Tracking people with probabilistic appearance models

Andrew Senior
aws@us.ibm.com
IBM T. J. Watson Research Center,
PO Box 704, Yorktown Heights, NY 10598

Abstract

This paper describes a real-time computer vision system for tracking people in monocular video sequences. The system tracks people as they move through the camera's field of view, by a combination of background subtraction and the learning of appearance models. The appearance models allow objects to be tracked through occlusions using a probabilistic pixel reclassification algorithm. The system is evaluated on the three test sequences of the PETS 2002 dataset, for which tracking results and processing time requirements are presented.

1. Introduction

The PETS2002 people tracking database presents a challenging tracking problem in the field of automated visual surveillance. Automated visual surveillance has a variety of potential applications:

- security and surveillance: limiting video storage to interesting events, and alerting guards to exceptions
- traffic flow design: analysing the use of spaces and designing better layouts for public places, museums and shops
- retail space instrumentation: analyzing the shopping habits of consumers
- video indexing: automatic annotation of video for subsequent retrieval
- sports video enhancement: automatically gathering statistics on sports video

All of these tasks have, to some extent, hitherto been carried out by people, but the labour intensity of such jobs and the tedium, resulting in errors, makes them prime targets for automation. In recent years the reduction in cost of cameras and computer processing power have enabled practical video processing in a much wider set of applications. Consequently, research interest in these areas has grown considerably. In practice though, the problems of tracking people in video remain difficult. Automatic approaches are hampered by the variability of human appearance, poor quality images, lighting variations, occlusions and unexpected situations.

The task for the PETS2002 workshop is to track the motion of pedestrians in video sequences of a shopping mall, and automatically determine their motion and activity. The data can be seen as representing a typical surveillance task, though there are a number of particular features that distinguish the data from data that might be acquired in other surveillance situations: the people are close to the camera, subtending a wide area; the people of interest are moving beyond a glass window which partially reflects objects behind the camera; the people being observed are walking on a reflective surface; though colour cameras are used there is little saturation in the colours; being indoors there is no noticeable change in the ambient lighting.

1.1 Previous work

A number of authors have previously tackled the problem of tracking people or objects in video, with a variety of different applications in mind [1, 2, 6, 8].

Tao *et al.* [9] develop a dynamic layer representation in which rigid objects (vehicles) are tracked through 2D translation and rotation with constant velocity prediction. Because of the nature of the overhead view, this system only deals with occlusions by fixed objects. Objects as small as 10x10 pixels are detected by motion segmentation. It is assumed that appearance changes are small. Zhao *et al.* [11] have also described an approach which uses appearance models very similar to those used here. They store a textural template and develop a foreground probability. These authors track with a Kalman filter, model cast shadows explicitly and model the variations in shape due to walking motion.

2. Tracking system architecture

The system that we present here follows the design of our previous PETS paper [7]. The major components are shown in figure 1. Video is segmented into background and foreground regions by a background subtraction algorithm described in section 3. These regions are associated into tracks (section 4) which are refined using the appearance models described in section 5. Occlusions are resolved using these models as described in section 5.2 and in section 6 we propose a method of learning and dealing with fixed objects which occlude the scene. Finally, in sections 7 and 8 we

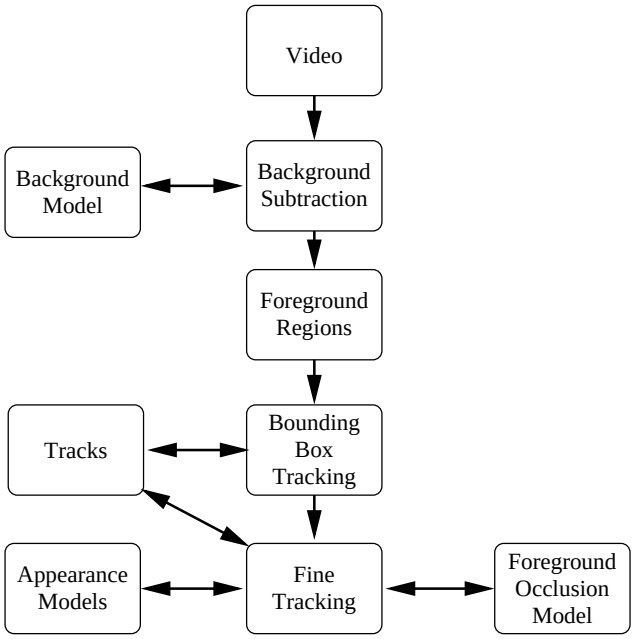

Figure 1: A schematic view of the components of the tracking system.

present experimental results on the PETS2002 datasets and summarize our approach and experiences in the evaluation.

