Active Monocular Tracking with Temporal Integration of Visual Cues

Artur Arsénio* , José Santos-Victor

Instituto de Sistemas e Robótica - Instituto Superior Técnico
Av. Rovisco Pais, nº1 - 1096 LISBOA Codex, PORTUGAL
Tel. : 351 - 1 - 8418294 Fax : 351 - 1 - 8418291
email: {aaa, jasv}@isr.ist.utl.pt

Abstract*: In this paper we address the problem of tracking a moving target by a monocular observer. Tracking a moving target is one of the basic functionalities of any active vision system and may be used to retrieve important information about the target/observer 3D motion, shape, etc. The strategy we present is based on the integration of correlation based techniques together with active contours, using a Kalman filtering approach. The active eye system consists on a camera mounted on a pan & tilt unit. We describe in detail the motion algorithms used and the control system design issues. Due to some simplifications of the motion algorithms, the overall operating frequency is about 6Hz. The system is robust both to changes in the illumination and multiple moving objects in cluttered environments. Results are presented and discussed.*

Keywords*: Active Vision, Vision based Control, Tracking, Deformable Contours, Kalman Filter.*

1. Introduction

Tracking a moving object is one of the main functionalities of an active vision system, and has attracted a significant amount of attention in the past few years. In particular the tracking system can provide information about the observer/target 3D motion and may simplify the retrieval of the object shape.

This work aims at developing a reliable monocular tracking system to keep a moving object centred in the image plane. The use of a closed loop control system, imposes strong constraints in the computational cost of the motion algorithms to be used. We integrate both correlation measurements and deformable contours [11] to enhance the tracking robustness. The performance of the tracking system is improved by the use of a Kalman-Bucy filter to predict the target position and velocity in the image plane together with uncertainty estimates. This information is used to speed up the algorithms in two ways. First, we can profit from having an estimate of the target position/velocity in the subsequent frame. Secondly, the uncertainty information is used to determine the image area where the target is likely to be found.

1.1. Related Work

Many methods have been presented to track a moving object over a sequence of images. The techniques must often used are optical-flow, correlation techniques and deformable contours.

The use of optical flow involves the segmentation of the object to be tracked from the scene background [17]. Since the observer is in motion, a velocity field is induced all over the image. This problem is far from being solved and the existing algorithms (typically segmenting various motion models) are very time consuming.

The approach followed by Murray et al. in [13] consists in first extracting corner points of the target image, which are then tracked using correlation. They assume that the observer motion is known and that the target moves at constant velocity relative to the camera. Correlation techniques have been widely used also in stereo [8,18].

Kass et al. [11,16] have presented models which relate to very general physical properties of objects such as the cohesion of matter and smoothness. Some extensions are presented in [14] where snakes are modeled as a dynamic system, and a Kalman-Bucy filter approach is used. Curwen and Blake [7] impose shape assumptions on the deformable contour; the drawback being that large target deformations are not allowed. They use a parametric representation of the curve to save time (B-Splines). Recent work is based in the automatic control of spatio-temporal scale[3].

The approach we follow is based on deformable models as a tracking methodology, since they can cope with the active change of the object shape. Although the use of snakes is very efficient from the point of view of global tracking, the mapping between edge points in successive images may be biased by the slippage of the snake control points along the image contour. This problem is important when performing eye saccades during the tracking procedure. To overcome this limitation we maintain an estimate of the snake geometric center through the use of a Kalman-Bucy filter. Then, the snake points are estimated given the expected position of the snake center.

* Artur Arsénio supported by a grant from PRAXIS XXI-BM/7153/95

As a consequence of this approach, the components of the dynamic system state are the position and velocity of the snake geometric center, rather than the positions of the snake control points. This approach is also significantly more efficient from a computational point of view.

1.2. Structure of the Paper

In Section 2 we present the dynamical model of the system, introduce some simplifying assumptions and discuss the model validity. The visual processing is discussed in Section 3. Section 4 details the structure of the Kalman-Bucy filter to be used. In Section 5 we introduce the controller and establish some considerations on the processing delays. The results are presented in Section 6 and in Section 7 we draw some conclusions and discuss further directions of work.

2. Tracking System

The vision system described comprises the following major processing modules: The image acquisition, the visual processing module and the Pan and Tilt Unit (PTU) controller. The system structure is shown in Figure 1. At every sampling instant, a new image is acquired by the moving camera, the new target position is computed and the control commands are sent. The visual processing module is responsible for the estimation of the target position in the present image. The closed loop controller is a PID controller that receives information from the pan-tilt unit and determines the appropriate control signals for the tracking.

In the next section, we describe the dynamical model that accounts for the target motion and embodies the relationship between the system state and the visual observations made at each sampling instant.

