Lip-motion events analysis and lip segmentation using optical flow

Stefan M. Karlsson and Josef Bigun
Halmstad University, IDE SE-30118 Halmstad, Sweden
Stefan.Karlsson@hh.se

Abstract

We propose an algorithm for detecting the mouth events of opening and closing. Our method is translation and rotation invariant, works at very fast speeds, and does not require segmented lips. The approach is based on a recently developed optical flow algorithm that handles the motion of linear structure in a stable and consistent way.

Furthermore, we provide a semi-automatic tool for generating groundtruth segmentation of video data, also based on the optical flow algorithm used for tracking keypoints at faster than 200 frames/second. We provide groundtruth for 50 sessions of speech of the XM2VTS database [16] available for download, and the means to segment further sessions at a relatively small amount of user interaction.

We use the generated groundtruth to test the proposed algorithm for detecting events, and show it to yield promising result. The semi-automatic tool will be a useful resource for researchers in need of groundtruth segmentation from video for the XM2VTS database and others.

1. Introduction

We use optical flow for temporal detection of motion events(opening and closing) without the need of segmenting the lips, nor apriori knowledge of their position in the region of interest(ROI). The approaches are fast enough to potentially be implemented in real-time on a smart-phone app, and the translation and rotation invariant properties of the method further implies this to be a promising future platform.

Furthermore, we present a semi-automatic lip-segmentation software for video, achieved by tracking points at 200 frames/second. This interactive software require regular user input for groundtruth in key-frames. Few keyframes are needed, and the tracking inbetween key-frames is done in a forward backward manner with appropriate smoothness imposed. The user updates the key-frames, or adds new ones on the fly, as manual inspection of the segmentation is judged. We test our proposed algorithm towards groundtruth of 50 sessions, and show it to provide accurate results on the XM2VTS database.

Our visual analysis in conjunction with voice recognition can be used when performing speaker recognition, much in line with the approach in [9]. Such application could include both improving the verification/identification rates as well as improved liveness detection.

In general, lip-events should correlate with auditory information, and if a video signal does not conform, perhaps it is a user attempting to (unsuccesfully) articulate with recorded audio. This also opens for the application of quality assesing dubbed video, even if the language is spoken in a different language.

For the application of audio-visual speech recognition, combining modalities of speech and audio improves recognition results both in human perception [15], as well as in advances towards digital systems [17].

Methods of visual speech dynamics description can be broadly divided into three approaches: Parametric models, regions of interest(ROI) and optical flow.

Parametric models are based on the assumption that the shape (3D mesh or 2D contour/binary image) of the lips can be extracted from the image sequence. The shapes of the lips can then be efficiently represented by a coarse set of parameters, e.g. splines [18] or Fourier descriptors[20] for contours and surface shape index [13] for 3D meshes.

The main drawback of parametric models is the difficulty in estimating them from the sequence, especially in conditions where lips are partially occluded, as in the presence of abundant facial hair. Humans, on the other hand, excel at doing this, and can with high degree of accuracy determine when a segmentation is wrong.

ROI approaches, on the other hand overcome some difficulties by only requiring that the lips are centralized in the region. The entire set of image pixels is used as features for a given time instant. Extensive systems are often constructed around this approach, using first reductionist transformations, such as the discrete cosine transform or wavelets expansion, and then combinations of linear discriminant analysis, principal components analysis, vector quantization, and/or multidimensional scaling[17]. The drawback of ROI methods is that pixel values are not qualitatively suited for lip dynamics description, nor are their derivative in time.

Optical flow based approaches are in theory well suited, given an algorithm that outputs a consistent flow field for analysis. Two main problems are: a) lip motion contains a high-degree of line motions (aperture problem for optical flow) and b) it is not clear how to process the flow field into a smaller set of robust features. We will address these issues by introducing an optical flow algorithm well suited for the task, and provide the methods of producing robust features efficiently. The importance of robust features and to deal with the case of linear motions for lips, is underlined in the work of [8]. A further problem in the case of lip motions lies with prescence of multiple motions, or motion boundaries, a topic we do not deal with in this work.

Groundtruth 2D countour is valuable for a database of visual speech. It can be used to evaluate lipsegmentation algorithms, as well as provide an upperbound for recognition algorithms that make use of lip segmentation.

State of the art optical flow algorithms focus on the highest accuracy possible for the displacements of points of topologically consistent (yet not necessarily rigid) bodies. Typical variational methods such as Horn and Schunk[10], penalize the discontinuities of the field. This is an expression of the assumption of topological constancy of the object observed, and therefore such methods are limited in their usefulness for lip dynamics. On the other hand, variational approaches hold the power to fill in the blanks often left by local flow estimation in areas of no structure, or only linear structure (as is the case for the Lucas and Kanade (LK) [14] method). While the variational approach is general enough to overcome these difficulties, it does so at increasingly higher computational costs and does not lend itself well for parallel implementations. Because computational efficency and consistency of estimation are in focus we choose to derive a local method. We focus on the first order derivatives of the spatio-temporal volume, but note that interesting complementary viewpoints may be found by considering higher orders of spatial derivatives in a lokal fashion[19].