3. Background estimation and subtraction

The most fundamental operation of the system is to distinguish objects of interest which are to be tracked from the background. Since the camera is fixed and conditions in the sequences are fairly stable, we rely on a background subtraction algorithm. The algorithm used is that of Horprasert *et al.* [3].

Briefly, this algorithm accumulates images for a period that shows typical variation in the background appearance and calculates statistics for each pixel across this period. Mean and variance of the pixel values are calculated and thresholds for the noise derived from these. In addition the system models lightness variations of pixels to account for lighting changes giving highlights and shadows when a pixel is not occluded by a foreground object. A final morphology and connected components step removes small foreground regions and fills small holes in bigger components. Ultimately the algorithm returns the set $\mathcal{F}$ of pixels considered to be part of foreground objects. The algorithm has been modified slightly to allow the background model to be created on sequences where there are foreground objects. During the training period regions which differ significantly in appearance from the same regions in the initial frame are assumed to be foreground objects and not added into the estimation of the background statistics. For the experiments here, the background model was trained on training set 2 between frames 120 and 269 which do not contain any moving objects.

On the test sequences, the background subtraction is found to work reasonably well. It has a tendency to under segment, so small objects are often missed, partly because the system has a lower limit (300 pixels) on the smallest connected component that will be returned as a foreground region. On the test data used here, there is very little colour information, and the moving objects are frequently of a similar colour to the background, so large regions of the objects are classified as background. This means that the objects are often fragmented into several connected components, and a small proportion of the foreground pixels are actually identified as such. In sequence 3 background subtraction sometimes fails in the lower right hand corner, so we discard foreground regions below the line $x + 4y = 360$.

4. High-level tracking

The tracking system used in this paper is an extension of the system described in our previous tracking work [7]. A number of extensions have been necessary to allow the system to cope with the PETS 2002 datasets — in particular to cope with the poor quality of the segmentation information coming from the background subtraction algorithm in these conditions.

Initial tracking under simple conditions is carried out by *bounding box tracking*. For this, in each frame, we form a list of the connected components found by the background subtraction algorithm. From preceding frames we have a list of tracks of objects which have been seen in recent frames, together with their centroids and a bounding box for the object. The velocity of each track is calculated and its centroid and bounding box in the current frame are predicted. Tracks are associated with the connected components by searching for overlap in the predicted areas of the track and the regions occupied by the connected components. We associate a bounding box with a track if their *boundary distance* (the shortest distance between the perimeters of two rectangles) is below some threshold (typically around five pixels).

In a simple scene, with well-separated tracks, the association gives a one-to-one mapping between tracks and connected components, and the system proceeds directly to a finer approach using an appearance model, described in section 5.

A number of other cases can occur:

- A foreground component has no corresponding track: here a new track is created to correspond to the foreground region.

- A track is associated with more than one foreground region: in this case the object is assumed to have been poorly segmented and both connected components are treated as foreground regions of the associated track.
- A track is not associated with any foreground region: The track is assumed to have been missed by background subtraction, and is kept alive. If a track has not been observed for a number of frames it is marked as inactive, and if only observed in a handful of frames, it is marked as invalid – attributed to noise, lighting variations or other artefacts of the background segmentation.
- Finally, if several tracks are associated with one or more foreground components, all the foreground components are treated as a single region and the appearance models are used to segment that region into subregions each associated with just one of the nearby tracks. This procedure is described in section 5.2.

Additional rules are used to manage tracks as follows:

- If the centroid of a track goes out of the image, and it is not associated with a foreground region, the track is marked inactive and not considered further.
- If a new track appears, but within 15 few frames is close to another track and its motion is similar, the tracks are assumed to be different fragments of the same object, and are merged into a single object.
- If a single track is explaining two regions which are moving apart and if the separation between them becomes large, the track is assumed to be two objects which were initially close together (such as two people who entered the scene together), and the track is split into two objects representing the two regions.

