Application of the Self-Organising Map to Trajectory Classification

Jonathan Owens* and Andrew Hunter
School of Computing and Engineering Technology
University of Sunderland
England
email: {jonathan.owens, andrew.hunter}@sunderland.ac.uk
*telephone: +44 (0)191 515 3293

Abstract

This paper presents an approach to the problem of automatically classifying events detected by video surveillance systems; specifically, of detecting unusual or suspicious movements. Approaches to this problem typically involve building complex 3D-models in real-world co-ordinates to provide trajectory information for the classifier. In this paper we show that analysis of trajectories may be carried out in a model-free fashion, using self-organising feature map neural networks to learn the characteristics of normal trajectories, and to detect novel ones. Trajectories are represented in 2D image co-ordinates. First and second order motion information is also generated, with moving-average smoothing. This allows novelty detection to be applied on a point-by-point basis in real time, and permits both instantaneous motion and whole trajectory motion to be subjected to novelty detection.

1. Introduction

This paper addresses the issue of detecting suspicious behaviour of pedestrians using a computerised grey-scale video surveillance system. The ultimate objective of the system is to act as a filter, detecting behaviour that appears out of the ordinary and drawing the attention of a human operator to such events.

The visual surveillance task is usually broken down into a modular pipeline, as illustrated in figure 1. This paper is primarily concerned with the implementation of the *Decision Maker* module, although our approach to the earlier stages in the pipeline is also briefly described.

The input to the Decision Maker module is the trajectory of an object, which consists of a series of centroid positions. The trajectory sequence is recoded into a trajectory description vector, which is used as input to a Self Organising Feature Map neural network [4]. The neural network is trained to recognise normal trajectories, using video footage of the target site (a car park) in normal use. During surveillance, the neural network identifies abnormal behaviour (that is, object trajectories unlike those encountered during the training phase) and highlights these. Abnormal trajectories may include those that enter areas that are not usually traversed, or those moving at unusual speeds or moving unusually erratically. A key aspect of the approach is that it is largely "model-free" – there is no explicit modelling of normal or abnormal behaviour, which is instead learned by the neural network. The system may also be used on a point-by-point basis, detecting any unusual structure in the partially complete trajectories.

Preliminary results indicate that the approach

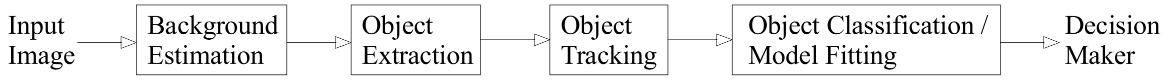

Figure 1. Typical visual surveillance processing pipeline

0-7695-0698-04/00 $10.00 © 2000 IEEE

is able to detect unusual behaviour with an acceptable level of accuracy, although we also find that it erroneously classifies some normal behaviour as suspicious. Further analysis demonstrates that those normal trajectories classified as abnormal are in fact not represented in the original training data, demonstrating that the approach requires comprehensive training data, and probably therefore on-line updating in a practical setting. This sensitivity to training data is a drawback of the model-free approach.

2. Pre-Processing Stages

This section discusses our approach to the Background Estimation, Object Extraction, Object Tracking and Object Classification/Model Fitting stages in the pipeline.

First, moving objects are identified by subtraction from a background image, but changes in lighting condition, movement of foliage and so forth imply that it is difficult to identify a static background; consequently, the background image needs to be adapted over time. We generate the reference image using an algorithm described by Makarov, A. [5]. The update operation performs a kind of temporal low-pass filtering of the input image pixels, where only slow changes that fall below a threshold are inserted into the background. It cannot cope with very rapid background changes, such as clouds moving across the sun. However, it is computationally light enough to be applied at real-time frame rates on a standard PC.

Stauffer, C. and Grimson, W. E. L. [9], describe a more complicated, but more robust, method. Each pixel is modelled by a mixture of Gaussians that are continuously updated. This method allows a multimodal representation of the reference image where, for example, moving foliage may result in a switching between two luminance levels, with both belonging to the background. Skifstad, K. and Jain, A. [8] give two methods for illumination independent object detection; however, these algorithms apply to pairwise image comparisons, rather than to the generation of an explicit reference image.

Object extraction is performed using standard techniques. The pixels in the difference image not belonging to the background are assigned to labelled, connected components. As the objective of this paper is to demonstrate classification of pedestrian motion, cars were rejected at this stage based upon measurements of size and aspect ratio. Very small objects are also rejected, and the properties of the remaining objects, including the centroids, are calculated.

