



Robust Facial Features Tracking Using Geometric Constraints and Relaxation

Désiré Sidibé #¹, Philippe Montesinos *², Alain Trémeau #³

*Université Jean Monnet, Laboratoire Hubert Curien - UMR CNRS 5516
18 rue du Professeur B. Laurus, 42000 Saint-Etienne, France*

¹dro.desire.sidibe@univ-st-etienne.fr

³alain.tremeau@univ-st-etienne.fr

* *LGI2P, Ecole des Mines Ales*

Parc Scientifique Georges Besse, 30035 Nimes cedex 1, France

²philippe.montesinos@ema.fr

Abstract—This work presents a robust technique for tracking a set of detected points on a human face. Facial features can be manually selected or automatically detected. We present a simple and efficient method for detecting facial features such as eyes and nose in a color face image. We then introduce a tracking method which, by employing geometric constraints based on knowledge about the configuration of facial features, avoid the loss of points caused by error accumulation and tracking drift. Experiments with different sequences and comparison with other tracking algorithms, show that the proposed method gives better results with a comparable processing time.

I. INTRODUCTION

Automatic human face analysis and recognition has received significant attention during the past decades, due to the emergence of many potential applications such as person identification, video surveillance, teleconferencing and human computer interaction. An automatic face recognition usually begins with the detection of face pattern, and then proceeds to normalize the face images using information about the location and appearance of facial features such as eyes and mouth. Therefore, detecting faces and facial features is a crucial step. Many methods for solving the face detection problem have been proposed in the literature [1] and most of them can be put into a two-stage framework. The first stage focuses on face candidates, i.e. regions that may contain a face are marked. In the second stage, the face candidates are sent to a “*face verifier*”, which will decide whether the candidates are real faces or not. Different methods put emphasis on one or the other of these stages [2].

In this work, we are interested in the detection and tracking of facial features in a human computer interaction application. Among all facial features, eyes can be considered the most salient and stable [3], [4]. We present a simple and efficient eye detection method based on both skin color modeling and geometric characteristics of human eyes. Based on the positions of eyes, other facial features such as nose and mouth are detected. The method is simple since it needs no training examples of eyes and is robust to face rotation in the image

plane. Once facial features are detected in a frame, they are tracked in the video sequence. We present a robust technique for tracking these facial features using geometric constraints about their configuration. Exploiting the knowledge that the features being tracked belong to a face, no points are lost during tracking. This knowledge is used through a relaxation scheme.

The remainder of the paper is organized as follows. In Section 2 we review some existing work about facial features detection and tracking. The proposed facial features detection method is described in Section 3. Then, the robust tracking algorithm is addressed in Section 4. Some experimental results showing the validity of the method, are given in Section 5. Finally, concluding remarks are given in Section 6.

II. RELATED WORK

Automatic detection of facial features in video sequences is generally performed after a face is detected in a frame. Many approaches have been proposed to detect faces [1]. Recently, methods based on machine learning techniques have proven to be very effective for face detection and the detector proposed by Viola and Jones is now one of the most popular [5]. Eyes can be considered the most salient and stable features in a human face in comparison with other facial features. Therefore, extraction of eyes is often a crucial step in many face detection algorithms [3], [4]. Many image-based eye detection techniques have been proposed recently and a review on eye detection techniques can be found in [6]. Han *et al.* [3] use morphological operations and a labeling process to search for potential face regions. Then, they use a trained backpropagation neural network to identify faces and their locations. Similar ideas are used by Wu and Zhou [2]. They employ size and intensity information to find eye-analogue segments from gray scale image, and exploit geometrical relationships to filter out the possible eye-analogue pairs. Huang and Wechsler [7] use genetic algorithms to evolve some finite state automata to discover the most likely eye locations. Then, optimal features are selected and a decision tree is built to classify whether the most salient locations identified earlier are eyes. Kawaguchi and Rizon [8] use intensity and edge

information to locate the iris. The main techniques they use are template matching, separability filter and Hough transform. Song et al. [6] use similar ideas to detect eyes based on multi-resolution wavelet transform.

