Refining Face Tracking with Integral Projections

Ginés García Mateos

Dept. de Informática y Sistemas, Universidad de Murcia
30.170 Espinardo, Murcia (Spain)
ginesgm@um.es

Abstract. Integral projections can be used, by themselves, to accurately track human faces in video sequences. Using projections, the tracking problem is effectively separated into the vertical, horizontal and rotational dimensions. Each of these parts is solved, basically, through the alignment of a projection signal –a one-dimensional pattern– with a projection model. The effect of this separation is an important improvement in feature location accuracy and computational efficiency. A comparison has been done with respect to the CamShift algorithm. Our experiments have also shown a high robustness of the method to 3D pose, facial expression, lighting conditions, partial occlusion, and facial features.

1 Introduction and Related Research

A wide range of approaches has been proposed to deal with the problem of human face tracking. Most of them are based on skin-like color detection [1]–[8], which is a well-known robust method to track human heads and hands. Color detection is usually followed by a location of individual facial features, for example, using PCA [2], splines [3] or integral projections [4], [5], [6]. However, in most of the existing research, projections are processed in a rather heuristic way. Other kinds of approaches are adaptations of face detectors to tracking, e.g. based on contour modelling [9], [10], eigenspaces [11] and texture maps [12]. These methods are computationally expensive and sensitive to facial expressions, so efficiency is usually achieved at the expense of reducing location accuracy.

In this paper[1], we prove that projections not only can be used to track human faces, but the dimensionality reduction involved in projection yields substantial improvements in computational efficiency and location accuracy. The technique has been implemented using Intel IPL and OpenCV libraries [13], and compared with the CamShift algorithm [1]. Each frame typically requires less than 5 ms. on a standard PC, achieving a location accuracy of about 3 mm.

2 Working with Integral Projections

An integral projection is a one-dimensional pattern, obtained through the sum of a given set of pixels along a given direction. Let $i(x, y)$ be an image and

[1] This work has been supported by the Spanish MCYT grant DPI-2001-0469-C03-01.

$R(i)$ a region in it, the vertical integral projection of $R(i)$, denoted by $P_{VR(i)}$ is given by: $P_{VR(i)}(y) = \overline{i(x,y)}$; $\forall (x,y) \in R(i)$. The horizontal projection of $R(i)$, denoted by $P_{HR(i)}$, can be defined in a similar way.

2.1 Integral Projection Models

In order to establish a formal framework for the use of projections, it is adequate to distinguish between integral projection *signals* and *models*. A *projection signal* is a discrete function $S : \{s_{min}, .., s_{max}\} \rightarrow \mathbf{R}$. A *projection model* describes a variety of signals, usually obtained from different instances of a kind of object. We propose a gaussian-style modelling, in which the model corresponding to a set of signals is given by the mean and variance at each point in the range of the signals. Thus, a projection model can be expressed as a pair of functions:

- $M : \{m_{min}, .., m_{max}\} \rightarrow \mathbf{R}$. Mean at each point in the range of the signals.
- $V : \{m_{min}, .., m_{max}\} \rightarrow \mathbf{R}$. Variance at each point.

Fig. 1 shows three typical examples of integral projection models, corresponding to the vertical projection of the whole face, and the horizontal projection of eyes and mouth regions. In this case, the training set contained 45 face instances.

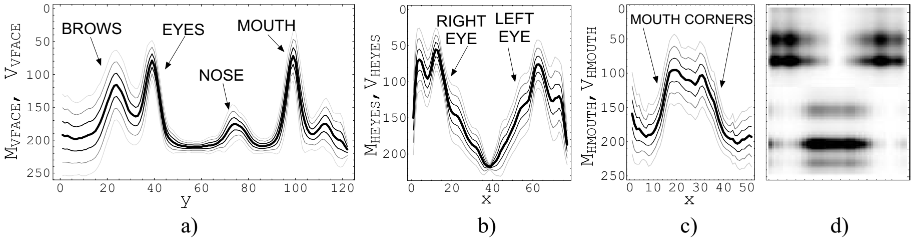


Fig. 1. Integral projection models and reprojection. a-c) Mean and variance of the vertical projection of the face a), and the horizontal projection of eyes b) and mouth c) regions. d) Reprojection of the model by matrix product

