*British Machine Vision Conference 2005*

# Robust Non-Rigid Object Tracking Using Point Distribution Models

Tom Mathes and Justus H. Piater
Department of Electrical Engineering and Computer Science
Institut Montefiore, University of Liège
Building B28, B-4000 Liège, Belgium
mathes@montefiore.ulg.ac.be, justus.piater@ulg.ac.be

**Abstract**

This paper presents a robust approach to non-rigid object tracking in video sequences. The object to track is described by a 2-dimensional point distribution model whose landmarks correspond to interest points that are automatically extracted from the object and described by their geometrical position and their local appearance. The approach is novel in that we describe the appearance locally instead of using the raw texture information. This provides a natural way to robustly handle partial occlusions. A second contribution is that we present a method that allows to learn the model automatically. Our algorithms have been successfully tested on several video streams taken from soccer games and video surveillance footage. They have been implemented with the aim of achieving near real-time performance.

## 1 Introduction

Object tracking plays a crucial role in many applications such as video surveillance, sports broadcasts, scene monitoring or medical image analysis. Most current methods are application dependent and many of them are based on blobs [10], color histograms [13, 12], points or contours [18], each of them having its own limitations. In this paper we present a new model-based approach able to deal with non-rigid objects, partial occlusions and non-static cameras. The advantage is that no background model needs to be used. We are particularly interested in highly dynamic scenes with many interacting targets.

We use a new method describing an object by a Point Distribution Model (PDM), using feature vectors for local appearance instead of the raw texture information. This situates our approach somewhere between Active Shape Models (ASMs) and Active Appearance Models (AAMs) [1, 2], thus providing a natural way to robustly handle severe partial occlusions. Another contribution is a method for automatic landmarking to construct the model while at the same time tracking the object, albeit somewhat less robust than the model-based. The user only needs to initialize the tracker in the first frame. After a fixed number of frames the tracker can automatically switch from the learning-phase tracking to the model-based tracking or save the model for later use.

Section 2 describes how we extract interest points and how we describe their local appearance. Section 3 introduces the concept of PDMs and explains how they can be exploited for tracking purposes, while Section 4 shows how PDMs can be constructed from

This work has been sponsored by the Région Wallonne under DGTRE/WIST contract 031/5439.

different instances of the same object. This requires automatic landmarking by tracking individual interest points, as described in Section 5. The complete tracking algorithm is detailed in Section 6 and tracking results obtained for real soccer and video surveillance sequences are presented in Section 7.

## 2 Interest points and local appearance

The basic idea of local feature-based approaches is that concentrating on sparse sets of especially salient image points both saves computation time and improves robustness. This is particularly interesting for highly non-rigid objects, such as human bodies. We will use the interest point detector introduced by Harris and Stephens [7], which has been made invariant to scale changes [3] and affine transformations [11].

We use an extension to color images [6] of the scale-space gray-level Harris corner detector used by Dufournaud et al. [3]. This color version basically takes the sum of the autocorrelation matrices of the 3 color channels:

$$\mathbf{M}(\mathbf{x}, s_h, \sigma_i, \sigma_d) = s_h^2 G(s_h\sigma_i) \otimes \begin{pmatrix} (r_x^2+g_x^2+b_x^2)(\mathbf{x}, s_h\sigma_d) & (r_x r_y+g_x g_y+b_x b_y)(\mathbf{x}, s_h\sigma_d) \\ (r_x r_y+g_x g_y+b_x b_y)(\mathbf{x}, s_h\sigma_d) & (r_y^2+g_y^2+b_y^2)(\mathbf{x}, s_h\sigma_d) \end{pmatrix} \tag{1}$$

where $\mathbf{x} = (x,y)$, $c_i(\mathbf{x},\sigma) = G_i(\sigma) \otimes c(\mathbf{x})$ with $c \in \{r,g,b\}$ and $i \in \{x,y\}$. $G_i(\sigma)$ is a Gaussian-derivative kernel and $s_h$ allows to select the scale at which to extract the corner points, because we want to detect the same points on the object, whatever its size in the image. Furthermore we define the cornerness $K = \det(\mathbf{M}) - k \cdot \mathrm{trace}(\mathbf{M})^2$ with $k = 0.04$ as proposed initially by Harris and Stephens. In each frame, we place a rectangular region of interest (ROI) around our object in which we select the $N$ best points, i.e., the $N$ points corresponding to $N$ highest local maxima of the cornerness function.

