Partial Observation vs. Blind Tracking through Occlusion

Ming Xu and Tim Ellis
Information Engineering Centre
City University, London, EC1V 0HB
{m.xu, t.j.ellis}@city.ac.uk

Abstract

This paper presents a framework for multi-object tracking from a single fixed camera. The region-based representations of each object are tracked and predicted using a Kalman filter. A scene model is created to help predict and interpret the occluded or exiting objects. Unlike the traditional blind tracking during occlusion, the object states are updated using partial observations whenever available. The observability of each object depends on the predictive measurement of the object, the foreground region measurement, and perhaps the scene model. This makes the algorithm more robust in terms of both qualitative and quantitative criteria.

1 Introduction

Tracking non-rigid targets in low-resolution images has long been realized as a region-based correspondence problem, in which each target is mapped from one frame to the next according to its position, dimension, colour and other contextual information. When multiple targets exist and their dimensions are not negligible in comparison with their velocities, occlusion or grouping of these targets is a routine event. This brings about uncertainty for the tracking, because the contextual information is only available for the group and individual targets cannot be identified.

Existing region-based tracking algorithms use either the measurement for the group or the prediction for each target to update the target estimate through grouping. Intille, Davis and Bobick [2] updated the centroid of a target using that of the group and held the velocity, size and colour estimates. Rosales and Sclaroff [5] modelled the two corners of each target's bounding box and updated their positions using the prediction of an Extended Kalman filter. Ellis and Xu [1] estimated the target, which is closer to the group in state distance, using the group measurement and updated the other targets with prediction. However, these algorithms all suffer from poor performance for target estimation during grouping or occlusion. To estimate a target with the group measurement, the estimate of the target is often seriously discontinuous at the start of grouping and may be so misleading as to fail to find a match at the end of grouping. Target updating using prediction is heavily reliant on the motion model and vulnerable to any violation of the underlying assumption during grouping, e.g. the target turning or accelerating, for a first-order motion model that assumes a linear trajectory and a constant velocity.

BMVC 2002 doi:10.5244/C.16.76

We realize that the targets in a group are often partially observable, because some of their bounding edges constitute the four bounding edges of the group. If these partial observations are fed into the estimation process during grouping, the tracker should be more robust and accurate than those without any observation. In this paper, our system assumes that each target has a constant height and width, and models the four bounding edges of each target by a Kalman filter. Once some edge is decided to be observable and its measurement is input to the tracker, its opposite edge could be roughly deduced because the two opposite edges share the "same" (though disturbed by confined noise) horizontal or vertical velocity according the constant height and width assumption. The deduction of unobservable variables from observable ones can be either direct or implicit. The decision of target observability is based on the group foreground measurement, target predictive measurement, and perhaps a simple scene model.

2 Foreground Measurement

Our system uses frame differencing for change detection in dynamic images. It compares each incoming frame with an adaptive background image and classifies those pixels of significant variation into foreground. To maintain a reliable background image, the probability of observing a value for each pixel is modelled by a mixture of Gaussians [7]. At each frame, every new pixel value is checked against the Gaussian distributions. For a matched distribution, the pixel measurement is incorporated in the estimate of that distribution and the weight is increased. For unmatched distributions, their estimates remain the same but the weights are decreased. If none of the existing distributions matches the current pixel value, either a new distribution is created, or the least probable distribution for the background is replaced. The distribution with the greatest weight is identified as the background.

The foreground pixels are filtered by a morphological closing (dilation-plus-erosion) operation and then clustered into foreground "blobs" using a connected component analysis. A minimum number of foreground pixels is set for each blob to rule out small disturbances. A foreground blob may correspond to an object, a group of objects due to dynamic occlusion, or part of an object due to static occlusion. It is represented by a **foreground measurement vector**, $\mathbf{b} = [r_c \;\; c_c \;\; r_1 \;\; c_1 \;\; r_2 \;\; c_2]^T$, where ($r_c$, c_c) is the centroid, (r_1, c_1) and (r_2, c_2) are the two opposite corners of the bounding box. r_1, c_1, r_2, c_2 represent the top, left, bottom and right bounding edges, respectively ($r_1 < r_2$, $c_1 < c_2$). In this paper, we use $\mathbf{b}(i)$ to represent the i-th element of the vector **b**, e.g. $\mathbf{b}(1) = r_c$.

