Robust Visual Tracking by an Active Observer

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. The ability to track a moving object has many applications in robotics, teleoperation, surveillance systems, human-machine interfaces, etc. Our goal was the development of a robust tracking system for practical (industrial) applications and therefore based on inexpensive hardware.

The strategy we present is based on the integration of correlation based techniques together with active contours, using a Kalman filtering approach. 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.

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.

Robust tracking is the basis of a wide range of applications such as advanced teleoperation, human-machine interfaces, virtual reality, surveillance and autonomous or semi-autonomous robotic systems.

This work aims at developing a reliable monocular tracking system to keep a moving object centred in the image plane. A major constraint on the complexity of the algorithms to use is imposed by the closed loop frequency required for the system operation.

Artur Arsénio was supported by a grant from PRAXIS XXI-BM/7153/95. This research was partially funded by projects JNICT-PBIC/TPR/2550/95 and EC-ESPRIT/LTR Project NARVAL.

To enhance the tracking robustness, we integrate both correlation measurements and deformable contours [12]. Additionally, we use 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.

Many methods have been presented to track a moving object over a sequence of images, based on optical flow, image correlation or deformable contours.

The use of optical flow involves the segmentation of the object to be tracked from the scene background [14]. This problem is far from being solved and the existing algorithms (typically segmenting various motion models) are very time consuming.

Murray et al. in [12] extract corner points of the target image, which are then tracked using correlation. They assume that the target moves with known constant velocity relative to the camera. Correlation techniques have been widely used also in stereo [15].

Kass et al. [10] presented models which relate to general physical properties of objects such as the cohesion of matter and smoothness. Some extensions are presented in [13] where snakes are modeled as a dynamic system, and a Kalman-Bucy filter approach is used. Curwen and Blake [8] impose shape assumptions on the deformable contour (B-Splines); although large target deformations are not allowed. Recent approaches use 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 changes 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. To overcome this limitation we maintain an estimate of the snake geometric center through the use of a Kalman-Bucy filter. Hence, the

snake points are estimated given the expected position of the snake center.

Then, 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.

2. Tracking System

The vision system comprises the image acquisition module, the visual processing module and the Pan and Tilt Unit (PTU) controller, as 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 generated. The visual processing module is responsible for the estimation of the target position in the current image. The closed loop controller determines the appropriate control signals for the tracking.

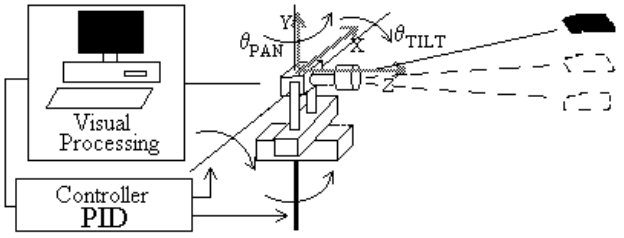


Figure 1. Model of the target pursuit system

2.1. Dynamical Model

The dynamical model accounts for the target motion and embodies the relationship between the system state and the visual observations made at each sampling instant.

Under perspective projection a 3D point $[X \quad Y \quad Z]^T$ is imaged in the pixel coordinates (x,y) given by:

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

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

The camera motion acts as 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 a 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 the 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* and 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.

When considering the Pan-Tilt head kinematics, we assume that the camera is rotating around the optical center. 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 :

$$V = -\frac{dP}{dt} = \left[\dot{X} \quad \dot{Y} \quad \dot{Z}\right]^T = -\omega \times P \tag{5}$$

where $\omega = [\omega_{TILT} \quad \omega_{PAN} \quad 0]^T$ expresses the camera angular velocity. In our experimental setup the camera cannot rotate around the optical axis (ciclotorsion).

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

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

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 :

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

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. We utilise two different methodologies for this purpose: Image Correlation and Active Snakes. The target centroid is computed in real-time assuming that the target is inside a small window in the centre of the visual field. Using the target centroid leads to a further reduction of information. It has the advantage of being more robust for tracking, since it results 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 target initial position is obtained by automatic thresholding the difference between consecutive images, and computing the centroid of the resulting region.

In this stage, we assume that the camera is initially stationary. An example of the initial motion detection step is shown in Figure 2.

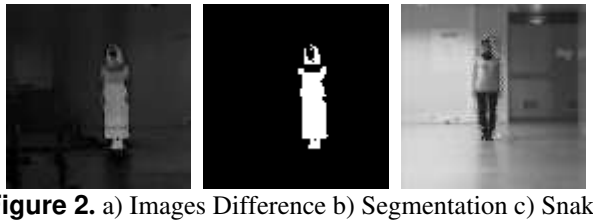


Figure 2. a) Images Difference b) Segmentation c) Snake initialization.

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*).

