A NEURAL SYSTEM FOR AUTOMATED CCTV SURVEILLANCE

A Hunter, University of Durham
J Owens, University of Sunderland
M Carpenter, Akeler Management Services Ltd.

INTRODUCTION

This paper overviews a new system, the "Owens Tracker," for automated identification of suspicious pedestrian activity in a car-park.

Centralized CCTV systems relay multiple video streams to a central point for monitoring by an operator. The operator receives a continuous stream of information, mostly related to normal activity, making it difficult to maintain concentration at a sufficiently high level. While it is difficult to place quantitative boundaries on the number of scenes and time period over which effective monitoring can be performed, Wallace and Diffley [1] give some guidance, based on empirical and anecdotal evidence, suggesting that the number of cameras monitored by an operator be no greater than 16, and that the period of effective monitoring may be as low as 30 minutes before recuperation is required.

An intelligent video surveillance system should therefore act as a filter, censuring inactive scenes and scenes showing normal activity. By presenting the operator only with unusual activity his/her attention is effectively focussed, and the ratio of cameras to operators can be increased.

The *Owens Tracker* learns to recognize environment-specific normal behaviour, and refers sequences of unusual behaviour for operator attention. The system was developed using standard low-resolution CCTV cameras operating in the car-parks of Doxford Park Industrial Estate (Sunderland, Tyne and Wear), and targets unusual pedestrian behaviour.

The *modus operandi* of the system is to highlight excursions from a learned model of normal behaviour in the monitored scene. The system tracks objects and extracts their centroids; *behaviour* is defined as the trajectory traced by an object centroid; *normality* as the trajectories typically encountered in the scene. The essential stages in the system are: *segmentation* of objects of interest; *disambiguation and tracking* of multiple contacts, including the handling of occlusion and noise, and successful tracking of objects that "merge" during motion; *identification* of unusual trajectories. These three stages are discussed in more detail in the following sections, and the system performance is then evaluated.

Figure 1
Background *B*, current frame *I*, and difference *D*

OBJECT SEGMENTATION

The system segments objects using the well-known background differencing technique: a background image is maintained, and subtracted from the current frame and thresholded to identify moving objects; see figure 1. There exists a range of techniques for calculation of the background image, *B*, from the difference image, *D*; we use a relatively simple approach [2] that updates each pixel of the background image using the equation:

$$B_i(t) = \beta_i I_i(t) + (1-\beta_i) B_i(t-1) \qquad (1)$$

where i indexes the image pixels, and β_i controls the update rate of the background image. β_i is set using:

$$\beta_i = \begin{cases} 0.1 & D_i(t) = 0 \\ 0.0001 & D_i(t) = 1 \end{cases} \qquad (2)$$

The background image follows the input if the pixel is judged to belong to the background. Pixels classified as foreground influence the background at a much-reduced rate, so that if a pixel is persistently mis-classified as foreground the background image will eventually update.

OBJECT TRACKING AND DISAMBIGUATION

The difference image must be processed to identify individual moving objects - a difficult process due to noise, occlusions, and object merging events. The approach we take is to apply some morphological noise removal, followed by connected components analysis and removal of very small components. The resulting image contains a number of "silhouettes" (significant connected components). A pedestrian may correspond to part of a

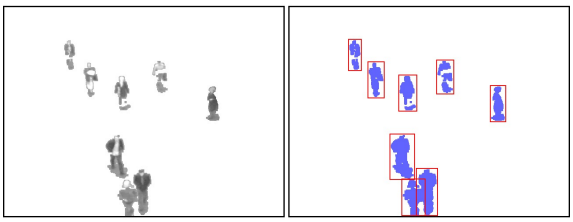

Figure 2
Tracking with merged and fragmented objects

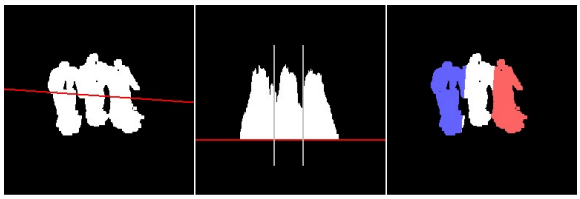

Figure 3
Splitting a merged silhouette along the principal axis

silhouette (e.g. if walking in a group), or to several silhouettes (break up due to low contrast), and some silhouettes may correspond to no foreground object at all (noise, shadows) or to irrelevant objects (e.g. cars).