3 Motion Model

A Kalman filter based on a first-order motion model is used to track each object according to the **object measurement vector,** $\mathbf{z} = [r_c \;\; c_c \;\; r_1 \;\; c_1 \;\; r_2 \;\; c_2]^T$. We distinguish object measurements from foreground measurements, because they are the same only for separate objects. Because our system aims to monitor pedestrians and vehicles, each target is assumed to move along a linear trajectory at constant velocity and with constant size. In practice, any minor violation of this assumption can be encoded in the process covariance matrix. The state vector used is $\mathbf{x} = [r_c \;\; c_c \;\; \dot{r}_c \;\; \dot{c}_c \;\; \Delta r_1 \;\; \Delta c_1 \;\; \Delta r_2 \;\; \Delta c_2]^T$, where

$(\Delta r_1, \Delta c_1)$ and $(\Delta r_2, \Delta c_2)$ are the **relative** positions of the bounding box corners to the centroid. They not only incorporate height and width information, but also accurately represent the bounding box even when the centroid is shifted away from the geometric centre of the bounding box, e.g. due to asymmetry or shadows.

The state and measurement equations are:

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{A}\mathbf{x}_k + \mathbf{w}_k \\ \mathbf{z}_k &= \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \end{aligned} \tag{1}$$

$\mathbf{w}_k$ and $\mathbf{v}_k$ are process noise and measurement noise, respectively; the state transition matrix, **A**, and measurement matrix, **H**, are

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_2 & T\mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 \\ \mathbf{O}_2 & \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 \\ \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{I}_2 & \mathbf{O}_2 \\ \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{I}_2 \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{O}_2 \\ \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{I}_2 & \mathbf{O}_2 \\ \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{I}_2 \end{bmatrix} \tag{2}$$

where $\mathbf{I_2}$ and $\mathbf{O_2}$ are 2×2 identity and zero matrices; T is the time interval between frames. The *a priori* estimate, $\hat{\mathbf{x}}_k^-$, and *a posteriori* estimate, $\hat{\mathbf{x}}_k^+$, are related by:

$$\begin{aligned} \hat{\mathbf{x}}_k^- &= \mathbf{A}\hat{\mathbf{x}}_{k-1}^+ \\ \hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-\right) \end{aligned} \tag{3}$$

where $\mathbf{K}_k$ is the Kalman gain matrix that is sought to minimize the *a posteriori* covariance in a least-square sense.

4 Scene Model

Because the camera is fixed, a scene model can be constructed for a specific camera position. Whilst this is currently done manually, the previous work on path learning [3] may be extended to derive an automatic method for learning the scene model. This helps reasoning about the termination and occlusion of objects by scene elements. Three types of static occlusions in a scene are identified (Fig. 1):

- **Border occlusions (BO)**, due to the limits of the camera field-of-view (FOV).
- **Long-term occlusions (LO)**, where objects may leave the scene earlier than expected, corresponding to the termination of a record in the object database. The long-term occlusion may exist at the border (e.g. buildings or vegetation) or in the middle of an image (e.g. at the doors of a building).
- **Short-term occlusions (SO)**, where an object may be temporarily occluded by a static occlusion, e.g. a tree or a road sign. Prior knowledge of these occlusions helps avoid missing existing objects and creating “new” objects.

Each occlusion is characterized by its type (BO, LO or SO) and bounding box representing its location and dimension. The overlap of these static occlusions with the predicted centroid of an object can be used to predict object termination and occlusion. After the *a priori* estimate of the state is determined, each object is subject to the status prediction based on the scene model and predictive measurement:

$$\hat{\mathbf{z}}_k^- = \mathbf{H}\hat{\mathbf{x}}_k^- \tag{4}$$

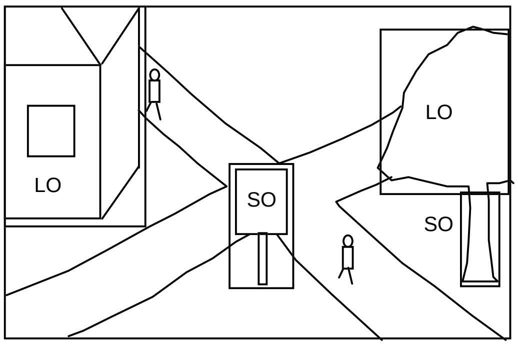

Fig. 1: Static occlusions in a scene.

