Robust Non-Rigid Object Tracking Using Point Distribution Manifolds

Tom Mathes and Justus H. Piater

Department of Electrical Engineering and Computer Science
Montefiore Institute, University of Liège
Building B28, B-4000 Liège, Belgium
mathes@montefiore.ulg.ac.be, justus.piater@ulg.ac.be

Abstract. We present an approach to non-rigid object tracking designed to handle textured objects in crowded scenes captured by non-static cameras. For this purpose, groups of low-level features are combined into a model describing both the shape and the appearance of the object. This results in remarkable robustness to severe partial occlusions, since overlapping objects are unlikely to be indistinguishable in appearance, configuration and velocity all at the same time. The model is learnt incrementally and adapts to varying illumination conditions and target shape and appearance, and is thus applicable to any kind of object. Results on real-world sequences demonstrate the performance of the proposed tracker. The algorithm is implemented with the aim of achieving near real-time performance.

1 Introduction

Typical object tracking applications include video surveillance for security or behaviour analysis, traffic-monitoring, sports analysis and human body tracking. In this work we develop a model-based technique able to cope with non-rigid objects in crowded scenes, involving many interacting targets with frequent mutual occlusions. We use single-view video streams taken by non-static cameras, which poses serious difficulties to tracking systems based on background models.

Many tracking approaches are based on more or less elaborate variants of background subtraction [1]. They can easily handle only static cameras, and object labels cannot be preserved throughout occlusions, except by using high-level scene interpretation algorithms. Most model-based object tracking methods use a fixed object representation, a so-called *template*, that describes the appearance or the shape of the tracked object. Most of these are based on colour histograms [2,3]. Such approaches tend to have problems with richly textured objects or multiple interacting objects having similar global appearance. Few convincing attempts have been made that track objects using feature points, although it is generally accepted that point-based methods should have some interesting properties. Some basic point-based solutions were developed by Arnaud and Mémin [4] by combining a Rao-Blackwellized particle filter with a model consisting of a noisy, planar cloud of points, and by Bevilacqua et al. [5] who perform smart point grouping based on self-organising maps. More sophisticated approaches include the work by Leordeanu and Collins [6] where feature pairs are coupled based

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

on their pairwise statistics, by Tang and Tao [7] where objects are modelled adaptively with an attributed relational graph, and by Mathes and Piater [8], where point distribution models are learnt non-incrementally for tracking planar objects.

In our approach, each tracked object is described by a point distribution model [9] using feature vectors for local appearance instead of raw texture information. Such a model combines local appearance information with global shape information. The model is learnt incrementally and continuously, enabling it to accommodate to appearance and illumination changes. Point features tend to flicker in noisy image sequences or disappear due to occlusions, but as long as a reasonable subset of all the model points is visible in each frame, tracking can be performed reliably. The model can dynamically add good new features or remove bad old features. During occlusion by other tracked objects, model updating is disabled, rendering our tracker even more robust. Point landmarking is performed automatically, so that user interaction is only required to initialise the tracker in the first frame.

The following section explains how we extract interest points and how we describe their local appearance. Section 3 introduces the concept of point distribution manifolds, and Section 4 explains how they can be used for tracking purposes. Experimental results are given in Section 5.

2 Interest Points and Local Appearance

We concentrate on sparse sets of local features because they are well-suited for non-rigid objects and tend to yield methods particularly robust to partial occlusions. Local features are extracted by a colour version [10] of a scale-space, grey-scale Harris corner detector [11]. This is illustrated in Fig. 1. On each detected interest point we describe the local appearance by the 11-dimensional feature vector

$$\mathbf{v} = (x, y, r, g, b, r_x, r_y, g_x, g_y, b_x, b_y)^T, \tag{1}$$

where $\mathbf{v} \in \mathbf{V}$ with $\mathbf{V} \subset \mathbb{R}^{11}$. $\mathbf{V}$ is called the *feature space*. $\mathbf{v}$ corresponds to the first-order local jet enhanced by the interest point position. Using colour images and a rotationally variant descriptor yields enough discriminative power to obtain reliable point matchings between frames.

