{M_i M_{i+1}}|} \quad \phi_{-\mathbf{i}-1} = \frac{\int_{M_{-\mathbf{i}-1}}^{M_{-i}} \mathbf{R_{top}} \cdot \mathbf{dn}}{|\mathbf{M_{-i-1} M_{-i}}|} \tag{3}$$

where $\mathbf{dn}$ is the vector orthogonal to the segment. The maximizations of $\phi_{-(i+1)}$ and ϕ_{i+1} are achieved by a systematic computation over a small set of candidates located in the search area.

When the snake reaches a predetermined number of points $2N+1$, the growth stops and the position of the new seed S^1 is computed. This is the jump phase of the jumping snake algorithm. Let $\{M_{-N},\ldots,M_{-1},S^0,M_1,\ldots,M_N\}$ be the points of the snake and let $\{\phi_{-N},\ldots,\phi_{-1},\phi_1,\ldots,\phi_N\}$ be the mean flows through the $2N$ segments. The new seed S^1 has to get closer to high gradient regions, i.e., high mean flow segments. We consider that S^1 is the barycentre of S^0 and the points which are in the highest gradient regions (the big dots in Fig. 2). If $\{i_1,\ldots,i_N\}$ are the indices associated with the N highest mean flows, then the vertical position of S^1 can be written as follows:

$$y_{S^1} = \frac{1}{2}\left(y_{S^0} + \frac{\sum_{k=1}^{N}\phi_{i_k}\cdot y(i_k)}{\sum_{k=1}^{N}\phi_{i_k}}\right) \tag{4}$$

where $y(i_k)$ is the vertical position of the point M_{ik}. The horizontal position x_{S1} of the seed is kept constant.

Then, a new snake grows from this new seed until it reaches the predetermined length and "jumps" again. This growth-jump process is repeated until the jump's amplitude becomes smaller than one pixel. Typically, four or five jumps are needed to achieve the convergence of the snake. In its final position, it lies on the upper lips boundary (the white line in Figs. 4 and 5).

Unlike classic snakes, the choice of the jumping snake parameters $(\Delta,\theta_{\mathrm{inf}},\theta_{\mathrm{sup}},N)$ is easy and intuitive. If there is no strong edge in the neighborhood of the snake, the overall directions of its left and right parts are dependent on the choice of θ_{inf}, and θ_{sup}, the angular limits of the search area. When $|\theta_{\mathrm{inf}}| = |\theta_{\mathrm{sup}}|$, the snake tends to be horizontal. If $|\theta_{\mathrm{inf}}| < |\theta_{\mathrm{sup}}|$, then the two branches tend to go upwards. At the opposite, they tend to go downwards when $|\theta_{\mathrm{inf}}| > |\theta_{\mathrm{sup}}|$. In our case, the initial seed S^0 is above the mouth and the snake has to fall down to get closer to the upper mouth boundary. Then we choose $|\theta_{\mathrm{inf}}| > |\theta_{\mathrm{sup}}|$.

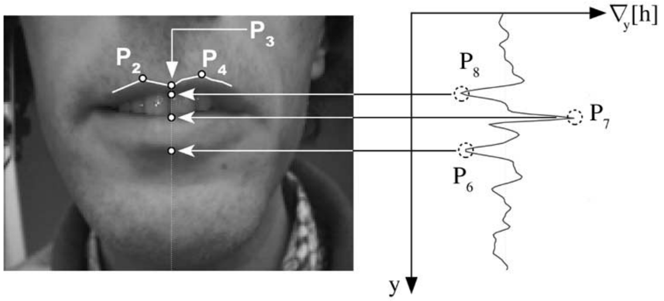


Fig. 5. Three upper points are found on the estimated upper boundary resulting from the jumping snake algorithm (white line). P_6, P_7, and P_8 are below P_3, on the extrema of $\nabla_y[h]$.

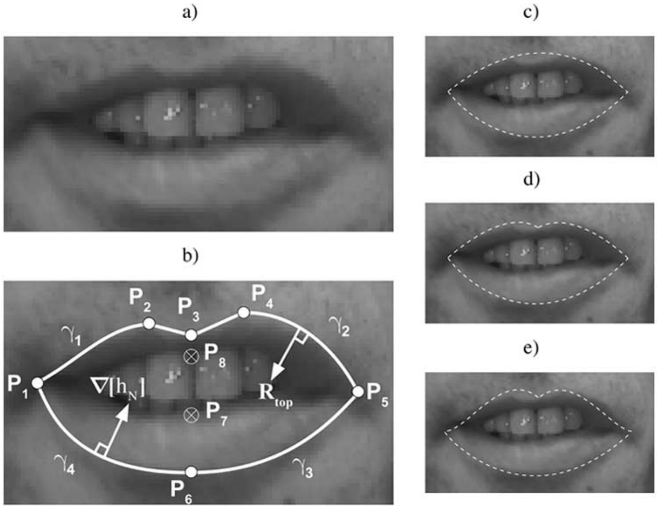


Fig. 6. (a) Asymmetric mouth shape. (b) The model we use is composed of four cubics γ_i passing by the six main keypoints (○). We also use two secondary keypoints (⊗). (c), (d) Parametric models using parabolas and (e) using quartics.

The horizontal distance Δ has a direct influence on the accuracy of the snake final position. For a small value of Δ, the detected upper edge is very detailed. However, for a given length of the snake, the computational cost is higher because more points have to be computed. On the other hand, a high value of Δ leads a rough estimation, but the convergence is achieved quicker. So, the choice of Δ is a compromise between speed and accuracy. The last parameter N gives the number of points of the snake. For a given Δ, a high value of N leads to a long snake.

In Section V, we will see that the convergence of the algorithm is ensured as long as several conditions are met. These conditions will be used to choose a unique set of parameters $(\Delta,\theta_{\mathrm{inf}},\theta_{\mathrm{sup}},N)$ that enables the segmentation of a wide variety of lips.

B. Upper and Lower Key Points Detection

The keypoints give important cues about the lip shape. They are used as fulcra for the computation of the model. We use six principal key points [see Fig. 6(b)]: the right and left mouth corners (P_1 and P_5), the lower central point (P_6) and the three points of the Cupid's bow (P_2, P_3, and P_4). We also use two secondary points located inside the mouth: P_7 and P_8. They are used to find the lower central point P_6. Moreover, they will be useful in a future work for the inner lip boundary segmentation.