Our approach for events detection is reminiscent of the work in [5]. However, we use concepts of flow processing, more in line with [12], and focus on optical flow differential invariants for novel problems here. Our approach can be interpreted as affine optical flow estimation[1], yet we do not leave the realm of local methods, nor do we need to deal with costly inversions of large matrices. We use divergence of the field at a coarse scale, to estimate the change in lip countour area(first temporal derivative of area). We show how this estimate is reasonable, by comparing with the manually generated groundtruth of 50 sessions of unique subjects of the XM2VTS database (uttered digits from 0-9).

2. Optical Flow Estimation

Optical flow is the apparent 2D motion in an image sequence, a $R^3 \rightarrow R^2$ flow-field, $\vec{u}(\vec{x},t) = \{u,v\}$. Any region Ω where flow is to be estimated with full degrees of freedom, must contain linearly independent $\nabla I(\vec{x_i})$. A region where gradients are linearly dependent is called linearly symmetric and it is impossible to estimate flow locally except for one component: that of the aligned direction (the so called aperture problem). The amount of linear symmetry is given by the eigenvalues of the 2D structure tensor:

$$G(\vec{x}) = \begin{pmatrix} m_{200} & m_{110} \\ m_{110} & m_{020} \end{pmatrix}$$

The quantity $\alpha = \frac{\lambda_2^{2D} - \lambda_1^{2D}}{\lambda_2^{2D} + \lambda_1^{2D}} \in [0,1]$ yields the amount of linear symmetry, and thus $1-\alpha$ tells how well distributed the spatial structure in the region is. Here we choose to express the 2D structure tensor in 3D spectral moments, similar to the investigation in [6]. These are calculated as e.g $m_{020} = \sum w I_y^2$ and $m_{101} = \sum w I_x I_t$, where w is a smooth window function covering the spatio-temporal region Ω.

Our approach to optical flow will be inspired by the 3D structure tensor [4]:

$$G^{3D} = \iiint_{\Omega} \nabla_3 I \nabla_3^T I = \begin{pmatrix} m_{200} & m_{110} & m_{101} \\ m_{110} & m_{020} & m_{011} \\ m_{101} & m_{011} & m_{002} \end{pmatrix} \quad (1)$$

where ∇_3 indicates the 3D gradient. The 3D tensor can be used to estimate optical flow directly [4] by considering its eigen system: $\{\vec{v_i}, \lambda_i\}$ for $\lambda_1 \leq \lambda_2 \leq \lambda_3$ (G^{3D} and G are positive semi-definite). We note that the m_{ijk} are scalar products, efficiently calculated in parallel and easily implemented on standard GPU architectures. As a local algorithm for optical flow estimation, the structure tensor is advantegous because it readily differentiates between the 3 important cases of motion of lines, motion of points(distributed structure) and presence of higher order terms (higher order motions and/or uncorrelated noise). The algorithm goes as follows[2] (where v_{ix} is the x component of the ith eigenvector, and $l_{thres} \in [0,1)$ is a threshold).
For every Ω of interest:

- if $\frac{\lambda_3 - \lambda_2}{\lambda_3 + \lambda_2} > l_{thres}$, then Ω contains line motion given by:
$$\vec{u}_l = \frac{v_{3t}}{v_{3x}^2 + v_{3y}^2} \begin{pmatrix} v_{3x} \\ v_{3y} \end{pmatrix} \quad (2)$$
- else if $\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1} > l_{thres}$ then Ω contains point motion given by:
$$\vec{u}_p = \frac{1}{v_{1t}} \begin{pmatrix} v_{1x} \\ v_{1y} \end{pmatrix} \quad (3)$$

- else if $\left(1 - \frac{2\lambda_2}{\lambda_1+\lambda_2} + \frac{2\lambda_2}{\lambda_2+\lambda_3}\right) > l_{thres}$ then Ω contains higher order terms
- else Ω contains no structure (close to a constant gray-value).