The main objective of tracking is to roughly predict and estimate the location of a target object in each frame of the image sequence. In the context of face tracking, proposed methods can be classified into two categories. On the one hand, if the target being tracked is the entire face region, then kernel-based methods such as Mean Shift [9] or CamShift (Continuously Adaptive Mean Shift) [10] are used and give good results. On the other hand, if individual features are tracked, then a point tracker is used. Several point trackers have been proposed. Among them, it is worth mentioning the Kanade-Lucas-Tomasi, KLT for short, tracker [11] [12]. KLT tracker is used to track facial features by Bourel *et al.* [13] and by Colmenarez *et al.* [14]. It is based on minimizing a measure of dissimilarity between the appearance of a feature in the current and the last frames. In general, for reliable and fast processing, the maximum interframe displacement is limited. Spors and Rabenstein [15] use a luminance-adapted block matching technique to track eyes in a video sequence. Block matching is performed using a reference eye pattern and the extracted eye region from the previous frame. To cope with temporal changes in brightness, after the matching process, each reference pattern is adapted by the difference between the detected eye region in the current frame and the reference pattern.

The main problem with these tracking methods is the loss of points during tracking. A point may be lost because of illumination variation, head motion, temporary occlusion, or gradual error accumulation. Moreover, if there is a large interframe displacement, error in features localization increases and may cause a drift away from the correct positions. To deal with these problems, Bourel *et al.* [13] propose a method to recover the lost points using their known positions in previous frames. In this work, facial features are recovered based on some empirical knowledge about the type of feature, i.e. lip corners, nose, etc.

The relaxation labeling technique, which was first introduced by Rosenfeld *et al.* [16], has been widely used in the computer vision community. The principal idea of a relaxation scheme is to use the information provided by the neighbourhood of each feature in order to improve consistency and reduce ambiguity. One main limitation of this technique is its high complexity when using a large number of features. However, it has been shown that this complexity can be significantly reduced employing a sparse matrix representation [17]. Moreover, in the context of facial features tracking, the number of features is generally very small.

III. FACIAL FEATURES DETECTION

Our facial features detection method is based on skin color information. A first step of skin detection is used to reduce the search region for facial features. Human skin color is widely used as an important cue for face detection. Many different

color spaces have been employed and Terrillon *et al.* [18] have shown that the tint-saturation-luma (TSL) space and the normalized RGB space provide best results for Gaussian models.

A. Skin color modeling and detection

Skin color distribution can be modeled by an elliptical Gaussian probability density function (pdf), defined as:

$$f(c|skin) = \frac{1}{2\pi|\Sigma_s|^{1/2}} e^{-\frac{1}{2}(c-\mu_s)^T \Sigma_s^{-1}(c-\mu_s)}, \quad (1)$$

where c is a color vector and (μ_s, Σ_s) are the distribution parameters. These parameters are estimated from a training sample. We used a set of 1,158,620 skin pixels, manually selected from different images. The images are chosen in order to represent people belonging to several ethnic groups, and a wide range of illumination conditions.

A more sophisticated model, a mixture model, is often used in the literature. It is a generalization of the single Gaussian model. The pdf in that case is the sum of several single Gaussians. However, Caetano *et al.* [19] have shown that both models give comparable results, except when a high true positive rate is needed (more than 80%). In that case, the performance of mixture models exceeds single model's performance. Since we use skin detection as an initial step to reduce the search region for facial features, a single Gaussian model is adopted.

Once the skin color distribution's parameters are obtained from the training sample, the Mahalanobis distance from a color vector c to mean vector μ_s , given the covariance matrix Σ_s , is used to measure how "skin like" the color c is:

$$\lambda_s(c) = (c - \mu_s)^T \Sigma_s^{-1} (c - \mu_s). \quad (2)$$

Given an input image, for each pixel x , $x = (r, g)$ in the normalized RGB color space, x is considered a skin pixel if the distance $\lambda_s(x)$ is less than a predefined threshold [18].