3 Point Distribution Manifolds

When using interest points to track objects, a natural approach is to use point distribution models, which are statistical models of shape and/or appearance. The shape of an object can be interpreted as all geometric information that remains when location, scale and rotational effects have been removed. Instead of using raw texture information, we describe the shape and appearance by constructing our model from a set of feature vectors that correspond to interest points that may lie anywhere on the object. Thus, each shape is represented by a vector

$$\mathbf{x} = \left({\mathbf{v}_x^1}^T, {\mathbf{v}_x^2}^T, \ldots, {\mathbf{v}_x^N}^T \right)^T \tag{2}$$

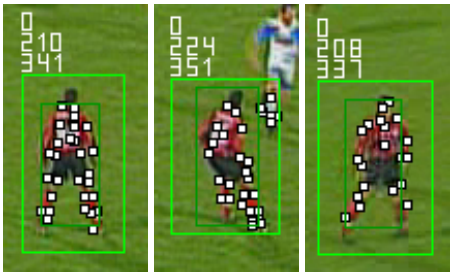

Fig. 1. Soccer player performing out-of-plane rotation (frames 0, 28 and 48).

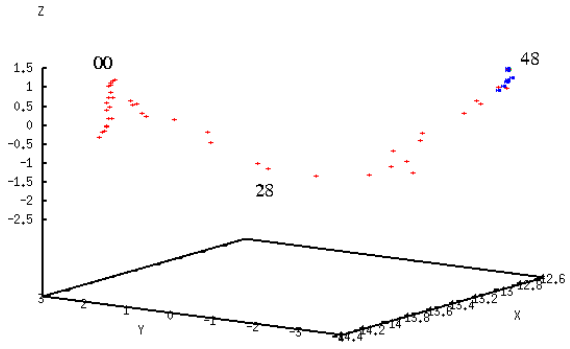

Fig. 2. Projection onto the three principal components of a non-linear low-dimensional manifold corresponding to the normalised shapes of the rotating soccer player of Fig. 1.

that is simply a concatenation of all the feature vectors extracted from the object in a given frame. These shape vectors lie in an $11N$-dimensional space, and more precisely on a low-dimensional, non-linear manifold $\mathcal{M}$ embedded in this high-dimensional space, because the interest points on a real non-rigid object are strongly correlated. The shape and dimensionality of $\mathcal{M}$ depend on the nature of the object deformations. Figure 2 shows the projection onto the three principal components of a typical manifold obtained for a rotating soccer player. It illustrates the potentially non-linear nature of the manifold.

3.1 Matching Interest Points

If $\mathcal{M}$ is sampled densely enough and if we assume that it is locally linear, new shapes can be generated by linear interpolation of neighbouring shapes. Shapes within the image are denoted by the letter $\mathbf{X}$, whereas shapes that are part of the model are denoted by the letter $\mathbf{Y}$. Let us suppose we have used the model to generate a shape $\mathbf{Y}$ superimposed onto the current video frame. In order to test if the current set of points $\tilde{\mathbf{X}}$ taken from the image is a valid shape, the points from the image, indexed by i, and the points from the model, indexed by j, have to be brought into correspondence. To do so, we compute a maximum-gain matching by using the *Hungarian method* [12]. We use the

gain function

$$g(\mathbf{v}^i_{\tilde{X}}, \mathbf{v}^j_Y) = 1 - \frac{d(\mathbf{v}^i_{\tilde{X}}, \mathbf{v}^j_Y)}{\theta}, \tag{3}$$

where $d(\mathbf{v}^i_{\tilde{X}}, \mathbf{v}^j_Y)$ is the distance between feature vectors $\mathbf{v}^i_{\tilde{X}}$ and $\mathbf{v}^j_Y$. All edges with negative weights are ignored, meaning that matchings with distances greater than θ are impossible. In this way, θ acts as a gating threshold for d. The squared distance d^2 between two N-sized vectors is computed as:

$$d^2(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \mathbf{y}), \tag{4}$$