2.1. Model

Consider a camera moving with relation to a fixed point in space, as show in Figure 1.

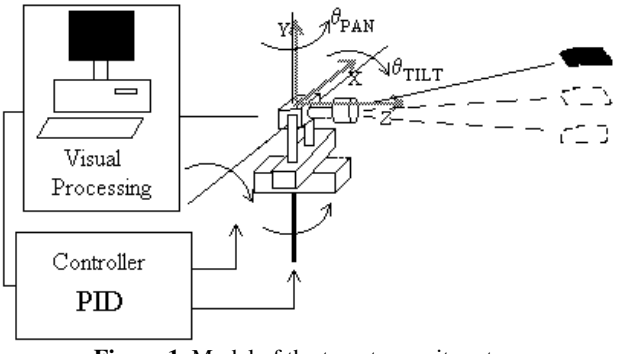

Figure 1. Model of the target pursuit system

We use the pinhole camera model [1,8]. According to this model, the location of a given image pixel (x,y) is determined by the perspective projection in the image plane of the corresponding 3D point, $[X \quad Y \quad Z]^T$:

$$x = \frac{f}{Z}X \quad , \quad y = \frac{f}{Z}Y \tag{1}$$

where *f* is the camera focal length.

The system state vector is defined as the position and velocity of the target center in the image plane according to :

$$X(k) = \left[x(k),\, y(k),\, \dot{x}_{obj}(k),\, \dot{y}_{obj}(k)\right]^T$$

As the camera is moving over time, the image velocity of the target is due both to the target 3D motion, and to the camera motion. In the system state, we consider only the component of the target image velocity due to the target 3D motion, by compensating for the observer egomotion.

$$\dot{x}_{obj} = \dot{x} - \dot{x}_{cam} \quad , \quad \dot{y}_{obj} = \dot{y} - \dot{y}_{cam} \tag{2}$$

where $\dot{x}$, $\dot{y}$ denote the target velocity in the image plane, $\dot{x}_{cam}$, $\dot{y}_{cam}$ are the components of the target motion in the image which is induced by the camera motion. Hence, $\dot{x}_{obj}$, $\dot{y}_{obj}$ are the target motion in the image plane solely due to the 3D motion of the object.

The camera motion is an external input to the system, inducing the image velocities $\dot{x}_{cam}$, $\dot{y}_{cam}$:

$$U(k) = \left[\dot{x}_{cam} \quad \dot{y}_{cam}\right]^T$$

The state equation is given by::

$$X(k+1) = FX(k) + GU(k) + v(k) \tag{3}$$

$$F = \begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } G = \begin{bmatrix} \Delta T & 0 \\ 0 & \Delta T \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

where *v(k)* is Gaussian, zero-mean, temporally uncorrelated noise vector with covariance Q, and ΔT is the sampling time interval.

At every sampled image, we observe the following variables:

$$M = \begin{bmatrix} x(k) \\ y(k) \\ \dfrac{x(k+1) - x(k)}{\Delta T} \\ \dfrac{y(k+1) - y(k)}{\Delta T} \end{bmatrix} = \begin{bmatrix} x(k) \\ y(k) \\ \dot{x}_{obj}(k) \\ \dot{y}_{obj}(k) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} U(k)$$

$$M(k) = CX(k) + DU(k-1) + w(k) \tag{4}$$

with C = I, the identity matrix, and

$$D = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T$$

where *w(k)* is Gaussian, zero-mean, temporally uncorrelated noise vector with covariance R.

2.2. Accounting for Head Motion

As mentioned before, the motion of the target perceived in the image plane is due to the target motion in the 3D space but also to the camera motion. As the state vector

depends on the component of the target image velocity that is caused by the target 3D motion, we have to determine the effect due to the camera motion so that it can be compensated.

In order to determine the image motion motivated by the observer movements, we have to introduce the Pan-tilt head kinematics. We will consider that the camera is rotating around the optical center to simplify the kinematic equations. Although this hypothesis might not be fulfilled in practice, we rely on the closed-loop control strategy to reduce the influence of these modeling approximations. Furthermore, this assumption makes the system independent of the kinematic parameters which are hard to calibrate.