The visual appearance of an image neighborhood can be described by a local Taylor series. The coefficients of this series constitute a feature vector that compactly represents the neighborhood appearance [8, 16]. This neighborhood characterization is called the *local jet* which up to order $n$ can be expressed for the point $\mathbf{x} = (x,y)$ as follows:

$$J^n(\mathbf{x}, s_h, \sigma_d) = \{c_{i_1 \ldots i_k}(\mathbf{x}, s_h\sigma_d) | k = 0, \ldots, n\} \tag{2}$$

where $c_{i_1 \ldots i_k}(\mathbf{x}, s_h\sigma_d)$ represents the Gaussian color channel derivative relative to the $i_1 \ldots i_k$ variables (with $i_j \in \{x,y\}$) and $s_h\sigma_d$ the size of the Gaussian smoothing applied during the derivative computation. Considering the local jet up to order 1, defining one local jet per color channel and including the interest point position, we obtain the 11-dimensional feature vector

$$\mathbf{v} = \begin{pmatrix} x & y & r & g & b & r_x & r_y & g_x & g_y & b_x & b_y \end{pmatrix}^T \tag{3}$$

where $\mathbf{v} \in \mathbf{V}$ with $\mathbf{V} \subset \mathbb{R}^{11}$. $\mathbf{V}$ is called the feature space. As we are using color images and as our point descriptors vary with rotation, the first-order local jet yields enough discriminative power to perform good point matching. A key problem in computing the local jet is to determine the scale $s_h\sigma_d$ at which to evaluate the image derivatives. The optimal scale of an interest point on an object depends on the global scale of the object in the image and of course on the interest point itself. In the present work, we ignore the second effect and define only a global scale for all points belonging to the object. This scale is the same as that used for computing the derivatives in the auto-correlation matrix of the corner detector.

# 3 Point Distribution Models

When using feature points to track non-rigid objects in video sequences, the most natural approach is to use PDMs, ASMs or AAMs, which have been introduced by Cootes and Taylor [1, 2]. They are statistical models of shape and/or appearance that have proven to be powerful tools for interpreting images, particularly when combined with algorithms for rapid matching of models to new images. The shape of an object can be interpreted as all geometric information that remains when location, scale and rotational effects are removed. There is a difference between ASMs that include only the positions of a set of points (or landmarks), and AAMs that add the texture of the object to the model. Most examples in the literature address medical imaging and face modelling, and most of the time, the landmarks are chosen to lie on the object borders.

We introduce here a somewhat new approach to modelling the objects. Our model can be considered as an AAM, but we don't use raw texture information to describe the appearance. We simplify the appearance of the object by constructing our model from a set of feature vectors which correspond to extracted points that do not necessarily lie on the object boundaries. Thus, each shape is represented by a vector

$$\mathbf{X} = \begin{pmatrix} \mathbf{v}_1^T & \mathbf{v}_2^T & \dots & \mathbf{v}_N^T \end{pmatrix}^T \tag{4}$$

that is simply a concatenation of the feature vectors at the interest points detected on the object in a given frame. As our feature vectors are 11-dimensional, the shape vectors lie in a $11N$-dimensional space, and more precisely on a low-dimensional manifold embedded in this high-dimensional space. The shape and the dimensionality of this manifold depend on the nature of the object deformations. For small deformations, we can approximate this manifold locally by a tangent hyperplane. Under a Gaussian assumption, this is equivalent to saying that the shape vectors form a low-dimensional Gaussian cloud in shape space.

Tracking is performed by matching the model points with a set of similar feature points extracted in each frame. This approach is robust to partial occlusions, because only a subset of the object points need to be visible in order to constrain the model parameters.

# 4 Constructing a Point Distribution Model

To construct the PDM of a given object, we use a set of shapes (interest point sets) that describe different instances of the object in question. First of all, the extracted shapes are aligned using a common approach known as *Procrustes Analysis* [5]. Shapes in the image reference frame are denoted by upper-case letters (e.g. $\mathbf{X}$), while the shapes in the model reference frame are denoted by lower-case letters (e.g. $\mathbf{x}$). Let's define the similarity transform $\mathbf{T}_{t_x,t_y,s,\alpha}$ specifying the position $(t_x, t_y)$, scale $s$ and orientation $\alpha$ of the model instance in the image. The translation and scaling are only applied to the $x$ and $y$ elements of the feature vectors, whereas the rotation must also be applied to the derivatives of the color channels. We determine a separate $\mathbf{T}$ for each extracted shape such that the sum of distances of each shape to the mean shape ($\sum_i |\mathbf{x}_i - \bar{\mathbf{x}}|^2$) in the model reference frame is minimized.