where $\mathbf{\Sigma}$ is the covariance matrix estimated from all interest points belonging to $\mathcal{M}$. For computational reasons, the cross-correlations in $\mathbf{\Sigma}$ are assumed to be equal to 0, in order to avoid costly matrix inversions. Once the matchings have been performed, the interest points of $\tilde{\mathbf{X}}$ can be rearranged so that they are in the same order as their correspondences in $\mathbf{Y}$. This new vector will be denoted by $\mathbf{X}$. In general, not all the points from $\tilde{\mathbf{X}}$ (resp. $\mathbf{Y}$) have a correspondence in $\mathbf{Y}$ (resp. $\tilde{\mathbf{X}}$). For this reason, we will have two vectors of equal size with missing elements, denoted by $\mathbf{X}^\bullet$ and $\mathbf{Y}^\bullet$.

3.2 Image-to-model and model-to-image similarity

Before adding a shape to the manifold $\mathcal{M}$, it is centred to the origin and scaled such that the mean distance of its points to the origin is equal to 1. This operation defines a similarity transform $\mathbf{T} = \mathbf{T}_{t_x, t_y, s, \alpha}$ that maps a shape from the manifold to the image reference frame. The inverse transform $\mathbf{T}^{-1}$ can be used to map it back to the manifold. 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 colour channels. 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. $\mathbf{x}$). Thus we have $\mathbf{x}^\bullet = \mathbf{T}^{-1}(\mathbf{X}^\bullet)$.

3.3 Computing the Weights

The model is used to reconstruct a shape as similar as possible to the current, rearranged and reprojected image shape $\mathbf{x}^\bullet$. Our approach is similar to one popularly used for locally linear embedding [13]. We begin by identifying the K nearest neighbours $\mathbf{y}_i$ of $\mathbf{x}^\bullet$ on $\mathcal{M}$ by applying the distance d defined earlier on the non-missing elements of $\mathbf{x}^\bullet$. Reconstruction errors are measured by the cost function

$$\epsilon(w) = \left| \mathbf{x}^\bullet - \sum_{i=1}^{K} w_i \mathbf{y}_i^\bullet \right|^2, \tag{5}$$

which is the squared distance between the image shape and its reconstruction and where again only the non-missing dimensions are considered. The weights w_i summarise the contribution of the ith model shape to the reconstruction. They are computed by minimising the cost function subject to two constraints: First, the image shape is reconstructed only from its neighbours, enforcing $w_i = 0$ if $\mathbf{y}_i$ does not belong to this set;

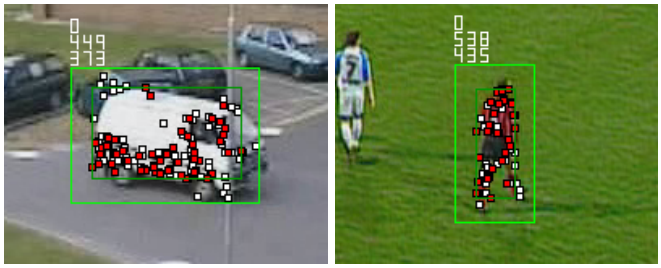

Fig. 3. A car and a soccer player with their current interest points (in white) and model points (in red) superimposed.

second, the weights must sum to one: $\sum_i w_i = 1$, thus enforcing the invariance to translation in the manifold space. The optimal weights subject to these constraints are found by solving the linear system of equations

$$\sum_j c_{ij} w_j = 1, \forall i, \tag{6}$$

where c_{ij} are the elements of the local covariance matrix defined by

$$c_{ij} = (\mathbf{x}^\bullet - \mathbf{y}_i^\bullet)^T (\mathbf{x}^\bullet - \mathbf{y}_j^\bullet). \tag{7}$$

For details on these calculations, see Appendix A of Saul and Roweis [13]. The weights are constrained to be larger than a small negative threshold to generate only shapes close to the convex hull of the K neighbouring shapes. We take $K = 13$ in our experiments.