The 3D motion of the point P can be expressed in the camera coordinate system as :

$$V = \frac{dP}{dt} = \begin{bmatrix} \dot{X} & \dot{Y} & \dot{Z} \end{bmatrix}^T = \omega \times P \tag{5}$$

where ω expresses the camera angular velocity. In our experimental setup the camera cannot rotate around the optical axis (ciclotorsion), and therefore the velocity of P can be written as :

$$\begin{bmatrix} \dot{X} & \dot{Y} & \dot{Z} \end{bmatrix}^T = \begin{bmatrix} \omega_{TILT} & \omega_{PAN} & 0 \end{bmatrix}^T \times \begin{bmatrix} X & Y & Z \end{bmatrix}^T$$

By introducing the perspective projection equation, this velocity can be expressed in the image plane as :

$$\begin{aligned} \dot{x}_{cam} &= 1/f \left[f^2 \omega_{PAN} - xy\omega_{TILT} + x^2 \omega_{PAN} \right] \\ \dot{y}_{cam} &= 1/f \left[f^2 \omega_{TILT} + xy\omega_{PAN} - y^2 \omega_{TILT} \right] \end{aligned} \tag{6}$$

A further simplification can be used due to the tracking constraint, by assuming that the target location is close to the image center, i.e. $(x,y) \approx (0,0)$. In this case, the motion equations can be simplified to :

$$\dot{x}_{cam} = f\omega_{PAN} \quad , \quad \dot{y}_{cam} = f\omega_{TILT} \tag{7}$$

In summary, when the camera is rotating with velocities ω_{PAN}, ω_{TILT}, the motion induced in the image plane is given by $\dot{x}_{cam}$, $\dot{y}_{cam}$.

3. Target Pursuit

In order to achieve smooth pursuit, the system must be able of measuring the target motion. In this work we utilise two different methodologies for this purpose: Image Correlation and Active Snakes. Our starting point, therefore, is the real time computation of the target centroid, assuming the target is in a small window in the centre of the field of view. Using the target centroid leads to a further reduction of information. A priori, it presents the advantage of being more robust for tracking, since it is the result of an averaging process.

3.1. Movement Detection

The initialization step consists in detecting a moving object in the scene, which will then be tracked over time.

The goal of the motion detection implemented is to perform a segmentation between static and dynamic regions between two consecutive images, in order to determine the initial target location. The target detection algorithm consists in segmenting the difference on two consecutive images with an automatic threshold. Then, we take the centroid of the resulting region and obtain the 2D target position in image coordinates.

Therefore, we assume that the camera is initially stationary. If the camera were moving during this period, there would be motion induced over all the image plane and the segmentation would become much more difficult. In such cases, one could use image stabilisation methods [12] or multiple motion estimation at various resolutions, as described in [10] where a coarse-to-fine strategy is employed on a relaxation process, over an image pyramid. An example of the initial motion detection step is shown in Figure 2.

Figure 2. a) Image differences b)Segmentation c) Snake initialization

3.2. Correlation

Once the initial position of the target has been found, we use correlation to track it over the sequence of images. We have implemented this algorithm in two versions: The Sum of Squared Differences (SSD) criterion and the Normalised Cross-Correlation (NCC).

3.2.1. Sum of Squared Differences

The sum of squared gray level differences can be used to quantify the "likelihood" of a matching decision between the target centroid on the previous and current frames:

$$SSD(x,y,u,v) = \sum_i \sum_j \left(I(i,j) - I_1(u+i, v+j) \right)^2 \tag{8}$$

where $I(p,q)$, $I_1(p,q)$ designates the pixel (p,q) gray level intensity of the image acquired at time (k-1) and k, respectively.

3.2.2. Normalised Correlation

$$NCC(x,y,u,v) = -\frac{\sum_i \sum_j \left(I(i,j) - \overline{I(x,y)} \right) \left(I_1(u+i, v+j) - \overline{I_1(x+u, y+v)} \right)}{\sigma(x,y)\sigma_1(x+u, y+v)} \tag{9}$$

where $\overline{I(p,q)}$, $I_1(p,q)$ and $\sigma(p,q)$, $\sigma_1(p,q)$ are the mean and standard deviation of the image (acquired at time (k-1) and k, respectively) intensity values at the point (p,q), and can be computed, for i =(k-1, k) by:

$$\overline{I_i(p,q)} = \frac{1}{(2N+1)(2P+1)} \sum_{i=-N}^{N} \sum_{j=-P}^{P} I_i(p+i, q+j) \tag{10}$$

$$\sigma_i^2(p,q) = \sum_{i=-N}^{N} \sum_{j=-P}^{P} \left(I_i(p+i,q+j) - \overline{I_i(p,q)} \right)^2 \qquad (11)$$

Notice that, due to the normalisation by σ and σ_1, NCC is normalized between -1 and 1 [8].

The minimum peak in the correlation surface is selected as the position of the feature vector in the new image. A considerable improvement in system performance in terms of both computing time and error rate is achieved by selecting a search window enclosing the target centroid, rather than considering the whole image.

3.2.3. Computing Uncertainty

Once computed the cost functional, that establishes a matching between two consecutive frames, a method must be chosen to estimate the state of the target.