2.1. Lucas and Kanade

The other classical approach to local optical flow, was suggested by Lucas and Kanade (LK) [14]. Originally, they did not formulate an analysis of spatio-temporal volumes, but rather considered template matching/registration between two images. However, it is a straightforward improvement to replace the original elements of the LK formulation, with that of spectral moments (as is done in [6]). The derivation of the LK algorithm starts with the optical flow constraint (first order approximation of the brightness constancy):

$$I_t = -\vec{u}^{\mathrm{T}} \nabla I$$

which yields the following error to minimize:

$$E_{LK} = \iiint_{\Omega} (I_t(\vec{x}) + \vec{u}^{\mathrm{T}} \nabla I(\vec{x}))^2 \tag{4}$$

For the case when distributed spatial structure exists ($\alpha < 1$), Eq. 4 is minimized by:

$$\vec{u}_{LK}(\vec{x}) = G^{-1} \vec{b}(\vec{x}) \tag{5}$$

for

$$\vec{b}(\vec{x}) = \begin{pmatrix} m_{101} \\ m_{011} \end{pmatrix} = \begin{pmatrix} \sum w I_x I_t \\ \sum w I_y I_t \end{pmatrix}$$

This corresponds to the special case of point motion for the structure tensor algorithm. This can be seen by rewriting Eq. 4 as

$$\begin{aligned} E_{LK} &= \iiint_{\Omega} \begin{pmatrix} \vec{u} \\ 1 \end{pmatrix}^{\mathrm{T}} \nabla_3 I^{\mathrm{T}} \nabla_3 I \begin{pmatrix} \vec{u} \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} \vec{u} \\ 1 \end{pmatrix}^{\mathrm{T}} G^{3D} \begin{pmatrix} \vec{u} \\ 1 \end{pmatrix} \end{aligned}$$

and recognizing that for a stable solution, the matrix must have a unique minimum eigenvalue, and that Eq. 3 then yields an equivalent. However, this connection between the 3D structure tensor and the LK algorithm does not mean that the two methods are equivalent. The structure tensor comes with additional information in terms of confidence measures and estimates, not only for point motion, but for line motion and presence of higher order terms as well. The LK algorithm requires regularization in order to be stable outside of any region except that of point motion. A common approach is to apply a diagonal Tikhonov regularization as is done in [7]. This adds the problem of how to set the regularization parameter, a factor that penalizes the magnitude of the estimated flow-vectors.

However, two clear benefits of the LK algorithm are also evident: firstly, it avoids making use of m_{002}, which is essentially the variance of the time derivative(an element commonly suffering from noise), and secondly no eigensystem analysis is required. While theoretically the two methods are identical within regions of well distributed structure, the LK algorithm is more practical (when factoring in limited hardware, such as fixed point representation), and usually faster to compute as well.

In what follows, we will combine both algorithms with the aim of retaining beneficial properties of both.

2.2. Lucas and Kanade meets the Structure Tensor

For the case of linear symmetry ($\alpha \approx 1$), G becomes singular and $\vec{u}_{LK}$ is undefined. For such cases, we wish to estimate the motion of line patterns, but we do not wish to resort to the full eigen-analysis of G^{3D}. Instead, we note that at a single point, the normal flow is given by

$$\vec{p}_n = \frac{\nabla I \nabla^{\mathrm{T}} I}{\nabla^{\mathrm{T}} I \nabla I} \vec{u} = -\frac{\nabla I}{|\nabla I|^2} I_t$$

Averaging the normal flow over a region yields the line flow, *iff* the region is linearly symmetric. By using a smooth window w we obtain the following expression:

$$\vec{u}_n(\vec{x}) = \sum w \vec{p}_n = -\frac{\vec{b}(\vec{x})}{\mathrm{Tr}(G(\vec{x}))} \tag{6}$$

The two methods $\vec{u}_{LK}$ and $\vec{u}_n$, by Eq. 5 and 6 have distinct domains of good performance, determined by α. This motivates the use of a weighting function $w(\alpha) \in [0, 1]$:

$$\begin{aligned} \vec{u}_l &= w(\alpha)\vec{u}_n & (7) \\ \vec{u}_p &= (1 - w(\alpha))\vec{u}_{LK} & (8) \\ \vec{u}_w &= \vec{u}_l + \vec{u}_p & \end{aligned}$$

We call $\vec{u}_l$ the line flow, and $\vec{u}_p$ the point flow. Normal flow should not be confused with line flow. Normal flow is defined for every pixel whereas line flow is the averaging of normal flow *iff* the region is linearly symmetric.