The three upper points are located on the estimated upper lip boundary resulting from the jumping snake algorithm. P_2 and P_4 are the highest points on the left and right of the seed. P_3 is the lowest point of the boundary between P_2 and P_4 (see Fig. 5).

The points P_6, P_7, and P$_8$ are found by analyzing $\nabla_y[h]$, the one-dimensional gradient of the pseudo hue along the vertical axis passing by P_3 (see Fig. 5). The pseudo hue is higher for the lips than for skin or teeth and tongue. So, the maximum of $\nabla_y[h]$ below the upper boundary gives the position of P_7. P_6 and P_8 are the minima of $\nabla_y[h]$ below and above P_7, respectively.

III. Contour Extraction

A. Polynomial Model

As mentioned in the Introduction, several parametric models for the lip boundary have been proposed. Tian [9] uses a model made of two parabolas. It is very easy to compute, but it is too simple to fit the edges with accuracy [see Fig. 6(c)]. Other authors propose to use two parabolas instead of one for the upper boundary [10] or to use quartics instead of parabolas [11]. This improves accuracy, but the model is still limited by its rigidity, particularly in the case of asymmetric mouth shape [see Fig. 6(d) and (e)].

The model we use is flexible enough to reproduce the specificities of very different lip shapes [17] and is composed of five independent curves. Each one of them describes a part of the lip boundary. Between P_2 and P_4, Cupid's bow is drawn with a broken line and the other parts are approximated by four cubic polynomial curves γ_i [see Fig. 6(b)]. Moreover, we also consider that each cubic has a null derivative at key points P_2, P_4 or P_6. For example, γ_1 has a null derivative on P_2.

B. Mouth Corners and Model Fitting

The model fitting and the mouth corners detection are tightly linked. If a human operator has to find the corners, he implicitly uses the global shape of the mouth. He follows the upper and lower edges, extends them when they are becoming indistinct, and finally puts the corners where they intersect. So, the corners and the boundaries are found in a single operation. We propose an algorithm that works the same way.

Basically, a cubic curve is uniquely defined if its four parameters are known. Here, each curve passes by, and has a null derivative on points P_2, P_4 or P_6. These considerations bring two constraint equations that decrease the number of parameters to be estimated from four to two for each cubic. So, only two more points of each curve are needed to achieve the fitting. These missing points are chosen in the most reliable parts of the boundary, i.e., near P_2, P_4 or P_6. Upper curves missing points have already been found by the *jumping snake* (see Section II-A). On the other hand, only one point (P_6) of the lower boundary is known. To get additional lower points, we make a snake grow from the seed P_6. The growth stops after a few points (the dots in Figs. 7 and 8).

Now that there are enough points for each part of the boundary, it should be possible to compute the curves γ_i passing by them and to find the mouth corners where these curves intersect. However, this direct and intuitive method provides very inaccurate results. The reliable points used to compute the model are much too close to each other. A very small displacement of one of them leads to a completely different curve. It is also possible to use more than two additional points (as shown in Fig. 7) to make the computation more reliable. In that case, the best curves are found by using a least squares minimization algorithm. Using three or four points instead of two improves the accuracy, but the result is still very sensitive to points position.

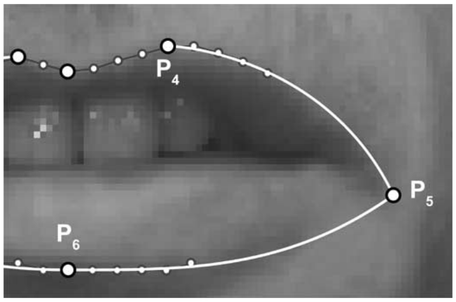


Fig. 7. Direct and intuitive approach leads to very inaccurate results. Several additional points (small dots) are used to compute the cubics (white lines). These cubics intersect far from the real mouth corner.

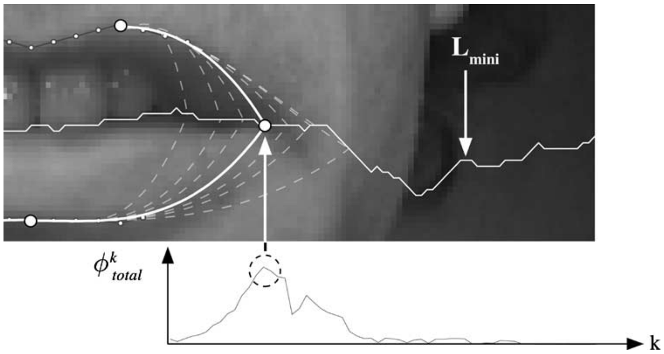


Fig. 8. Maximum of ϕ^k_{total} gives the corner position on $\mathbf{L}_{\mathrm{mini}}$. The dashed lines are the cubics associated to several tested corners number k along $\mathbf{L}_{\mathrm{mini}}$.

The direct method doesn't provide acceptable results. So, we have adopted a slightly different approach. We still use several reliable additional points to compute the curves, but we also suppose that the corners (P_1, P_5) are known. This consideration brings another constraint equation that decrease the number of parameters to be estimated from two to one for each curve. Therefore, the least squares minimization is achieved very quickly. Moreover, the resulting curves are much less sensitive to points positions. In other words, a given corner position corresponds to a unique and easily computed set of curves. So, the fitting is achieved by finding the corners that give the best curves. Obviously, an exhaustive test over all of the pixels of the image would be too long. But, a very simple hypothesis can help reduce the search area to several pixels. We consider that the mouth corners are located in dark areas. Thus we look for the minima of luminance for each column between the upper and lower boundaries. It gives the line of minima $\mathrm{L}_{\mathrm{mini}}$ (see Fig. 8). We suppose that the corners are located on this line. A given corner corresponds to a unique couple of upper and lower curves (the dashed curves in Fig. 8). So, the fitting is achieved by finding the corners that give the best couple of curves.