One method usually used is to fit a quadratic surface to a neighborhood of the minimum correlation point. Another method is to fit two one dimensional quadratic curves, both in the x and y directions, which seems to be more robust to noise. Due to the increased robustness to noise and simplicity, we adjust two 1 dimensional quadratic curves in each direction, in a neighborhood of the minimum correlation point, and its equivalent to approximate the cost functional near the minimum by a quadratic curve [18]:

$$p(n) = an^2 + bn + c \qquad (12)$$

where *a*, *b* and *c* are estimated from data, and *n* is a variable that represents points of the curve. Using 3 correlation measurements we have 3 equations with 3 unknown variables to compute *a*, *b* and *c* for *both x* and *y*. Once the analytical expression for the error surface has been found, the minimum point can be determined with sub pixel resolution, yielding the optimal subpixel position estimate:

$$n^* = -\frac{b}{2a} \qquad (13)$$

whenever the coefficient *a* is different from zero.

The uncertainty of the estimate is related to the shape of the correlation error surface, since the sensitivity of the computation of the minimum point depends on the error surface curvature. Uncertainty estimates are related to second order terms, describing the error surface curvature. In each direction, we will approximate the estimate variance by:

$$\sigma_n^2 = \left(\frac{\partial^2 p(n^*)}{\partial n^2} \right)^{-2} p(n^*) \qquad (14)$$

The first term of (14) expresses the decrease of the uncertainty with the increase of the error surface curvature, the second term is a normalising factor dependent of the minimum value of the correlation surface.

The estimated variance is given by (15), using equations (12), (13) and (14):

$$\sigma_n^2 = -\frac{1}{4a^2}\left(c - \frac{b^2}{4a}\right) \qquad (15)$$

As explained, the analysis of the shape of the correlation surface provides an estimate of the uncertainty associated to the measurement of the target position in the image plane. Therefore, it can be used to determine the covariance matrix associated to the observations in our dynamical model. Referring to equation (4), we have

$$R = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 \\ 0 & 0 & \dfrac{\sigma_x^2}{\Delta T^2} & 0 \\ 0 & 0 & 0 & \dfrac{\sigma_y^2}{\Delta T^2} \end{bmatrix}$$

3.3. Snakes - Active Deformable Contours

Snakes (active contour models) are deformable contours that have been used in many image analysis applications, including the image-based tracking of rigid and nonrigid objects [2,9,11,15].

This deformable contour moves under the influence of image forces. We can define a deformable contour by constructing a suitable deformation energy $P_i(z)$, where z represents the contour points. The external forces on the contour result from a potential $P_e(z)$. A snake is a deformable contour that minimises the energy [11]

$$P(z) = P_e(z) + P_i(z), \quad z(s) = (x(s), y(s)) \qquad (16)$$

For a simple (linear) snake, the internal deformation energy is

$$P_i = \int_0^T \frac{\alpha(s)}{2} |\dot{z}(s)|^2 + \frac{\beta(s)}{2} |\ddot{z}(s)|^2 ds \qquad (17)$$

where the differentiation is in respect to s. This energy function models the deformation of a stretchy, flexible contour z(s) and includes two physical parameter functions: $\alpha(s)$ controls the *tension* and $\beta(s)$ the *rigidity* of the contour.

To attract the snake to edge points we specify the external potentials $P_e(z)$:

$$P_e(z) = \int_0^T Pimage(z(s)) ds \qquad (18)$$

where *Pimage(x,y)* is a scalar potential function defined over the image plane and which is typically computed through image processing. The local minima of Pimage can be viewed as snake *attractors*. Hence, the snake will have an affinity to intensity edges if

$$Pimage(x,y) = -\left[G_\sigma * \left| \nabla I(x,y) \right| \right] \qquad (19)$$

It represents the convolution of the image gradient with a gaussian smoothing filter whose characteristic width, σ, controls the spatial extent of the attractive *depression* of *Pimage*. However, this convolution is very time

expensive, but one can convolve the image with two one-dimensional gaussians, since the 2D gaussian is a separable filter. Instead, we convolve only 3N points with the gaussian, that are the N control points and two neighbors necessary to compute Pimage gradient. This is possible because our numerical solution is performed only in one iteration.

3.4. Integration of Visual Clues

We can use the dynamic model defined in Section 2.1 to integrate both correlation and snake-based visual measurements. This is achieved by extending the observation vector in (4) to contain the two types of data.

4. Kalman-Bucy Filtering

After describing how to measure the motion of the target, to achieve smooth pursuit, we will introduce the Kalman-Bucy filter to estimate the target position and velocity over time.

The filtering equations comprise a prediction step and an update/filtering step. Centroids are tracked from frame to frame using a constant velocity Kalman-Bucy filter. In the prediction stage, we predict the position of the target in the actual frame based on past information and on the dynamic model.

