Tracking by cluster analysis of image differences

Arthur E.C. Pece[⋆]

Department of Computer Science
University of Reading
Reading RG6 6AY
England

Abstract. The EM algorithm is used to track moving objects as clusters of pixels significantly different from the corresponding pixels in a reference image. Simple statistical principles are used for the initialisation, termination, merging and splitting of clusters. The system is competitive with other tracking algorithms based on image differencing.

Many image-based tracking systems include the four stages of image differencing, thresholding of the difference image, morphological filtering, and connected-component labelling. These stages are used to identify distinct targets and attribute each image pixel to one of the targets. Further processing stages use this information for detection of target features, Kalman filtering, *etc.* Several examples of this approach can be found in [3]. This approach can be quite effective, but thresholding and morphological operators involve information loss; this information loss is the result of assigning each pixel unambiguously either to the background or to one (and only one) target, ignoring the uncertainty of these assignments.

For most purposes, this attempt is unnecessary: the output that is required from a tracker is information on how many targets are present and the approximate location and size of the targets. This seems an appropriate task for cluster analysis. Tracking algorithms based on clustering have been proposed in [4, 8]. These algorithms generate clusters based on colour alone [4] or in combination with image location [8], in both cases without image differencing; as a consequence, these algorithms can operate with a moving camera. On the other hand, clustering applied to image differences should be faster and more reliable, when used with a fixed camera.

The algorithm presented in this paper was developed as a component of a model-based tracking system [7]: every time a new target appears, initial estimates of its location and size are provided by the centroid and covariance matrix of the corresponding cluster, projected onto the ground plane. It was soon discovered that the cluster-tracking algorithm is remarkably effective on its own, especially considering its conceptual simplicity.

This paper describes the theory behind the algorithm and presents results obtained with the PETS2000 image sequences [3]. Apart from image differencing, other innovations of our approach with respect to previous cluster-tracking algorithms [4, 8] are: (1) the algorithm is based on a probabilistic generating model for the clusters, which allows maximum-likelihood estimation of the cluster parameters by the EM algorithm; (2) most important, the number of clusters is not fixed *a priori*, but dynamically updated on the basis of evidence from the image: in this respect, the new method is similar to layers of support [1].

1 Description of the algorithm

The principle behind the tracking algorithm is simple: a moving target will produce a cluster of pixels in the difference image. The probabilities that a pixel belongs to the background cluster or to one of the target clusters can be estimated from the location of the pixel and the value of its grey-level difference with respect to the reference image. Cluster analysis can be used to improve rough initial estimates of cluster parameters.

Each pixel of the difference image is considered an observation, including a 2-vector of spatial coordinates $\mathbf{u}$ (used to index the pixel) and a scalar grey-level difference value $\delta(\mathbf{u}) = i(\mathbf{u}) - r(\mathbf{u})$, where $i(\mathbf{u})$ and $r(\mathbf{u})$ are the grey-level values of pixel $\mathbf{u}$ in the current image and the reference image, respectively.

⋆ Email: aecp@cs.rug.nl

Once the probabilities that pixels belong to clusters are specified, it is straightforward to apply the EM algorithm [2, 6]: the probabilities of pixels belonging to each cluster are estimated by using the current parameter estimates, and then the parameters are re-estimated by summing the evidence from all pixels for each cluster, weighted by the probabilities that the pixels belong to that cluster. The iterative procedure is guaranteed, under mild assumptions, to converge to a local maximum of the likelihood of the cluster parameters [6].

1.1 Statistical model

The model is a mixture of clusters, not all of them Gaussian. We assume that the current image model includes n target clusters plus the background cluster: the method used to determine n will be detailed below. The cluster parameters have a subscript indicating the index of the cluster: the background cluster has index 0 and the target clusters have indexes $j > 0$. The set of parameters for cluster j is indicated by Θ_j. The clusters generate a fixed total of m image pixels for each frame of the image sequence; the percentage of pixels generated from cluster j is w_j (one of the cluster parameters).