Many possibilities are available for a weighting function. For example, using the brick wall thresholding function ($w = \alpha > l_{thres}$) will give near identical output to the structure tensor, concerning the differentiation between line and point flow. Other possibilities include a soft thresholding, using some sigmoidal function as a fuzzy delimiter. However, we have found promising results using the following weighting function:

$$w = \alpha^2 = 1 - 4\frac{|G(\vec{x})|}{\mathrm{Tr}(G(\vec{x}))^2} \tag{9}$$

This weighting function also provides for an especially effecient implementation, which we can see by substituting Eqs 7 and 8 with Eq. 9 and get:

$$\vec{u}_l = -\frac{(m_{200}-m_{020})^2+4m_{110}^2}{(m_{200}+m_{020})^3}\begin{pmatrix} m_{011} \\ m_{101} \end{pmatrix} \quad (10)$$

$$\vec{u}_p = \frac{4}{(m_{200}+m_{020})^2}\begin{pmatrix} m_{101}m_{110}-m_{011}m_{200} \\ m_{011}m_{110}-m_{101}m_{020} \end{pmatrix} \quad (11)$$

The combination of line and point flow using the weighting function of Eq. 9, can be seen as a local regularization of the LK method. Eq. 5 is ill-posed for the case of linear symmetry (the top left pane of Fig. 1 illustrates this). A common local approach is to apply a diagonal Tikhonov regularization, which in our case means adding a small positive value to the elements m_{200} and m_{020}. This will give more stable results (lower left pane of Fig. 1), which was noted in the work of [7] but still leaves the problem of separating the point flow from the line flow and causes the new problem of how to set the regularizing parameter(we manually adjusted for best result in Fig. 1. A fundamental problem with Tikhonov regularization is that it introduces a bias for smaller solutions, by changing the error minimized. Fig. 1 also shows the output of the classical 3D structure tensor (upper right pane), where line motions and point motions are differentiated automatically. Our suggested algorithm is illustrated in the lower right.

A problem comes about for large displacements in the sequence. This can be solved by processing images in a multi-scale pyramid. Coarser scales are then investigated first, and if motion is detected beyond the tolerance of lower levels, then one uses the coarser estimate (contending with a lower resolution). Better results can be achieved by local warping methods, but this requires an iterative procedure that does not lend itself well to parallel implementations. For the results of this paper, however, we found that a pyramid was not necessary. That is, for our data, one scale of interest was sufficient, as our algorithm provides good stability over the limited scale variability.

All algorithms visualized in Fig. 1 are local in nature, and require no iterative procedures, they all easily run in real-time in our Matlab implementation for resolutions up to 256 by 256 pixels (optimized implementations involving the GPU will greatly improve this). For the results that follow in this paper, it is done with our Matlab implementation on regions of interest of 128x128 size images, yielding a 41x41 size flow field at 200 frames/second on a laptop with Intel dual-core 1 GHz processor, 8 GB of RAM (roughly the same as the Tikhonov regularized LK version). The implementation is not dependent on any specialized hardware(the GPU is not used), although c code was constructed as a mex-module for fast calculation of the gradient fields.

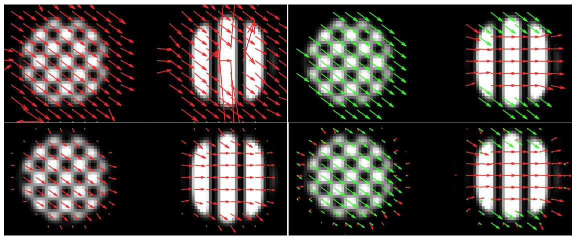

Figure 1. Estimated optical flow overlayed on a test sequences of two circular regions on the move. True motion is uniform translation in -40 degrees. Top left: the Lucas and Kanade(LK) algorithm with no regularization. Bottom left: LK with Tikhonov regularization(manually adjusted parameter). Top right: classical 3D structure tensor algorithm, with differentiated line(red) and point(green) motions. Bottom right: our suggested algorithm, with differentiated line and point motions.

3. Divergence as a Low-level lip-dynamics feature

In this paper, we investigate local features and take an approach that is inspired by classical shape from motion methodology from Koenderink and van Doorn [12], and investigate the 3 invariants: divergence(div), curl and deformation(def), that are by construction invariant to translation and static rotation. Locally, the flow is described by ($\vec{u} = \{u, v\}$):

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \begin{pmatrix} u_x & u_y \\ v_x & v_y \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}$$

where u_0 and v_0 is the translation. We can then write:

$$\begin{pmatrix} u_x & u_y \\ v_x & v_y \end{pmatrix} = \frac{div}{2}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \frac{curl}{2}\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} + \frac{def}{2}\begin{pmatrix} \cos 2\phi & \sin 2\phi \\ \sin 2\phi & -\cos 2\phi \end{pmatrix}$$

for:

$$\begin{aligned} div &= u_x + v_y \\ curl &= u_y - v_x \\ def &= \sqrt{(u_x - v_y)^2 + (u_y + v_x)^2} \end{aligned}$$

The approach of affine optical flow[1] estimates the differential invariants of the flow directly in a single minimization(essentially regression) procedure. This approach leads to a 6 by 6 matrices at every image position and is the basis of more costly iterative algorithms. We instead take the approach of estimating the invariants using derivatives of the underlying (non-affine) flow field.