To align the shapes, interest points from different instances of the same object must be brought into correspondence. In many applications such as medical imaging, this *landmarking* is performed by a human operator, whereas in object tracking, it is possible

to exploit the temporal coherence between object instances in consecutive frames using a motion correspondence algorithm, as described in the next section.

As a second step, to reduce the dimensionality of the data, a *Principal Component Analysis* (PCA) is performed on the shapes. This is done by computing the eigenvalues $\lambda_i$ and the eigenvectors $\phi_i$ of the covariance matrix of the data $\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$ and retaining only the eigenvectors corresponding to the $t$ largest eigenvalues by choosing $t$ such that $\sum_{i=1}^{t}\lambda_i \geq f_v V_T$ where $f_v$ defines the proportion of the total variance one wishes to explain and $V_T$ is the sum of all the eigenvalues. We use $f_v = 0.90$. This ignores the small eigenvalues that in general correspond to noise and very small deformations. For each shape instance, we can generate the corresponding shape parameter vector

$$\mathbf{b} = \Phi^T(\mathbf{x} - \bar{\mathbf{x}}) = \Phi^T(\mathbf{T}^{-1}(\mathbf{X}) - \bar{\mathbf{x}}) \tag{5}$$

where $\Phi = (\phi_1|\phi_2|...|\phi_t)$. Similarly, plausible shapes can be generated with

$$\hat{\mathbf{X}} = \mathbf{T}(\hat{\mathbf{x}}) = \mathbf{T}(\bar{\mathbf{x}} + \Phi\mathbf{b}) \tag{6}$$

Limits of $\pm 3\sqrt{\lambda_i}$ are usually applied to the parameters $b_i$ to avoid unlikely shapes.

In general, it is impossible to track all the feature points through all the frames of the training sequence. What's more, the correspondence algorithm described in section 5 is likely to create small gaps in the tracks due to disappearing points. This is a problem because PCA cannot handle this missing data. The gaps created by the correspondence algorithm are simply filled in by linear interpolation. If the tracks don't cover the whole training sequence however, the missing track ends must be estimated by some other method. The simplest method would be to replace the missing values by the mean of the data, but this underestimates the variance of the data. Other possibilites include maximum likelihood estimation. We use the iterated PCA algorithm proposed by Rogers and Graham [15].

# 5 Tracking interest points

The output of a motion correspondence algorithm is a set of tracks, where each track ideally corresponds to a unique point on the object, specifying its position in every frame from entry to exit in the scene. Most methods first define a motion model and then use some optimization technique to maximize (minimize) a gain (cost) function based on that motion model. The methods differ in the choice of their motion model, their optimization technique, and/or their gain function, while their common property is that they use only two frames for establishing the correspondences.

Better performance is achieved by using multiple frames to establish the correspondences. Methods such as the Joint Probabilistic Data-Association Filter [4] and Multiple Hypothesis Tracking [14] suffer from several drawbacks, justifying our choice of a new algorithm proposed by Shafique and Shah [17], in which the $k$-frame correspondence problem is formulated as a graph theoretical problem.

At time $t$, we consider a window of the $k$ most recent frames. From all the detected points in these $k$ frames, a weighted directed graph $D$ is constructed, on which an optimization algorithm is applied, which maximizes the gain over the $k$ frames and is general enough to be used for a large variety of motion models and cost functions. It is able to handle all possible combinations of detection errors, occlusion or absence for a maximum

of $k-2$ frames. The optimization algorithm is equivalent to finding a maximally-weighted path cover of the directed graph $D$. This can easily be performed by transforming $D$ into a bipartite weighted graph [17], for which a maximum-gain matching can be determined using for example the *Hungarian method* [9], which is $O(n^3)$ with $n = kN$ in our case, assuming that $N$ points are detected in each frame. We use the gain function