Useful information can be obtained from simple heuristic analysis of integral projections, e.g. searching for global minima [4], applying fuzzy logic [6] or thresholding projections [5], without the need of using explicit models. However, these techniques are normally very sensitive to outliers. Working with explicit integral projection models entails the following advantages:

- Models are learnt from examples, avoiding the *ad hoc* nature of heuristic methods. This allows an easier adaptation to similar applications.
- A distance function can be defined to measure the likelihood that a certain signal S is an instance of a particular model (M, V). In our case, this distance has the form of a mean squared difference,

$$d(S,(M,V)) = \frac{1}{||C||} \sum_{i \in C} \frac{(S(i) - M(i))^2}{V(i)} \;, \tag{1}$$

with,

$$C = \{s_{min}, \ldots, s_{max}\} \cap \{m_{min}, \ldots, m_{max}\} \ . \tag{2}$$

- The model can be visualized and interpreted by a human observer and, more interesting, an approximate reconstruction can be obtained by reprojecting the model. For example, if vertical and horizontal projections are applied, the reprojection can be computed through a matrix product. An example of reprojection from a projection model is shown in Fig. 1d).

2.2 Alignment of Integral Projections

Alignment is the basic concern when dealing with projections; corresponding features in the images should be projected into the same locations. The problem of aligning 1-D signals is equivalent to object location in 2-D images. However, using projections involves a reduction in the number of freedom degrees; in practice, only two parameters have to be considered: scale and translation.

Assuming signal S is an instance of a given model (M, V), the alignment problem can be formulated as finding the values of scale d and translation e, such that after aligning S, corresponding pixels are projected in S and (M, V) into the same locations. The aligned S, denoted by S', is given by,

$$S' : \{(s_{min} - e)/d, \ldots, (s_{max} - e)/d\} \rightarrow \mathbf{R} \ ; \ S'(i) = S(di + e) \ . \tag{3}$$

Note that the distance function defined in (1, 2) requires S and (M, V) to be properly aligned. Furthermore, if S is an instance of the model, this distance can be used as a goodness of alignment measure: a low value will be obtained for a good alignment, and a high value otherwise. Thus, substituting S in (1) with S' in (3), the alignment problem is reformulated as finding the pair of values (d, e) which minimize,

$$\frac{1}{||C||} \sum_{i \in C} \frac{(S(di + e) - M(i))^2}{V(i)} \ . \tag{4}$$

3 Face Tracking Using Projections

The tracking algorithm is part of an iterative process, which recalculates the location of the face and facial features in a sequence of images. The face detection technique described in [8] is used as initialization. For each face being tracked, a bounding ellipse and the location of eyes and mouth are computed.

The input to the tracker is a face model, the state of tracking in the previous image, i_{t-1}, and a new image i_t. The face model consists of a set of projection models of the whole face and some parts in it, where the locations of facial features are known. This model is computed using the first image of the sequence, where eyes and mouth locations are given by the face detector, as in Fig. 2a).

Basically, the algorithm consists of three main steps in which the vertical, horizontal and orientation parameters of the new location are estimated independently. This process is explained in more detail in the following subsections.