To know if a curve fits well to the lip boundary, we use an edge criterion. If the upper curves γ_1 and γ_2 fit perfectly to the edge, they are orthogonal to the $\mathbf{R_{top}}$ gradient field, as shown in Fig. 6). On the other hand, the curves γ_3 and γ_4 have to be orthogonal to the $\nabla[h_N]$ gradient field. We compute $\phi_{\mathrm{top},i}$ and $\phi_{\mathrm{low},i}$, the mean flows through the upper and lower curves, respectively, as follows:

$$\begin{cases} \phi_{\mathrm{top},i} = \dfrac{\int_{\gamma_i} \mathbf{R_{top}}\cdot\mathbf{dn}}{\int_{\gamma_i} \mathbf{ds}} \\ i \in \{1, 2\} \end{cases} \quad \begin{cases} \phi_{\mathrm{low},i} = \dfrac{\int_{\gamma_i} \nabla[h_N]\cdot\mathbf{dn}}{\int_{\gamma_i} \boldsymbol{ds}} \\ i \in \{3, 4\} \end{cases} \tag{5}$$

where $\mathbf{dn}$ and $\mathbf{ds}$ are the vector orthogonal to the segment and the curvilinear abscissa, respectively. We consider n possible positions along L_{mini} for each point P_1 and P_5, respectively. The best position gives a high $\phi_{\mathrm{top},i}$ and a very negative $\phi_{\mathrm{low},i}$. So, on each side we have to maximize ϕ^k_{total}, computed as follows:

$$\phi^k_{\mathrm{total}} = \phi^k_{\mathrm{top,normalized}} - \phi^k_{\mathrm{low,normalized}}, \quad k \in \{1, \ldots, n\} \tag{6}$$

where

$$\phi^k_{\mathrm{top,normalized}} = \frac{\phi^k_{\mathrm{top}} - \min_{j\in\{1,\ldots,n\}}\{\phi^j_{\mathrm{top}}\}}{\max_{j\in\{1,\ldots,n\}}\{\phi^j_{\mathrm{top}}\} - \min_{j\in\{1,\ldots,n\}}\{\phi^j_{\mathrm{top}}\}}. \tag{7}$$

ϕ^k_{top} and ϕ^k_{low} are associated with the tested corner number k. $\phi^k_{\mathrm{top,normalized}}$ and $\phi^k_{\mathrm{low,normalized}}$ are their normalized values over the whole tested set. When ϕ^k_{total} is high, the corner position is reliable because the corresponding curves fit well to the lip boundaries. So, as stated previously, the boundaries and the corners are found in a single operation. Fig. 8 shows the evolution of ϕ^k_{total} for different values of k. The maximum of ϕ^k_{total} gives the position of the corner along L_{mini}.

IV. Following Images

A. Keypoints Tracking

To increase the robustness and the speed of the segmentation, the keypoints are tracked from one image to the other. Their positions are obtained by using a variant of the Kanade–Lucas algorithm [18] adapted to the particular geometry of the mouth.

The neighborhoods of the points being tracked are assumed to have only translation movements from image I to next image J as follows:

$$J(x, y) = I(x - \alpha, y - \beta) + n(x, y) \tag{8}$$

where $(\alpha, \beta)^T$ are the components of the displacement vector $\boldsymbol{d}$ and $n(x, y)$ is the noise level for the pixel (x, y). $I(x, y)$ and $J(x, y)$ are scalars, for example, the luminance value of the pixel (x, y). Fig. 9 illustrates (8) for a monodimensional signal. The vector $\boldsymbol{d}$ is chosen to minimize the residue factor ε, computed on the neighborhood window $\mathcal{W}$, around the pixel (x, y), as follows:

$$\varepsilon = \int\int_{\mathcal{W}} [I(\mathbf{x} - \mathbf{d}) - J(\mathbf{x})]^2 \omega(\mathbf{x})\, d\mathbf{x} \tag{9}$$

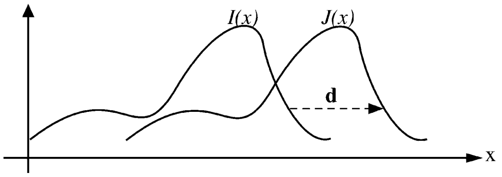


Fig. 9. Neighborhood in image I may be found in the next image J by applying a displacement vector $\boldsymbol{d}$.

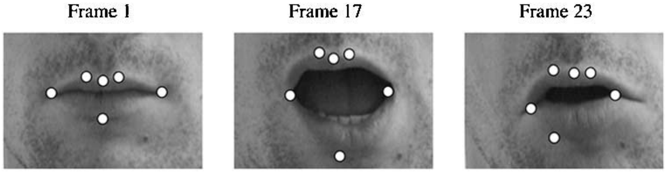


Fig. 10. Results of tracking using shifted analysis windows. After several frames, the points positions become unreliable, especially for the lower point.

where $\mathbf{x} = (x, y)^T$ and $\omega(\mathbf{x})$ is a weighting function usually constant and equal to 1.

The resolution of (9) (detailed in [19]) leads to the following 2×2 linear system of equations:

$$\mathbf{G}\,\mathbf{d} = \mathbf{e} \tag{10}$$

where

$$\begin{cases} \mathbf{G} = \int\int_{\mathcal{W}} \mathbf{g}(\mathbf{x})\, \mathbf{g}^T(\mathbf{x})\, \omega(\mathbf{x})\, d\mathbf{x} \\ \mathbf{e} = \int\int_{\mathcal{W}} (I(\mathbf{x}) - J(\mathbf{x}))\, \mathbf{g}(\mathbf{x})\, \omega(\mathbf{x})\, d\mathbf{x} \\ \mathbf{g}^T = \begin{pmatrix} \frac{\partial I(\mathbf{x})}{\partial x} & \frac{\partial I(\mathbf{x})}{\partial y} \end{pmatrix}. \end{cases} \tag{11}$$

Equation (10) gives an estimation of the displacement vector $\mathbf{d}$ and therefore of the characteristic points' location in image J.

During the tracking process, the neighborhoods of characteristic points are supposed to move only through translations. More realistic models (affine models) taking into account deformation movements have been proposed [19]. However, for consecutive images of a video sequence, the deformation of the window $\mathcal{W}$ is small. Therefore, using affine models considerably slows down the tracking process without any significant gain in results accuracy. Furthermore, the interior of the mouth is a highly deformable area in which the similarity calculation from one image to the next one is very difficult to achieve. Teeth and tongue can appear or disappear very quickly during apertures or closures. Analysis windows are thus shifted in such a way that they do not encroach upon the interior of the mouth [20].

Fig. 10 shows results obtained on a videosequence using shifted analysis windows of size 14×14. The Kanade–Lucas algorithm enables a good tracking of the keypoints from one image to the following one. However, the tracking is not perfect and after several frames the points' positions become unreliable, especially for the lower point P_6 that lies on an horizontal edge. In that case, the first component of $\boldsymbol{g}$ [in system (11)] is small and the estimated horizontal displacement of the point is not reliable. This is the well-known "aperture problem" [21].