B. Eye detection

The eye detection method starts by considering as potential eye regions, the non-skin regions within a detected face region. Obviously, eyes should be within a face region and eyes should not be detected as skin by the skin detector. The use of skin color information makes a clear difference with the work of Wu and Zhou [2] and Han *et al.* [3] where eyes are detected based on the assumption that they are darker than other parts of the face. In these methods, eye-analogue segments are found in the entire image resulting in a high number of possible pairs to check. On the contrary, we have to find eye-analogue pairs among a reduced number of potential eye regions.

An ellipse is fitted to each potential eye region using connected component analysis. Let R_k be a potential eye region and (x_k, y_k) its centroid. Then R_k , reduced to an ellipse, defines a_k , b_k and θ_k which are, respectively, the length of the major axis, the length of the minor axis and the orientation of the major axis of the ellipse. Finally, a pair of potential eye regions is considered as eyes if it satisfies

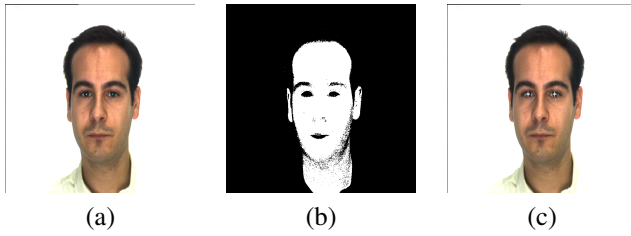


Fig. 1. Skin and eye detection example. a) Original image. b) Skin region detected. c) Eye detection result (eyes centers are depicted by white crosses).

some constraints based on anthropological characteristics of human eyes. Let R_i and R_j be two potential eye regions. Then (R_i, R_j) corresponds to a pair of eyes if the following equations are satisfied:

- $$\begin{cases} 1 < \frac{a_i}{b_i} < 3 \\ 1 < \frac{a_j}{b_j} < 3 \end{cases} \quad (3)$$

- $$|\theta_i - \theta_j| < 20^\circ \quad (4)$$

- $$\frac{a_i + a_j}{2} < d_{ij} < 3 \frac{a_i + a_j}{2} \quad (5)$$

The parameters in (3) and (5) are chosen according to the fact that for human eyes, if we denote by w_e and h_e respectively the width and the height of an eye, the average value for w_e/h_e is 2 and, averagely, the distance between two eyes is $d_{ij} = 2w_e$ [2]. Equation (4) is based on the fact that the two major axis should have the same orientation. A final constraint is the alignment of the two major axis, i.e. for two eye regions they belong to the same line.

Using these rules, the algorithm sometimes detects not only eyes, but also eyebrows. To discard regions corresponding to eyebrows, we use the fact that the center part of an eye region is darker than other parts. Then a simple histogram analysis of the region is done for selecting eye regions, since, an eye region should exhibit two peaks while an eyebrow region shows only one. An example of eye detection result is shown in Fig. 1. Once eyes are detected, the position of the nose is obtained based on the symmetry of facial features.

IV. ROBUST TRACKING ALGORITHM

As mention before, the main problem with points tracking methods such as KLT or block matching, is the loss of points during tracking. A point may be lost because of different reasons. In the context of facial features tracking, the main causes include illumination variation, head motion, temporary occlusion, or gradual error accumulation. Moreover, in case of fast head motion, there is a large interframe displacement and the correct positions of facial features are lost rapidly. Nevertheless, lost points can be detected and their correct positions can be recovered. This strategy has been adopted by Bourel *et al.* [13] who propose a method to recover lost points using their known positions in previous frames and some knowledge about the type of each individual feature.

Instead of recovering lost points, our goal is to design a robust tracking method which avoid imprecisions in facial features localization.

A. Using geometric constraints through a relaxation scheme

Imprecisions of tracking algorithms are due to the fact that the features are tracked individually. Thus, a facial feature's position is estimated from frame to frame independently from the positions of other features. However, one knows that the positions of facial features are not independent and are related by some geometric constraints. These constraints are imposed by the configuration of human faces and can be employed in the tracking algorithm through a relaxation scheme.