- An object is labelled as PREDICT_TERMINATED if its predicted centroid, ($\hat{\mathbf{z}}_k^-(1)$, $\hat{\mathbf{z}}_k^-(2)$), is within a long-term occlusion (LO) or the outer limit of the border occlusion (BO).
- An object is labelled as PREDICT_OCCLUDED if its predicted centroid is within a short-term occlusion (SO).

5 Partial Observability

For tracking multiple objects in a complex scene, it is noted that the object measurement, $\mathbf{z}_k$, may be either partly unavailable or completely unavailable. This occurs due to dynamic occlusion between objects, static occlusion, or just the failure of foreground detection. Fig. 2 shows some examples of partial observation.

5.1 Deciding Observability

We decide the observability of the objects based on the predictive measurement, $\hat{\mathbf{z}}_k^-$, the foreground measurement, $\mathbf{b}_k$, and perhaps the scene model. The outcome is represented by the observability vector, $\mathbf{m}_k$, which has the same dimension as the object measurement vector, $\mathbf{z}_k$ (their elements are in one-to-one correspondence). Each element of $\mathbf{m}_k$ has only two possible values: 1 for OBSERVABLE and 0 for UNOBSERVABLE. The centroid is determined as OBSERVABLE only when all the four bounding edges (r_1, c_1, r_2, c_2) are OBSERVABLE.

(1) Observability in grouping (Figs. 2(a) and (b))

- For each foreground blob, all the objects that have their predicted centroid, ($\hat{\mathbf{z}}_k^-(1)$, $\hat{\mathbf{z}}_k^-(2)$), within the blob bounding box are counted.
- If the count is more than 1, the relevant objects form a group and are all associated with that blob.
- Within such a group, if an object has the minimum $\hat{\mathbf{z}}_k^-(3)$ and/or $\hat{\mathbf{z}}_k^-(4)$, its r_1 and/or c_1 become(s) OBSERVABLE; otherwise it is UNOBSERVABLE.
- Within such a group, if an object has the maximum $\hat{\mathbf{z}}_k^-(5)$ and/or $\hat{\mathbf{z}}_k^-(6)$, its r_2 and/or c_2 become(s) OBSERVABLE; otherwise it is UNOBSERVABLE.

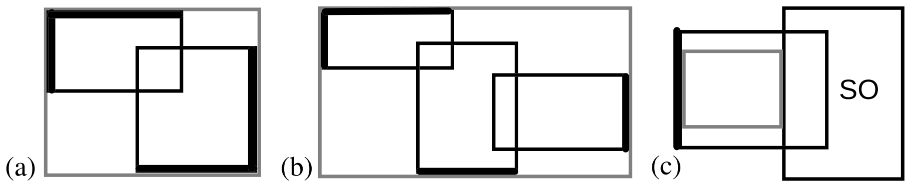

Fig. 2: Partial observations when objects are grouped (a)(b) or behind a static occlusion (c). Grey lines represent the foreground bounding boxes; thick and thin black lines represent observable and unobservable bounding edges of objects, respectively.

(2) Observability in occlusion (Fig. 2(c))

- For each object, check each of its predicted bounding edges. If either corner delimiting that edge is within a short- or long-term occlusion, that edge becomes UNOBSERVABLE; otherwise it is OBSERVABLE.

It is noted that the observability of an object also depends on the observability of its associated foreground measurement, $\mathbf{n}_k$, which has the same dimension as the object measurement, $\mathbf{z}_k$. If the bounding box of this foreground measurement touches the border of the FOV, its relevant bounding edge becomes UNOBSERVABLE and thus inhibits (masks) the relevant observability for the associated object(s), i.e.:

$$\mathbf{m}_k = \mathbf{m}_k \ \& \ \mathbf{n}_k \tag{5}$$

where & represents the logical AND between corresponding elements in two vectors.

5.2 Using Observability

For a completely unobservable object, its state is updated using its predictive state, $\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^-$. For a partially unobservable object, a pseudo measurement vector is constituted, which members can be classified into two inter-correlated blocks (r_c, r_1, r_2) and (c_c, c_1, c_2). The inter-block variables are bound by the constant height (Δr_1 and Δr_2) and constant width (Δc_1 and Δc_2) assumption. Within each block, if all the variables are unobservable, the only clue for their measurements are the prediction, which reflects the constant velocity assumption; if part of its variables is observable, the unobservable measurements can be jointly deduced from the observable measurement, reflecting the constant size assumption, and the prediction. Suppose the observability matrix, **M**, is a diagonal matrix whose main diagonal is the observability vector $\mathbf{m}_k$. The pseudo measurement vector is estimated by:

$$\mathbf{z}_k = \mathbf{M}\mathbf{b}_k + (\mathbf{I} - \mathbf{M})\left[\alpha \mathbf{d}_k + (1-\alpha)\hat{\mathbf{z}}_k^-\right] \tag{6}$$

where $\mathbf{d}_k$ is the directly deduced measurements of unobservable variables from observable measurements, and α controls the combination weights between the directly deduced measurements (constant size assumption) and the prediction (constant velocity assumption). If all the variables in an inter-correlated block are unobservable, $\mathbf{d}_k = \hat{\mathbf{z}}_k^-$ for that block and this is equivalent to $\alpha = 0$. The height and width information in the *a priori* state estimate, $\hat{\mathbf{x}}_k^-$, is used to compute $\mathbf{d}_k$.

6 Tracking Algorithm

Our system works in a frame-based loop, which is summarised in the following:

- Compute the foreground measurement, $\mathbf{b}_k$, and decide its observability, $\mathbf{n}_k$.
- For each tracked object, compute its Mahalanobis distance with each foreground blob. If the predicted centroid of the object is within the bounding box of the foreground blob, the Mahalanobis distance is set to zero. Select the best-matched foreground blob for each object according to the Mahalanobis distance.
- If multiple objects have the same best-matched blob, keep the one(s) with the smallest or zero Mahalanobis distance, and attribute the other(s) an inhibiting Mahalanobis distance.
- Detect object grouping. Decide the observability vector, $\mathbf{m}_k$, and the measurement, $\mathbf{z}_k$, for each object.
- For each object with an allowable minimum Mahalanobis distance to some foreground blob, estimate its state with the measurement, $\mathbf{z}_k$, if it is at least partially observable; otherwise update its state using the predictive state, $\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^-$.
- For each unmatched object
  (1) It is assumed to be terminated, if it is PREDICT_TERMINATED or has been unmatched for *k* frames.
  (2) It is assumed to be behind static occlusions or lost in foreground detection. Update it using the predictive state, $\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^-$. If not PREDICT_OCCLUDED, it is subject to termination after *k* frames.
- For each unmatched foreground blob, a new object is created and its state is initiated by $\mathbf{b}_k$ and zero velocity.
- Compute the *a priori* estimate of each object for *k*+1 and predict its status.

7 Results

To evaluate the performance of our tracking algorithm, we have tested it on a range of image sequences and compared it with other two algorithms using blind tracking through occlusion. To distinguish the effect of using partial observation, both algorithms were designed to be the same as the new one (e.g. the same Kalman tracker), except their treatment to objects in grouping or occlusion:

- **Algorithm 1** — The object with the smallest Mahalanobis distance to the group foreground blob is estimated with the measurement of the group, $\mathbf{z}_k = \mathbf{b}_k$; the others are updated using prediction, i.e. $\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^-$, as in [1].
- **Algorithm 2** — All the objects in a group are updated using prediction, i.e. $\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^-$, as in [5].

The image sequences used for the demonstration in this paper are the testing dataset 1 (CAM1 and CAM2) for PETS'2001 [4]. We processed frames 1 to 2681 at a temporally sub-sampled rate of 5 (simulating 5 fps) and at the half frame size (384×288). *k*=5 in our experiments. In the image results shown below, black and white boxes represent foreground blob measurement, $\mathbf{b}_k$, and object *a posteriori* estimate, $\hat{\mathbf{x}}_k^+$, respectively. A white dotted box represents a partly or completely unobservable object. A white curve