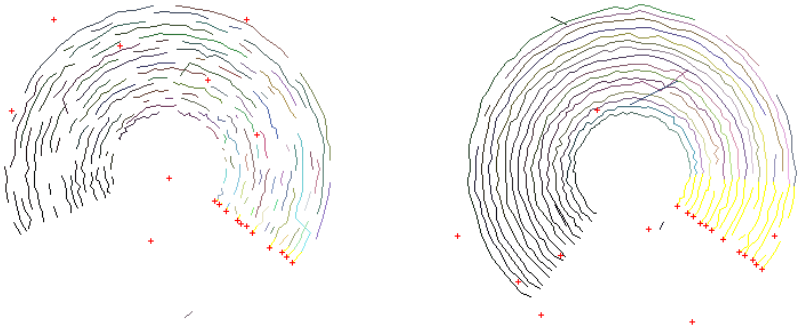
$$g(X_a^i, X_b^j) = 1 - \frac{d_M(\mathbf{v}_a^i, \mathbf{v}_b^j)}{\theta}. \tag{7}$$

where $d_M(\mathbf{v}_a^i, \mathbf{v}_b^j)$ is the distance between feature vectors $\mathbf{v}_a^i$ and $\mathbf{v}_b^j$. All edges with negative weights are ignored, meaning that matchings with distances above $\theta$ are impossible. In this way, $\theta$ acts as a threshold for the distance $d_M$. Furthermore in our formulation we do not use a motion model for the points.

When comparing $\nu$-dimensional data vectors $\mathbf{v}_i$ (our 11-dimensional feature vectors) drawn from a Gaussian distribution, the most natural metric is the Mahalanobis distance

$$d_M^2(\mathbf{v}_1, \mathbf{v}_2) = (\mathbf{v}_1 - \mathbf{v}_2)^T \mathbf{C}^{-1} (\mathbf{v}_1 - \mathbf{v}_2) \tag{8}$$

where $\mathbf{v}_1, \mathbf{v}_2 \in \mathbf{V}$. A frequent situation is when $\mathbf{v}_2$ is equal to the cluster mean $\overline{\mathbf{v}}$. The most critical part in the Mahalanobis distance is the computation of the covariance matrix $\mathbf{C}$. An analytical expression is not always available which is why it must often be estimated from the available data. The covariance matrix takes into account magnitude differences, possible correlations between the different vector components and of course the noise affecting them. The quality of this estimate strongly influences the results. There are different approaches to estimation that can mainly be classified into two categories:



Window size 2, corresponding to the classical situation of the two-frame correspondence problem.

Window size 9. It is possible now to overcome gaps in the tracks (up to a size of 7 frames in this example), yielding much longer tracks.

Figure 1: Synthetic example for the feature point tracking algorithm for different window sizes. 16 points rotate clockwise about the image center. The points' local appearance is randomly chosen (between 100 and 150 for the rgb values and between –10 and 10 for the derivatives) and a strong Gaussian noise is added (standard deviation of 5.0 for the rgb values and of 1.0 for the positions and the rgb derivatives). Additionally, the points may disappear in each frame with a probability of 0.2.

1. Define one unique $\mathbf{C}$ and estimate it from all the available data. This is equivalent to saying that all the feature vectors are drawn from one unique Gaussian distribution. This simple solution generates non-discriminative weights and gives only a rough model of noise. This is the approach we use for learning the model.

2. A more sophisticated way is to create a distinct covariance matrix $\mathbf{C}_i$ for each point. This is equivalent to saying that $k$ different interest points generate $k$ Gaussian clusters in feature space. In this case the covariance matrices can either be computed analytically using an appropriate noise model, or each one can be estimated from the feature vectors belonging to the interest point in question. We use this method during the model-based tracking, described in the next section.

Under a Gaussian assumption, the Mahalanobis distances are drawn from a $\chi^2_\nu$ distribution, where $\nu$ is the dimensionality of the feature space. As our feature vectors are 11-dimensional, for the rest of this paper we will be using $\chi^2_{11}$. Outliers are rejected by fixing the threshold $\theta$ on $d_M$.

Figure 1 demonstrates the performance of the point tracking algorithm.

# 6 Tracking with Point Distribution Models

Once we have learned the model, we can use it to perform tracking according to the following algorithm which is applied to each frame:

1. In the current frame extract interest points that lie inside the ROI from the previous frame.

2. Initialize the pose and the shape parameters with the final values obtained at the previous frame.

3. Match every model point with the best nearby interest point in the image. As in Section 5, we use the Mahalanobis distance and the Hungarian method to perform the matchings. Contrary to the training phase, each model point is represented by a Gaussian cluster, each having its own covariance matrix $\mathbf{C}_i$ as explained in Section 5. The $\mathbf{C}_i$ are simply extracted from the global covariance matrix $\mathbf{S}$.