3.4 Shape Generation

If we assume that the manifold is locally linear, then we can predict the position of missing points. We compute the K nearest neighbours and the corresponding weights of the current image shape using only the dimensions corresponding to the model points that could be matched to image points, as described in the previous section. These same weights can then be used to predict the missing feature vectors, generating a complete vector $\mathbf{x}$ equal to $\mathbf{x}^\bullet$, but with the missing values filled in by the corresponding values of $\sum w_i \mathbf{y}_i$. This is a very important step in our method because it solves the problem of flickering feature points. The generated shape $\mathbf{y} = \mathbf{x}$ is then added to the manifold.

Due to opaque objects rotating in depth or object deformations, some feature points become hidden because they move behind the object. When generating shapes from the model, such hidden points should not be projected into the image. Therefore, for every feature point of every shape on the manifold there is a flag that indicates whether that point was visible at the moment the shape was added to the manifold. When we generate a new shape, a point is taken to be visible if among the K neighbouring shapes at least one flag is set. Figure 3 shows a car and a soccer player with the image interest points (in white) and the model points (in red) superimposed.

4 Tracking

4.1 Cleaning the Manifold

To make the model adaptive to changing object appearance or shape, we need a mechanism to add and remove points from it. Points appear and disappear usually when an object performs out-of-plane rotations or undergoes strong non-rigid deformations.

We call *new* points all those feature points in the current frame that cannot be matched to a model point. Each of these points is added to the model and matched to image points from frame to frame, but is not yet used to update the model parameters. This happens only when the point has proven to be stable, meaning that it has been matched a minimum number of times. We call these the *active* points. A similar methodology is applied to *inactive* points, which are model points that have not been matched for some time.

As we are interested in tracking objects in crowded scenes, our model was designed to be very robust to occlusions. Nevertheless, due to its incremental nature, bad points (points not belonging to the object) could be added to the model, especially if the background is very cluttered or the occluding object has similar velocity and/or appearance. The addition and deletion of points are therefore disabled as soon as the regions of interest of the two objects intersect each other.

In some situations (cluttered background or occluding object not being tracked), it still happens that a bad point is likely to be added to the model. We therefore discard points that lie too far from the 4-dimensional Gaussian cluster $\mathcal{N}$ formed by the 4-D points $\mathbf{q} = (x, y, v_x, v_y)$, where (v_x, v_y) is the velocity vector of (x, y). This means that we consider as outliers those points for which $d^2(\mathbf{q}, \mathcal{N}) > \gamma^2$ and

$$d^2(\mathbf{q}, \mathcal{N}) = (\mathbf{q} - \mu)^T \Sigma_{\mathcal{N}}^{-1} (\mathbf{q} - \mu), \tag{8}$$

where μ is the mean vector and $\Sigma_{\mathcal{N}}$ is the covariance matrix of $\mathcal{N}$. We use $\gamma = 3.0$. This is analogous to the gating commonly used with Kalman filtering.

For computational reasons it is not possible to add new shapes indefinitely. We therefore generally limit the size of the manifold to a maximum of 30 shapes. When this limit is reached, the oldest shape is simply discarded. Keeping more than 30 shapes on the manifold doesn't improve the tracking results considerably.

4.2 Kalman Filtering

A Kalman filter is applied to the model-to-image similarity parameters. In our state vector $\mathbf{p} = (t_x, t_y, s, \alpha, v_x, v_y) \in \mathbb{R}^6$, the position is governed by a first-order process (constant-velocity model), whereas s and α are governed by a zeroth-order process, giving $\mathbf{p}_t = \mathbf{A}\mathbf{p}_{t-1} + \mathbf{u}_{t-1}$, where $\mathbf{A}$ is the state transition matrix. The corresponding measurement vector $\mathbf{z} = (t_x, t_y, s, \alpha) \in \mathbb{R}^4$ is provided by $\mathbf{z}_t = \mathbf{H}\mathbf{p}_t + \mathbf{v}_t$, where $\mathbf{H}$ is the measurement matrix. The random vectors $\mathbf{u}_t$ and $\mathbf{v}_t$ represent the process and measurement noise at time t respectively. They are assumed to be independent of each other, white and with normal probability distributions. In soccer or video surveillance the filter is tuned in order to allow only slow variations of scale and angle. In sequences with more chaotic movements, the scale and the angle can be made more flexible.