The probability[2] $f_j(\mathbf{u})$ that cluster j generates a pixel value at image location $\mathbf{u}$ is assumed to be a separable function of the image coordinates and the grey-level difference observed at that location:

$$f_j(\mathbf{u}) = g[\mathbf{u} \,|\, \Theta_j] \cdot h[\delta(\mathbf{u}) \,|\, \Theta_j] \tag{1}$$

Correlations between neighbouring pixels are neglected because they are assumed to arise from the clusters themselves, *i.e.* grey-level values are conditionally independent, given the number of clusters and the cluster parameters.

Background cluster: By treating the background as a cluster, it becomes unnecessary to threshold grey-level differences. The probabilistic model for the background is of course different from the models for the target clusters.

The probability $f_0(\mathbf{u})$ that the background cluster generates a pixel at location $\mathbf{u}$ is a function of the grey-level difference alone, since the background is present (even if occluded) at every location in the image. In other words, $g[\mathbf{u} \,|\, \Theta_0]$ is uniform over the image: $g[\mathbf{u} \,|\, \Theta_0] = 1/m$.

Clearly, the larger (in absolute value) the grey-level difference, the smaller the probability that the pixel belongs to the background. Also, it does not matter whether the grey-level difference is positive or negative: even though grey levels are bounded between 0 and 255, grey-level differences in the background are unlikely to reach the limits of this range. In order to reduce the influence of outliers, we assume a two-sided exponential distribution, rather than a Gaussian distribution, for the grey-level differences in the background cluster:

$$h[\delta(\mathbf{u}) \,|\, \Theta_0] = \frac{1}{2\mu_0} \exp(-|\delta(\mathbf{u})| \,/\mu_0) \tag{2}$$

where μ_0 is the mean (absolute value) grey-level difference for the pixels in the background cluster.

Target clusters: The target clusters can have any value of image difference, changing from one region of the cluster to the other and from one image frame to the next. For simplicity, we assume that $h[\delta(\mathbf{u}) \,|\, \Theta_j]$ is a uniform density: $h[\delta(\mathbf{u}) \,|\, \Theta_j] = 1/q$ for $j > 0$, where q is the number of observable grey-level differences (usually 256).

We assume that the probability is a Gaussian function of the distance of the pixel from the centroid of the cluster:

$$g[\mathbf{u} \,|\, \Theta_j] = \frac{1}{2\pi\sqrt{|\Sigma_j|}} \exp\left(-\frac{1}{2}\Delta\mathbf{u}_j^T \cdot \Sigma_j^{-1} \cdot \Delta\mathbf{u}_j\right) \tag{3}$$

where $\Delta\mathbf{u}_j = \mathbf{u} - \mathbf{c}_j$ is the vector distance of pixel $\mathbf{u}$ from the centroid $\mathbf{c}_j$ of cluster j, Σ_j is the covariance matrix of cluster j and $|\Sigma_j|$ is the determinant of Σ_j.

Clusters generated from visual targets of interest are usually not Gaussian, but better approximated as top-hat distributions of the form

$$g[\mathbf{u} \,|\, \Theta_j] = \begin{cases} \left(4\pi\sqrt{|\Sigma_j|}\right)^{-1} & \text{if} \quad \Delta\mathbf{u}_j^T \cdot \Sigma_j^{-1} \cdot \Delta\mathbf{u}_j < 4 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

[2] We use probabilities, instead of probability densities, because the images are discretised both in space and in grey-level values.

The "top-hat ellipsoid" defined by Eq. 4 is very useful both for display purposes (since it normally contains almost all the pixels likely to belong to the cluster) and for testing whether a cluster does indeed contain only one target (see Subsection 1.3). However, a top-hat distribution gives zero weight to pixels belonging to the target, but lying outside the estimated radius of the distribution: such pixels arise both because of errors of parameter estimates before convergence, and because of the irregular shapes of the targets. If the grey-level differences of these outlying pixels are significantly large, they will be captured by a smooth (Gaussian) cluster model (but not by a top-hat model) even if they are quite far from the centroid. Other reasons to use a Gaussian model are

- a Gaussian blur is an effective model of the spread of uncertainty for the cluster position from frame to frame;
- the objective function (log-likelihood) to be optimised is a smooth function of cluster parameters when the cluster is Gaussian.