Let us consider two sets of features $u = \{u_1, \dots, u_n\}$ and $v = \{v_1, \dots, v_m\}$, from two different images I_t and I_{t+1} . If we define for each feature u_i a set of initial probabilities $p_i^0(k), k = 1, \dots, m$; $p_i^0(k)$ being the probability that u_i is associated with v_k , then, a relaxation process is designed to update the probabilities until a consistent distribution is reached [16]. One standard updating rule is:

$$p_i^{t+1}(k) = \frac{p_i^t(k)q_i^t(k)}{\sum_k p_i^t(k)q_i^t(k)}, \quad (6)$$

where

$$q_i^t(k) = \sum_j w_{ij} \left[\sum_l p_{ij}(k, l) p_j^t(l) \right], \quad (7)$$

and $p_{ij}(k, l)$ is the probability that feature u_i can be associated with feature v_k under the condition that feature u_j is associated with v_l . The scalars w_{ij} are weights that indicate the relative influence of neighbouring features on each other.

The conditional probabilities, $p_{ij}(k, l)$, represent the contextual information that helps improving consistency and, thus, insure that correct features are associated. In our application of facial features tracking, we can use the constraints imposed by the configuration of human faces to compute these conditional probabilities. For example, if we are tracking the two eyes and the nose, then the triangle formed by these three features must be preserved from frame to frame. In other words, the displacement of all three features are interrelated. The geometric constraints can be expressed, in terms of conditional probabilities, as follows:

$$p_{ij}(k, l) = \prod_{\substack{j' \neq i; j' \neq j \\ l' \neq k; l' \neq l}} f(|(\overrightarrow{u_{j'} u_i}, \widehat{\overrightarrow{u_{j'} u_j}}) - (\overrightarrow{v_{l'} v_k}, \widehat{\overrightarrow{v_{l'} v_l}})|) e^{-\left(|\frac{d_{kl'}}{d_{ij'}} - \frac{d_{ll'}}{d_{jj'}}|\right)} \quad (8)$$

where the function f is defined by:

$$f(x) = \begin{cases} \frac{1-x}{\eta} & \text{if } x < \eta \\ 0 & \text{otherwise} \end{cases}$$

The different terms in (8) have following meanings:

- $p_{ij}(k, l)$ is the probability that feature u_i can be associated with feature v_k under the condition that its neighbour u_j is associated with v_l .

- d_{ij} is the Euclidean distance between feature u_i and feature u_j .
- $\widehat{(\vec{u}_j' u_i^t, \vec{u}_j' u_j^t)}$ is the angle formed by the two vectors $\vec{u}_j' u_i^t$ and $\vec{u}_j' u_j^t$.

Therefore, for any feature u_i and for each of its neighbour u_j , the triangles formed by u_i, u_j and any other feature u_j' , are preserved in the image sequence. Note that since the number of features being tracked is generally small, all features are considered to be neighbours with one another.

B. Self-reinitialization

In the proposed tracking method, as well as in the KLT or block matching methods, one has to specify a search window size. In order to make this parameter independent from the considered sequence, we set the search window size to be equal to the distance between the two eyes. This has the effect of automatically scaling the window for every sequence. However, a problem arises if the person moves towards or away from the camera. In that case, the window size is not adapted since the distance between eyes in the current frame is different from the distance computed in the first frame.

To overcome this issue, we introduce a reinitialization of the tracking procedure using the eye detector described in Section III-B. For each frame I_t , we define a measure Q_t of the precision of the tracking method by:

$$Q_t = \min\{sc(u_i^t, v_i^{t-1}), i = 2, \dots, n\}, \quad (9)$$

where $sc(u_i^t, v_i^{t-1})$ is the correlation coefficient between a window centered at feature u_i in I_t and a window centered at its corresponding position v_i in frame I_{t-1} . If Q_t exceeds a defined threshold, then the tracking procedure is reinitialized using the eye detector to get the new positions and search window size.

V. EXPERIMENTAL RESULTS

In this section, we show experimental results obtained by the proposed detection and tracking methods. For facial features detection, we use the AR face database [20] in order to compare our results with those described by Song *et al.* [6] and Kawaguchi and Rizon [8], who used the same database. For tracking, we used a set of 5 image sequences showing small, medium and large interframe displacement. The sequences are in MPEG1 format, 320x240 color pixels and at 25 frames per second.