4. Update the model parameters $(t_x, t_y, s, \alpha, \mathbf{b})$ such as to minimize $|\mathbf{Y}_\bullet - \mathbf{T}(\bar{\mathbf{x}}_\bullet + \Phi_\bullet \mathbf{b})|^2$ where $\mathbf{Y}_\bullet$ is a vector combining only the currently matched image feature vectors and where $\bar{\mathbf{x}}_\bullet$ and $\Phi_\bullet$ contain only the rows of $\bar{\mathbf{x}}$ and $\Phi$ corresponding to the matched model points.

In the first frame, the tracking algorithm is initialized manually or with pose and shape parameters obtained in the last frame of the training sequence. Altough we use a least-squares approach, if the model is well initialized, the algorithm is robust to outliers, because outliers are not likely to be matched and the minimization is only performed on the correctly matched points. As no filter is used, the best predictions of the ROI position and the model parameters are those from the previous frame. Note that during tracking, the matchings are not performed between different frames, but between the current frame and the model.

# 7 Experimental Results

We demonstrate the performance of our tracking system on a challenging corner sequence of a soccer game. The model construction of a soccer player from this sequence is illustrated in Fig. 2. Image (a) shows the initial frame in which the region of interest is selected manually. Image (b) shows point tracks after a training sequence of 40 frames. Section 5 explained how these tracks are created. The length of the training sequence is fixed a priori. Twenty tracks of lengths between 25 and 33 frames are retained to construct the model. Shorter tracks are discarded. During the training sequence, the player is tracked automatically by centering the region of interest on the center of gravity of the matched points. Image (c) displays the positions of the points of the aligned shapes and images (d) to (f) show the first mode of variation of a total of 10 modes retained for this particular example. On each point is superimposed a small patch indicating its local appearance.

The sequence illustrating the tracking has a length of 400 frames. During the whole sequence, the camera is almost static, but the tracked player undergoes various occlusions which become very severe after frame 220. Our tracker is able to recover from each of those occlusions, because if the current estimate of the model is not too far from the real points, the model is attracted by them. It is important to note that our tracker continues to follow the right player in situations where methods based on blobs, histograms or contours would be likely to fail.

Figure 3 shows 10 key frames of the sequence. Towards the end of the sequence, the tracker becomes unstable, meaning that it doesn't match the target very well. The first reason for this is that the tracked player performs an out-of-plane rotation (not learned by our model) and the second reason is that the occlusions become too severe and the movements very fast and chaotic. The first problem can be solved by replacing the 2-dimensional model by a 3-dimensional one whereas the second one could be addressed by applying an appropriate filter on the models pose and shape parameters.

From Fig.4(a), displaying the temporal evolution of the player position, it can be stated that the extracted trajectories are very smooth, and this without any filtering of the parameters. This is because our model-based tracker has a self-stabilizing effect, due to the simultaneous tracking of many different points on the object. The model scale is shown on Fig.4(b). Its mean value decreases slighty towards the end of the sequence, because the player walks away from the camera. The scale increases strongly during the

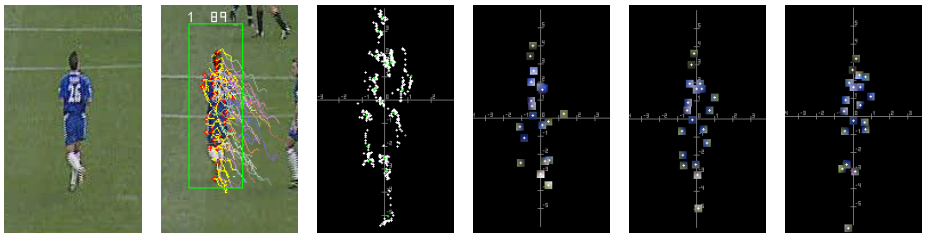


Figure 2: Creating the model (from left to right): (a) the initial frame, (b) the last frame of the training sequence with the point tracks, (c) the aligned shapes, (d)–(f) the first mode of variation where a small patch indicates the local appearance of each point.