Under these conditions, an adjustment of the points positions is necessary. Theoretically, the reliability of a tracking can be estimated by the residue error ε (9). If ε is important, then the point is considered to be mistracked. However, in practice, it is

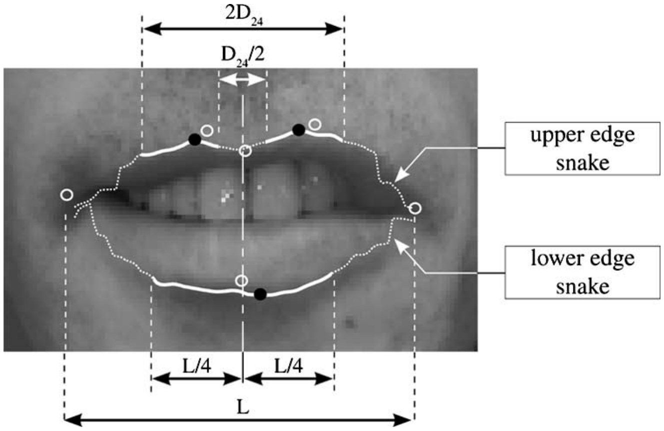


Fig. 11. Keypoints $P_i'(n)$ provided by the tracking algorithm (white circles) are adjusted by using snakes. The highest and lowest points of the snakes in the permitted areas (thick lines) are the upper and lower adjusted positions respectively (black dots).

very difficult to set a confidence threshold on ε. Sometimes a point that seems well tracked is associated with an important value of ε because luminance suddenly changed or some wrinkles appeared. So, in our case the Kanade–Lucas algorithm can be considered as a "blind tracker" whose predictions have to be refined by an adjustment process. In the following sections, these predictions will be denoted $\{P_1'(n), \ldots, P_6'(n)\}$ in image n. The adjusted keypoints positions will be denoted $\{P_1(n), \ldots, P_6(n)\}$.

B. Adjustment of the Keypoints Positions

As seen in the previous section, the keypoints positions provided by the Kanade–Lucas algorithm have to be refined. First, we adjust the upper points $P_2'(n)$ and $P_4'(n)$ by making an active contour converge toward the upper lip edge. As shown in Fig. 11, the highest points of the final contour in the neighborhood of $P_2'(n)$ and $P_4'(n)$ are the final points, denoted $P_2(n)$ and $P_4(n)$, respectively.

A snake or active contour is a parameterized curve $\boldsymbol{\nu}$ defined [see (12)] by its Cartesian coordinates x and y along the curvilinear abscissas which evolves through the minimization of its functional ϕ as follows:

$$\boldsymbol{\nu}(s) = [x(s), y(s)], \quad s \in [0,1] \tag{12}$$

$$\phi\colon \boldsymbol{\nu}(s) \rightarrow \int_0^1 [E_{\text{int}}(\boldsymbol{\nu}(s)) + E_{\text{ext}}(\boldsymbol{\nu}(s))]\, ds. \tag{13}$$

The internal energy (14) is a second-order regularization term derived from Tikhonov ill-problems theory. It controls the curve smoothness via the weighting parameters α and β. α controls the snake tension and β its curvature. External energy represents the fitting of the image data to the current curve. We focus on the upper lip boundary. Therefore, we use the $\boldsymbol{R}_{\mathbf{top}}$ norm as an external energy, given as folllows:

$$E_{\text{int}}(s, \boldsymbol{\nu}, \boldsymbol{\nu}', \boldsymbol{\nu}'') = \alpha|\boldsymbol{\nu}'(s)|^2 + \beta|\boldsymbol{\nu}''(s)|^2 \tag{14}$$

$$E_{\text{ext}}(s, \boldsymbol{\nu}) = -|\mathbf{R_{top}}|^2. \tag{15}$$

This leads us to the classic dynamic scheme [3]

$$\mathbf{V}(t) = (\gamma\mathbf{I_d} + \mathbf{A})^{-1}(\gamma\mathbf{V}(t-1) - \mathbf{F}(\mathbf{V}(t-1))) \tag{16}$$

where $\boldsymbol{I}_d$ is the identity matrix, $\boldsymbol{A}$ the Toeplitz snake matrix, $\boldsymbol{V}$ is the snake control points vector, $\boldsymbol{F}$ is the force derived from external energy, and γ is the time step coefficient. The matrix $(\gamma\boldsymbol{I}_d + \boldsymbol{A})$ is called the stiffness matrix. One of the most challenging problem of the classic snakes is the adjustment of parameters (α, β). Thus to get free from this supplementary constraint, we do not use any interior force. This leads to a simpler dynamic scheme with no stiffness matrix, shown as follows:

$$\mathbf{V}(t) = \mathbf{V}(t-1) - \frac{1}{\gamma}\mathbf{F}(\mathbf{V}(t-1)). \tag{17}$$

Obviously, as shown in Fig. 11, the resulting contour is very noisy, especially in the low-gradient value areas, because there is no elasticity or smoothness constraint any more. However, our goal is not to extract the whole upper lip edge. We just need to know it in the neighborhood of the upper points, where the gradient values are high and the snake is reliable.

The initial position of the snake is computed by using the segmentation achieved in the previous frame. The lip contour model of image $n-1$ is stretched and twisted in a way it fits on the current keypoints positions $P_i'(n)$. So, each cubic of image $n-1$ (denoted $\gamma_i(n-1)$) is deformed by using a weighted sum of its associated keypoints displacements. For example, the displacements of the points P_1 and P_2 are used to compute the displacement vector of each point Q of $\gamma_1(n-1)$ as follows:

$$\begin{aligned}\mathbf{d}_Q = \mathbf{d}_{\mathbf{P}_1}&\left(1 - \frac{|P_1(n-1)Q(n-1)|}{|P_1(n-1)P_2(n-1)|}\right)\\ &+ \mathbf{d}_{\mathrm{P}_2}\left(1 - \frac{|P_2(n-1)Q(n-1)|}{|P_1(n-1)P_2(n-1)|}\right)\end{aligned} \tag{18}$$

where

$$\begin{cases}\mathbf{d}_Q = \overrightarrow{Q(n-1)Q(n)}\\ \mathbf{d}_{\mathbf{P}_1} = \overrightarrow{P_1(n-1)P_1'(n)}\\ \mathbf{d}_{\mathbf{P}_2} = \overrightarrow{P_2(n-1)P_2'(n)}.\end{cases}$$

$\boldsymbol{d}_Q$, $\boldsymbol{d}_{P_1}$, and $\boldsymbol{d}_{P_2}$ are the displacement vectors of Q, P_1, and P_2, respectively. This transformation provides the curves $\gamma_i'(n)$ which are used for the snake initialization. They are close enough to the edges to enable a quick and robust snake convergence. The highest points of the snake in the permitted areas are the adjusted positions $P_2(n)$ and $P_4(n)$ (see Fig. 11). The size and location of these permitted areas are based on the mouth corners' positions and the distance between $P_2(0)$ and $P_4(0)$ in the first frame (denoted D_{24}).