The posterior probability $p_j(\mathbf{u})$ that an observed pixel at location $\mathbf{u}$ is generated from cluster j is given by Bayes' theorem:

$$p_j(\mathbf{u}) = \frac{w_j f_j(\mathbf{u})}{f(\mathbf{u})} \tag{5}$$

where w_j is the prior probability that cluster j generates a pixel, *i.e.* the fraction of image pixels generated from cluster j; and the prior probability of the pixel being generated is given by

$$f(\mathbf{u}) = \sum_{j=0}^{n} w_j f_j(\mathbf{u}) \tag{6}$$

Effectively, Bayes' theorem is used to normalise the sum of probabilities to unity for each pixel.

1.2 Cluster parameter estimation

All the probabilities defined above can be computed from the cluster parameters $\Theta_j = (w_j, \mu_j, \mathbf{c}_j, \Sigma_j)$. These parameters are estimated by maximising their log-likelihood: the log-likelihood $L(\underline{\Theta}|D)$ of the parameters is the logarithm of the probability of the difference image $D = \{\delta(\mathbf{u})\}$, given the cluster parameters $\underline{\Theta} = \{\Theta_j | 0 \leq j \leq n\}$:

$$\begin{aligned} L(\underline{\Theta}|D) &= \log \prod_{\mathbf{u}} f(\mathbf{u}) \\ &= \sum_{\mathbf{u}} \log \sum_{j} w_j f_j(\mathbf{u}) \end{aligned} \tag{7}$$

At each iteration, the parameters are re-estimated for each cluster by using the current estimates of the probabilities (see section 2.7.2 in [6]). For instance, the updated estimate of the average (absolute-value) grey-level difference μ_j for cluster j is computed as

$$\mu_j^{(k+1)} = \frac{\sum_{\mathbf{u}} |\delta(\mathbf{u})| \cdot p_j^{(k)}(\mathbf{u})}{\sum_{\mathbf{u}} p_j^{(k)}(\mathbf{u})} \tag{8}$$

where the superscripts indicate iteration number.

The EM algorithm converges to a local maximum of the likelihood of the model parameters, given the data. The convergence criterion is that the log-likelihood changes by less than $\epsilon = 10^{-5}$ from one iteration to the next. Since the target clusters are well separated in most frames, convergence usually requires less than 10 iterations.

The estimates for centroids and covariances at a given frame in an image sequence are taken as the initial estimates for the EM algorithm applied to the next frame. If a Kalman filter were applied, the centroids would be shifted depending on the current estimate of centroid velocities and the covariances would be increased. However, the tracking algorithm, as implemented, is meant to be general, without prior assumptions about the physical properties of the targets: this limits the scope for estimating target velocities. As for increasing covariances, this can be counter-productive, since clusters with larger covariances tend to "capture" pixels which belong to other, new clusters, or pixels which are outliers in the background.

The probabilities of pixels belonging to the background cluster are also used to update the reference image. Good results have been obtained by weighting the pixel update by the product of prior and posterior probabilities of its belonging to the background cluster:

$$r^{(t+1)}(\mathbf{u}) = r^{(t)}(\mathbf{u}) + \alpha\, f_0(\mathbf{u})\, p_0(\mathbf{u})\, \delta^{(t)}(\mathbf{u}) \tag{9}$$

where α is the time constant of adaptation to slow changes in the background, the superscripts refer to frame number and the probabilities are those obtained at convergence of the EM algorithm applied to the current frame.

1.3 Estimating the number of clusters

Deciding how many distinct clusters are present in the data is usually the hardest problem in cluster analysis. In the present application, the problem is simplified by the fact that the pixels to be clustered are in a low-dimensional space. On the other hand, for real-time applications it is practically impossible to experiment, at each image frame, with different numbers of clusters before deciding which number best fits the data. Much faster processing can be achieved by attempting to fit each image frame with the same clusters as in the previous frame, and then adding or removing clusters, possibly by splitting or merging, if the fit does not satisfy some statistical criteria. Therefore, the algorithm is started with only the background cluster at the first frame of the image sequence; thereafter, the number of clusters is updated according to the following criteria.