Prediction : (20)

$$\hat{X}(k|k-1) = F\hat{X}(k-1|k-1) + GU(k-1)$$
$$P(k|k-1) = FP(k|k)F^T + Q$$

where $\hat{X}(k|k-1)$ is the predicted state value for time k, based on the data available up to time $(k-1)$. The state covariance matrix is denoted by P(k|k-1) and Q is the covariance matrix of the noise in the motion equation.

The predicted state covariance estimate can be used to determine the size of the search windows when performing correlation to locate the target. As the covariance increases, we have to enlarge the dimension of the search space, whereas when the target position is predicted with low uncertainty, a smaller search window is sufficient. Hence, we use the information on the predicted state covariance to control an adaptive correlation search window.

At time k, when a new measurement becomes available, the predicted state can be updated, in the filtering step

Filtering : (21)

$$K(k) = P(k|k-1)C^T\left[CP(k|k-1)C^T + R\right]^{-1}$$
$$\mathrm{P(k|k)} = \mathrm{[I - K(k)C]P(k|k-1)}$$
$$\hat{X}(k|k) = \hat{X}(k|k-1) + K(k)\left[M(k) - C\hat{X}(k|k-1) - DU(k-1)\right]$$

The initial conditions are $\hat{X}(0|-1)$ e $P(0|-1)$, which are the initial state and state uncertainty estimates, respectively. K(k) is the Kalman gain and R is the covariance matrix of the measurement equation noise.

5. Visual Control

The control system will use visual measurements of the target position in the image to drive the pan, tilt degrees of freedom of the head. The goal consists in keeping the target position centered in image.

The error signal is defined as the difference between the target image position and the reference position $(0,0)$:

$$e_k = \begin{bmatrix} x_k & y_k \end{bmatrix}^T$$

where x_k, y_k are the filtered components of the state vector updated by the Kalman-Bucy filter.

As explained in Section 2.3, for a given image position of the target, we can determine the motion (Pan and Tilt) that is required to perform in order to recenter the target in the image. Hence, a PID controller can be defined as:

$$\Delta\theta^{\,k} = \begin{bmatrix} \Delta\theta^{\,k}_{PAN} & \Delta\theta^{\,k}_{TILT} \end{bmatrix}^T$$
$$= K_p\left\{e(k) + K_i^T\sum_n e(k-n) + K_d^T(e(k) - e(k-1))\right\} \qquad (22)$$

The pan-tilt head that is used in this work allows not only to specify the desired joint positions but also the velocity to be used from an initial joint position to the final one. Then, the position commands and joint speed sent to the head corresponds to:

$$\begin{aligned} \theta^k_{PAN} &= \theta^{k-1}_{PAN} + \Delta\theta^k_{PAN} \quad , \quad \theta^k_{TILT} = \theta^{k-1}_{TILT} + \Delta\theta^k_{TILT} \\ \dot{\theta}^k_{PAN} &= \Delta\theta^k_{PAN}/\Delta T \quad , \quad \dot{\theta}^k_{TILT} = \Delta\theta^k_{TILT}/\Delta T \end{aligned} \qquad (23)$$

5.1. Dealing with Delays in the Feedback System

A typical problem in vision based control is the delay introduced by the image processing algorithms. Although stability can often be achieved, this latency will lead to a delayed response of the system.

To overcome this problem, one could predict the target trajectory [4,6]. In our case, the predicted state computed by the Kalman filter, or a simples α - β or the α - β - γ predictor could be used to compensate for the processing delay. However, this approach may lead to system instabilities, and at the current stage we chose to use a simpler PID controller. A review of control of systems with delay in the context of vision can be found in [6].

6. Experimental Results

The experimental setup used in this work consist in a Directed Perception PTU-46-17.5 pan-tilt unit (see Fig.3) and a EDC-1000L digital camera with a 12.5mm lens. The pan-tilt unit is controlled from the hosts via a serial link. The main limitation found throughout the work is that the minimum speed of the pan-tilt unit is quite high (1.5 deg./s). The hosts system is a 486 DX4/120MHz PC.

We have performed different experiments with different targets and using various tracking strategies. To illustrate the results we present every 12^{th} image acquired during a

tracking experiment. The image size is 138×276 pixels and a window of 92×92 pixels is processed.

Figure 3. Camera mounted in the Pan and Tilt unit

Figure 4 shows an image sequence obtained while tracking a person, using correlation and the Kalman filter approach. The correlation window used is 9x9 pixels. Since there are homogeneous regions in the target image, the correlation measurements are rather noisy, particularly in the vertical direction.

Figure 4. Results of the visual pursuit for correlation algorithm using a Kalman-Bucy filtering approach.