5. Appearance models

To resolve more complex structures in the track lattice produced by the bounding box tracking, we use appearance-based modelling. Here, for each track we build an appearance model, showing how the object appears in the image. The appearance model is an RGB colour model with an associated probability mask. The colour model, $M_{RGB}(\mathbf{x})$, shows the appearance of each pixel of an object, and the probability mask, $P_c(\mathbf{x})$, records the likelihood of the object being observed at that pixel. For simplicity of notation, the coordinates $\mathbf{x}$ are assumed to be in image coordinates, but in practice the appearance models model local regions of the image only, normalized to the current centroid, which translate with respect to the image coordinates. However, at any time an alignment is known, allowing us to calculate P_c and M_{RGB} for any point $\mathbf{x}$ in the image, $P_c(\mathbf{x})$ being zero outside the modelled region.

When a new track is created, a rectangular appearance model is created with the same size as the bounding box of the foreground region. The model is initialized by copying the pixels of the track's foreground component into the colour model. The corresponding probabilities are initialized to 0.4, and pixels which did not correspond to this track are given zero initial probability.

On subsequent frames, the appearance model is updated by blending in the current foreground region. The colour model is updated by blending the current image pixel with the colour model for all foreground pixels, and all the probability mask values are also updated with the following formulae ($\alpha = \lambda = 0.95$):

$$M_{RGB}(\mathbf{x},t) = M_{RGB}(\mathbf{x},t-1)\alpha + (1-\alpha)I(\mathbf{x}) \text{ if } \mathbf{x} \in \mathcal{F} \quad (1)$$

$$P_c(\mathbf{x},t) = P_c(\mathbf{x},t-1)\lambda \text{ if } \mathbf{x} \notin \mathcal{F} \quad (2)$$

$$= P_c(\mathbf{x},t-1)\lambda + (1-\lambda) \text{ if } \mathbf{x} \in \mathcal{F} \quad (3)$$

In this way, we maintain a continuously updated model of the appearance of the pixels in a foreground region, together with their observation probabilities. The latter can be thresholded and treated as a mask to find the boundary of the object, but also gives information about non-rigid variations in the object, for instance retaining observation information about the whole region swept out by a pedestrian's legs.

Test 1 fr. 570 Test 2 fr. 756

Test 2 fr. 1703 Test 3 fr. 590

Figure 2: Selected appearance models during testing. Each model is represented by two images, the left-hand being P_c, with grey scale from black to white indicating the probability range [0,1]. The right hand image shows M_{RGB} for those pixels where P_c¿0.4. In particular, note the fragmented nature of the models. The model for sequence 2, frame 756 is for two people which have not yet been distinguished. The probability mask shows striations where the models are occluded by a reflection. The right-hand model of frame 570 however, shows a white patch where such a reflection, classified as foreground after being filled in by morphology, is incorporated into the appearance model.

5.1 Appearance-based tracking

The appearance models are used to refine the tracking of objects. Given the location of an object predicted by the first-order model, the location is refined by finding the maximum likelihood location of the appearance model. The appearance of the object is approximated by a spherical Gaussian colour distribution for each pixel. The colour model gives the mean for each pixel so we can calculate the likelihood of the image data, I, given the model and a particular location.

$$p(I, \mathbf{x}, M) = \prod_{\mathbf{y}} p_{RGB}(\mathbf{x}+y) P_c(\mathbf{x}+y) \quad (4)$$

$$p_{RGB}(\mathbf{x}) = (2\pi\sigma^2)^{-\frac{3}{2}} e^{\frac{-\|I(\mathbf{x})-M(\mathbf{x})\|^2}{2\sigma^2}} \quad (5)$$

We evaluate this likelihood over a small search region (± 2 pixels), and fitting a quadratic surface, predict the location of the maximum likelihood. The procedure is repeated around that peak, with a finer step size, to ultimately arrive at sub-pixel localization of the object.

5.2 Multi-object segmentation

The appearance models also provide the key to the segmentation of foreground regions made up of several objects. Here the task is to label the pixels of the foreground region according to which object produced them.

At the heart of this segmentation is a probabilistic pixel classification algorithm that uses the appearance models to calculate the likelihood that a pixel in a foreground region belongs to a particular object. This is an extension of the classification algorithm used in our previous paper. As with the one-to-one case, the algorithm first begins by predicting the location of the foreground objects with a constant velocity model. Then the objects are also aligned to the data using the maximum likelihood fit and quadratic interpolation. This step is complicated by the fact that the objects may well be overlapping and so many of the foreground pixels can only be explained by only one of several models that overlie the pixel. In recognition of this, for objects that overlapped in the previous frame we take advantage of the previous depth ordering (calculated as described below) to guide the fitting and alignment processes.

