Proc. ICPR 2002. Vol II, pp. 697-700, IEEE. Quebec, Canada, August 2002

Real-time tracking and estimation of plane pose*

José Miguel Buenaposada, Luis Baumela
Departamento de Inteligencia Artificial
Universidad Politécnica de Madrid
Campus de Montegancedo s/n, 28660 Boadilla del Monte (Madrid). SPAIN
e-mail: jmbuena@dia.fi.upm.es, lbaumela@fi.upm.es

Abstract

In this paper we present a method to estimate in real-time the position and orientation of a previously viewed planar patch. The algorithm is based on minimising the sum of squared differences between a previously stored image of the patch and the current image of it. First a linear model for projectively tracking a planar patch is introduced, then, a method to compute the 3D position and orientation of the patch is 3D space is presented. In the experiments conducted we show that this method is adequate for tracking not only planar objects, but also non planar objects with limited out-of-plane rotations, as is the case of face tracking.

1. Introduction

Tracking planar patches is a subject of interest in computer vision, with applications in augmented reality [5], mobile robot navigation [7], face tracking [1], or the generation of super-resolution images [2]. Traditional approaches to tracking are based on finding correspondences in successive images. This can be achieved by computing optical flow [4] or by matching a sparse collection of features [6]. In flow-based methods, a velocity vector is computed for each pixel, while in feature-based methods, image features, such as points and lines are matched across all frames in the sequence. Feature-based methods minimise an error measure based on geometrical constraints between a few corresponding features, while direct methods minimise an error measure based on direct image information collected from all pixels in the image, such as image brightness.

The tracking method presented in this paper belongs to the first group of methods. It is based on minimising the sum-of-squared differences (SSD) between a previously stored image of the tracked patch (template image) and the current image of it. It extends Hager's SSD tracker [3] by introducing a projective motion model and a procedure to compute the position and orientation of the tracked patch in each frame.

*Work funded by CICyT under project number TIC1999-1021

2. SSD plane tracking

The main idea of Hager's SSD tracking procedure is as follows. After a movement of the planar object P, the grey level of a pixel $\bar{x}$ on P will change, as a different region of P is now projected over it. This information will be used to estimate the motion parameters that best explain, in a least square sense, the grey level changes produced by the motion of P.

Assuming no changes in the illumination,

$$I(\bar{x}, t_0) = I(f(\bar{x}, \bar{\mu}), t_n) \forall x \in P, \tag{1}$$

where $I(\bar{x}, t_0)$ is the template image and $I(f(\bar{x}, \bar{\mu}), t_n)$ is the rectified image at time t_n, with motion parameters $\bar{\mu}$.

From equation (1) the motion parameters $\bar{\mu}$ can be estimated by minimising the difference between the template and the rectified image:

$$\min_{\bar{\mu}} (\sum_{\bar{x} \in P} (I(f(\bar{x}, \bar{\mu}), t_n) - I(\bar{x}, t_0))^2) \tag{2}$$

The minimisation problem can be solved linearly by computing $\bar{\mu}$ incrementally while tracking. This can be achieved by making a Taylor series expansion of (2) about $(\bar{\mu}, t_n)$ and computing the increment, $\delta\mu$, between two time instants [3]:

$$\delta\bar{\mu} = -(M^t M)^{-1} M^t [I(f(\bar{x}, \bar{\mu}), t_n) - I(\bar{x}, t_0)] \tag{3}$$

where M is the Jacobian matrix of the image. It must be computed in each frame as it depends on $\bar{\mu}$. This is computationally expensive, as it is of dimension $N \times n$, where

N is the number of template pixels and n is the number of motion parameters. In order to simplify this computation the structure of M must be analysed. It can be written as

$$M(\bar{\mu}) = \begin{pmatrix} \nabla_x I(\bar{x}_1, \bar{\mu}_0)^t f_x(\bar{x}_1, \bar{\mu})^{-1} f_\mu(\bar{x}_1, \bar{\mu}) \\ \nabla_x I(\bar{x}_2, \bar{\mu}_0)^t f_x(\bar{x}_2, \bar{\mu})^{-1} f_\mu(\bar{x}_2, \bar{\mu}) \\ \vdots \\ \nabla_x I(\bar{x}_N, \bar{\mu}_0)^t f_x(\bar{x}_N, \bar{\mu})^{-1} f_\mu(\bar{x}_N, \bar{\mu}) \end{pmatrix}, \quad (4)$$

where $\nabla_x I$ is the template image gradient, f_x is the derivative of the motion model with respect to the pixel coordinates and f_μ is the derivative of the motion model with respect to the motion parameters.