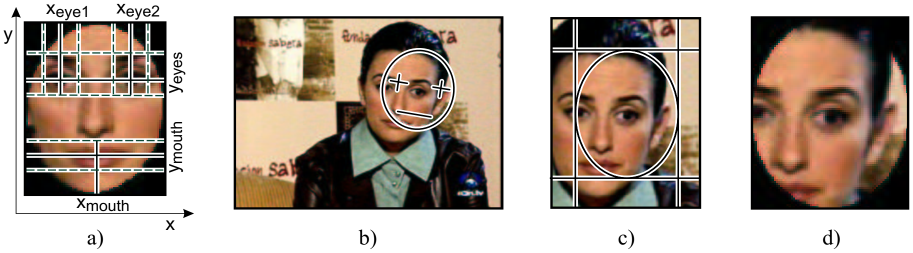


Fig. 2. Face model and preprocessing step. a) Sample face used to compute a face model. b) Expected location of the bounding ellipse and facial features in a new image, using locations in the previous one. c) Face region extracted from b), using $r_{tolerance}$=25%. d) Face region wrapped and segmented, according to the model

3.1 Preprocessing Step

Kalman filters [2], [3] and skin-color detection [1],[4],[5] have been commonly applied in face tracking as prediction filters. In our case, we have proved that integral projections can be used to solve the problem by themselves. Thus, a null predictor is used: the locations of the bounding ellipse and the facial features in $i_{t-1}(x, y)$ are taken as the expected locations in $i_t(x, y)$.

In the preprocess, a rectangle in $i_t(x, y)$ containing the bounding ellipse and rotated with respect to the eye-line (the line going through the expected locations of both eyes) is wrapped into a predefined size rectangle given by the face model, see Fig. 2c). In fact, an area bigger than just the bounding ellipse is wrapped. The size of this additional area is a percentage of the model size, denoted by $r_{tolerance}$ (set to 25% in the experiments). The inner part of the ellipse is segmented, as shown in Fig. 2d), and taken as an input for the vertical alignment step.

3.2 Vertical Alignment Step

In this step, using vertical projections, the vertical translation and scale of the face in the new image are estimated. Firstly, the vertical projection of the segmented face region, P_{VFACE}, is computed, see Fig. 3c). This signal is aligned with respect to the vertical projection model (M_{VFACE}, V_{VFACE}), see Fig. 3d). Finally, the alignment parameters (d, e) are used to align the face vertically.

3.3 Horizontal Alignment Step

After the vertical alignment step, y coordinates of mouth and eyes are known[2], so eyes and mouth regions are segmented (including the tolerance area) according to the model, see Fig. 3g). Using horizontal projections of these regions, the algorithm estimates the horizontal translation and scale of the face. Actually,

[2] Although both eyes might not have exactly the same y coordinate, since the face could be rotated. Anyway, a small rotation is supposed in the worst case.

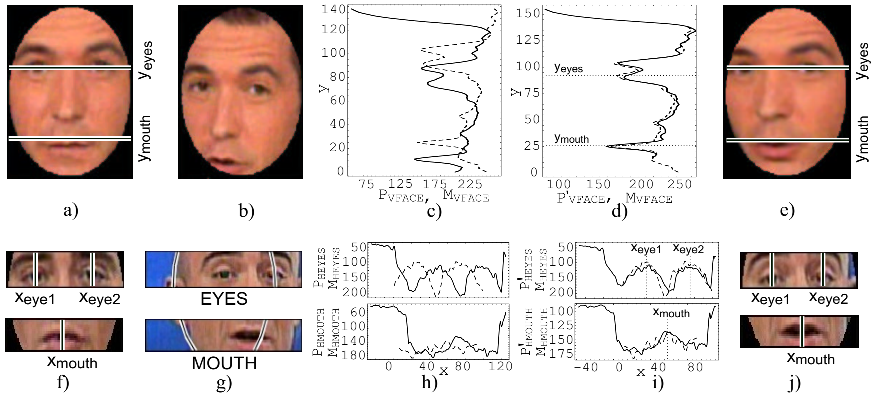


Fig. 3. Vertical and horizontal alignment steps. Upper row: vertical alignment. Lower row: horizontal alignment. a),f) Images used to compute the model. b),g) Segmented face, eyes and mouth regions in $i_t(x, y)$, using the locations in $i_{t-1}(x, y)$. c),h) Vertical and horizontal projections of the models (dashed lines) and the regions in b),g) (solid lines). d),i) The same projections after alignment. e),j) Vertical and horizontal alignment of b),g), using alignment parameters obtained in d),i), respectively

only the horizontal projection of the eyes region, P_{HEYES}, is considered. As can be seen in Fig. 3i), P_{HEYES} is usually more stable than P_{HMOUTH}, producing a more reliable alignment.