To fit and align the objects, we proceed in depth order, fitting the front-most model first. After finding its maximum-likelihood position, pixels that appear to have come from that model (i.e. whose likelihoods exceed a threshold) are deleted from the foreground object. Subsequent fitting operations only seek to fit deeper models to the so-far-unexplained pixels which remain. Finally, any objects for which no depth ordering is available are fitted to the data. When there was no overlap in the previous frame, there is no depth ordering information available, but the extent of overlap is likely to be small.

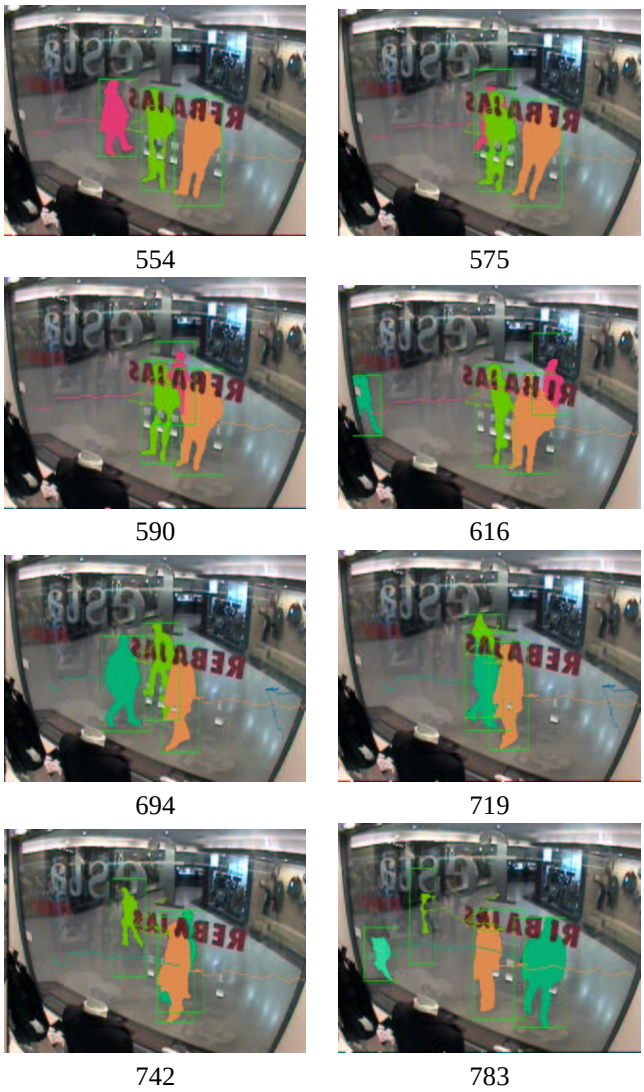

Figure 3: The progress of segmentation during two parts of test sequence 3, with frame numbers.

Given this alignment of all the appearance models, the algorithm can make a better classification of which pixels came from which model. This is formulated as a maximum likelihood classification, with the likelihood $p_i(\mathbf{x})$ of a pixel being generated by foreground model i calculated as follows:

$$p_i(\mathbf{x}) = p_{RGB_i}(\mathbf{x}) P_{c_i}(\mathbf{x}) P_{NO}(i) \quad (6)$$

The non-occlusion probability $P_{NO}(i)$ is either available from a previous frame (described below), or is assumed to be one. Choosing the track i with maximum $P_i(\mathbf{x})$ gives us an object label for the pixel. If the maximum likelihood is very low or is not much higher than the likelihood of belonging to another object, the pixel is marked as "ambiguous" and subsequent processes are used to classify it.

While assigning the pixels in this way, a count is made of the number of "disputed pixels", i.e. pixels for which

two appearance models have high observation probabilities. $P_{c_i}(\mathbf{x})$. These pixels are areas where occlusions are occurring. For each object we record, in a row of an *occlusion matrix*, how many times it occluded pixels of each other object. From the occlusion matrix, we can calculate the non-occlusion probabilities and the depth ordering used in subsequent frames.

Because of the nature of the data, and the approximate nature of our appearance models, there are necessarily misclassifications of foreground pixels, so a number of heuristics are employed to 'clean' the segmentation.

First, we apply a connected components algorithm to the segmented object's regions. Small holes are filled and labelled as belonging to their largest neighbour region. Regions which are entirely surrounded by another component are also assigned to the enclosing component. Finally, if two well-separated regions are disputed between two foreground objects, small areas of pixels in one region assigned to the object which 'owns' the other region are reassigned to the owner of the neighbouring region instead. Figure 3 shows the results of tracking three people through two separate occlusions in test sequence 3. Each person is labelled with a separate colour, and the bounding box of each person's appearance model is drawn as a rectangle. It can be seen that despite the almost total occlusion, the tracking is maintained.