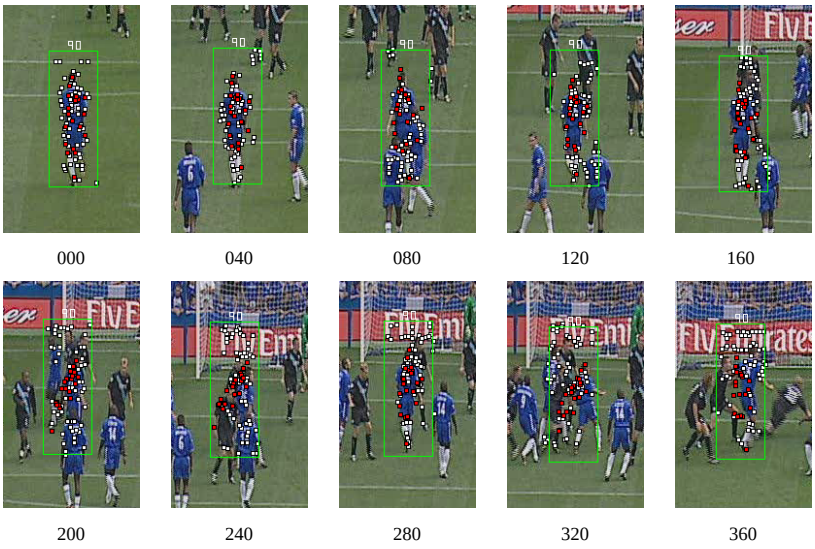



Figure 3: Tracking a soccer player. The images correspond from left to right and from top to bottom to every 40th frame of a 400 frame sequence. In each image, the red patches indicate the position of the model points, the white patches indicate the currently extracted points and the green rectangle represents the current region of interest.

partial occlusion around frame number 100, because some model points are matched with the occluding players, as they are very similar both in pose and appearance. This effect is generally not seen if the occluding objects are different. Tests on other sequences have shown that the object scale can be correctly estimated even if it is partially occluded. Currently the scale of the corner detector $s_h$ is not linked to the model scale $s$. Doing so would probably further improve the tracking results. Figure 4(c) demonstrates that $\alpha$ is close to 0 all over the sequence, except during the full occlusions. In particular cases, such as walking or running people, the tracking performance can be improved by constraining $\alpha$ to a value close to 0. From Fig. 4(d) it can be seen that the incorrect values of $s$ and $\alpha$ are correlated with low numbers of matches. As the model has 4 pose parameters and $t$ shape parameters, yielding a total of $t+4$ degrees of freedom, we require at least $(t+4)/2$ matches (7 in our case) in order to constrain the model.

Figure 5 contains another tracking example taken from the first 2001 PETS sequence. It demonstrates that the tracker can be used for any kinds of objects, such as cars for example. During the occlusion by the car, the occluded points of the person are correctly predicted by the model.

Our implementation of the tracking algorithm reaches a processing speed of 6 to 8 frames/second on a standard 1.7 GHz computer, depending on the size of the region of interest. Optimizing the code should allow for real-time performance.

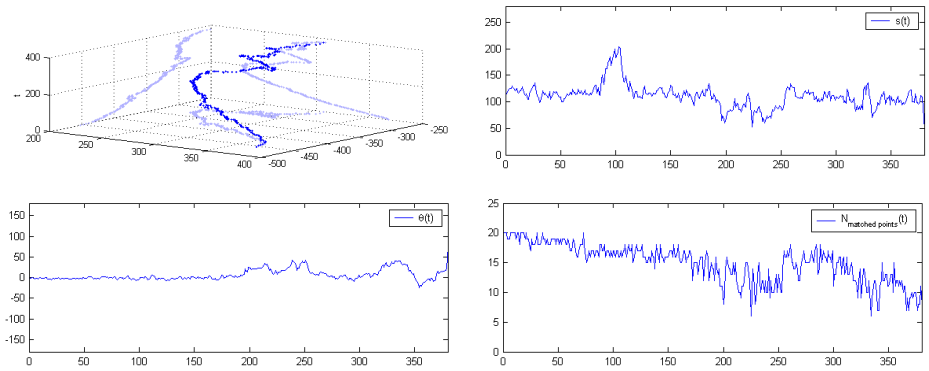



Figure 4: Evolution of the different parameters over the whole sequence. Top left: Player position $(x, y, t)$. Top right: Scale $s(t)$. Bottom left: Angle $\alpha(t)$. Bottom right: Number of matched model points.