The lower point $P_6(n)$ is found by using the same method. The curves $\gamma_3'(n)$ and $\gamma_4'(n)$, provided by the deformation of $\gamma_3(n-1)$ and $\gamma_4(n-1)$, are used for the initialization of an active contour. This contour has to fit the lower edge, so its exterior energy is $-|\nabla[h_N]|^2$. After the convergence, the lowest point of the snake in the permitted area is the adjusted position $P_6(n)$. This area is defined by using the mouth corners positions, as shown in Fig. 11.

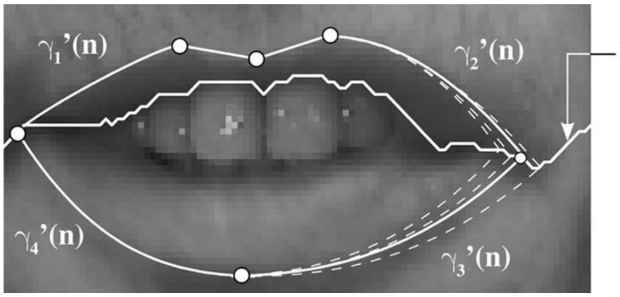


Fig. 12. To adjust the corners, several possible positions along the line $L_{\rm mini}$ are considered. The dashed lines are the corresponding deformed curves.

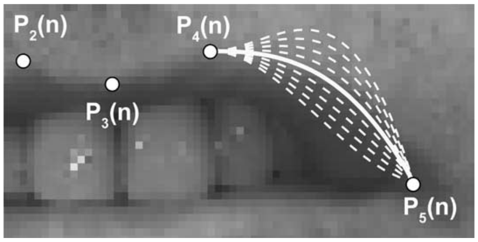


Fig. 13. Several cubics with different slopes on the mouth corner point are tested (dashed) for the optimization of the right upper curve. The best one is $\gamma_2(n)$ (plain). It maximizes $\phi_{\rm top,2}$.

To find the upper central point $P_3(n)$, we suppose that it is located at equal distance of $P_2(n)$ and $P_4(n)$. So, we test several positions in the neighborhood of $P'_3(n)$ along the mediator of $[P_2(n)\ P_4(n)]$. The best point maximizes the $\mathbf{R_{top}}$ mean flow through the broken line $[P_2(n)\ P_3(n)\ P_4(n)]$.

The mouth corners positions cannot be adjusted by using a snake because most of the time their neighboring edges are not very distinguishable. As explained in Section III-B, the lip boundaries and the mouth corners are implicitly found in a single operation. Here, we already have predictions of the corners positions ($P'_1(n)$ and $P'_5(n)$). Moreover, we can compute good approximations of the lip boundaries by using the (18). So, like in Section III-B, we consider several possible corners along the line of luminance minima $\rm L_{mini}$. However, as we already have estimations of the corners positions, only a reduced number of tested points is needed. Typically, considering the four or five closest points to $P'_1(n)$ and $P'_5(n)$ on $\rm L_{mini}$ is sufficient to achieve a good adjustment. For each one of these tested points, the corresponding curves $\gamma'_i(n)$ are computed by (18). The best corners positions $P_1(n)$ and $P_5(n)$ maximize $\phi^k_{\rm total}$ [see (6)] on the left and right sides, respectively. Fig. 12 illustrates this adjustment process for the right corner. The deformed curves $\gamma'_i(n)$ fit the edges quite well. However, they are no longer cubic. So, in order to keep the model consistency, the last step of the segmentation is the computation of the fitting cubics associated with the adjusted keypoints $P_i(n)$.

C. Cubics Fitting

The last step of the algorithm is the lip contour extraction. It consists in the computation of the fitting cubics $\gamma_i(n)$ passing by the keypoints. As in Section III-B, we use an edge criterion. The upper boundary is found by maximizing $\phi_{\rm top,1}$ and $\phi_{\rm top,2}$ [see (5)], and the lower boundary by the minimization of $\phi_{\rm low,3}$ and $\phi_{\rm low,4}$. Each cubic is uniquely described if its four parameters are known. However, the hypothesis used to build the model (see Section III-A) brings three constraint equations that decrease the number of parameters to be estimated from four to one for each curve. The last parameter can be found if the slope of the cubic on the mouth corner is known. So, the fitting is achieved by a systematic computation over a discrete set of slopes. Fig. 13 shows several cubics (dashed lines) associated with $P_4(n)$ and $P_5(n)$, with varying slopes on $P_5(n)$. The best one (plain line) maximizes the $\boldsymbol{R_{top}}$ mean flow.

To reduce the number of tested slopes and to enhance the robustness of the segmentation, we use the segmentation achieved in the previous image. As we suppose that the interframe contour deformation is small, the slopes in consecutive images are close. So, we test only slopes that are close to the one computed in the previous frame.

V. Results

Figs. 14–16 show representative results of our algorithm for different speakers. The resulting lip shapes are very realistic and fit the edges with accuracy (segmented video sequences available [22]). The model is able to reproduce the specificities of very different speakers' lips. Moreover, the method is robust even in challenging cases such as a bearded speaker (first column of Fig. 14), shadowy images (third and fifth columns), or if teeth or tongue are visible. The high flexibility of the model enables the segmentation of very asymmetrical lips (second row of Fig. 15).

The white dots on the first images of Fig. 14 are the initial seed S^0 of the jumping snake algorithm. For the moment, the initialization is done by hand. But it should be possible to get S^0 automatically by using an eyes detection algorithm. In that case, the initial seed can be deducted geometrically from the eyes positions. It is also possible to ask the speaker have his mouth in a predetermined area of the image during the initialization step. After the computation of the initial contour (which takes less than 1 s), the speaker could move freely.