5.3 Background vs foreground segmentation

One disadvantage of most background subtraction approaches is that they are attempting to solve a two-class classification problem where one class is undefined. A background subtraction algorithm must decide if a pixel is background or not. The background's appearance variation can be modelled more-or-less well by observing that pixel over time and making assumptions about such things as lighting variations. However, the alternative category, 'not background', cannot be well modelled *a priori*, (however see *e.g.* Isard *et al.* [4] who do this). Indeed, there is nothing to prevent an object of exactly the same appearance as the background passing in front of the observed pixel. While we rely on background subtraction for our initial determination of foreground vs background, once we have built models for the moving objects in a scene (and for a given pixel, if it is unlikely for a new object to appear there), then we can formulate the problem as a two-class classification problem distinguishing between the background and the (one or more) foreground objects that might occlude the pixel. This should give a more accurate classification as can be seen from the following example: Suppose a background pixel, modelled as colour C_b, is observed to have colour C_o in a particular frame, when we predict that it should be occluded by a foreground object pixel of colour C_f. If C_o is very similar to C_f, and dissimilar to C_b even though it is similar enough to C_b to be deemed background by our background subtraction algorithm, it would be reasonable to assume that it was in fact a foreground pixel.

Thus, after alignment of the foreground objects, every pixel in the area modelled by any foreground object is reclassified according to the most likely of all the overlapping models, including the background model. Pixels are only reclassified when the probabilistic evidence is strong.

5.4 System failures

The main cause of failure in the system is a failure to correctly segment a group of people. Segmentation failures can be divided into two main categories: (1) Under segmentation. The tracking algorithm used is unable to distinguish two people who enter the scene walking together until they separate into two foreground regions. Since an individual is of variable shape, with different parts moving relative to each other and the foreground extraction is erratic, there is no salient feature that distinguishes one person from two people walking together. An explicit person model might resolve this, but would probably require better quality images. The segmentation when two people do separate should be carried out sooner. (2) Mis-segmentation. When two tracks come together the algorithm fails to allocate the pixels to the correct model because of similarities in appearance, and tracking is lost, particularly when the depth-ordering is incorrectly estimated. When the people do separate, additional tracks are created, so the system can end up with more than one track for a single person.

6. Foreground occlusion modelling

In section 5.2 we have dealt with the problem of occlusion of one tracked object by another tracked object, but another problem in many computer vision problems is when a tracked object is occluded by a static object. In our situation, we have no special model of the static objects in the scene — they are simply modelled by the background subtraction algorithm — and it is consequently tacitly assumed that the background model pixels will be occluded when they overlap with the pixels of a foreground object.

In both the PETS 2001 and 2002 datasets, this assumption is violated— moving objects pass behind objects modelled by the background model, whether these be buildings, parked cars, or in the PETS2002 dataset, the window frame or the text appearing on the window. The previously described modelling will fail at these locations, and pixels are quickly "forgotten" (P_c approaches zero) by the appearance model when they are not detected as foreground. In fact these pixels are being occluded, which should not lead to changes in the appearance model. To improve the modelling of such scenes we create a *foreground occlusion map*.

In complex scenes predicting whether a moving object will be occluded by the static "background" objects requires

a model of the depth of the moving object and of every point in the background model. This can be learnt over time, with enough data [5] but for the short sequences here it is not practicable. However the data in PETS2002 can be simply divided into three depth categories.

- Background pixels: These are the background pixels that are occluded by moving objects and which are modelled by the background model.
- Moving objects: The tracked foreground objects for which we create appearance models. They occlude pixels in the background but are occluded by *foreground-occluding pixels:*
- Foreground-occluding pixels: These pixels are static and modelled by the background model, but are never occluded, being closer to the camera than all the moving objects.

This three-layer model holds well for these data-sets, the only exception being people passing behind the shop opposite the camera. These people are, however, too small to be detected reliably by our background subtraction, so the exception is not important.