Depending on the motion model, M may be factored into the product of two matrices,

$$M(\bar{\mu}) = \begin{pmatrix} \nabla_x I(\bar{x}_1, \bar{\mu}_0)^t \Gamma(\bar{x}_1) \\ \nabla_x I(\bar{x}_2, \bar{\mu}_0)^t \Gamma(\bar{x}_2) \\ \vdots \\ \nabla_x I(\bar{x}_N, \bar{\mu}_0)^t \Gamma(\bar{x}_N) \end{pmatrix} \Sigma(\bar{\mu}) = M_0 \Sigma(\bar{\mu}) \quad (5)$$

a constant matrix M_0 of dimension $N \times n$ and a matrix Σ of dimension $n \times n$, that depends on $\bar{\mu}$. As M_0 can be precomputed, this factorisation reduces the on line computation to the inversion of the matrix Σ:

$$\delta\bar{\mu} = -\Sigma^{-1}(M_0^t M_0)^{-1} M_0^t [I(f(\bar{x}, \bar{\mu}), t_n) - I(\bar{x}, t_0)]. \quad (6)$$

The matrix M_0 represents *a priori* knowledge about the target structure, that is, how the grey level value of each pixel changes as the object moves. It summarises the information provided by each template pixel to the tracking process. It is important to note that we can not track any object with this method, as a non singular $M_0^t M_0$ matrix is needed.

3. Projective model for plane tracking

In this section we are going to introduce a projective model of target motion. Introducing a new motion model in Hager's framework consists on finding the Jacobian matrix decomposition that arise from this model.

Let $\bar{x} = (u, v)^t$ and $\bar{x}_h = (r, s, t)^t$ be respectively the Cartesian and projective coordinates of an image pixel. They are related by:

$$\bar{x}_h = \begin{pmatrix} r \\ s \\ t \end{pmatrix} \rightarrow \bar{x} = \begin{pmatrix} r/t \\ s/t \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} \forall \bar{x} \in I(\bar{x}) \quad (7)$$

The equation f that describes the motion of a planar region is then a 2D projective linear transformation,

$$f(\bar{x}_h, \bar{\mu}) = H\bar{x}_h = \begin{pmatrix} a & d & g \\ b & e & h \\ c & f & 1 \end{pmatrix} \begin{pmatrix} r \\ s \\ t \end{pmatrix}, \quad (8)$$

where, now, the motion parameter vector is $\bar{\mu} = (a, b, c, d, e, f, g, h)^t$.

Next we express the Jacobian matrix decomposition of the projective motion model in terms of the elements of equation (4):

$$\nabla_{x_h} I(\bar{x}_h, \bar{\mu}_0)^t = \left(\frac{\partial I}{\partial u}, \frac{\partial I}{\partial v}, -\left(u\frac{\partial I}{\partial u} + v\frac{\partial I}{\partial v} \right) \right) \quad (9)$$

$$f_x(\bar{x}_h, \bar{\mu})^{-1} = H^{-1} \quad (10)$$

$$f_\mu(\bar{x}_h, \bar{\mu}) = \begin{pmatrix} r & 0 & 0 & s & 0 & 0 & t & 0 \\ 0 & r & 0 & 0 & s & 0 & 0 & t \\ 0 & 0 & r & 0 & 0 & s & 0 & 0 \end{pmatrix} \quad (11)$$

Introducing (9), (10) and (11) into (4) M can be factored according to (5):

$$f_x(\bar{x}, \bar{\mu})^{-1} f_\mu(\bar{x}, \bar{\mu}) =$$
$$H^{-1} \left(\begin{array}{ccc|c} rI_{3\times3} & | & sI_{3\times3} & | \\ & & & \hline \end{array} \begin{array}{c} tI_{2\times2} \\ \hline 0_{1\times2} \end{array} \right) =$$
$$\left(\begin{array}{ccc|c} rH^{-1} & | & sH^{-1} & | \end{array} tH_{12}^{-1} \right) = \Gamma(\bar{x}_h)\Sigma(\bar{\mu}),$$

where H_{12}^{-1} is the matrix composed with the first two columns of H^{-1}, $I_{q\times q}$ is the $q \times q$ identity matrix and

$$\Gamma(\bar{x}_h) = \begin{pmatrix} rI_{3\times3} & | & sI_{3\times3} & | & tI_{3\times3} \end{pmatrix},$$
$$\Sigma(\bar{\mu}) = \begin{pmatrix} H^{-1} & 0 & 0 \\ 0 & H^{-1} & 0 \\ 0 & 0 & H_{12}^{-1} \end{pmatrix}.$$