Unlike the classic snakes, the initial seed can be located quite far away from the final edge. As long as S^0 is located in the gray areas of Fig. 14, the convergence is ensured. So, because of its wide convergence area, the jumping snake is a very efficient tool for the upper lip edge localization.

Moreover, the parameters of the jumping snake $(\Delta, \theta_{\rm inf}, \theta_{\rm sup}, N)$ are easy to choose. Although the size of the images and the lips are different, the parameters have been kept constant and we took $(\Delta, \theta_{\rm inf}, \theta_{\rm sup}, N) = (5, -\pi/3, \pi/6, 5)$. Actually, the convergence of the jumping snake is easy to achieve as long as several conditions are met. First, $|\theta_{\rm inf}|$ has to be greater than $|\theta_{\rm sup}|$ to make the snake "fall" on the upper lip boundary. Moreover, the angular sector $[\theta_{\rm inf}, \theta_{\rm sup}]$ has to be wide enough otherwise the snake will not be able to fit the edge of lip. So, we take $\theta_{\rm inf} < -\pi/4$ and $\theta_{\rm sup} > \pi/8$. Second, the snake must be long enough to enable the detection of P_2 and P_4 and to provide the additional points needed for the cubics computation. So, at the end of a growth phase, the horizontal length of the snake $2N\Delta$ must be greater than the distance between P_2 and P_4, denoted D_{24} (i.e., $2N\Delta > D_{24}$). By using this condition, it is possible to obtain a maximum value of D_{24}. If $\Delta = 5$ and

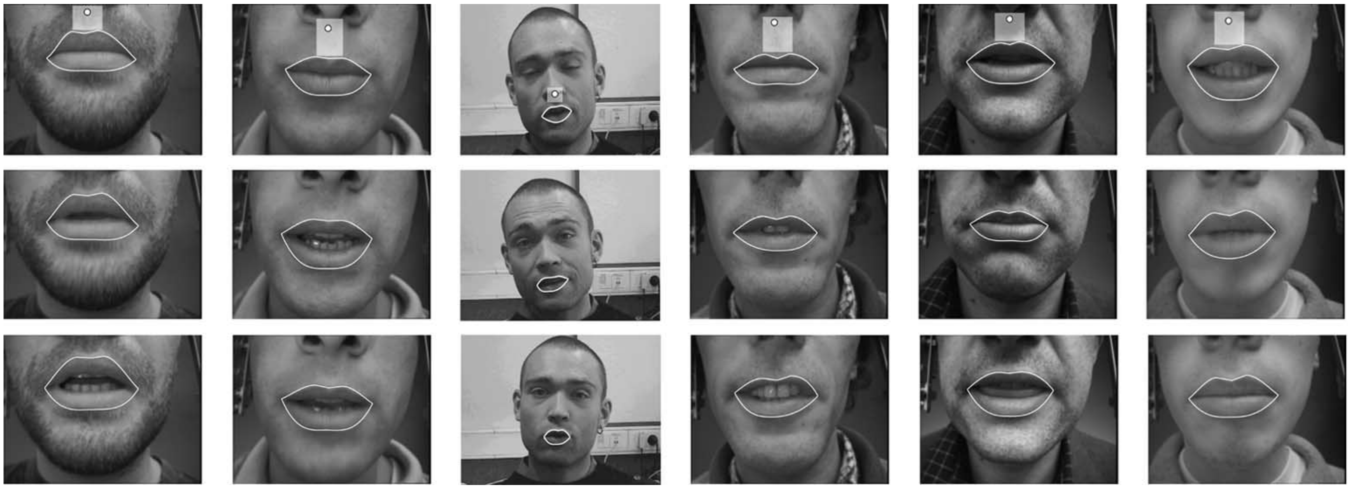
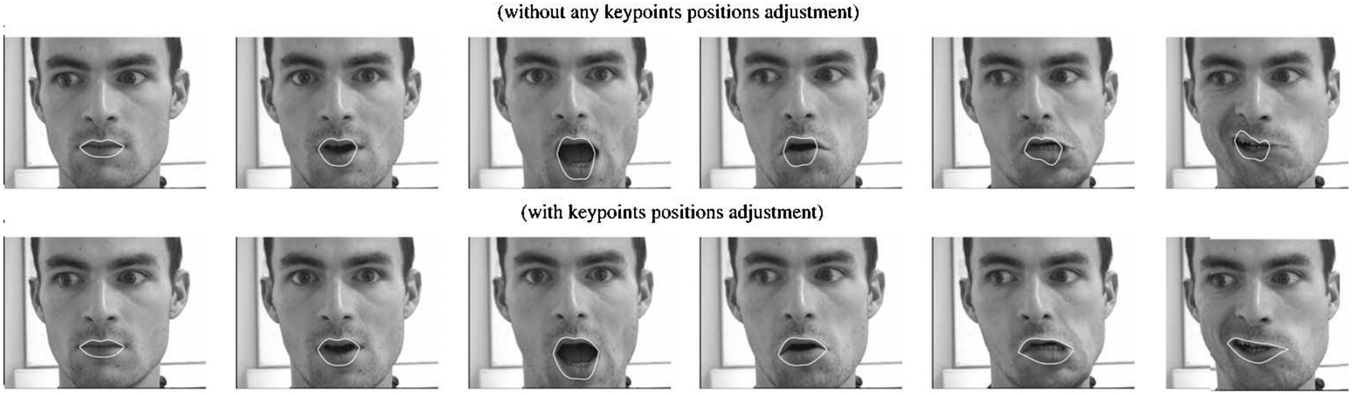


Fig. 14. Several results on various speakers. The white dots on the first images are the initial seed S^0, and the gray areas are the approximative convergence areas of the *jumping snake* algorithm. The model is able to reproduce the specificities of very different mouth shapes.



(without any keypoints positions adjustment)
(with keypoints positions adjustment)

Fig. 15. Results of segmentations without keypoints positions adjustment (top row) and with keypoints positions adjustment (bottom row).

$N = 5$, then $D_{24} < 50$. Finally, the points of the snake should not be too far away from each other. We consider that there must be at least one point between P_2 and P_4 to ensure an accurate keypoints detection. So, Δ must verify $2\Delta < D_{24}$. A minimum value of D_{24} can be deducted from this last condition. If $\Delta = 5$, then $D_{24} > 10$. Therefore, theoretically the set of parameters that we take enables the convergence of the snake as long as we have $10 < D_{24} < 25$. In our tests, the convergence is ensured (with a single set of parameters) for values of D_{24} spanning from 15 to 35, which corresponds approximately to mouth widths spanning from 50 to 90 pixels.