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:

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

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.

The *NCC* cost functional is given by:

$$NCC(x,y,u,v) = 1 - \frac{\sum_i \sum_j \left(I(i,j) - \overline{I(x,y)}\right)\left(I_1(u+i, v+j) - \overline{I_1(x+u, y+v)}\right)}{\sigma(x,y)\sigma_1(x+u, y+v)} \quad (9)$$

where $\overline{I(p,q)}$, $\overline{I_1(p,q)}$ 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). Notice that, due to the normalisation by σ and σ_1, the *NCC* is normalized between 0 and 2. 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.

3.3. Computing the Target Position and Uncertainty

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

To determine the target position, we adjust two one-dimensional quadratic curves to the correlation surface in the direction of the image axis, in a neighborhood of the minimum [15]:

$$p(x) = ax^2 + bx + c \quad (10)$$

where a, b and c are estimated from data, and x is a variable that represents points of the curve. Using three correlation measurements we can 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:

$$x^* = -\frac{b}{2a} \quad (11)$$

whenever a is different from zero.

The uncertainty of the estimate is related to the curvature of the correlation error surface, which determines the sensitivity of the computation of the

minimum point. In each direction, we will approximate the estimate variance by:

$$\sigma_x^2 = \left(\frac{\partial^2 p(x^*)}{\partial x^2} \right)^{-2} \qquad p(x^*) = -\frac{1}{4a^2}(c - \frac{b^2}{4a}) \quad (12)$$

The first term of (12) expresses the decrease of the uncertainty when the error surface curvature increases, the second term is a normalising factor dependent of the minimum value of the correlation surface.

The estimate of the uncertainty associated to the measurement of the target position in the image plane can be used to determine the covariance matrix associated to the observations in our dynamical model. Referring to equation (4), we have:

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

3.4. 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 [2,10]. 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 [10]:

$$\mathrm{P(z)} = P_e(z) + P_i(z), \qquad \mathrm{z(s)=(x(s),y(s))} \quad (13)$$

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

$$P_i = \int_0^T \frac{\alpha(s)}{2}|\dot{z}(s)|^2 + \frac{\beta(s)}{2}|\ddot{z}(s)|^2 ds \quad (14)$$

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 $P_e(z)$:

$$P_e(z) = \int_0^T Pimage(z(s))ds \quad (15)$$

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

$$Pimage(x,y) = -\left[G_\sigma * |\nabla I(x,y)| \right] \quad (16)$$

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 the gradient.

3.5. Integration of Visual Cues

Depending on the scene structure, correlation can outperform the snakes-based approach and vice-versa. We can use the dynamic model defined in Section 2.1 to integrate both correlation and snake-based visual measurements, extending the observation vector in (4) to contain the two types of data. Hence, the integration of visual cues is achieved by duplicating the observation equation (4) in the state space model. Then, the Kalman filter described below provides a natural framework for the state estimate from these multiple measurements.

4. Kalman-Bucy Filtering

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

We assume that the target is moving with constant velocity. In the prediction stage, we estimate the position of the target in the current frame based on past information and on the dynamic model.

Prediction : (17)

$$\hat{X}(k|k-1) = F\hat{X}(k-1|k-1) + GU(k-1)$$

$$P(k|k-1) = FP(k|k)F^T + Q$$

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)*.

The predicted state covariance estimate is used to determine the size of an adaptive search window when performing correlation to locate the target. As the covariance increases, we have to enlarge the dimension of the search space.

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

Filtering : (18)

$$K(k) = P(k|k-1)C^T \left[CP(k|k-1)C^T + R \right]^{-1}$$

$$\mathrm{P(k|k)} = \left[\mathrm{I - K(k)C} \right] \mathrm{P(k|k-1)}$$

$$\hat{X}(k|k) = \hat{X}(k|k-1) + K(k)\left[M(k) - C\hat{X}(k|k-1) - DU(k-1) \right]$$

The initial state and uncertainty estimates are denoted by $\hat{X}(0|-1)$ and $P(0|-1)$. *K(k)* is the Kalman gain.

5. Visual Control

The control system uses visual measurements of the target position in the image to drive the pan and 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 2.2, 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:

$$\Delta\theta^{\,k} = \begin{bmatrix} \Delta\theta^{\,k}_{PAN} & \Delta\theta^{\,k}_{TILT} \end{bmatrix}^T$$

$$= K_p \left\{ e(k) + K_i^T \sum_n e(k-n) + K_d^T (e(k) - e(k-1)) \right\} \quad (19)$$

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 [4,5,7]. In our case, the predicted state is computed by the Kalman filter, but it could be done by a simple α - β or α - β - γ predictor. However, this approach may lead to system instabilities. A review of control of systems with delay in the context of vision can be found in [7].