A. Facial features detection

The AR face database contains color images of frontal view faces with different facial expressions, illumination condition and occlusions. For a direct comparison, we used the same subset of the database employed in [6] and [8], which contains 63 images.

A commonly used criterion for the performance evaluation of an eye detection method is *the relative error* defined by [21]:

$$err = \frac{\max(d_l, d_r)}{d_{lr}}, \quad (10)$$

where d_l is the left eye disparity, i.e. the distance between the manually detected eye position and the automatically detected position, d_r is the right eye disparity, and d_{lr} is the Euclidean distance between the manually detected left and right eye positions. In [6], the detection is considered to be correct if $err < 0.25$.

Using this criterion, the proposed detection method achieves a performance of 100% correct detection. An example of detection result is shown in Fig. 1c.

For comparison, Kawaguchi and Rizon [8] reported a correct detection rate of 96.8%, and Song *et al.* [6] a correct detection rate of 98.4%. The methods in [8] and [6] can deal with gray scale images but they need the detection of the reflected light dots as a cue for eye localization, which is not needed by our approach.

B. Facial features tracking

Given a sequence, we use the detection method described in Section III-B to detect eyes in the first frame. Then, three facial features, the two eyes and the nose, are tracked using the tracking method described in Section IV. We used five sequences whose length varies between 60 and 600 frames. The first three sequences show small interframe displacement, while the two last sequences contain fast head motion and, therefore, large interframe displacement.

For all of the 5 sequences used in our experiments, facial features have been correctly tracked with no loss despite large head motion in some of the sequences. Figure 2 shows an example of tracking results obtained by three methods, i.e. the KLT tracker, the block matching method and our relaxation based tracking, using Sequence #3. As it can be seen, the KLT tracker gives good results for the first frames of the sequence, but due to error accumulation, the features gradually drift from their correct positions and are totally lost at frame 130. The block matching method gives very poor results because it is based on correlation only. On the contrary, considering geometric constraints through the tracking procedure, our method can correctly tracks all feature in the entire sequence.

For a quantitative evaluation, let's define a tracking error for each facial feature in a sequence as follows:

$$\epsilon = \frac{1}{N} \sum_{i=1}^N disp_i, \quad (11)$$

where N is the length of the sequence, and $disp_i$ is the Euclidean distance between the manually detected position of the feature and its tracked position in frame i of the sequence.

For each sequence, we consider the mean value of the errors for all facial features as the tracking error. Table I shows tracking errors obtained by the three methods. As it can be seen, our relaxation-based tracker gives the best results. In particular, when the interframe displacement, i.e. the distance between the position of the features in two successive frames, is large, both the KLT tracker and the block matching method lead to important localization errors (Sequences #4 and #5). In contrast, our approach gives very good results in such cases as shown by an example in Fig 3.

TABLE II
AVERAGE PROCESSING TIME WITH A 1.7 GHZ CPU

	Block-Matching	KLT tracker	Our tracker
time per frame (ms)	44	66	46
# frames per second	23	15	22

Another important issue appears when one or both eyes are closed. In that situation, the correlation based block matching method and the KLT tracker fail because patterns in consecutive frames do not correspond. Considering the geometric relationship between features, makes our method robust to such intensity variation. Thus, closed eyes are correctly tracked.

The results shown in Table II summarize the average processing time needed for each of the three tracking methods on a 1.7 GHz CPU personal computer. As it can be seen, one important aspect of our method is that considering geometric constraints through a relaxation scheme does not increase the processing time per frame. This is because the number of tracked features is very small. So our method provides better results with a comparable processing time. Note that the processing time for the KLT tracker in Table II includes time needed to compute image derivatives.