We model the foreground-occluding pixels with a separate probability map, $P_f(\mathbf{x})$, which records the probability of a pixel, $\mathbf{x}$, being in this category. The map is updated with the following equation:

$$P_f(\mathbf{x}) \quad = \quad \frac{\sum_{t \in T} P_c(\mathbf{x}) + k_n}{\sum_{\forall t} P_c(\mathbf{x}) + k_d} \tag{7}$$

where T is those times when pixel $\mathbf{x}$ is assigned to the background. The sums are updated for every frame, changing only when an object overlies a given pixel. Constants $k_n = 0.4$ and $k_d = 2$ are used to initialize the model to give initial estimates, and to prevent the model from saturating with small amounts of data. The map is initialized with the three training sequences, and then loaded before operating on the test sequences, though it continues to be updated during the test sequences.

The foreground occlusion model can now be used in the update function for the appearance models. The probability mask update rule now becomes:

$$P_c(\mathbf{x}, t) = P_c(\mathbf{x}, t-1)(1 - \lambda P_f(\mathbf{x})) \text{ if } \mathbf{x} \notin \mathcal{F} \tag{8}$$

$$= P_c(\mathbf{x}, t-1)\lambda + (1 - \lambda) \text{ if } \mathbf{x} \in \mathcal{F} \tag{9}$$

(Figure 4 shows examples of the foreground-occlusion model during training. The occluding regions of text on the window and the window frame, as well as the strong reflections on the glass, can be clearly seen in the model as having high probability of occluding the foreground appearance models. In areas where moving objects have been frequently seen, the probability is close to zero. In less-well travelled areas, the probabilities take intermediate values, often too high because of early failures in tracking and where the foreground model probability mask, being temporally smoothed, is only a rough guide to the appearance of the foreground region coming from background subtraction. The foreground-occlusion model also highlights areas where background subtraction is prone to failure, such as the band to the top-right of the image. This is desirable as a consistent failure of background subtraction is equivalent to an occlusion.

While the foreground-occlusion models generated do match our intuition of what should be learned, and they do result in appearance models which are less eroded by passing behind foreground-occluding pixels, it was found that using them led to poorer tracking performance. Since large areas of the field of view are "expected to be occluded", these areas inhibit diminution of model observation probabilities. This means that objects are mistakenly tracked into these areas, but these failures are not penalized as before when foreground pixels are not observed. Consequently, this feature was not used in the final experiments.

7. Experimental results

The system described above was applied to the sequences provided in the PETS 2002 person tracking datasets. The MPEG videos provided were used without modification. A background model and foreground occlusion model were trained on the training sequences and the former information was used in the runs on the test sequences.

The PETS2002 task requires the following measures to be evaluated, which we provide on a frame-by-frame basis:

- Number of people in the scene
- Number of people in front of the window
- Number of people looking at the window
- Processor time

Timing is discussed in section 7.1. The other measures are shown plotted in figure 5. The number of people in the scene is just the number of tracks active at a given time, though there may not actually be any foreground pixels assigned to the track for the current frame (due to occlusions or failure of background subtraction). We have created ground truth data for the entry and exit times for all people, regardless of size or occlusion by observing the videos. The number of people visible in the scene (or occluded by the lettering) is plotted in the graphs of figure 5.

On sequence one, results are essentially correct, except the failure to detect people when they are distant — a person who enters at frame 144 is not detected until after frame 300, and another person entering at around frame 333, is never detected. On sequences 2&3, failures to correctly segment are relatively frequent when there are several people in the scene, but the graphs shows that our estimate

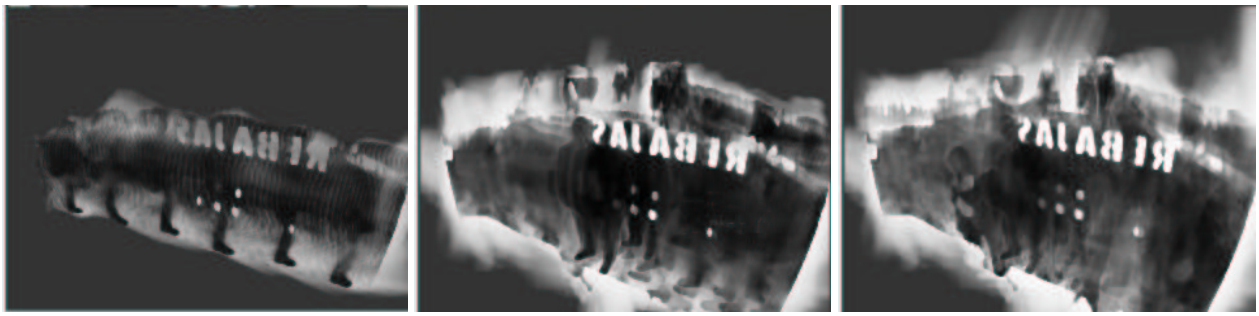

Figure 4: Evolution of the foreground-occlusion model $P_f(\mathbf{x})$ during training. (a) after 119 frames from training sequence 2. (b) after all 1420 frames of sequence 2 (c) after training on all three training sequences, prior to being used for testing.

of the number of people is within 1 of the correct answer for sequence 2, and nearly always within 2 for sequence 3, though after about frame 800 the tracking is almost completely wrong.