Detecting new clusters: New clusters are detected by locating maxima in the difference image, after accounting for the clusters already known to be present in the previous frame of the image sequence. Specifically, at each frame, one iteration of the EM algorithm is carried out with the old clusters, following which a "corrected" image difference is computed, in which each grey-level difference is weighted by the probability of its belonging to the background cluster:

$$\delta_0(\mathbf{u}) = \delta(\mathbf{u}) p_0^{(0)}(\mathbf{u}) \tag{10}$$

where $p_0^{(0)}(\mathbf{u})$ is the estimate of $p_0(\mathbf{u})$ obtained with the cluster parameters carried over from the previous frame. This corrected image difference is averaged within square cells of linear size equal to L pixels, and one pass of smoothing is applied to the coarse-grained difference image obtained in this way. Local maxima in the smoothed coarse-grained image are assumed to arise from new targets if the corresponding (locally-averaged) value of $\overline{\delta_0}(\mathbf{u})$ is greater than a threshold:

$$\overline{\delta_0}(\mathbf{u}) > \kappa_\mu \mu_0 \tag{11}$$

Eliminating clusters: The criteria for eliminating a cluster j are related to the method used to generate new clusters:

1. the average absolute value of grey-level differences for cluster j is less than κ_μ times the value for the background cluster, *i.e.*

$$\mu_j < \kappa_\mu \mu_0 \tag{12}$$

2. the prior probability w_j is less than L^2/m (where L is the cell size used for the detection of new clusters) *i.e.* less than L^2 pixels in the image are likely to be generated from cluster j;

A cluster is eliminated if one or both criteria are satisfied. This test is applied at each iteration of the EM algorithm, to every cluster: the motivation is that the speed of the algorithm is approximately linear in the number of clusters, and therefore it is convenient to remove clusters as soon as possible.

Merging clusters: The two criteria for merging two clusters i and j are based on the covariances of the clusters and the distance between the centroids. Essentially, the two criteria require that the two cluster are close to each other and that, if merged, the resulting cluster would have an approximately ellipsoidal shape. The criteria are as follows:

1. The centroid of the two clusters must be within a given Mahalanobis distance from each other, where the Mahalanobis distance is defined by the covariance of either of the clusters:
$$\sqrt{\Delta\mathbf{c}_{ij}^T \cdot \Sigma_i^{-1} \cdot \Delta\mathbf{c}_{ij}} < \kappa_M \quad \text{OR} \quad \sqrt{\Delta\mathbf{c}_{ij}^T \cdot \Sigma_j^{-1} \cdot \Delta\mathbf{c}_{ij}} < \kappa_M \tag{13}$$
where $\Delta\mathbf{c}_{ij} = \mathbf{c}_i - \mathbf{c}_i$;
2. the two clusters have approximately the same width (within a factor of κ_T) in the direction orthogonal to the distance between the centroids:
$$\kappa_T^{-1} < \frac{s_i}{s_j} < \kappa_T \tag{14}$$
where
$$s_i^2 = \frac{1}{\|\Delta\mathbf{c}_{ij}\|^2} \Delta\mathbf{c}_{ij}^T \cdot \mathbf{R}^T \cdot \Sigma_i^{-1} \cdot \mathbf{R} \cdot \Delta\mathbf{c}_{ij} \tag{15}$$
where $\mathbf{R}$ is a 2×2 rotation matrix which rotates $\Delta\mathbf{c}_{ij}$ by 90 degrees.

Both criteria must be satisfied for merging to take place. The second criterion is needed to ensure that two clusters forming a T-junction are not merged; such T-junctions can be seen *e.g.* when a pedestrian approaches a car from a direction not far from the line of sight (see *e.g.* frame 1265, bottom right of Fig. 1).

This test is applied once per image frame, after convergence of the EM algorithm, to every pair of clusters. Although the resulting computational complexity is quadratic in the number of clusters, the computational cost is negligible when compared to the cost of even a single EM iteration.