VI. CONCLUSION

In this paper an efficient and robust facial features tracking method is proposed. It is based on, first, detecting facial features using a robust skin region detector which provides face candidates. Then using some simple rules derived from anthropological characteristics, eyes are selected within face regions. Once facial features are detected, they are tracked in the video sequence. We have shown how considering geometric constraints between facial features can make the tracking algorithm more robust and, thus, avoid the loss of points and tracking drift.

Experiments with different sequences show that our tracking method performs better than the well known KLT tracker with comparable execution time, in particular when the interframe displacement is large. It is also worth to mention that the proposed tracking algorithm is not limited to facial features tracking. It can be useful for general point tracking application.

However, both the detection and tracking algorithms searches for facial features in frontal views. So, the method do not apply for out of plane rotation. Trying to solve this overcome could be an interesting future research direction.

REFERENCES

- [1] M.-H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 34–58 (2002)
- [2] J. Wu and Z. H. Zhou, "Efficient face candidates selector for face detection," *Pattern Recognition*, vol. 36, pp. 1175–1186 (2003)
- [3] C. C. Han, H. Y. M. Liao, G. J. Yu, and L. H. Chen, "Fast face detection via morphology-based pre-processing," *Pattern Recognition*, vol. 33, pp. 1701–1712 (2000)
- [4] R. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detecting in color images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 696–706 (2002)
- [5] P. Viola and M. J. Jones, "Robust real-time face detection", *International Journal of Computer Vision*, vol. 57(2), pp. 137-154 (2004)

- [6] J. Song, Z. Chi, and J. Liu, "A robust eye detection method using combined binary edge and intensity information," *Pattern Recognition*, vol. 39, pp. 1110–1125 (2006)
- [7] J. Huang and H. Wechsler, "Visual routines for eye location using learning and evolution," *IEEE Transaction on Evolutionary Computation*, vol. 4, no. 1, pp. 73–82 (2000)
- [8] T. Kawaguchi and M. Rizon, "Iris detection using intensity and edge information," *Pattern Recognition*, vol. 36, pp. 549–562, (2003).
- [9] D. Comaniciu, V. Ramesh, and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift", *IEEE Proc. of International Conference on Computer Vision and Pattern Recognition*, pp. 142–149 (2000)
- [10] G.R. Bradski, "Computer video face tracking for use in a perceptual user interface", *Intel Technology Journal*, Q2 (1998)
- [11] J. Shi and C. Tomasi, "Good features to track", *IEEE Proc. of International Conference on Computer Vision and Pattern Recognition*, pp. 593–600 (1994)
- [12] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", *Proc. International Joint Conference on Artificial Intelligence*, pp. 674–859 (1981)
- [13] F. Bourel, C. C. Chibelushi, and A. A. Low, "Robust facial features tracking", *Proc. of 11th British Machine Vision Conference, Bristol, England*, pp. 232–241 (2001)
- [14] A. Colmenarez, B. Frey, and T. S. Huang, "Detection and tracking of faces and facial features", *IEEE Proc. of International Conference of Image Processing*, pp. 657–661 (1999)
- [15] S. Spors and R. Rabenstein, "A Real-Time Face Tracker For Color Video", *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1493–1496 (2001)
- [16] A. Rosenfeld, R. Hummel, and S. Zucker, "Scene labeling by relaxation operations", *IEEE Trans. Systems, Man and Cybernetics*, vol. 6, pp. 420–433 (1976)
- [17] S. Sidibe, P. Montesinos, and S. Janaqi, "Fast and robust image matching using contextual information and relaxation," in *Proc. of the 2nd International Conference on Computer Vision Theory and Applications (VISAPP)*, pp. 68–75 (2007)
- [18] J. Terrillon, M. N. Shirazi, H. Fukamachi, and S. Akamatsu, "Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images," in *Proc. IEEE International Conference on Face and Gesture Recognition*, pp. 54–61 (2000)
- [19] T. S. Caetano, S. D. Olabarriaga, and D. A. C. Barone, "Do mixture models in chromaticity space improve skin detection?," *Pattern Recognition*, vol. 36, pp. 3019–3021 (2003)
- [20] A. M. Martinez and R. Benavente, "The AR face database," Tech. Rep. 24, CVC, June 1998.
- [21] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust face detection using the hausdorff distance," in *Proc. of the Third International Conference on Audio and Video-based Biometric Person Authentication*, pp. 90–95, (2001).