As shown in Figure 5a, the target initial location is very far from the image center. After performing a saccade, the system starts tracking the moving person. The target moves both in the vertical and horizontal directions and there is another person crossing the visual field during the experiment. Fig. 5b shows the target image speed while Figure 5c shows the evolution of pan-tilt angles. Fig. 5d shows the reference pan-tilt angles sent to the head.

Another experiment was conducted using dynamic contours as the visual input to the tracking system The snake is initialized manually. The same initial snake was used to tracking various moving persons, based on 36 control points and a gaussian smoothing by a 9x9 kernel. The processing time varies linearly with the number of control points used and the number of gaussian kernel elements.

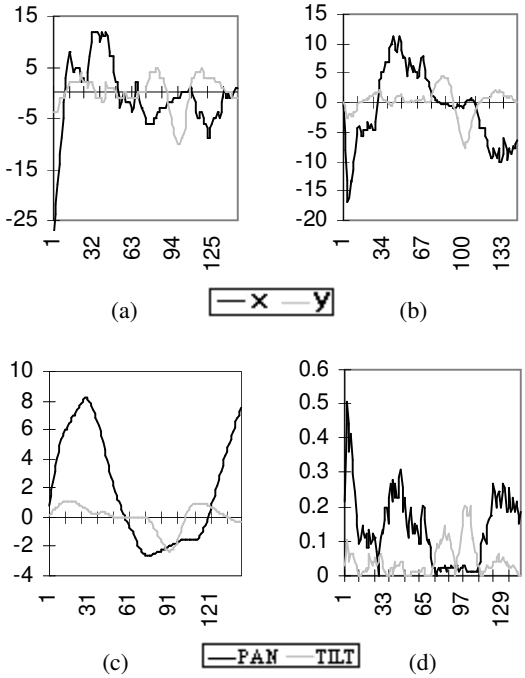

Figure 5. State evaluation for the image sequence in Figure 4, for all frames in the horizontal axis, using correlation. (a) Position in images (pixels). (b) Target velocity in images (pixels/frame). (c) Pan and Tilt camera position in Degrees. (d) Desired Pan and Tilt velocity in Degrees/frame.

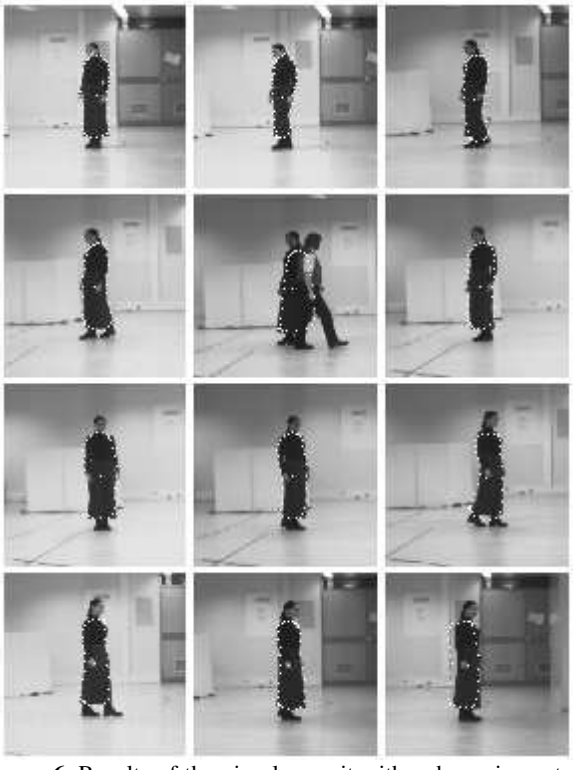

Figure 6. Results of the visual pursuit with a dynamic contour in a Kalman-Bucy filtering approach.

The tracking sequence is shown in Figure 6, where the temporal evolution of the snake is shown. Again, another moving person enters the visual field, partially occluding the target, without significantly disturbing the tracking.

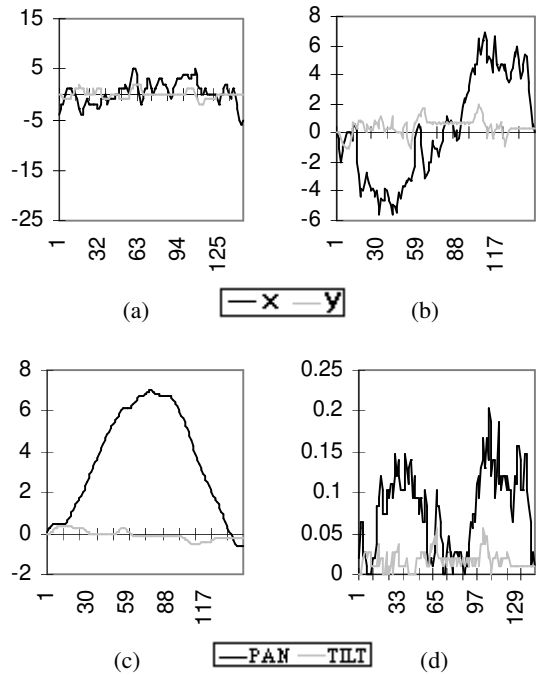

Figure 7. State evaluation for the images sequences in Figure 6, for all frames, using a snake. (a) Position in images (pixels). (b) Target velocity in images (pixels /frame). (c) Pan and Tilt camera position in Degrees. (d) Desired Pan and Tilt velocity in Degrees/frame.