A popular approach [3] [4] maintains a "pool" of competing Kalman filters, with the best matches persisting, which allows objects to be tracked through occlusions and segmentation failure. However, when objects merge into a single silhouette in the difference image, the system only assigns one of the tracks, and the other track is lost if the objects do not separate before the non-matched filter is destroyed. Our multiple tracking module (Owens et al [5]) addresses the issues of merging and break-up, using a novel algorithm that maintains a correspondence between tracked objects and silhouettes, allowing multiple objects per silhouette and *vice versa*; see figure 2. In contrast with the alternative approaches discussed above the predictive step is very simple, and yet the algorithm is very effective, indicating that sophisticated prediction is not the core issue in multiple tracking.

Each object or silhouette is characterized by a feature vector containing its pixel area, bounding box height and width, and 16-bin intensity histogram. Central to the algorithm described below is a cost function for matching a silhouette and an object's feature vectors; this is calculated by summing the normalized differences for the area, height, width and histogram. Normalized differences are calculated for the three scalars by dividing the absolute difference by the values for the object; the normalized difference of the histograms is calculated by treating the histograms as vectors, and dividing the difference histogram vector's magnitude by the object histogram vector's magnitude. The object-silhouette match is maintained in an $n \times m$ binary matrix, M, where n is the number of tracked objects and m is the number of segmented silhouettes.

$$M = \begin{array}{c} \\ Q_0 \\ Q_1 \\ \vdots \\ Q_n \end{array} \begin{array}{c} \begin{array}{cccc} S_0 & S_1 & \cdots & S_m \end{array} \\ \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \end{array} \qquad (3)$$

On each time-step, our tracking algorithm reconstructs the match matrix, M, using several steps. Initially there are no instantiated objects, and the algorithm moves directly to the new object instantiation step, described below. Otherwise the match matrix is created, and initialised by assigning each object to the silhouette nearest to its predicted position, provided that silhouette is within a maximum allowable distance; a single silhouette may acquire more than one object. The object predicted position is given by the sum of the previous location and velocity. In ideal cases this naïve matching step may unambiguously track objects. The remaining steps compensate for various matching problems.

First, we deal with *match conflicts*, where a silhouette is assigned to more than one object. If a transient object (one time-step old) has a match-conflict with a non-transient, we remove the former match; this addresses various problems with brief noise sources such as reflections of pedestrians on vehicles and shadows. If two non-transients match-conflict, we test whether they should be regarded as merged. The feature vector of the union of the two objects is compared with the silhouette vector. If this has lower cost than the lowest cost single object matched to the silhouette a merge event has occurred, and the double match is retained. Otherwise, the higher cost match is broken, and that object is assigned to its next lowest cost match. This process is iterated until all match conflicts have been resolved. Following the conflict resolution stage there may be some merged silhouettes; we split these by projection onto the principal axis of the silhouette; see figure 3.

Fragmented objects are handled next. Unassigned silhouettes are assigned to nearby objects, and if this reduces the current match cost the assignment is accepted. This process is carried out first with non-transient objects, then transients; favouring assignment to non-transients helps to contain noise problems.

The reassignment algorithm having completed, any objects that lack a match are deleted, and new objects are instantiated for sufficiently large unassigned silhouettes.

The multiple tracker was evaluated on live video gathered during the morning of two days from 8:00am to 11:00am, comprising a total of 6 hours, spanning a range of activity levels, from peak activity to quiescent periods. Table 1 shows the number of actual versus system-identified

	Number of Pedestrians/Vehicles Tracker				
Actual		0	1	2	3
	0	–/-	9/0	1/0	0/0
	1	3/0	120/139	27/1	2/0
	2	0/0	3/0	4/0	0/0
	3	0/0	0/0	1/0	1/0

Table 1
Performance of the Multiple Tracker

contacts; ideally the table should contain no off-diagonal terms. Modes of failure included vehicles and pedestrians fragmenting, and false-positive pedestrian contacts caused by reflections or shadows. All of these error events were of short duration; as our system only submits trajectories to the novelty detection stage if they persist beyond a threshold period, none of them caused novelty detection errors.