3.4 Orientation Estimation Step

Face orientation[3] is computed through the estimation of the eye-line. While in the previous steps both eyes were supposed to be located along y_{eyes}, here the y coordinate of the right and left eyes (y_{eye1} and y_{eye2}, respectively) are estimated independently. This process is illustrated in Fig. 4.

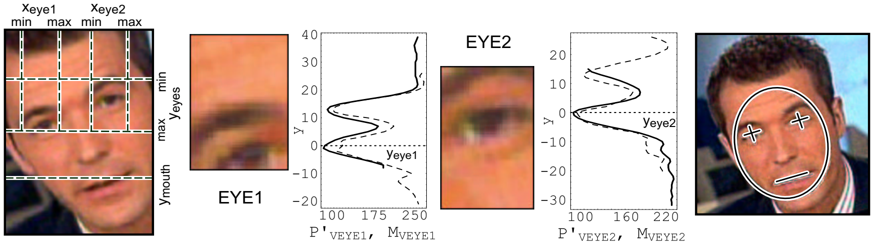
After steps 2 and 3, the locations of the eyes are approximately known, so regions *EYE1* and *EYE2* can be segmented. Aligning the vertical projections of these regions, the values of y_{eye1} and y_{eye2} are computed.

4 Experimental Results

The purpose of these experiments was to assess the location accuracy of the integral projection method, its robustness to a variety of non-trivial conditions, and to compare its computational efficiency with respect to other approaches.

The tracking algorithm described in this paper has been implemented using Intel IPL and OpenCV image processing libraries [13]. OpenCV provides a color-based face tracker, called CamShift [1], which has been used for comparison. This

[3] Orientation is here considered in a planar sense, i.e. rotation with respect to the image plane.



a) b) c) d) e) f)

Fig. 4. Orientation estimation step. a) A sample rotated face, after vertical and horizontal alignment. b),d) Right and left eyes segmented, respectively. c),e) Vertical projections of the right and left eyes (solid lines), after alignment with the model (dashed lines). f) Final location of the facial features and the bounding ellipse

algorithm, by itself, only computes the bounding ellipse, but not facial features locations. To solve this problem, we have assumed that eyes and mouth move coherently with respect to the bounding ellipse.

Both algorithms have been applied to four video sequences[4], adding up a total of 1915 frames ($\sim$ 72 sec.). The first two sequences were captured from TV (news programs) at a 640x480 resolution; another from an inexpensive videoconference camera; and the last one was extracted from a DVD movie. All the sequences contain wide changes in facial expression and 3D pose; the last two also include samples of partial occlusion, varying illumination and faster movements.

The obtained location accuracy and execution times are summarized in Table 1. Fig. 5 shows feature location errors for the first two sequences; signal-to-model distances (see Sect. 2.2) of the resultant aligned projections are also shown. Finally, some sample frames for the worst error cases are presented in Fig. 6. Errors are expressed as Euclidean distances in millimeters, in the face plane, assuming an interocular distance of 70 mm. and using a manual labelling of facial features as ground-truth.