Figure 8. Results of the target visual pursuit with integration of correlation measurements with a dynamic contour using a Kalman-Bucy filtering approach.

In Figure 7, we show the position of the geometric center of the snake (7a) which is close to the image center; the target image velocity (7b); the evolution of the pan-tilt angles (7c) and the control signals (7d).

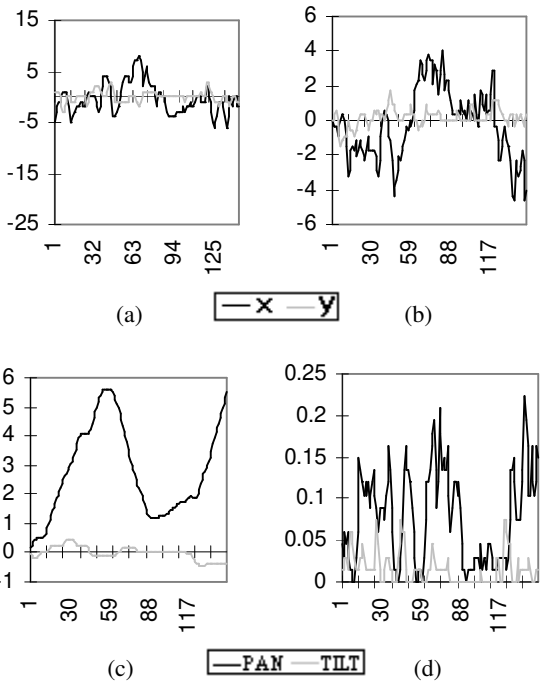

Figure 9. State evaluation for the images sequences in Figure 8, for all frames, integrating data. (a) Position in images (pixels). (b) Target velocity in images (pixels /frame). (c) Pan and Tilt camera position in Degrees. (d) Desired Pan and Tilt velocity in Degrees/frame.

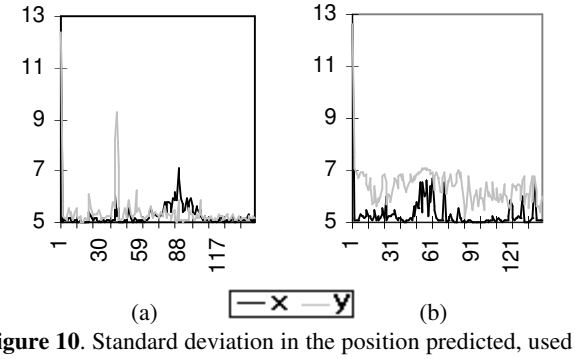

Figure 10. Standard deviation in the position predicted, used to select the search correlation window. (a) Using correlation (b) Using a Kalman-filter for data integration.

The final experiment was obtained by integrating both the correlation and snake-based approaches using the Kalman filter structure. With this integration one could profit from the benefits of each of the methodologies, thus improving the overall system robustness.

The results are shown in the image sequence of Figure 8. As before the system is tracking a moving person. During the sequence, another person crosses the visual field.

In Figure 9, we show the target position (9a), pan-tilt angles (9b), target velocity (9c) and reference angles (7d). In general the system succeeds in keeping the target close to the image center and can cope with another moving object crossing the visual field. It can be seen that most of the motion is carried out in the horizontal direction although a slight action is performed by the tilt joint.

Whenever correlation is used as a tracking input, we use an adaptive search windows (see Figure 10) whose size is determined by the uncertainty of the predicted state of the Kalman filter. Hence, if the position uncertainty increases in one direction, the window size increases in the same direction.

Using correlation is very effective when the target is highly textured. The snake has the disadvantage of requiring a prior initialization but can be very efficient when the target contours are well defined. By combining both approaches, the system can deal with situations where any of the methods alone would fail. Of course, the integration of both methodologies increases the computation time.

6.1. Time Performances

It is important to compare various vision algorithms not only in terms of their relative performance (accuracy, etc) but also the required computational effort.