In this case the matrix Σ is 9×8 so it has no inverse. Then the equation to compute $\delta\mu$ is as follows:

$$\delta\bar{\mu} = -(\Sigma^t M_0^t M_0 \Sigma)^{-1} \Sigma^t M_0^t [I(f(\bar{x}, \bar{\mu}), t_n) - I(\bar{x}, t_0)]. \quad (12)$$

With this factorisation we can projectively track a planar patch with the computational cost of inverting an 8×8 matrix and some matrix multiplications on each frame.

4. 3D pose estimation

The tracking model presented in the previous section computes the homography H_0^n between the present image and the stored template. In this section we are going to show that it is possible to estimate the pose of the tracked patch from H_0^n, if the camera is calibrated.

So far we have only computed 2D information. In order to have 3D information we have to compute two more homographies: one from P to the image plane at t_0, H_W^0, and another from P to the image plane at t_n, H_W^n, see Fig. 1.

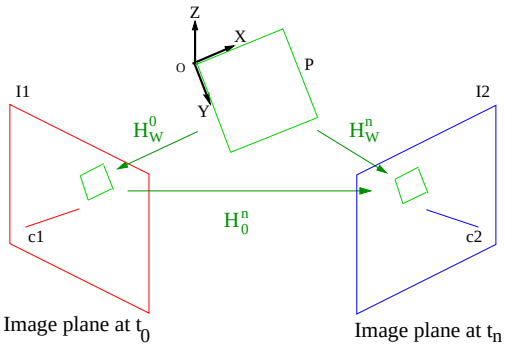

Figure 1. Projective transformations involved in 3D plane tracking.

In order to simplify the equations we choose the scene coordinate system to have the X and Y axis on the plane P and the Z axis perpendicular to it (see Fig. 1).

Homography H^0_W can be computed off line using the projection of, at least, four known points on P. Let $(X_P, Y_P)^t$ be the Cartesian coordinates of a known point in P and let $(x_0, y_0)^t$ be Cartesian coordinates of the projection of $(X_P, Y_P)^t$ onto I at t_0 (i.e. at the template image). H^0_W can be computed from:

$$\begin{pmatrix} x_0 \\ y_0 \\ 1 \end{pmatrix} = H^0_W \begin{pmatrix} X_P \\ Y_P \\ 1 \end{pmatrix}. \tag{13}$$

On the other hand, the projection of point $(X_P, Y_P)^t$ onto I at time instant t_n is given by

$$\begin{pmatrix} x_n \\ y_n \\ 1 \end{pmatrix} = \lambda K[R|t] \begin{pmatrix} X_P \\ Y_P \\ 0 \\ 1 \end{pmatrix}, \tag{14}$$

where R and t are respectively the orientation and the position of P in the camera coordinate system, λ is a scale factor and K is the camera instrinsics matrix.

Introducing in (14) the fact that all points of P have coordinate $Z = 0$, H^n_W can be written as:

$$\begin{pmatrix} x_n \\ y_n \\ 1 \end{pmatrix} = \lambda K[r_1 r_2 t] \begin{pmatrix} X_P \\ Y_P \\ 1 \end{pmatrix} = H^n_W \begin{pmatrix} X_P \\ Y_P \\ 1 \end{pmatrix} \tag{15}$$

where r_i is the ith column of the matrix R.

Now, from (13) and (15)

$$\begin{pmatrix} x_n \\ y_n \\ 1 \end{pmatrix} = \underbrace{H^n_W (H^0_W)^{-1}}_{H^n_0} \begin{pmatrix} x_0 \\ y_0 \\ 1 \end{pmatrix} =$$

$$\underbrace{\lambda K[r_1 r_2 t](H^0_W)^{-1}}_{H^n_0} \begin{pmatrix} x_0 \\ y_0 \\ 1 \end{pmatrix} \tag{16}$$

From which we obtain the relation between the homography computed in the previous section, H^n_0, and the pose of P. So, if the intrinsics K and the homographies H^0_W and H^n_0 are known, we can compute H^* [5],

$$H^* = K^{-1} H^n_0 H^0_W = \lambda[r_1 r_2 t] \tag{17}$$

The translation is obtained directly from the third column of H^* but in order to obtain the rotation matrix we still have to impose some constraints:

- $\| r_1 \| = \| r_2 \| = 1$, as R is a rotation matrix. In this way we get $\widehat{r}_1$ and $\widehat{r}_2$.
- $r_3 \perp r_1$ and $r_3 \perp r_2$, from where we get $\widehat{r}_3$.