# 8 Conclusions and future work

We have introduced a new tracking approach based on a model that lies somewhere between ASMs and AAMs, because it contains only local appearance information. The proposed approach is particularly robust to partial occlusions and insensitive to camera motions and scale changes. No background modelling is required. The model is learned automatically by tracking individual interest points during the training phase. User interaction is only required at initialization. Our method suffers from some limitations that we are trying to overcome in our future research:

- Our current system uses only a 2-dimensional model of the objects. For that reason it is unable to cope with objects turning around (rotation axis parallel to the image plane). We are currently extending our model to the general 3D case.
- At the moment, no filtering is used to predict the pose parameters. Our research focuses on filtering combined with matrix perturbation theory to correctly model the process and measurement covariances. Filtering will make the model stable enough so that we can link the corner detector scale to the model scale, thus eliminating the constraint of a roughly constant target size.

# References

[1] T. F. Cootes and C. J. Taylor. Statistical models of appearance for computer vision. Technical report, University of Manchester, 2001.

[2] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models – their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[3] Y. Dufournaud, C. Schmid, and R. Horaud. Matching images with different resolutions. In *Proc. of the Conf. on Computer Vision and Pattern Recognition*, pages 612–618, Juin 2000.

[4] T. E. Fortmann, Y. Bar-Shalom, and M.Sheffe. Sonar tracking of multiple targets using joint probabilistic data assocation. *IEEE J. Oceanic Eng.*, 8(3):173–184, 1983.

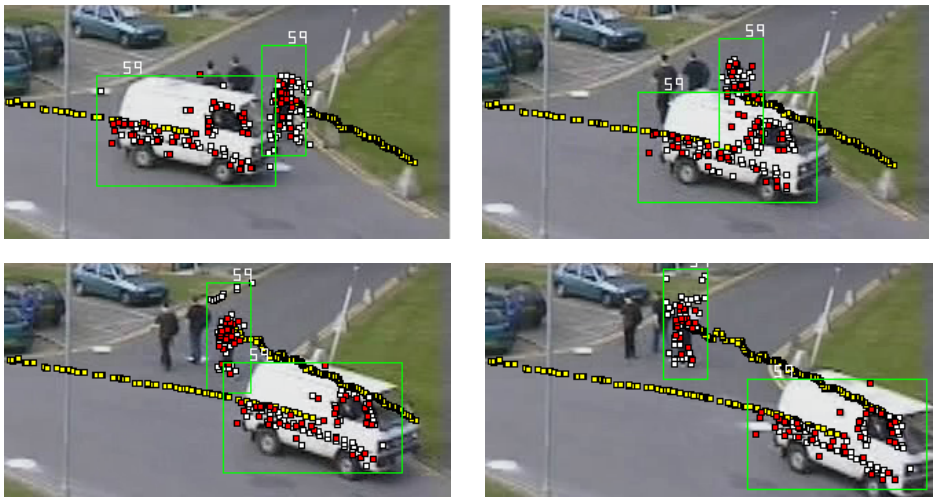


Figure 5: Tracking of a car and of a person partially occluded by that car.


[5] C. R. Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society*, 53:285–339, 1991.

[6] V. Gouet and N. Boujemaa. About optimal use of color points of interest for content-based image retrieval. Technical report, INRIA Rocquencourt, 2002.

[7] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[8] J.J. Koenderink and A.J. van Doorn. Representation of local geometry in the visual system. *Biological Cybernetics*, 55(6):367–375, 1987.

[9] H. W. Kuhn. The Hungarian method for solving the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[10] R. Mégret and J-M. Jolion. Tracking scale-space blobs for video description. *IEEE MultiMedia*, 9(2):34–43, 2002.

[11] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *European Conference on Computer Vision*, pages 128–142. Springer, 2002.

[12] K. Nummiaro, E. Koller-Meier, and L. J. Van Gool. An adaptive color-based particle filter. *Image Vision Comput.*, 21(1):99–110, 2003.

[13] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV (1)*, pages 28–39, 2004.

[14] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Trans. Automatic Control*, 24(6):843–854, 1979.

[15] M. Rogers and J. Graham. Structured point distribution models: Model intermittently present features. In *BMVC*, 2001.

[16] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. on Pattern Analysis & Machine Intelligence*, 19(5):530–534, 1997.

[17] K. Shafique and M. Shah. A noniterative greedy algorithm for multiframe point correspondence. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(1), January 2005.

[18] P. Tissainayagam and D. Suter. Object tracking in image sequences using point features. *Pattern Recognition*, 38(1):105–113, 2005.