Table 1. Average and maximum feature location errors (eyes and mouth), and execution time (per frame, not including input/output), using integral projections and the CamShift algorithm. Computer used: AMD Athlon processor at 1.2 GHz

			Face size	Int. Projections		CamShift	
Sequence file name	Source	Length (frames)	X x Y (pixels)	Location error avg./max.(mm)	Time (ms)	Location error avg./max.(mm)	Time (ms)
tl5-02.avi	TV	281	97x123	3.41 / 13.0	4.02	10.3 / 28.8	8.17
a3-05.avi	TV	542	101x136	1.95 / 9.76	4.62	9.22 / 24.1	7.52
ggm2.avi	QuickCam	656	70x91	1.83 / 9.29	3.69	12.4 / 30.8	5.45
13f.avi	DVD	436	146x176	3.61 / 14.2	8.35	Unable to work	

[4] Test videos and results available at: `http://dis.um.es/~ginesgm/fip/demos.html`

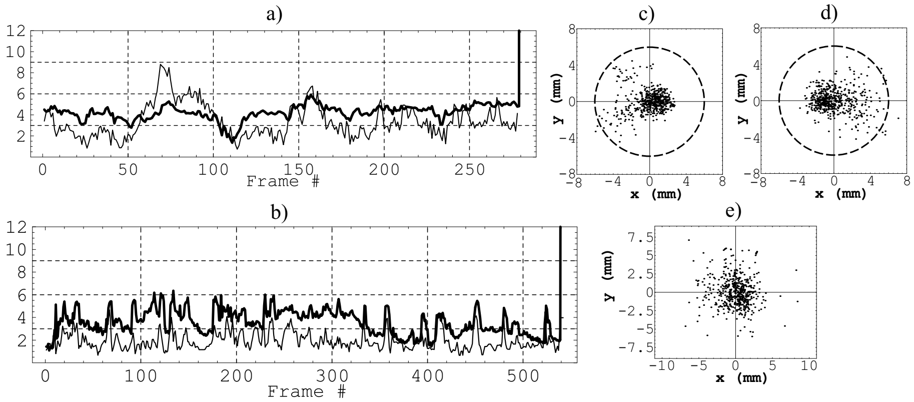


Fig. 5. Integral projection face tracker results. a),b) Average feature location error in mm. (thin lines) and signal-to-model distance (thick lines), for each frame in the first two sequences, respectively. c),d),e) Location errors for the right c) and left d) eyes, and mouth d), for the second sequence. The dashed circle represents the iris size

In all the sequences, the integral projection tracker exhibits a clearly better performance, both in time and accuracy. The average location error –always below 4 mm– contrasts greatly with the error obtained by CamShift, above 10 mm. In the last sequence, CamShift could not be applied due to the presence of skin-like color background. The CPU time required for each frame depends on the face size and is typically below 5 ms. This allows real-time processing without much CPU usage. Moreover, the algorithm is between 103% and 48% faster than CamShift, even when the last is only applied to a part of the images.

Another interesting result from the experiments is the possibility of using the signal-to-model distance as a reliability degree of tracking. As shown in Fig. 5, a direct relationship exists between this value and the location error. Furthermore, a very high value (usually over 20) is obtained when the face disappears. Thus, its value has been used to detect the end of tracking in a sequence.

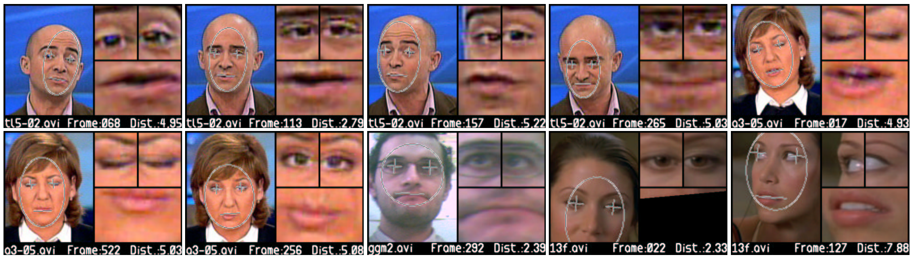

tl5-02.avi Frame:068 Dist.:4.95
tl5-02.avi Frame:113 Dist.:2.79
tl5-02.avi Frame:157 Dist.:5.22
tl5-02.avi Frame:265 Dist.:5.03
o3-05.avi Frame:017 Dist.:4.93
o3-05.avi Frame:522 Dist.:5.03
o3-05.avi Frame:256 Dist.:5.08
ggm2.avi Frame:292 Dist.:2.39
13f.avi Frame:022 Dist.:2.33
13f.avi Frame:127 Dist.:7.88