Determining if a person was in front of the window, was carried out with two simple conditions: if the x coordinate of the centroid is between 140 and 490 and the lower edge of the bounding box for this ordinate is below the line joining (110,140) with (490,50). People were deemed to be looking into the window if their centroid speed is below 1 pixel per frame. For these measures in particular, some temporal smoothing would give more useful results.

We did investigate using head pose detection to more accurately determine the direction of gaze, with a method similar to that of Wu and Toyama [10], but the person location, and the data quality and resolution were not good enough to return meaningful answers.

Since the results are reported on a whole sequence, the observation counts are actually calculated after the sequence has been processed. This permits a small amount of post-processing that means the results can be reported more accurately. Specifically, tracks which were seen to split into two people can be retrospectively labelled as representing two people, back to the time the people entered the scene. Similarly, with tracks which were deemed to be separate parts of a single person, or deemed spurious can be correctly reported with this "hindsight". The postprocessing takes a negligible amount of CPU time.

7.1 Processing time

For experimental purposes, the background subtraction and the tracking were carried out separately, and we have evaluated the processing time for the two components separately. Figure 6 shows graphs of the time used for each frame of the three sequences. The upper line in each graph shows the total amount of time (in ms) used by the system, excluding disk access and software decoding of the MPEG video, and the lower line shows the portion of this time required for the background subtraction. All experiments were performed on a machine with a single 1.8GHz Pentium 4 processor.

It can be seen that during sequence 1 the system operates at at least 10fps on the 640x240 images supplied. Performance is lower when there are many people are present tracking starts to fail. Significant speed-up can be achieved by down-sampling the images or not processing all the frames. In figure 6(2) we show times for processing sequence 2 recoded as an AVI file at half x resolution, (but full y resolution *i.e.* 320x240) and using only alternate frames. The time for each frame is halved and tracking works as well. The worst-case performance ($\tilde{7}$5ms/frame) on this sequence is faster than real-time (12.5fps). Further downsampling in resolution or time should give approximately linear speed-increases.

8. Summary and conclusions

We have described a system that tracks people in video, and is able to track them in real time despite a number of difficult conditions found in the test sets used. Non-rigid objects, occlusions, similarly coloured objects and reflections are all handled by the tracking system.

Further improvements to the algorithms could be made by more detailed analysis of pixel reclassification with a foreground model, as the system is still heavily reliant on the results of the background model before the detailed foreground models are considered. Tracking, perhaps with image gradients, and incorporating object edges might also yield improvements. Finally, it is hoped that the foreground occlusion model can be made to yield better results, though for most practical situations if such a model is to be of use, it will need to allow arbitrary depths, not the simple three-plane model that we have implemented. For a practical system to handle the scenario presented in the PETS data, the reflections could be avoided by better camera placement or polarization, and the modelling could be improved by larger amounts of training data.

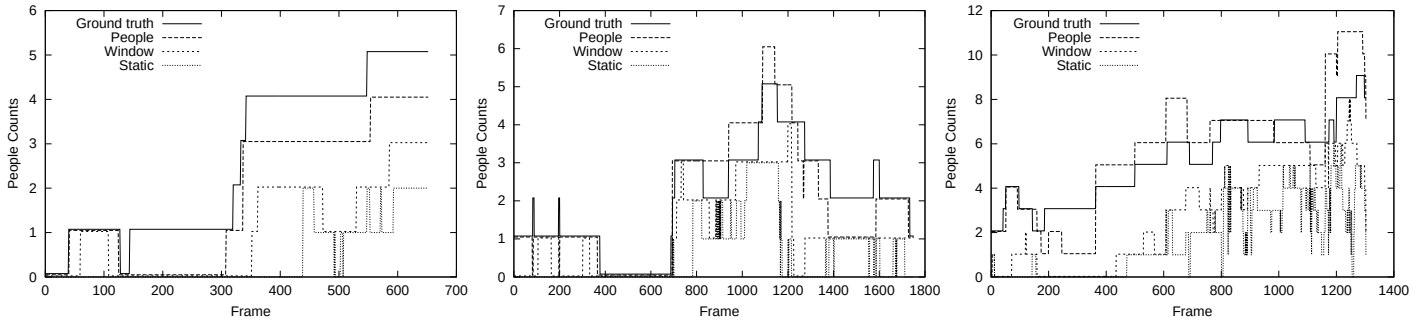

Figure 5: Person counts for the three test sequences, in order. The graphs show our ground-truth estimate of the number of people in the scene (solid). In addition, the graphs show (dashed) the number of people in the scene (upper line), the number considered to be in front of the window (middle) and the number of people considered to be stopped in front of the window (lower). The lines are displaced vertically for clarity.