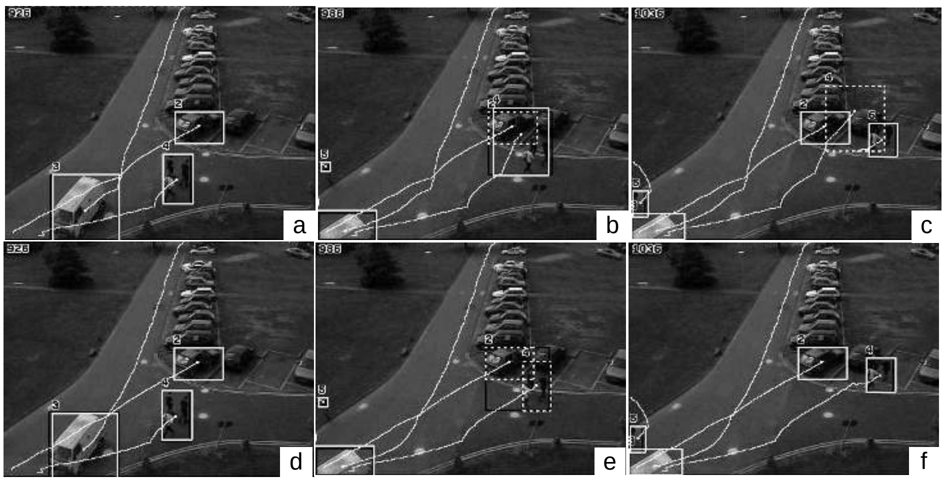

Fig. 3: Example 1 of object tracking through occlusion, (a)-(c) using Algorithm 1 and (d)-(f) using partial observation ($\alpha = 0$).

represents the centroid trajectory of an active object.

7.1 Qualitative Performance

Fig. 3 shows an example of tracking through occlusion. In this example, a group of people (object 4) walk toward and then pass by a stationary car (object 2). At frames 946-991, both the objects are grouped and segmented as a large foreground blob (Fig. 3(b)). Algorithm 1 matches the group foreground blob with object 4, the size of which is then gradually adapted to that of the group measurement. After the group of people is split from the car and segmented as a separate, smaller foreground blob (Fig. 3(c)), Algorithm 1 rejects the match between this blob and object 4, due to the great difference in size and position, and creates a new object (Object 6). Therefore, the group of people changes its label after the occlusion. By using the partial observation, the location and size of object 4 is more accurately estimated during occlusion (Fig. 3(e)). Finally, object 4 is correctly matched to the separate foreground blob (Fig. 3(f)).

Fig. 4 shows another example of tracking through occlusion. In this example, a dark car (object 11 in top row and object 10 in bottom row) moves toward a stationary white van (object 3) and finally occludes it. At frames 2246-2496, both the targets are segmented as a large foreground blob. Algorithm 2 uses linear prediction to update the estimate of object 11 during the grouping. Because object 11 moves in a non-linear trajectory, there exist some errors between the estimate and the foreground blob measurement (Fig. 4(b), see the unfitted bottom and right bounding edges). These estimation errors accumulate and object 11 finally fails to match the corresponding foreground blob (Fig. 4(c)). Using the partial observation, the bottom and right bounding edges of object 10 are closely fit to the foreground blob (Figs. 4(e) and (f)). Object 10 even has a non-linear trajectory during grouping (Fig. 4(f)), which indicates the linear motion model has been continuously adapted to the non-linear motion. It is also noted that the estimate of the top bounding edge of object 10 is not accurate (Fig. 4(f)), because it has been unobservable for a long time.

For all the 9 grouping events (19 objects involved) in which objects merge and then

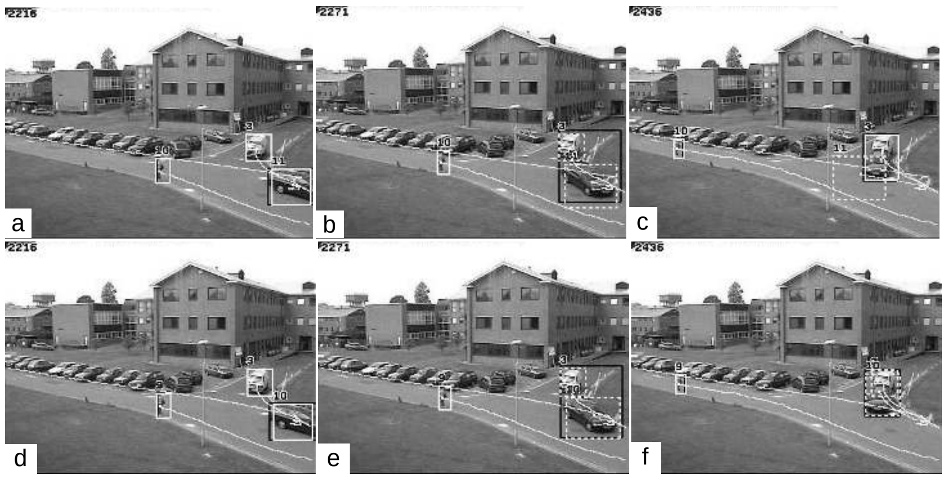

Fig. 4: Example 2 of object tracking through occlusion, (a)-(c) using Algorithm 2 and (d)-(f) using partial observation ($\alpha = 0$).

split, we counted the mis-tracking events in which any object in a group changes its label after splitting. The result shown in Table 1 indicates that the new algorithm performs more reliably than Algorithms 1 and 2.