Splitting clusters: The criterion for splitting clusters is related to the criteria for merging clusters: a cluster is split if its shape is significantly different from an ellipsoid. Obviously, before splitting a cluster, it is not possible to say whether the two resulting clusters are close to each other, or whether they form a T-junction. Instead, we use a χ^2 measure of the deviations of the observed density from the expected density of pixels belonging to the cluster. As pointed out above, the expected density for a target is well approximated by an ellipsoidal top-hat distribution (Eq. 4), using the same centroid and covariance as estimated with the Gaussian model. To test whether the observed density is significantly different from this expectation, the ellipsoid is divided into nine sections orthogonally to its main axis and the squared deviation of the observed density from the expected density is computed for each section. The sum of the ratios of the squared deviations to the expected densities can be used for a χ^2 test of the hypothesis that the cluster is well-approximated by a top-hat distribution. This value could be converted to a P-value, but since the number of sections, and hence the number of degrees of freedom, is the same for all clusters, it is simpler to set a threshold for the χ^2 value: the cluster is split if

$$\sum \frac{(\text{observed -expected})^2}{\text{expected}} > \kappa_\chi \tag{16}$$

This test is applied once per frame, after convergence of the EM algorithm, to each cluster.

When two clusters are merged, the new cluster is assigned the identity of the larger of the two merged clusters. Similarly, when a cluster is split into two, the larger of the new clusters is assigned the identity of the parent cluster. For each frame in which one or more clusters have been split, or two or more clusters have been merged, the EM algorithm is applied again to optimise the parameters of the new set of clusters.

1.4 Parameters of the algorithm

The number of clusters is dynamically updated at each image frame and all the cluster parameters are determined by the EM algorithm, which is parameter-less. The parameters that remain to be specified are (1) the image-sampling parameters; (2) the initial estimates of cluster parameters for new clusters; (3) the thresholds for generating, removing, merging or splitting clusters; (4) the convergence criterion.

The image sequences were downsampled 3×3 times in space and 5 times in time (*i.e.* the algorithm operated on 5 frames/second). For the detection of new clusters, the linear size of the cells, within which the grey-level differences were averaged, was $L = 8$ pixels of the downsampled image (*i.e.* $L = 24$ pixels of the original image).

The initial centroid of a new cluster is determined by the image position of the corresponding local maximum of grey-level differences. The initial estimate of cluster covariance is given by an isotropic Gaussian with standard deviation equal to L pixels. The initial probability for a new target cluster is fixed at $\hat{w}_j = L^2/m$. Before the first frame, the prior probability of the background cluster is set to $\hat{w}_0 = 1$.

The initial estimate for μ_0 (at the first frame) is obtained by observing a few frames in which no target clusters are visible.

The other parameters of the method are listed in Table 1.

Parameter	Introduced in	Value
α	Eq. 9	10^{-2} frames^{-1}
ϵ	Subsection 1.2	10^{-5}
κ_μ	Eq. 12	6
κ_M	Eq. 13	2.5
κ_T	Eq. 14	2
κ_χ	Eq. 16	80

2 Results

The algorithm was tested on the PETS2000 image sequences (http://www.cs.reading.ac.uk/ftp/PETS2000/). All the vehicles and people visible in the test sequence were detected and tracked until they left the field of view, or until the end of the image sequence. A bird appearing in three frames of the test sequence was also detected. Some frames, showing the tracks of the visible targets and the "top-hat ellipsoids" defined above (Eq. 4), can be seen in Fig. 1: the ellipsoids can be seen to fit quite well the actual targets; the fit is even better if the shadows of the targets on the ground are taken into account.

People entering the field of view from the top are not detected until they get closer to the camera: some more sophisticated thresholding technique might result in earlier detection. However, the main problem becomes evident in frame 1220 (Fig. 1, bottom left): two pedestrians are "merged" into a single cluster; when they are at a sufficient distance to be split again (frame 1265, bottom right of Fig. 1), the identity of the pedestrian who had been tracked from frame 700 is lost. A movie with the tracking results will be shown at the symposium.