TABLE I
COMPARISON OF TRACKING ERRORS FOR FIVE SEQUENCES

	Sequence #1	Sequence #2	Sequence #3	Sequence #4	Sequence #5
Block-Matching	2.42	8.05	21.14	33.77	39.86
KLT tracker	1.24	5.81	8.52	14.27	23.19
Our tracker	0.13	0.89	1.37	2.16	2.84

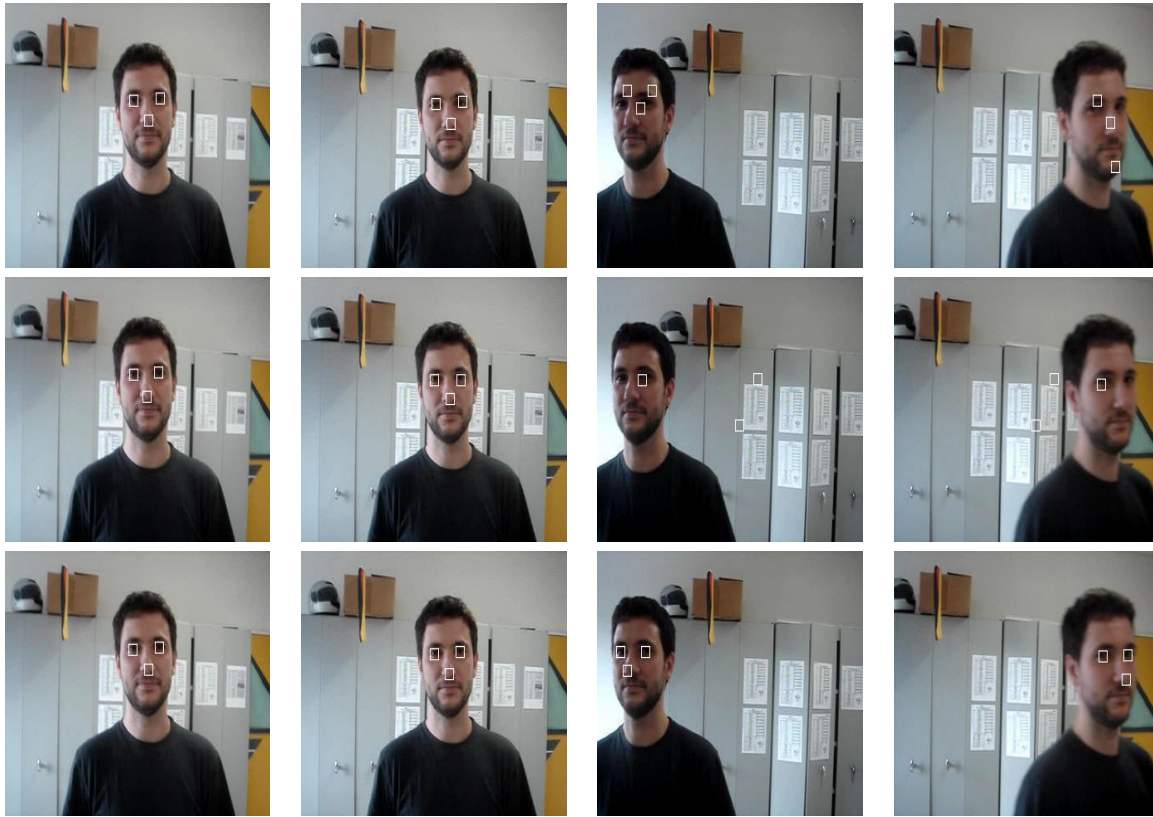


Fig. 2. Facial features tracking results. Each row shows the frames 1, 30, 80 and 130 of the sequence #3. First row: KLT tracker; Middle row: block matching method; Last row: our tracking method.



Fig. 3. A example of tracking result with our approach in the presence of large interframe displacement. Frames 1, 40, 60 and 110 of the sequence #5 are shown.