	Algorithm 1	Algorithm 2	New ($\alpha = 0$)	New ($\alpha = 1$)
Count of Errors	1	2	0	0

Table 1: Counts of erroneous tracking in 9 grouping-and-splitting events.

7.2 Quantitative Performance

The advantages of partial observation are not only reflected in the qualitative comparisons as above, but also exist in some quantitative measures applied on the tracking results in which both Algorithms 1 and 2 succeed. The first measure is the tracking error between actual and predictive measurements, i.e. $e_k = \left\| \mathbf{z}_k - \hat{\mathbf{z}}_k^- \right\|$. For objects updated using prediction, this error is set to zero.

The second quantitative measure is the path coherence, which represents a measure of agreement between the derived object trajectory and the motion smoothness constraints [6]. Suppose $\mathbf{s}_k$ is the segment between the centroid estimates at two consecutive frames, $\mathbf{s}_k = \left(\hat{\mathbf{x}}_k^+(1) - \hat{\mathbf{x}}_{k-1}^+(1),\, \hat{\mathbf{x}}_k^+(2) - \hat{\mathbf{x}}_{k-1}^+(2) \right)$. The path coherence function used is:

$$\Phi_k = w_1 \left[1 - \frac{\left| \mathbf{s}_k \cdot \mathbf{s}_{k+1} \right|}{\left\| \mathbf{s}_k \right\| \left\| \mathbf{s}_{k+1} \right\|} \right] + w_2 \left[1 - 2 \frac{\sqrt{\left\| \mathbf{s}_k \right\| \left\| \mathbf{s}_{k+1} \right\|}}{\left\| \mathbf{s}_k \right\| + \left\| \mathbf{s}_{k+1} \right\|} \right] \tag{7}$$

where the weights w_1 and w_2 control the importance of direction coherence and velocity coherence ($w_1 = 0.5$ and $w_2 = 0.5$ in this paper), and $\Phi_k \in [0,1]$.

These two quantitative measures were selected because they are the basis of most existing motion correspondence algorithms that usually assume the smoothness of motion. These measures are demonstrated using the example shown in Fig. 5, which is

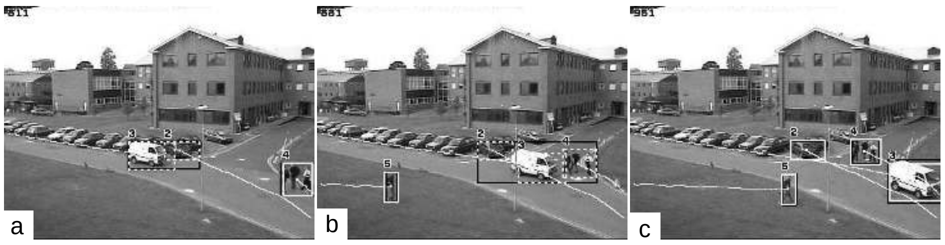

Fig. 5: Example 3 of object tracking using partial observation ($\alpha = 0$).

overlaid by the tracking result using partial observation. In this example, a white van (object 3) first passes by a newly stationary dark car (object 2), heads toward and occludes a group of people (object 4), decelerates and stops separately at the right border of the FOV. Object 3 has been grouped at frames 806-941. Due to the linear trajectories for the objects involved, Algorithms 1 and 2 also succeed in this example. However, these two algorithms have different performance based on our quantitative measures.

Fig. 6 shows the tracking errors and path coherence values for object 3 in Fig. 5, resulting from all the three algorithms. There are two points that should be noted. Firstly, the centroid estimation error only accounts for about one third of entire tracking error, because the latter also includes the errors for two bounding corners. Secondly, the zero values in the tracking error and coherence function for Algorithms 1 and 2 arise from grouping and state updating using prediction, representing uncertainty rather than perfect tracking. Therefore to be fair, our comparison is concentrated on the measures just after the end of grouping (frame 946). At that time objects 3 and 4 split and are re-tracked; the tracking error and coherence are expected to have a peak value.