NEURAL NOVELTY DETECTION

The purpose of the novelty detection stage is to learn to recognise normal trajectories in the observed scene, and to process new trajectories to recognise whether they are normal. This is a challenging problem, not least because of the variable length of trajectories. One extreme approach is to diagnose on the basis of instantaneous information such as position alone [6] or first-order velocity [7], or to combine these two into a single feature vector, as in many motion detection systems [3] [8]. The other extreme is to classify entire trajectories; intermediate approaches use a limited-term trajectory history. In our experience these two types of novelty detection are best treated quite differently; consequently our system contains two novelty-detection components.

Our first novelty detection module is a topological map (SOFM), with an augmented input feature vector that contains the current position and instantaneous velocity, together with time-smoothed (trace decay) versions of these that provide a short recent history. This is similar in motivation to the time-smoothing layer used in the specialized neural network of [6], but provides a simpler model with somewhat improved performance. Training is performed by gathering a large number of trajectories during a normal monitoring period (see figure 4), extracting feature vectors, then applying the standard Kohonen training algorithm. This component recognises novel trajectories defined by objects in unusual positions, moving at unusually low or high speed for a given position, meandering, and making unusual changes of direction; it is described in more detail by Owens et al [9].

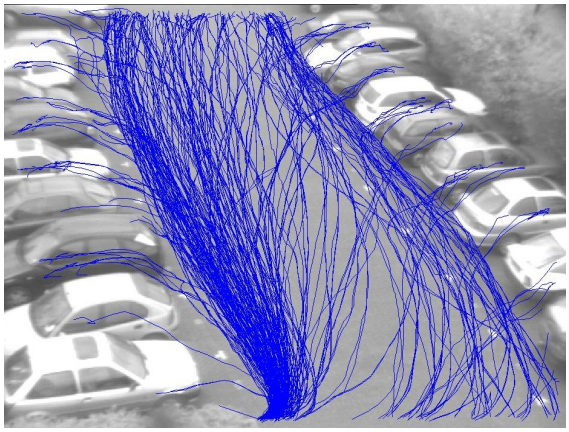

Figure 4
Scene trajectories in training set

Our second novelty-detection component is aimed at diagnosing longer-scale novelty, such as a pattern of movement that visits several cars. Such trajectories are characterized by an unusual ordered co-occurrence of otherwise normal short-term trajectory sequences [3]; the details of these sub-sequences are not important. Several approaches to handling such long-term recognition have been suggested in the literature, but they often suffer from the drawback that a full trajectory must be traced before novelty detection can occur [10]. In contrast, our system is able to detect excursions from normality immediately they occur.

The system uses a specialized self-organizing neural network that is designed to "chunk" trajectories into self-organized segments, and to build a model of ordered co-occurrence of those chunks. To achieve trajectory chunking in a self-organizing fashion, the hierarchical network has units with overlapping receptive fields; see figure 5. A trace rule [11] is used to maintain a recent movement history. Together with competition between hidden units, and Hebbian learning between pre-synaptic traces and the winning hidden neuron, this allows the hidden neurons to learn to respond to specific local sub-trajectories. Subsequent hidden layers with overlapping receptive fields repeat the process at a larger spatio-temporal scale, progressively chunking the trajectories. Figure 6 illustrates the trajectories corresponding to hierarchical network output units, produced by back-projecting their activation sensitivity to the input layer.

Associated with each chunking neuron in the final level is a second *novelty-accumulator* neuron. During learning, lateral connections between these novelty-accumulator neurons are set to indicate ordered co-occurrence; that is, a connection $w_{i,j}$ is set to 1 if neuron i is ever active before neuron j in a specific trajectory sequence. During execution, novelty accumulator neurons receive excitatory input from chunking neurons, and inhibitory input from co-occurring novelty accumulator neurons. Hence, a high novelty signal indicates a novel co-occurrence, and an alarm can be raised.