Fig. 15 exhibits the crucial role of the keypoints positions adjustment. It presents the segmentation achieved with (first row) and without (second row) the adjustment step. As discussed in Section IV-A, the Kanade–Lucas tracking algorithm is a "blind tracker." Without any adjustment the tracking errors accumulate and the points progressively shift to wrong positions. After a few frames, the estimated contour becomes very inaccurate. At the opposite, the segmentation achieved with an adjustment of keypoints is reliable even in the challenging case of very asymmetrical mouth. Moreover, the adjustment ensures a good stability of the algorithm. It enables a good mouth tracking even after hundreds of frames, as shown on the TV news sequence of Fig. 16.

The six keypoints that we chose are also used by the MPEG-4 standard to describe the mouth movements. Our algorithm can be used to track these points. It can therefore be integrated in a MPEG-4 based system. To test the tracking performance of the algorithm, the keypoints of 300 images (extracted from 11 sequences) have been manually marked several times by different human operators. For each image n, we consider that the ground-truth keypoints $P_{i,\mathrm{ref}}(n)$ are the mean of the hand-checked positions. Then we compute the normalized mean tracking error as follows:

$$\varepsilon_{i,\mathrm{tracking}} = \frac{1}{\mathrm{T}} \sum_{n=1}^{T} \frac{|P_{i,\mathrm{ref}}(n) P_{i,\mathrm{tracking}}(n)|}{|P_{1,\mathrm{ref}}(n) P_{5,\mathrm{ref}}(n)|} \tag{19}$$

where $P_{i,\mathrm{tracking}}(n)$ is the estimated position of point i in image n, and T is the number of tested images. $|P_{1,\mathrm{ref}}(n) P_{5,\mathrm{ref}}(n)|$ is the mouth width in image n. Moreover, we compare these tracking errors to human mean normalized errors computed as follows:

$$\varepsilon_{i,\mathrm{human}} = \frac{1}{\mathrm{T}} \sum_{n=1}^{T} \left(\frac{1}{\mathrm{K}} \sum_{k=1}^{K} \frac{|P_{i,\mathrm{ref}}(n) P_{i,\mathrm{human}}(n,k)|}{|P_{1,\mathrm{ref}}(n) P_{5,\mathrm{ref}}(n)|} \right) \tag{20}$$

where K is the number of manual checks and $P_{i,\mathrm{human}}(n,k)$ is the kth manually estimated position of point i in image n. Table I

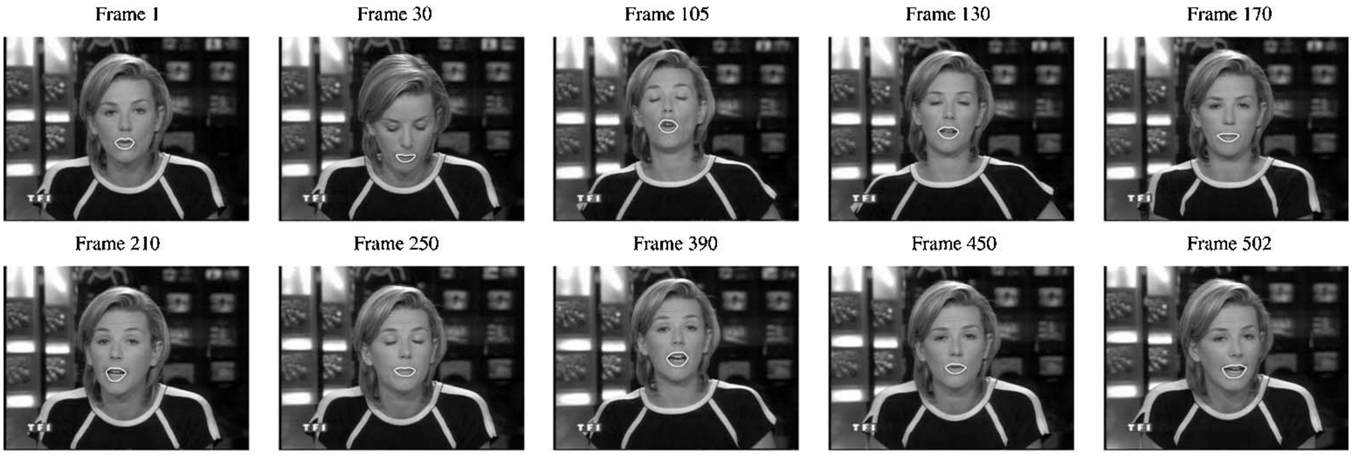


Fig. 16. Results of segmentations on a long sequence (TV news). Even after hundreds of images, the segmentation is achieved accurately.

TABLE I
MEAN NORMALIZED TRACKING AND HUMAN ERRORS FOR THE SIX KEYPOINTS (THE REFERENCE IS THE MOUTH WIDTH)

	P_1	P_2	P_3	P_4	P_5	P_6
$\varepsilon_{i,tracking}$	0.040	0.024	0.018	0.019	0.033	0.036
$\varepsilon_{i,human}$	0.010	0.008	0.008	0.009	0.011	0.011

TABLE II
COMPUTATION TIMES FOR THE INITIALIZATION STEP (WITH QCIF IMAGES)

Gradients and hybrid edges computation	0.04 s
Jumping snake convergence	0.36 s
Upper and lower points detection	0.2 s
Contour extraction	0.2 s
Total initialization time	**0.8 s**

shows the values of the tracking and human mean normalized errors for the six keypoints. Like the human errors, they are more important for the corners (P_1 and P_5) and the lower point (P_6). To visualize more easily what represent these errors, it can be helpful to know that the diameter of the circles in the Fig. 11 is approximately 3.5% of the mouth width. The maximum mean tracking error is 4% of the mouth width, which is comparable to usual mean human errors. This makes our algorithm very suitable for applications that require an accurate detection of the mouth keypoints.

For the moment, our method is implemented under MATLAB. The initialization step takes about 0.8 s (on a Pentium IV 2.4-GHz) for a 144 × 188 image (QCIF format). The contour tracking takes about 0.24 s. Detailed computation times are presented in Tables II and III. The tracking is quicker than the initialization because the analysis areas are reduced to the keypoints neighborhoods. Moreover, the snakes that we use to achieve the adjustment of the upper and lower points do not need any stiffness matrix inversion. This increases the snakes convergence speed and leads to a fast adjustment process. With a C++ implementation, it should be possible to meet the requirements of real time processing.