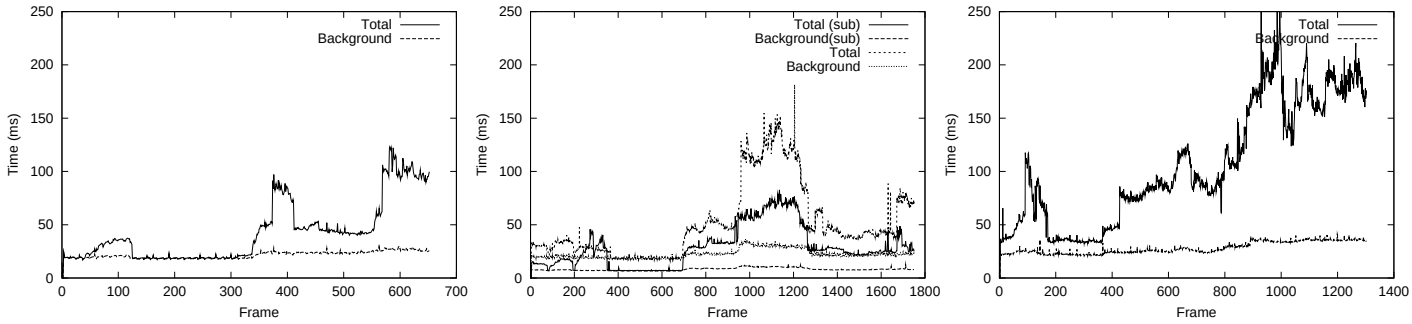

Figure 6: Timing results for the three test sequences, in order. For each frame we plot the total processing time in milliseconds for background subtraction and tracking, and (lower line) the time spent on background subtraction only. In the middle graph we also plot the times required ('sub') for processing alternate frames at half resolution.

Acknowledgments

The author would like to thank Ruud Bolle and the other members of IBM's PeopleVision project: Sharath Pankanti, Arun Hampapur, Lisa Brown and Ying-Li Tian for their assistance throughout this work; the reviewers for many useful comments; and Ismail Haritaoğlu of IBM Almaden Research for the original background subtraction code.

References

[1] I. Haritaoğlu and M. Flickner. Detection and tracking of shopping groups in stores. In *CVPR*, 2001.

[2] I. Haritaoğlu, D. Harwood, and L. S. Davis. W^4: Real-time surveillance of people and their activities. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):809–830, August 2000.

[3] T. Horprasert, D. Harwood, and L. S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *ICCV'99 Frame-Rate Workshop*, 1999.

[4] M. Isard and J. MacCormick. BraMBLe: A Bayesian multiple-blob tracker. In *International Conf. on Computer Vision*, volume 2, pages 34–41, 2001.

[5] A. Schödl and I. Essa. Depth layers from occlusions. In *Conference on Computer Vision and Pattern Recognition*, 01.

[6] J. Segen and G. Pingali. A camera-based system for tracking people in real time. In *Proc. International Conference on Pattern Recognition*, pages 63–67, 1996.

[7] A. Senior, A. Hampapur, Y.-L. Tian, L. Brown, S. Pankanti, and R. Bolle. Appearance models for occlusion handling. In *Second International workshop on Performance Evaluation of Tracking and Surveillance systems*, 2001.

[8] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):747–757, August 2000.

[9] H. Tao, H. S. Sawhney, and R. Kumar. Object tracking with Bayesian estimation of dynamic layer representations. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(1):75–89, January 2002.

[10] Y. Wu and K. Toyama. Wide-range person- and illumination-insensitive head orientation estimation. In *Face and Gesture*, pages 183–188, 2000.

[11] T. Zhao, R. Nevatia, and F. Lv. Segmentation and tracking of multiple humans in complex situations. In *Conference on Computer Vision and Pattern Recognition*, 2001.