Static rotation is described by ϕ, and could also be called a constant roll angle offset of the camera or face, whereas

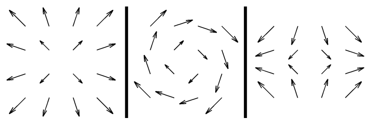

Figure 2. Regions of pure divergence(left), pure curl(center) and pure deformation(right). Size is 4 by 4.

dynamic rotation is the change in the roll during measurement by some angular speed of rotation(e.g. head wobbling). In our case, the static rotation is done away with by descriptors that do not depend on (ϕ), which is true for div, $curl$ and def. For dynamic rotation, its effect will vary, depending on how far away from the center of rotation we are taking measurements. If we take measurements over the center of the dynamic rotation, then it will affect only curl, otherwise it will affect a combination of translation and curl coefficients. We therefore contend to use only div and def as our local invariants.

Fig. 2 shows the appearance of the three patterns that the invariants are tuned to. Divergence is a measure correlating to the opening(positive) or closing(negative) of the mouth, and the deformation its asymmetry. The curl pattern on the other hand, dominates in the case of camera/facial rolling.

Interestingly, the invariants and the theory of the generalized structure tensor[3] are fundamentally connected. To see this we encode the flow field as a complex number, $\hat{u} = u + \mathbf{i}v$. On this we apply complex filters called symmetry derivatives:

$$\Gamma^{\{p\}}(\vec{x}) = (D_x + \mathbf{i}D_y)^p \frac{1}{2\pi\sigma^2} e^{-\frac{|\vec{x}|^2}{2\sigma^2}} \tag{12}$$

where p and σ are the order and the scale of the filter respectively. For filters of order $p = 1$ and $p = -1$ (symmetry derivatives $(D_x + \mathbf{i}D_y)$ and $(D_x - \mathbf{i}D_y)$ respectively) we have:

$$\begin{aligned} div &= \text{Re}\, \Gamma^{\{-1\}} * \hat{u} \\ curl &= \text{Im}\, \Gamma^{\{-1\}} * \hat{u} \\ def &= |\Gamma^{\{1\}} * \hat{u}| \\ \phi &= 2\angle\left(\Gamma^{\{1\}} * \hat{u}\right) \end{aligned}$$

where $*$ indicates convolution. Illustration of how these two filters look like in practice are found in Fig 3. To our knowledge, no such application of symmetry derivatives has been used in local shape from motion. Two fundamental benefits come to light using the symmetry derivatives:

Firstly, it offers a way of generalizing to higher orders, by considering other values for p (although in our application, we have found these less descriptive).

Secondly, by inspecting the complex filters as magnitude and angle, instead of as coarse scale partial derivatives, we see a possibility for tuning. It is the angle of the filters that

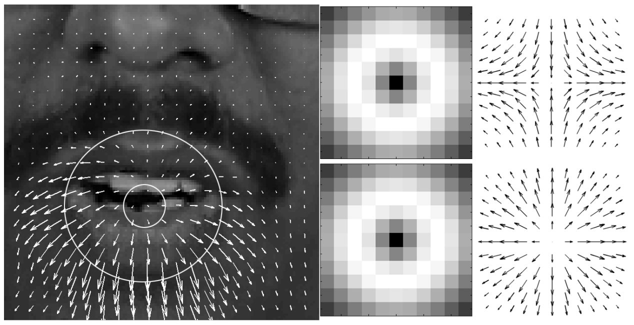

Figure 3. Left pane: a frame (128 by 128 pixels) of a manually extracted ROI, of user '002' of the XM2VTS database. The actual resolution used for experiments where 64 by 64 pixels. White arrows are the estimated optical flow (21 by 21 vectors). The annulus region delimited by the two white circles indicates the shape and size of the symmetry derivative filters used ($p = 1$ and $p = -1$ have the same size). The position of the filter is determined automatically by the maximum of divergence and deformation ($\vec{x}_{obj}(t)$). Right pane, top line: illustrates the actual size of the $p = -1$ filter as magnitude(left) and full complex values(right).Right pane, bottom line: same as top line for $p = 1$.

pick up on the relevant structure, and the annulus shaped magnitude indicates an area of interest(i.e. the support of the features). In our application, we found the best results for the actual derivatives of Gaussians (yielding the rotationally symmetric annulus of Fig 3). However, if one abandons the demand for rotation invariance, design of asymmetric magnitude filters could be produced that are more tuned to the average shape of lips.