TABLE III
COMPUTATION TIMES FOR THE CONTOUR TRACKING (WITH QCIF IMAGES)

Gradients and hybrid edges computation	0.04 s
Points tracking	0.08 s
Keypoints adjustment	0.06 s
Contour extraction	0.06 s
Total tracking time	**0.24s**

VI. CONCLUSION

In this paper, we have presented a robust and accurate quasi-automatic lip segmentation method. We introduce a new kind of active contour: the "*jumping snake*." We show that, unlike the classic snakes, it can be initialized quite far away from the final contour and its parameters are easy to adjust. Associated to an "hybrid edge," it ensures an accurate lip boundary localization in the first image of a video sequence. Then, a five-polynomial-curves model is fitted to the outer lips boundary. Its high flexibility enables very realistic results that can be used, for example, for high-quality clone synthesis. To achieve the segmentation in the following images, we use an interframe tracking of keypoints and curves parameters. This enhances the speed and the robustness. However, we show that the results of this tracking have to be refined. Thus , we propose an adjustment process that enables an accurate tracking of keypoints even in challenging cases. Finally, we show that the mean keypoints tracking errors are comparable to manual points selection errors. This makes this algorithm very suitable for applications that require an accurate detection of the mouth keypoints, such as MPEG-4 based applications or joint audio-video speech coding systems. For the moment, we still have to manually select the starting seed in the first image. We are currently trying to make this automatic, which would enable a fully automatic mouth segmentation.

REFERENCES

[1] X. Zhang, R. M. Mersereau, M. A. Clements, and C. C. Broun, "Visual speech feature extraction for improved speech recognition," in *Proc. ICASSP*, 2002, pp. 1993–1996.

[2] A. Nefian, L. Liang, X. Pi, L. Xiaoxiang, C. Mao, and K. Murphy, "A couple HMM for audio-visual speech recognition," in *Proc. ICASSP*, 2002, pp. 2013–2016.

[3] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comput. Vis.*, vol. 1, no. 4, pp. 321–331, Jan. 1988.

[4] D. Terzopoulos and K. Waters, "Analysis and synthesis of facial image sequences using physical and anatomical models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 569–579, June 1993.

[5] P. Delmas, P.-Y. Coulon, and V. Fristot, "Automatic snakes for robust lip boundaries extraction," in *Proc. ICASSP*, 1999, pp. 3069–3072.

[6] P. S. Aleksic, J. J. Williams, Z. Wu, and A. K. Katsaggelos, "Audio-visual speech recognition using MPEG-4 compliant visual features," *EURASIP J. Appl. Signal Processing*, vol. , pp. 1213–1227, Sept. 2002.

[7] J. Luettin, N. A. Tracker, and S. W. Beet, "Active Shape Models for Visual Speech Feature Extraction," Univ. of Sheffield, Sheffield, U.K., Electronic System Group Rep. 95/44, 1995.

[8] A. Yuille, P. Hallinan, and D. Cohen, "Feature extraction from faces using deformable templates," *Int. J. Comput. Vis.*, vol. 8, no. 2, pp. 99–111, 1992.

[9] Y. Tian, T. Kanade, and J. Cohn, "Robust lip tracking by combining shape, color and motion," in *Proc. ACCV*, 2000, pp. 1040–1045.

[10] T. Coianiz, L. Torresani, and B. Caprile, "2D deformable models for visual speech analysis," *NATO Advanced Study Institute: Speech reading by Man and Machine*, pp. 391–398, 1995.

[11] M. E. Hennecke, K. V. Prasad, and D. G. Stork, "Using deformable templates to infer visual speech dynamics," in *Proc. 28th Annu. Asilomar Conf. Signals, Systems, and Computers*, 1994, pp. 578–582.

[12] N. Eveno, A. Caplier, and P. Y. Coulon, "Jumping snake and parametric model for lip segmentation," in *Proc. ICIP*, Barcelona, Spain, 2003, pp. 867–870.

[13] M. O. Berger and R. Mohr, "Toward autonomy in active contour models," in *Proc. ICPR*, 1990, pp. 847–851.

[14] N. Eveno, A. Caplier, and P. Y. Coulon, "Key points based segmentation of lips," in *Proc. ICME*, 2002, pp. 125–128.

[15] A. Hulbert and T. Poggio, "Synthesizing a color algorithm from examples," *Science*, vol. 239, pp. 482–485, 1998.

[16] N. Eveno, A. Caplier, and P. Y. Coulon, "A new color transformation for lips segmentation," in *Proc. MMSP*, 2001, pp. 3–8.

[17] ——, "A parametric model for realistic lip segmentation," presented at the ICARCV, 2002.

[18] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. IJCAI'81*, Vancouver, BC, Canada, 1981, pp. 674–679.

[19] C. Tomasi and T. Kanade, "Detection and Tracking of Point Features," Carnegie Mellon Univ., Tech. Rep. CMU-CS-91-132, 1991.

[20] P. Delmas, N. Eveno, and M. Lievin, "Toward robust lip tracking," in *Proc. ICPR*, 2002, pp. 528–531.

[21] E. C. Hildreth, *The Measurement of Visual Motion*. Cambridge, MA: MIT Press, 1984.

[22] Nicolas Eveno's Page, Research Results [Online]. Available: http://www.lis.inpg.fr/pages_perso/eveno/recherche_en.htm

Nicolas Eveno received the M.S. degree in signal and image processing and the Ph.D. degree in image processing from the Institut National Polytechnique de Grenoble, France, in 2000 and 2003, respectively.

His research interests involve human gesture analysis.

Alice Caplier received the Ph.D. degree in image processing from the Institut National Polytechnique de Grenoble, France, in 1995.

She has been a Permanent Researcher with the Laboratoire des Images et Signaux, Grenoble, France, since 1997. Her main interests concern human motion analysis and interpretation. She is currently involved in facial emotions recognition, human detection and tracking, and head motion analysis.

Pierre-Yves Coulon received the Ph.D. degree in automatic control from the Institut National Polytechnique de Grenoble, Grenoble, France, in 1982.

He has worked in different areas such as target tracking and architectures for vision machines. Since 1996, he has been a Professor with Laboratoire des Images et Signaux, Grenoble, and his current research interest is image segmentation applied to lip detection and characterization.