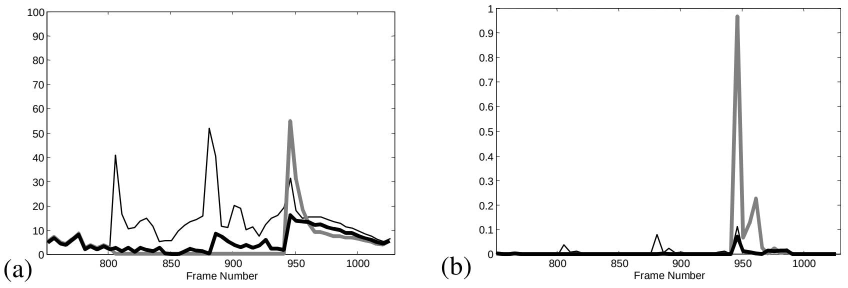

Fig. 6: (a) Tracking errors in pixels and (b) path coherence, for object 3 in Fig. 5, using Algorithm 1 (thin black lines), Algorithm 2 (thick grey lines), and partial observation ($\alpha = 0$, thick black lines).

For the 12 objects involved in all the 6 grouping-and-splitting events in which all the three algorithms succeed, the peak values of the new algorithm are lower than those of Algorithms 1 and 2 in each case; the average peak values are shown in Table 2. The quantitative measures for the new algorithm is much lower than those for Algorithms 1 and 2, indicating its improved performance. The reason is, even fed by partial observation only during grouping, objects could deduce their unobservable bounding edges according to the built-in relation among members of the measurement vector. For example, on the assumption of constant size, the left and right edges of an object should share a horizontal velocity, and the top and bottom edges should share a vertical velocity. The deduction of unobservable variables can be either direct ($\alpha = 1$) or implicit when the Kalman filter seeks the optimal solution for the *a posteriori* estimate ($\alpha = 0$). In the latter case, the measurements of the observable variables are propagated to the unobservable variables. Therefore, even with an incomplete measurement input, the objects still have the estimates of all its four bounding edges adapted to the new, partial measurement. This is partly reflected by the non-zero tracking errors of object 3 during grouping (Fig. 6(a)), which prevents the tracking errors from accumulating and makes object 3 adaptive to the deceleration. The after-grouping peak measures of the new algorithms using $\alpha = 1$ fluctuate around those using $\alpha = 0$. Their relative values depend on whether the constant size assumption ($\alpha = 1$) or the constant velocity assumption ($\alpha = 0$) is better fit to the practical situations in the testing sequences.

	Algorithm 1	Algorithm 2	New ($\alpha = 0$)	New ($\alpha = 1$)
Tracking errors	28.27	29.96	19.33	16.81
Path coherence	0.2765	0.2461	0.0923	0.0863

Table 2: Quantitative measures of the tracking algorithms.

8 Conclusions

We have presented a tracking algorithm utilizing partial observation of each target through grouping or occlusion. The unobservable variables can be estimated by a Kalman filter based on the measurement of observable variables, the state prediction, as well as the scene model. This makes target estimation adaptive to small changes of direction and accelerations during grouping or occlusion. The new algorithm has advantages over traditional blind tracking schemes in terms of lower tracking errors and better path coherence.

Acknowledgements

This work is supported by the EPSRC under grant No. GR/M58030. The authors would like to thank James Black for the discussion on the Kalman filter.

References

[1] T. Ellis and M. Xu, “Object detection and tracking in an open and dynamic world”, *IEEE CVPR workshop on PETS*, Hawaii, 2001.

[2] S. S. Intille, J. W. Davis and A. F. Bobick, “Real-time closed-world tracking”, *Proc. CVPR’97*, 1997.

[3] D. Makris and T. Ellis, “Finding paths in video sequences”, *Proc. BMVC'01.*

[4] PETS'2001, *http://www.visualsurveillance.org/PETS2001.*

[5] R. Rosales and S. Sclaroff, “Improved tracking of multiple humans with trajectory prediction and occlusion modelling*”, IEEE CVPR workshop on the Interpretation of Visual Motion*, Santa Barbara, 1998.

[6] I. K. Sethi and R. Jain, “Finding trajectories of feature points in a monocular image sequence”, *IEEE Trans. PAMI* : 9(1), 56-73, 1987.

[7] C. Stauffer and W. E. Grimson, “Adaptive background mixture models for real-time tracking”, *Proc. CVPR’99*, 1999.