Fig. 6. Sample frames, showing worst error cases. Eyes and mouth regions (right) are segmented according to the tracker results. *Dist* refers to signal-to-model distance

5 Conclusions

We have presented a new approach to human face tracking in video sequences, which is exclusively based on the alignment of integral projections. The tracking problem is decomposed into three main steps, where vertical, horizontal and orientation parameters are estimated independently. Although this separability is questionable in the general case of object location, our experiments have extensively proved its feasibility in face tracking.

The proposed technique exhibits a very high computational efficiency, while achieving a better location accuracy than other more sophisticated existing trackers, without losing track of the face in any frame. Each frame typically requires less than 5 ms. on a standard PC, with an error of about 3 mm. Our experiments have also shown a high robustness to facial expressions, partial occlusion, lighting conditions and 3D pose.

Another two interesting advantages, over color-based trackers, are that the algorithm is not affected by background distractors, and it can be applied on grey-scale images or under changing lighting or acquisition conditions.

References

1. Bradski, G. R.: Computer Vision Face Tracking For Use in a Perceptual User Interface. Intel Technology Journal Q2'98 (1998)
2. Spors, S., Rabenstein, R.: A Real-Time Face Tracker for Color Video. IEEE Intl. Conference on Acoustics, Speech, and Signal Processing, Utah, USA (2001)
3. Kaucic, R., Blake, A.: Accurate, Real-Time, Unadorned Lip Tracking. Proc. of 6th Intl. Conference on Computer Vision (1998) 370–375
4. Sobottka, K., Pitas, I.: Segmentation and Tracking of Faces in Color Images. Proc. of 2nd Intl. Conf. on Aut. Face and Gesture Recognition (1996) 236–241
5. Stiefelhagen, R., Yang, J., Waibel, A.: A Model-Based Gaze Tracking System. Proc. of IEEE Intl. Symposia on Intelligence and Systems (1996) 304–310
6. Pahor, V., Carrato, S.: A Fuzzy Approach to Mouth Corner Detection. Proc. of ICIP-99, Kobe, Japan (1999) I-667–I-671
7. Schwerdt, K., Crowley, J.L.: Robust Face Tracking Using Color. Proc. of 4th Intl. Conf. on Aut. Face and Gesture Recognition, Grenoble, France (2000) 90–95
8. García-Mateos, G., Ruiz, A., López-de-Teruel, P.E.: Face Detection Using Integral Projection Models. Proc. of IAPR Intl. Workshops S+SSPR'2002, Windsor, Canada (2002) 644–653
9. Isard, M., Blake, A.: Contour Tracking by Stochastic Propagation of Conditional Density. Proc. 4th Eur. Conf. on Computer Vision, Cambridge, UK (1996) 343–356
10. Vieren, C., Cabestaing, F., Postaire, J.: Catching Moving Objects with Snakes for Motion Tracking. Pattern Recognition Letters, 16 (1995) 679–685
11. Pentland, A., Moghaddam, B., Starner, T.: View-Based and Modular Eigenspaces for Face Recognition. Proc. CVPR'94, Seattle, Washington, USA (1994) 84–91
12. La Cascia, M., Sclaroff, S., Athitsos, V.: Fast, Reliable Head Tracking Under Varying Illumination: An Approach Based on Registration of Texture-mapped 3D Models. IEEE PAMI, 22(4), (2000) 322–336
13. Intel Corporation. IPL and OpenCV: Intel Open Source Computer Vision Library. `http://www.intel.com/research/mrl/research/opencv/`