We use div and def in our scheme to both find and describe a position of interest:

$$\vec{x}_{obj}(t) = \arg\max_{x} \{|div(\vec{u}(\vec{x},t))| + def(\vec{u}(\vec{x},t))\}$$

One such position is illustrated in the left pane of Fig 3. This is not a conventional mouth tracking algorithm. Only when visual speech is occurring does $\vec{x}_{obj}(t)$ give an estimate on the position of the lips.

This reasoning is similar to that of [5] where optical flow events are analyzed by trained spatio-temporal patterns. Our approach is intended to be more expressive as we use true invariants for 'spatio-temporal patterns', that are by construction orthogonal and measure physical quantities rather than trained on a limited set of data.

Of the two invariants selected, we have clearly noted that the divergence is the more important one, and we will explore using it for estimating the change in lip shape, and show it to offer a robust method for detecting opening and closing events of visual speech and semi-automatic lip segmentation.

4. Application of flow divergence

We denote with $A(t)$ the area of the outer lip contour, and expect $\dot{A} \propto div(\vec{x}_{obj}, t)$. Thus,

$$A(t) = a + b \int_0^t div(\vec{x}_{obj}, \tau)\,\mathrm{d}\,\tau$$

for some initial condition a, and constant of proportionality b. For practical estimation of A, we can use an accumulator variable that is updated at each time instant. For stability, one can introduce an equilibrium area (A_{eq}), and ensure that the estimate has a tendency to converge to equilibrium in the absence of divergence. Additionally, one should also enforce $A \geq A_{min} \geq 0$. This motivates the use of the following algorithm:

$$\begin{aligned}
A_{div}(0) &= a \\
& \quad \text{for each } t > 0: \\
accum &= A_{div}(t-1) + b\, div(\vec{x}_{obj}, t) \\
A_{div}(t) &= \max\left(accum + c(A_{eq} - accum), A_{min}\right)
\end{aligned}$$

where $c \in [0, 1)$ is a constant determining the speed of convergence to equilibrium. Fig. 6 illustrates typical results using this algorithm on speakers in the XM2VTS database [16].

The parameters a, b and c can be estimated empirically from the data set by a simple procedure. The parameter $a = A_{div}(0)$ is the initial state of the mouth opening. Assuming that the user is initially not speaking, with relaxed closed lips in the beginning, we set a to equal the A_{eq}, which can be estimated from the groundtruth data as the average lip area at rest. Too high value of c will introduce premature mouth closings, whereas too low allows for drift in the signal. One can measure the drift as a slow varying change in the mean of the signal. One can set c as small as possible to avoid such drift. Finally, after setting a and c, the parameter b is a scalar for multiplying the entire sequence, and can be set from the training data by a regular least squares fitting.

4.1. Semi-automatic Lip segmentation

We used this application to manually segment 50 subjects lip contours from the XM2VTS database [16]. One session per user, out of their total of 8 sessions available segmented. With the XM2VTS database the output of an automatic lip segmentation software is provided based on the work in[18]. For the majority of the subjects in the database this algorithm performs quite well as seen by visual inspection. On certain subjects however, such as ’002’ illustrated in fig. 5, the segmentation is done less well. Also, there is a noticable lack of temporal smoothness in the segmentation so that even if the square error is low per image, the

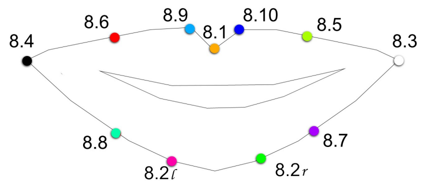

Figure 4. The standard point notation with color coding used in the software, slightly different from MPEG-4 as more detail is provided by lower lip points.

temporal derivatives of the keypoint position display irregularity (this especially effects the usability of such segmentation for the case of our algorithm for lip events detection, that is based on the derivate of shape over time). With our semi-automatic segmenter it is usually enough with 2 or 3 provided keyframes of groundtruth to segment the full sequence near perfect as seen by visual inspection. This is made even more efficient by the function to initiate the software with output of the segmentation of [18], which can then be easily adjusted. A simple user interface was developed where control points of the curve can be dragged around manually in a click and drag interface. The control points confirm to the MPEG 4 standard of facial parameter description for the outer lip[11] (point set 8 of the standard) except that two points are used to replace the lowest point of the lower lip (thus points 8.2l and 8.2r are positioned left and right of MPEG standard point 8.2) as was done in [18]. A color coding has been selected in the software to distinguish the points from maximally different pure saturation colors in HSV color space, except for points 8.3 and 8.4(right and left corners of outer lip contour) who are coded as black and white respectively. Fig 4 illustrates the convention used.