6. Experimental Results

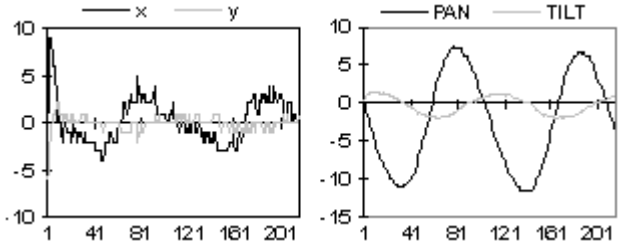
The experimental setup used consist in a Directed Perception PTU-46-17.5 pan-tilt unit (see Figure 3) and a EDC-1000L digital camera with a 12.5mm lens. The pan-tilt unit is controlled from the host via a serial link. The main limitation found throughout the work is that the minimum speed of the pan-tilt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

Figure 3. Camera mounted in the Pan and Tilt unit

We have performed different experiments with different targets and using various tracking strategies. The results are presented every 16^{th} image. The image size is 138×276 pixels and a window of 92×92 pixels is processed.

The results were 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 first experiment (Figure 4a) was performed with a target moving in circles on a plane. Due to the observer angle of view, the image trajectory is an ellipsis, with the major axis in the Pan direction. The target is moving with a relatively large velocity (the highest value is about 40 pixels/sec). When this velocity is approximately constant (in the linear zone of Figure 4b), the positional error can be disregarded, as the target is moving according to our dynamical model. This error grows when the aceleration of the target increases (after changing the velocity sign). As shown in Figure 4a, the absolute position error is under five pixels. The peak at the beginning is due to the saccade movement performed to centre the target when it is first detected.



(a) (b)

Figure 4. State evaluation (a) Position in images (pixels). (b) Pan and Tilt camera position in Degrees.

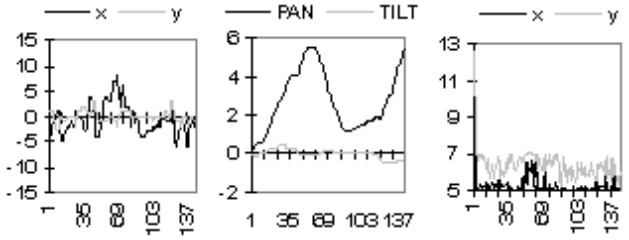
Another experiment was performed with the system tracking a moving person. The results are shown in the image sequence of Figure 5. In Figure 6, we show the target position (6a), and pan-tilt angles (6b). In general the system succeeds in keeping the target close to the image center and can cope with 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 window (see Figure 6c) whose size is determined by the uncertainty of the predicted state of the Kalman filter.

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.

Figure 5. Results of the target visual pursuit with integration of correlation measurements with a dynamic contour using a Kalman-Bucy filtering approach.



(a) (b) (c)

Figure 6. State evaluation for the images sequences in Figure 5, for all frames, integrating data. (a) Position in images (pixels). (b) Pan and Tilt camera position in Degrees. (c) Standard deviation in the position predicted, used to select the search correlation window.

Currently, the computation time is 50ms for Snakes with 36 control points and a 9×9 gaussian window, 160ms with Kalman filter, 160ms for correlation with Kalman filter, and 270ms for the integration of both methods. Hardware limitations impose an overhead of about 110ms for image acquisition and sending commands. The visual processing time, when considered alone, would allow fast sampling frequencies.

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. We have presented a correlation and a snake based approaches, which are 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 use B-Splines instead of snakes, that may prove more robust against slippage in the contour control points. We intend to model the uncertainty of this type of measurement, and, as we did for correlation, perform a spatiotemporal control in the tracking of contours. Finally, the initial motion detection algorithm will be extended to cope with multiple moving objects in the scene during startup.

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 an inexpensive computational platform, which may have an important impact in practical applications.

8. References

[1] D. Ballard and 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, nº 2, 127-145, 1993.

[4] C. Brown, “Gaze Controls with Interactions and Delays”, in IEEE Trans. on Systems, man, and Cybernetics, 20(1), 1990.

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

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

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

[8] R. Curwen and A. Blake, “Dynamic Contours: Real-time Active Splines”, in Active Vision, MIT Press, 1992

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

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

[11] 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.

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

[13] D. Terzopoulos and R. Szeliski, “Tracking with Kalman Snakes”, in Active Vision, MIT Press, 1992

[14] J. Santos-Victor and G. Sandini, “Visual Behaviours for Docking”, Computer Vision Image Understanding, 1997.

[15] J. Santos-Victor and J. Sentieiro, “A 3D Vision System for Underwater Vehicles: An Extended Kalman-Bucy Filtering Approach”, Acoustic Signal Processing for Ocean Exploration, 567-580, 1993.