5. Experiments

We have conducted our experiments on a GNU/Linux system with an AMD K7 750MHz. We acquire images from a Sony VL500 digital camera with IEEE 1394 interface. In the present implementation, the system is able to process 24 frames per second.

Until now we have ignored the efect of illumination changes. To alleviate the problem somehow, we equalize the image template and the rectified image . Doing so, we can remove the efect of intensity changes but not a change in the direction of the incomming light (the result is the appearence of shadows which changes the object structure).

We are going to validate the 3D plane tracking algorithm with three experiments (See videos in http://www.dia.fi.upm.es/∼jmbuena): in the first one the target is a planar object designed by us (see Fig. 2), in the second sequence we track a book hardcover (see Fig. 3) and in the third sequence we show that the algorithm can also track to some extent non planar objects, for example the human face (see Fig. 4).

We have validated our algorithm by overlaying the plane coordinate system over the image, in this way we can get an indirect perception of the accuracy of pose estimation. As can be seen in the results presented in Figs. 2, 3, and 4, in all cases the axes over the image are coherent with the plane movement, except when the SSD homography estimation has less precision, as in the case of the head motion up and down.

6. Conclusions

We have introduced a plane tracker which permits, given knowledge of camera calibration, the estimation of the 3D

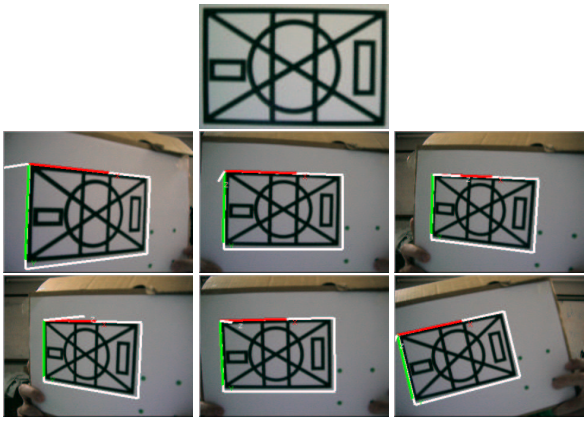

Figure 2. In the first row: template image. Second and third rows, frames 15, 85, 160, 200, 235, 280 and 330 of 400. The white quadrangle is the position in 2D estimated from the projective SSD tracker. The 3D pose estimation: Z axis in white (out of plane), Y axis in blue and X axis in green.

Figure 3. In the first row: template image. Second and third rows, frames 35, 84, 115, 170, 200 and 250 of 350. The white quadrangle is the position in 2D estimated from the projective SSD tracker. The 3D pose estimation: Z axis in white (out of plane), Y axis in blue and X axis in green.

Figure 4. In the first row: template image. Second and third rows, frames 37, 99, 130, 193, 233 and 289 of 350. The white quadrangle is the position in 2D estimated from the projective SSD tracker. The 3D pose estimation: Z axis in white (out of plane), Y axis in blue and X axis in green.

pose of a plane in real-time. The system used in testing works at 24 frames per second in an AMD K7 750 MHz.

References

[1] M. J. Black and Y. Yacoob. Recognizing facial expressions in image sequences using local parameterized models of image motion. *Int. Journal of Computer Vision*, 25(1):23–48, 1997.

[2] C. T. F. Dellaert and S. Thrun. Super-resolved texture tracking of planar surface patches. In *Proceedings Intelligent Robots and Systems*, pages 197–203. IEEE, 1998.

[3] G. D. Hager and P. N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analisys and Machine Intelligence*, 20(10):1025–1039, 1998.

[4] M. Irani and P. Anandan. All about direct methods. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and practice*. Springer-Verlag, 1999.

[5] G. Simon, A. Fitzgibbon, and A. Zisserman. Markerless tracking using planar structures in the scene. In *Proc. International Symposium on Augmented Reality*, October 2000.

[6] P. H. S. Torr and A. Zisserman. Feature based methods for structure and motion estimation. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and practice*, pages 278–295. Springer-Verlag, 1999.

[7] F. L. V. Ayala, J.B. Hayet and M. Devy. Visual localization of a mobile robot in indoor environments using planar landmarks. In *Proceedings Intelligent Robots and Systems, 2000*, pages 275–280. IEEE, 2000.