Method	Visual Processing	Total	Average
Correlation (search in x)	50 ms	160 ms	160 ms
Correlation using Kalman filter, search window 17x17	220 ms	330 ms	346 ms
Correlation using Kalman filter with an adaptive window	160 ms	270 ms	291 ms
Snakes	50 ms	160 ms	197 ms
Snakes and Kalman filter	160 ms	270 ms	231 ms
Fusing snakes and correlation with an adaptive window	270 ms	380 ms	393 ms

Table 1. Time performances for all methods. In column 2 are the processing times. Column 3 shows the time needed to acquire, processing and send commands. The last column shows an averaged time in 10 samples.

The computation time for the various methods is shown in Table 1. Notice that the image acquisition takes about 60ms plus 50ms when the camera cleans the CCD. The control commands take about 50ms to be sent. This limitations are exclusively due to the hardware constraints both of the digital camera and the pan-tilt unit controller. The visual processing time, when considered alone, is relatively low and allow fast sampling frequencies, if the hardware does not impose such long time delays. In this section we compare the different methods in terms of the computing time required for tracking and pursuit a target.

7. Conclusions

We have presented a monocular tracking system to follow a target moving in the scene. We have modeled the target dynamics and used a Kalman filter to update the target image position over time. As the visual measurements we have presented a correlation and a snake based approaches. These two strategies have also been integrated using the Kalman filter to improve the overall system performance.

We presented results on a number of experiments using the various methodologies proposed in the paper. The running frequency is currently highly dependent on the delays introduced by the digital camera hardware and pan-tilt unit controller. In general the results illustrate the robustness of the approach and the system can deal with illumination changes and multiple moving objects.

As future improvements, we plan to address the problem of coping with the delay introduced by the image processing, when designing the closed loop controller. Furthermore the used of B-Splines, instead of snakes, may prove more robust against slippage in the contour control points. Finally, the initial motion detection algorithm will be extended to cope with the existence of multiple moving objects in the scene, even during the startup phase.

As a final comment we would like to stress the performance of the system both in terms of the accuracy obtained and the running frequencies achieved using a modest computational platform.

8. References

[1] D. Ballard, C. Brown, “Computer Vision”, Prentice-Hall, 1982

[2] B. Bascle and R. Deriche, “Energy-based methods for 2D curve tracking, reconstruction and refinement of 3D curves and applications”, SPIE Vol. 2031, 1993.

[3] A. Black, R. Curwen and A. Zisserman, “A framework for Spatiotemporal Control in the Tracking of Visual Contours”, IJCV, 11, nº2, 127-145, 1993.

[4] C. Brow, D. Coombs and J. Soong, “Real-time Smooth Pursuit Tracking”, in Active Vision, MIT Press, 1992

[5] R. Cipolla and A. Black, “Surface Orientation and Time to Contact from Image Divergence and Deformation”, in Active Vision, MIT Press, 1992

[6] D. J. Coombs, “Real-time Gaze Holding in Binocular Robot Vision”, PhD Thesys.

[7] Rupert Curwen and Andrew Blake, “Dynamic Contours: Real-time Active Splines”, in Active Vision, MIT Press, 1992

[8] Olivier Faugeras, “Three-Dimensional Computer Vision - A Geometric Viewpoint”, the MIT Press, 1992

[9] Donald B. Gennery, “Visual Tracking of Known Three-Dimensional Objects”, IJCV, 7:3, 243-270 ,1992

[10] Sylvia Gil, R. Milanese and T. Pun, “Feature selection for object tracking in traffic scenes”, in SPIE, Boston, Oct.31-Nov.4, 1994

[11] M. Kass, A. Witkin and D. Terzopoulos, “SNAKES: Active Contour Models”, IJCV, Vol 1, No. 4, January 1988

[12] P. Nordlund and T. Uhlin, “Closing the Loop: Detection and Pursuit of a Moving Object by a Moving Observer”, Computational Vision and Active Perception Lab., 1995.

[13] I.D. Reid and D.W. Murray, “Active tracking of foveated feature clusters using affine structure”, IJCV, 18, 41-60, 1996.

[14] Demetri Terzopoulos and Richard Szeliski, “Tracking with Kalman Snakes”, in Active Vision, MIT Press, 1992

[15] Demetri Terzopoulos and Dimitri Metaxas, “Tracking Nonrigid 3D Objects”, in Active Vision, MIT Press, 1992

[16] D. Terzopoulos, A. Witkin and M. Kass, “Constraints on Deformable Models: Recovering 3D Shape and Nonrigid Motion”, AAAI-87 Conference

[17] José Santos-Victor and Giulio Sandini, “Visual Behaviours for Docking”, Computer Vision and Image Understanding, 96 (in press).

[18] José Santos-Victor and João Sentieiro, “A 3D Vision System for Underwater Vehicles: An Extended Kalman-Bucy Filtering Approach”, ASPOE, 567-580, 1993.