The following table lists all the events (changes in the set of clusters) detected in the PETS2000 test sequence:

Frame number	Event
115	First car (saloon) detected
330	Saloon leaves the field of view
340	Bird detected
350	Bird leaves the field of view
375	Second car (hatchback) detected
700	Pedestrian detected
790	Van detected
970	Van leaves the field of view
985	First passenger splits from hatchback
1060	First passenger leaves the field of view
1215	Second passenger splits from hatchback
1220	Second passenger and pedestrian merged into a single cluster
1265	Second passenger and pedestrian split into separate clusters
1395	Pedestrian leaves the field of view

It goes without saying that the algorithm did not recognise vehicles and pedestrians as such: the description of the events is provided by the author. However, in this image sequence, the sizes and image locations of the clusters, together with knowledge about the pose of the camera with respect to the ground plane, would be sufficient to distinguish between vehicles and pedestrians.

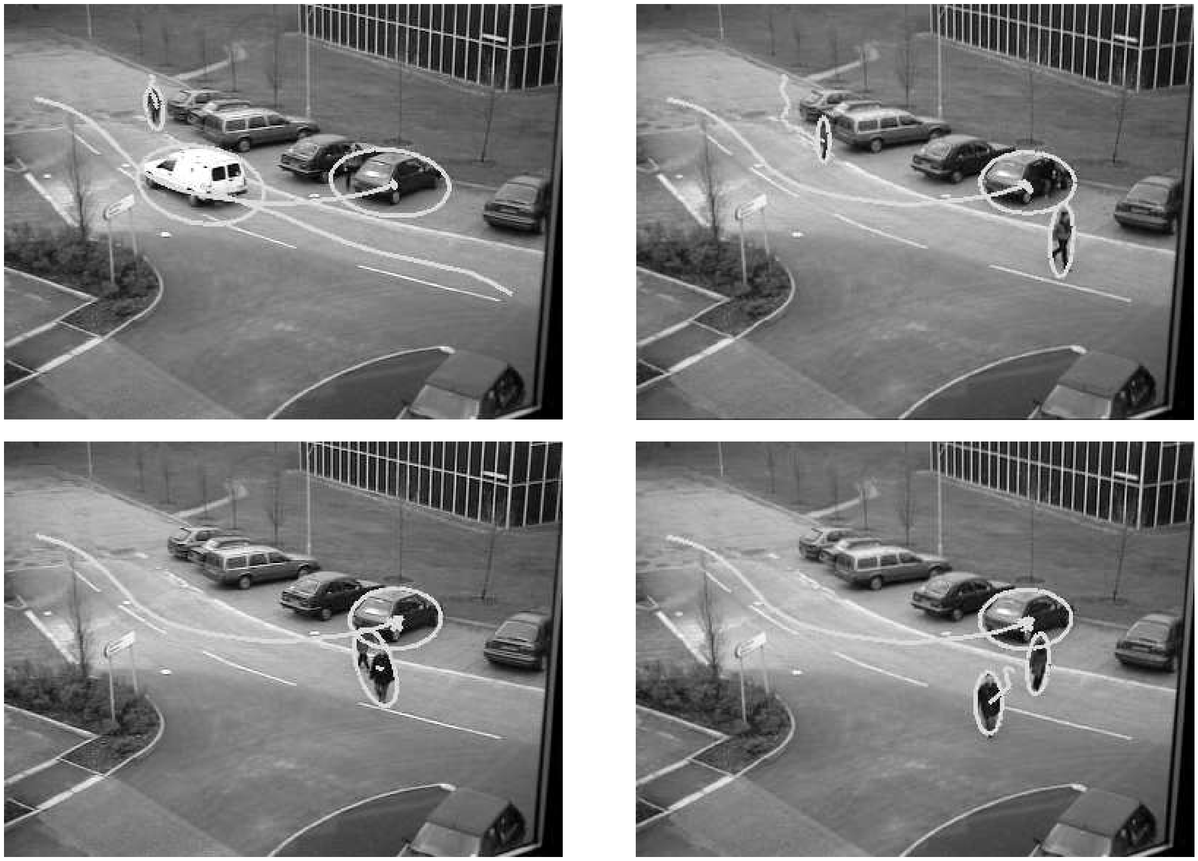

Fig. 1.: Some results obtained with the PETS 2000 test sequence. Top left: frame 840; top right: frame 985; bottom left: frame 1220; bottom right: frame 1265.