4.3 The Algorithm

Tracking is performed by applying the following algorithm to each frame:

1. In the current frame (time t), extract interest points that lie inside the ROI from the previous frame, giving the current shape $\tilde{\mathbf{X}}_t$. The ROI is equal to the smallest rectangle enclosing the model shape from the previous frame plus a small border.
2. Project the model shape from the previous frame into the current frame by using the predicted similarity resulting from the previous time update of the Kalman filter: $\mathbf{Y}_{t-1} = \hat{\mathbf{T}}_{t-1}(\mathbf{y}_{t-1})$.
3. Match the points of $\mathbf{Y}_{t-1}$ with the points of $\tilde{\mathbf{X}}_t$. This defines a vector $\tilde{\mathbf{X}}_t^\bullet$ and a vector $\mathbf{Y}_{t-1}^\bullet$ containing only the matched (and active) points of $\tilde{\mathbf{X}}_t$ and $\mathbf{Y}_{t-1}$. Let $\mathbf{X}_t^\bullet$ be the rearranged version of $\tilde{\mathbf{X}}_t^\bullet$.
4. Compute the new similarity $\mathbf{T}_t$ that minimises $|\mathbf{X}_t^\bullet - \mathbf{T}_t(\mathbf{y}_{t-1}^\bullet)|^2$.
5. Compute the K nearest neighbours of $\mathbf{x}_t^\bullet = \mathbf{T}_t^{-1}(\mathbf{X}_t^\bullet)$ on $\mathcal{M}$ and the corresponding weight vector $\mathbf{w}_t$.
6. Use these weights $\mathbf{w}_t$ to complete $\mathbf{x}_t^\bullet$ in order to generate what is the most probable current image shape $\mathbf{x}_t$.
7. Add this completed shape $\mathbf{y}_t = \mathbf{x}_t$ to the manifold.
8. Use the computed similarity $\mathbf{T}_t$ as measurement for the Kalman filter and do a time update which gives $\hat{\mathbf{T}}_t$.
9. Clean the manifold as described in Section 4.1.

Our method can be directly applied to sequences that do not contain many background feature points. If the background is highly cluttered, meaning that it gives rise to large numbers of feature points, a pre-filtering stage may be required, e.g. to remove all static points in the case of a static camera or to remove all the image-to-model homography inliers in the case of a moving camera. This approach is different from traditional background subtraction, because it is performed only locally and does not require a background model.

5 Results

We present tracking results on several challenging video sequences taken from a soccer game and from the PETS 2001 video surveillance data. Objects can be correctly tracked through scale, appearance and shape changes, as long as they exhibit sufficient texture. The tracker is not specific to people, but can also be used to track cars for example. Object labels are not lost during severe partial occlusions, even if the interacting targets look very similar. In all the examples, user interaction is only required in the first frame in order to initialise the targets to be tracked.

Example 1 is a difficult 150-frame sequence taken from a soccer game. The camera performs rotations and zooms whereas the players undergo drastic non-rigid deformations and very rapid movements, causing motion blur in some subsequences. In this sequence, four players are tracked. If their regions of interest intersect, their respective model learning is disabled, indicated by a red region of interest. The size of the regions of interest automatically adapts to the target size. The trackers are not disturbed by the

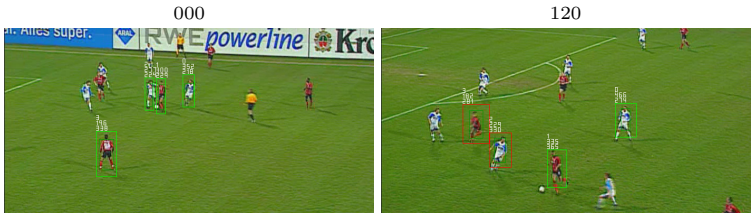

Fig. 4. Example 1: Global views of the sequence, illustrating the camera movement. The camera and the players perform fast movements, which causes some motion blur.