When points are modified, they are then tracked forward and backward in the sequence using our optical flow algorithm as the basic input, determining the time dependent curve. Interpolation occurrs only inbetween the new keyframe, and the neighouring keyframes. In addition to simply tracking the keypoints using the flow vectors, we implemented 3 optional sources of tracking information/constraints (available as tick boxes in the software, although affecting the timeliness of the software drastically):

- **Static feature based** where correlation between local 2D image features of the nearest keyframe provide information, (Option of Gabor and HOG type of features can be used).

- **MPEG 4 key-point constraints** where the nature of the different key-points are factored in (this option should be turned off for non-mouth region segmentation). The right and left corners of the contour remain unconstrained, but the others gain a constraint of

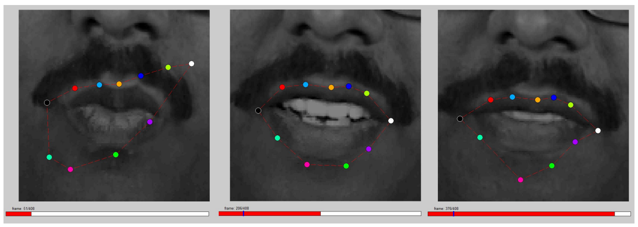

Figure 5. Screen shots of the software for semi-automatic lipsegmentation application. Left: the result of automatic lipsegmentaion provided from the XM2VTS homepage (average results are better than this user). Center frame: 150 frames later in the same sequence, after the frame in the left pane was manually segmented (tracking by flow only). Blue line on the red search bar indicates the key-frame where the user provided groundtruth. Right frame: close to the end of the session, the tracking has diverged from true segmentation, and user should provide a new key-frame.

slowly varying distance ratio between points (length of edges to nearest keypoints), and

- **Global Area Constraint** where the divergence measure of a filter centered over the polygon is used to estimate the area change, and this area change is used to uniformly scale all the estimates.

As a further option in the segmentation application is a gamma correction term, which scales the magnitude of the flow vectors as a pre-processing step, and or the option to make use of a two-level pyramid for handling faster motions.

We tested point trackers in the professional softwares of Adobe After Effects, Nuke by The Foundry, and imagineer systems Mocha PRO. Even though Mocha PRO is a planar tracker not specifically designed for this type of problem, it performed best both in speed and performance for our task (out of the three commercial systems). In all subjects tried, our algorithm provided a better segmentation (by visual inspection) than Mocha PRO, with fewer keyframes provided by the user, and with faster tracking results when the three options mentioned above were turned off. Considering that our software is fully implemented in matlab, we consider this to be very promising result. However, these are qualitative rather than quantitative results on the perfomance. Quantifying these findings are difficult to do as we are dealing with a semi-automatic task. Factors for the task is not only the finished fit of the curve, (including temporal consistency), but the number of times the user gave input, and the amount of input the user gave each time. Because our segmenter finishes the task of updating the segmentation faster than Mocha, those who have used our system tends to provide more keyframes, because the cost in terms of personal inconvenience is lower for doing so.

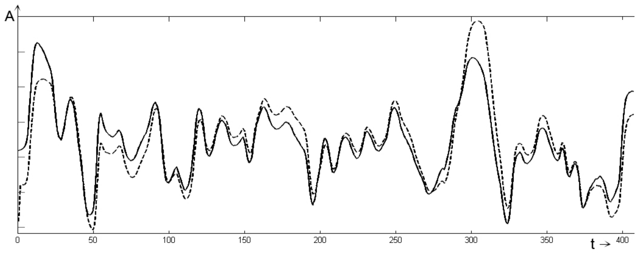

Figure 6. Results on estimating the lip-contour area of user '002', session 1 of the XM2VTS DB. Dotted line: estimated area. Solid line: Ground-truth area from manual mouth segmentation.

4.2. Lip Events Detection

In Fig. 6 we can clearly observe visual events, in particular the exact occurence of mouth opening and closing. They are defined by the dominant extrema. Both opening and closing have $\dot{A} = div(\vec{x}_{obj}) = 0$, but opening has $\ddot{A} = \dot{div}(\vec{x}_{obj}) < 0$ while closing has $\ddot{A} = \dot{div}(\vec{x}_{obj}) > 0$.

A threshold value ϵ_1 is used for detection of the temporal positions of the events by $|div(\vec{x}_{obj})| < \epsilon_1$. A simple two point derivative filter is then applied on the divergence estimation where a second test is performed to make sure it is not a local plateu point by $|\dot{div}(\vec{x}_{obj})| > \epsilon_2$. Assuming both tests are passed, the point is classified as the event given by the sign of $\dot{div}(\vec{x}_{obj})$. The short segments of the signal where detection is positive, are replaced by their temporal averages (centroids) as the output of the detection and compared with groundtruth.