In the PETS2000 training sequence, all vehicles and pedestrians were also detected and tracked successfully, except for another temporary “merging” of two pedestrians into a single cluster (frame 1515 in Fig. 2). Another problem was that a passenger leaving a car in frame 3765 “took over” the identity of the car (although the two new clusters fit quite well the pedestrian and the car, once they are well separated, see frame 3950 in Fig. 2). There were also a few “false alarms”, due to changes in the intensities of reflections from car windows; also, one car, which was included in the reference image, leaves behind a “ghost” (the cluster at the top left of frame 3950, Fig. 2) when it moves out of its parking place. On the positive side, the algorithm can cope pretty well with reversals of the driving direction, as can be seen *e.g.* in frame 2140 (Fig. 2).

3 Discussion

The aim of this paper is to illustrate the capabilities and the limitations of an algorithm based only on image-differencing and cluster analysis. The capabilities are already comparable to those of more sophisticated tracking system (see *e.g.* [3]). The main limitation is that an algorithm that ignores the history of the clusters, 3D geometry, and any other source of information, cannot be relied upon to maintain the identities of two targets when they merge into a single cluster. Several improvements can be made to the algorithm. The most obvious is that, in the first frame in which a new target is detected, grey-level statistics for the new target are potentially available and could be used for clustering in the next frame. The grey-level statistics should of course be updated at each subsequent frame, since they depend on the current view of the target. This improvement should be sufficient to preserve the identity of clusters if each target has distinctive colours and there is only limited occlusion.

The method can be described as model-based for three different reasons:

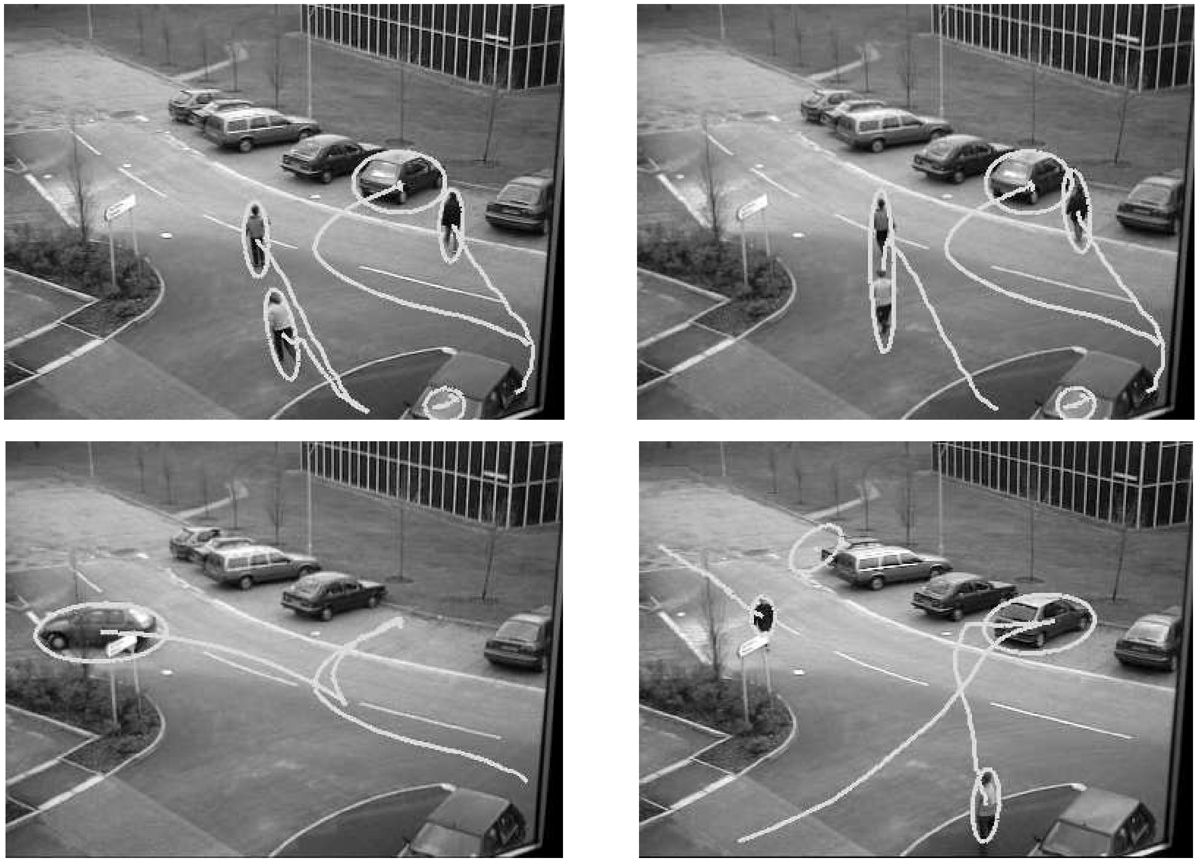

Fig. 2.: Some results obtained with the PETS 2000 training sequence. Top left: frame 1505; top right: frame 1515; bottom left: frame 2140; bottom right: frame 3950.

- the estimation of cluster parameters is based on a probabilistic generative model: only the techniques used to update the number of clusters are somewhat *ad hoc* and offer some scope for improvement;
- the current model parameters are used in the process of model evaluation, due to the application of the EM algorithm.
- the objective function that is being optimised, *i.e.* the likelihood of cluster parameters, is defined on the image-difference data; therefore, the optimisation requires top-down information flow, from current parameter estimates down to the image level.

On the other hand, the method could also be described as image-based, in the sense that the generative model makes no use of 3D geometry, and more generally includes very little knowledge about real-world objects: indeed, the statistical model is specifically designed to work with any type of target.

The EM algorithm is well known to be computationally expensive in terms of numbers of iterations, compared *e.g.* to Newton's method [6]. However, in our application, about 5 iterations per frame are usually sufficient to reach the convergence criterion and also to approximate the target shapes very well, as judged by visual inspection.

For each iteration, computing the probabilities of pixels belonging to clusters is the most time-consuming step, because of the exponential forms of the probabilities. The exponentiations could be avoided by using the K-means algorithm, instead of the EM algorithm. However, the attractive feature of the application of the EM algorithm is that it produces maximum-likelihood estimates which have a simple Bayesian interpretation. As a consequence, data fusion with the outputs of other algorithms is straightforward. Such data fusion is likely to be the best way to preserve the identity of targets when the corresponding clusters are merged or split.

Acknowledgements: This work has been made possible by the software environment for computer-vision research created and maintained by Anthony Worrall at the University of Reading. Many thanks are due to Herold Dehling and Rob de Bruin (University of Groningen, the Netherlands) for helpful discussions and suggestions on the statistical techniques.

References

1. T Darrell, AP Pentland, Cooperative robust estimation using layers of support. *IEEE Trans. P.A.M.I.* 17(5): 474-487.
2. AP Dempster, NM Laird, DB Rubin, Maximum likelihood from incomplete data via the EM algorithm (with discussion). J. Roy. Stat. Soc. B 39: 1-38.
3. JM Ferryman (ed.), Proceedings of the first IEEE workshop on performance evaluation of tracking and surveillance (PETS'2000). Grenoble (France), March 31, 2000.
4. B Heisele, U Kreβel, W Ritter, Tracking non-rigid, moving objects based on color cluster flow. IEEE Conf. Comp. Vision Pattern Rec. (CVPR'97), *pp*.257-260, 1997.
5. P Mahalanobis, On the generalized distance in statistics. Proc. of the Nat. Inst. of Sci. Calcutta 12: 49-55.
6. GJ McLachlan, T Krishnan, The EM Algorithm and Extensions. Wiley: New York 1997.
7. AEC Pece, AD Worrall, Tracking without feature detection. In [3], *pp*.29-37.
8. B Schiele, Model-free tracking of cars and people based on color regions. In [3], *pp*.64-71.