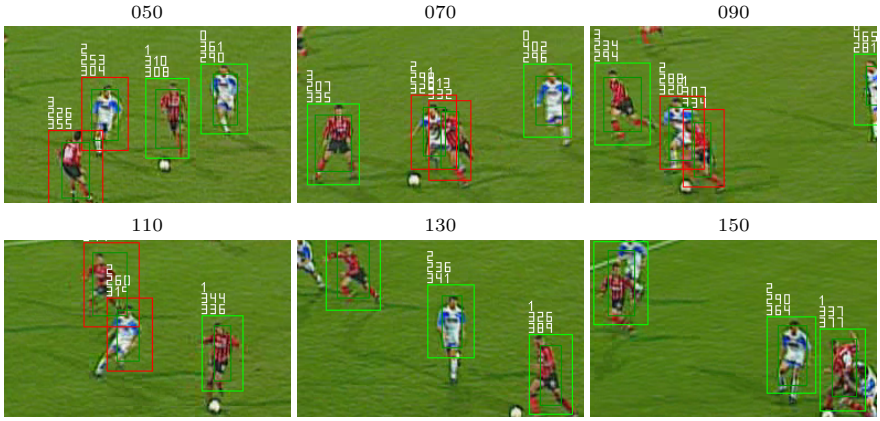

Fig. 5. Example 1: Local enlargements of interesting keyframes of the sequence.

partial occlusions. Figure 4 shows four keyframes of the sequence to illustrate the camera movements. Figure 5 contains six enlargements from another subset of keyframes of the same sequence.

Example 2, illustrated in Fig. 6, is similar to the previous one, but with much more severe occlusions. The occlusions in this example are typical occlusions our tracker is able to handle without confusing target labels. In this sequence, five soccer players and the referee are tracked. Between frames 060 and 110 there is a very complicated occlusion situation between the referee and 3 other players, which is correctly handled by the tracker.

Example 3 illustrates the ability of our tracker to handle appearance and scale changes. A person walking away from the camera is correctly tracked for more than 300 frames. Three keyframes of this sequence are shown in Fig. 7. After frame 290, the tracker fails, mainly for two reasons: First, the target becomes very small and no longer generates enough interest points; secondly, in its current form, our algorithm still has some problems with very slowly-moving targets in front of highly cluttered backgrounds. A possible solution to this problem might be to eliminate static interest points (required only locally, inside the region of interest) in the case of a static camera or the inter-frame homography inliers in the case of a rotating and zooming camera.

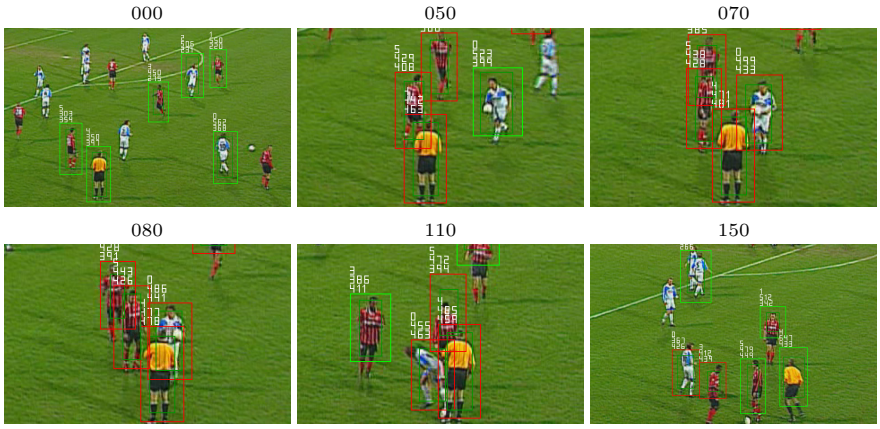

Fig. 6. Example 2: 160-frame soccer sequence with 6 tracked targets undergoing complex mutual occlusions. Each player is correctly tracked throughout the sequence without labels being lost. Frame 000 is the frame in which the targets are initialised.