Given that ϵ_1 and ϵ_2 are set in a reasonable range, our algorithm finds all events as given in the groundtruth data, with an average deviation of about 40 ms in temporal alignment. In real life situations, a correct ϵ value depends both on the speed of speaking, as well as the size of the user mouth region/and or distance to the camera. For a given stretch of signal of div and def, the epsilon can be given as proportional to the energy in the signal, assuming the user mode of speech varies more slowly than the variation within shorter durations of visemes.

5. Discussion and Conclusion

Our approach to optical flow estimation has been shown to be useful for the tasks of semi-automatic lip segmentation of the XM2VTS database, and has potential to be a useful tool for researchers in vision sciences for generating groundtruth segmentations from video. It has been shown to perform considerably better and faster than commercial software, while still being implemented in the matlab environment.

The software can be made to perform even better by further optimizing its implementations of three optional sources of tracking information listed in section 2, especially the tracking of static 2D based features. A natural step to increase the speed would be to port it into a more suitable setting, such as openCV. However, a Matlab implementation holds value initself, as it is still widely used throughout the wider vision community, especially where highly optimized software is less needed over reliability and ease of use (as is often the case in perceptional studies).

The suggested application of lip based articulatory events detection has been shown to perform well on the XM2VTS database for the cases of mouth opening and closing events, as defined by the first and second order time derivative of outer lip contour area. Also the area itself can be estimated, but with less accuracy and with the open question of how to set the three parameters of the algorithm in an optimal way.

In general, the robustness and speed of computation, together with invariance properties of the features, opens up the possibility of implementations on smart phones with GPU architectures with applications ranging from audio-visual speech and speaker recognition and real-time avatar lip-synchronization.

References

[1] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. *ECCV*, 15:237–252, 1992.

[2] J. Bigun. *Vision with Direction*. Springer, Heidelberg, 2006.

[3] J. Bigun, T. Bigun, and K. Nilsson. Recognition by symmetry derivatives and the generalized structure tensor. *IEEE-PAMI*, 26:1590–1605, 2004.

[4] J. Bigun, G. Granlund, and J. Wiklund. Multidimensional orientation estimation with applications to texture analysis and optical flow. *IEEE-PAMI*, 13(8):775–790, 1991.

[5] M. J. Black. Explaining optical flow events with parameterized spatio-temporal models. In *CVPR*, pages 326–332, 1999.

[6] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *IJCV*, 61:211–231, 2005.

[7] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *In European Conference on Computer Vision*. Springer, 2006.

[8] M. I. Faraj and J. Bigun. Lip biometrics for digit recognition. In *International Conference on Computer Analysis of Images and Patterns, CAIP2007, LNCS 4673*, pages 360–366. Springer, 2007.

[9] M. I. Faraj and J. Bigun. Speaker and digit recognition by audio-visual lip biometrics. In *The international conference on Biometrics (ICB) 27-29 Aug, Seoul, S. Korea*, pages 1016–1024. Springer, LNCS, 2007.

[10] B. Horn and B. Schunck. Determining optical flow. *Artificial intelligence*, 17:185–203, 1981.

[11] ISO-IEC. *International standard ISO/IEC 14496, Information technology generic coding of moving pictures and associated audio information: Video. Ref. No.: ISO/IEC:14496*. ISO, 1996.

[12] J. J. Koenderink and A. van Doorn. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta*, pages 773–791, 1975.

[13] J. J. Koenderink and A. J. van Doorn. Surface shape and curvature scales. *Image and Vision Computing*, 10:557 – 565, 1992.

[14] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the seventh Int. Joint Conf. on Artificial Intelligence, Vancouver*, pages 674–679, 1981.

[15] H. McGurk and J. MacDonald. Hearing lips and seeing voices. *Nature*, 264:746748, 1976.

[16] K. Messer, J. Matas, J. Kittler, and J. Luettin. Xm2vtsdb: The extended m2vts database. *In Second International Conference of Audio and Video-based Biometric Person Authentication, $ICSLP'$96*, pages 72–77, 1999. [Segmentation results online; accessed 31-September-2011].

[17] G. Potamianos, C. Neti, G. Gravier, A. Garg, and A. Senior. Recent advances in the automatic recognition of audiovisual speech. *Proceedings of the IEEE*, 91(9):1306–1326, 2003.

[18] M. R. Sanchez. *Aspects of facial biometrics for verification of personal identity*. PhD thesis, University of Surrey, 2000.

[19] M. Tistarelli. Multiple constraints to compute optical flow. *IEEE-PAMI*, 18:1243–1250, 1996.

[20] S. G. Zekeriya, S. Gurbuz, Z. Tufekci, E. Patterson, and J. N. Gowdy. Application of affine-invariant fourier descriptors to lipreading for audio-visual speech recognition. In *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pages 177–180, 2001.