		System Verdict	
		Normal	Unusual
Actual	Normal	320	40
	Unusual	1	63

Table 2
Overall system performance

Activity Characteristics	Events
Tracking Failure	20
Low representation in data set	10
Unpredictable object interaction	9

Table 3
Break-down of false positive failures

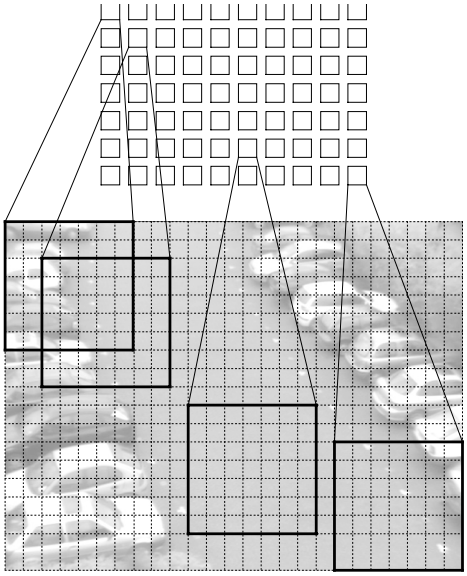

Figure 5
Receptive fields for the hierarchical network

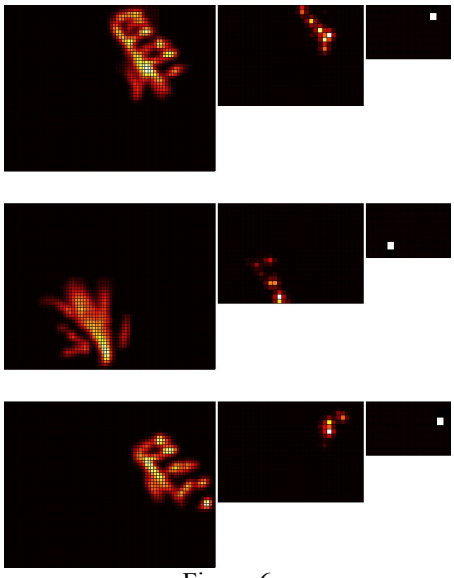

Figure 6
Back projection illustrating trajectory recognition

Further details of the hierarchical network execution and learning algorithms are given in Owens et al [12].

SYSTEM PERFORMANCE

The overall system is quite complex, and errors can occur within each stage, so that careful design is required to prevent error rates from being cumulative. This is achieved partly by the incorporation of heuristic rules; for example, the novelty detection module is not invoked until an instantiated object has persisted for at least three time-steps, thus eliminating errors caused by transients in the latter propagating into false alarms in the former; and specialized rules help to handle common occurrences such as cars stopping to allow passengers to alight.

The system was trained and tested on data gathered from two different car-park scenes, one of which was not used during system design; performance was broadly comparable on the two; we report results from only one. For this site, training data was gathered during the period 8.00am to 10.00am over five days, with a total of 308 normal pedestrian trajectories, containing over 20,000 centroid points, assigned to training the networks; figure 4 shows the training set. The operator is required to screen the training trajectories, purging any that are errors or atypical. For live testing, a further week's data was gathered, consisting of 424 recorded events, of which 261 were pedestrians, 64 of which were “unusual” trajectories, 58 deliberately generated by the author. The data included ample opportunity for the system to test performance on events such as pedestrians alighting from vehicles, and multiple objects in view simultaneously and interacting.

Table 2 summarizes system performance. The single false negative was generated by an otherwise normal trajectory with two brief periods of stationarity (about one second each), which proved sub-threshold; we do not deem the failure to respond in this case significant. However, the number of false positives is a cause for concern. These may be broken down according to the cause of failure; see table 3.

Tracking failure refers to cases where the fragmentation and merging algorithms did not respond reliably; consequently, the centroids of some tracked objects "jumped around," causing an erratic trajectory that triggered the novelty detector. These are genuine failures, and indicate that the multiple tracker is a key component governing performance. We note, in contrast, that the

novelty detection modules performed acceptably in all cases, both positive and negative, where a correct trajectory was generated.