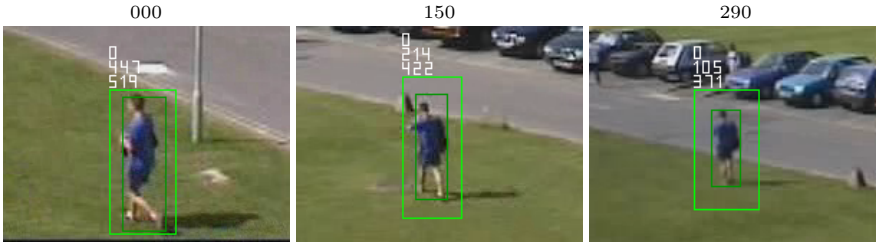

Fig. 7. Example 3: three representative frames from a 320-frame sequence taken from video surveillance. The tracked person undergoes strong appearance and scale changes.

All experiments were performed on a 1.7 GHz Celeron processor. Our current, non-optimised implementation runs at around 1.5 to 3.0 frames per second, depending on the image sizes (704×576 for the soccer sequences and 768×576 for the PETS sequences), and linearly in the number of tracked targets. The tracking itself is very fast; the current bottleneck of our implementation is the Harris detector, which can be sped up dramatically using efficient implementations. The speed of each tracker depends on the maximum number of shapes on the manifold and on the number of interest points per shape.

6 Conclusion

We presented a novel, robust approach to tracking non-rigid, textured objects in crowded scenes. An incremental model is learnt that combines groups of feature points. This allows us to handle highly non-rigid targets such as running people. Our method behaves

very well during partial occlusions in that target labels are generally preserved, and the objects' centres of gravity are correctly predicted. This latter property is essential for metric applications where the target position has to be mapped to the ground plane.

Our method is robust, because it takes into account the local appearance and the spatial configurations of feature points. It is highly unlikely that two targets look exactly the same, move at the same speed and are very close together. As the model is learnt automatically and incrementally, we can track any kind of object.

In contrast to histogram-based methods, our method works with any kind of object texture and can even handle objects that look similar to the background or to other tracked objects. Another advantage over background-subtraction methods is that we can easily work with non-static cameras. Our method does not necessarily work well with untextured objects, as it is based on feature points, although in many situations there are enough border points. Due to the incremental nature of our tracker, slowly-moving targets in front of cluttered backgrounds can also be lost. We will address this problem by efficient methods for removing background points.

References

1. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: Proceedings IEEE Conf. on Computer Vision and Pattern Recognition. (1999) 1
2. Nummiaro, K., Koller-Meier, E., Gool, L.J.V.: An adaptive color-based particle filter. Image Vision Comput. **21**(1) (2003) 99–110 1
3. Okuma, K., Taleghani, A., de Freitas, N., Little, J.J., Lowe, D.G.: A boosted particle filter: Multitarget detection and tracking. In: ECCV (1). (2004) 28–39 1
4. Arnaud, E., Mémin, E.: An efficient rao-blackwellized particle filter for object tracking. (2005) 1
5. Bevilacqua, A., Stefano, L.D., Vaccari, S.: Using local and global object's information to track vehicles in urban scenes. In: IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS). (2005) 1
6. Leordeanu, M., Collins, R.: Unsupervised learning of object features from video sequences. In: Proc. of CVPR. (2005) 1
7. Tang, F., Tao, H.: Object tracking with dynamic feature graph. In: Proc. of ICCV. (2005) 2
8. Mathes, T., Piater, J.: Robust non-rigid object tracking using point distribution models. In: Proc. of British Machine Vision Conference (BMVC'05). (2005) 2
9. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models – their training and application. Computer Vision and Image Understanding **61**(1) (1995) 38–59 2
10. Gouet, V., Boujemaa, N.: About optimal use of color points of interest for content-based image retrieval. Technical report, INRIA Rocquencourt (2002) 2
11. Dufournaud, Y., Schmid, C., Horaud, R.: Matching images with different resolutions. In: Proc. of the Conf. on Computer Vision and Pattern Recognition. (2000) 612–618 2
12. Kuhn, H.W.: The Hungarian method for solving the assignment problem. Naval Research Logistics Quarterly **2** (1955) 83–97 3
13. Saul, L.K., Roweis, S.T.: An introduction to locally linear embedding. (2001) 4, 5