Low representation refers to trajectories that, although regarded as normal by the observer, were in fact novel with respect to the training data set. This is an inherent limitation of the approach, but should be mitigated if the data set is dynamically updated and the system retrained over a longer period of time.

Unpredictable object interaction occurs when pedestrians interact (e.g. veer over the car-park to accompany each other to the building), or interact with cars. In our view it is debatable whether these instances should be regarded as false-positives for an attention-focussing filter system, as it seems reasonable to request operator attention during such events.

The false positive rate is therefore 10%, if we regard all three modes as failure; 5.5% for genuine tracking failures only.

CONCLUSION

We have presented the Owens Tracker, a complete hybrid neural pre-filtering system for tracking pedestrians in a car park, and raising operator attention when unusual activity (defined by pedestrian trajectories) is detected.

The system uses a combination of background differencing to detect moving objects, a specialized multiple tracking algorithm to maintain object records, and a two-part neural novelty detection module to detect novel trajectories defined both by short-term and long-term characteristics.

The system was developed using data from a commercial industrial park. Experiments demonstrate that it is very robust; it detects and discounts the movement of cars, and can handle problems such as car drop-offs, noise, shadows and reflections. It reliably detected virtually all unusual pedestrian trajectories, but raised a number of false positive alarms. Approximately 50% of these were due to tracking failures, indicating that some improvement in that component of the system would be useful; the others are tolerable, given the system's target deployment as an attention-focussing filter. However, even in the current form the system has the potential to dramatically reduce the burden of user monitoring, as only about 20 minutes footage from ten hours was identified as needing operator attention - a 30-fold decrease in effort.

REFERENCES

1. Wallace E, Diffley C, "CCTV: Making it Work," Police Scientific Development Branch of the Home Office (PSDB) publication 14/98, 1998.
2. Makarov A, "Comparison of Background Extraction Based Intrusion Detection Algorithms." IEEE International Conference on Image Processing. Switzerland, pp. 521-524, 1996.
3. Stauffer C, Grimson W E L, "Learning Patterns of Activity Using Real-Time Tracking." IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 22, No. 8, pp. 747-757, 2000.
4. Foresti G L and Roli F, "Learning and Classification of Suspicious Events for Advanced Visual-Based Surveillance." In: "Multimedia Video-Based Surveillance Systems: Requirements, Issues and Solutions," pp. 84-93. Foresti G L, Muhnen, P and Regazzoni C S (Eds.), Kluwer Academic Publishers, 2000.
5. Owens J, Hunter A, Fletcher E, "**A** Fast Model-Free Morphology-Based Object Tracking Algorithm," Proc. British Machine Vision Conference (BMVC 2002). Cardiff, Wales, pp. 767-776, 2002a.
6. Foresti G L, "A Real-Time System for Video Surveillance of Unattended Outdoor Environments", IEEE Transactions on Circuits and Systems for Video Technology. Vol. 8, No. 6, pp. 697-704, 1998.
7. Boghossian B A, Velastin S A, "Image Processing System for Pedestrian Monitoring Using Neural Classification of Normal Motion Patterns", Measurement and Control. Vol. 32, No. 9, pp. 261-264, 1999.
8. Johnson N, Hogg D C, "Learning the Distribution of Object Trajectories for Event Recognition", Image and Vision Computing. Vol. 14, pp. 609-615, 1996.
9. Owens J, Hunter A, "Application of the Self-Organising Map to Trajectory Classification", Proc. 3rd IEEE International Workshop on Visual Surveillance, Dublin, Ireland, pp. 77-83, 2000.
10. Srinivasa N, Ahuja N, "A Topological and Temporal Correlator Network for Spatiotemporal Pattern Learning, Recognition and Recall", IEEE Transactions on Neural Networks. Vol. 10, No. 2, pp. 356-371, 1999.
11. Földiák P, "Learning Invariance from Transformation Sequences", Neural Computation. Vol. 3, pp. 194-200, 1991.
12. Owens J, Hunter A, Fletcher, E. "Novelty Detection in Video Surveillance Using Hierarchical Neural Networks", Proc. International Conference on Artificial Neural Networks (ICANN 2002), Madrid, Spain, pp. 1249-1254, 2002.