In real-world scenes, a passive object tracker may produce partial, noisy trajectories due to environmental interaction. For example, a person may be partially or completely occluded by walking behind parked vehicles. Object tracking deals with these issues. A common approach is to use predictive tracking. The parameters of the object (e.g. size and location) are predicted for the next frame (using Kalman filters [1, 2, 9] or similar approaches), and the prediction is refined using the observed object. Therefore, even if an object is temporarily occluded, an estimate of its position is still given using the parameter values in the last step. This maintains a smooth trajectory and is able to cope with object fusion/separation when tracking multiple objects.

Our system actually uses a very naive approach (objects are matched to the nearest object in the next frame) which is suitable only for low activity images without significant occlusion. If two or more objects merge, for example, the system will only maintain the tracking of one of the merged objects. However, the technique is simple enough to allow the tracker to be implemented along with the background generator on a standard PC.

Most model-based systems translate the object position into world co-ordinates and track trajectories in that space, whereas our system uses image co-ordinates (the camera is located high above the car park, so that we still get reasonable results). However, our approach to trajectory novelty detection is compatible with the motion data produced by more sophisticated tracking algorithms.

In some cases, an object needs to be classified before having model parameters matched to it. Remagnino, P. et al [6] fit a 3D model to cars and a 2D model to people. The 3D model aids in trajectory classification, by assigning stable orientation and dimensions to the tracked object. A 2D contour is fit to each pedestrian, and although the contour does not include orientation information, the centroid can be reliably tracked even when only part of the person is visible. Stauffer, C. and Grimson, W. E. L. [9] use a binary threshold of the time-averaged aspect ratio of an object to classify it as a pedestrian or vehicle. In this case, classification is a post-processing step that is used to annotate the database of recorded trajectories, rather than guiding a decision as to the model to fit to the object. Foresti, G. L. [1] uses a transformation invariant shape description to classify tracked objects by comparing them with a large database of pre-classified vehicles.

Again, our system does not include these more sophisticated modelling approaches, but is compatible with the trajectory data produced by them.

3. Trajectory Classification

The *Decision Maker* module uses the trajectories generated by the Object Tracker to classify the behaviour observed in the image sequences. We are particularly interested in classifying behaviour that is suspicious, and may be indicative of criminal intent (e.g. pedestrians examining cars, speeding) or just unusual (e.g. parking in a non-parking area).

A variety of approaches have been suggested in the literature. Remagnino, P. et al [7] use Bayesian belief networks. For each object tracked, a belief network is assigned to examine the location, heading, speed and trajectory, using the image evidence to produce the most likely description of the object's behaviour. When the distance between two objects falls below a threshold, a belief network is instantiated to examine the interaction of the two objects, using the behaviour of each object to give an interpretation of the situation. The belief networks provide a textual description of the scene (such as "Pedestrian 2 passing by Vehicle 43") which may be classified by an operator, or a separate automated system.

Alternatively, a trajectory is commonly encoded by sequences of flow vectors, which describe the position and instantaneous velocity of the tracked object, $\mathbf{f} = [x, y, dx, dy]$.

Grimson, W. E. L. et al [2] cluster trajectories based on the flow vector, **f**, and object size. The vectors are clustered by a K-means algorithm. The input data is replaced by the flow vector assigned to the nearest cluster centre, giving a discrete description of the feature vector. The co-occurrence of these descriptor states is then used to classify future sequences as familiar or novel.

Johnson, N. and Hogg, D. [3] describe a model-free method for representing trajectory distributions. Flow vectors, generated from recorded trajectories, are used to train the distribution prototype vectors of a competitive neural network. Sequences of prototype flow vectors are fed to a network of neurons with short-term memory (STM), which retain traces of previous activations, capturing the recent history of each flow vector in the sequence. These time-smoothed vector sequences are used to train another competitive network, which can be used to classify future instances of learned trajectories. As the traces held in the short-term memory are trajectory specific, the activations must be reset to zero during learning to avoid interference between trajectory representations. This implies that, during classification, a separate network must be instantiated for each trajectory under consideration, to prevent the memory traces from interfering in STM.

We present a novel approach to model-free novelty detection, based on converting trajectories to a fixed length trajectory vector, which can be submitted to a Self-Organising Feature Map [4]. This is similar to the approach of Johnson, N. and Hogg, D. [3] in that the neural network in essence learns the distribution of training trajectories, and classifies outlying trajectories as novel. The key difference is that in our network, smoothing is performed as a pre-processing step, resulting in a simpler network structure, which is optimised to work with partial trajectories, allowing real-time novelty detection. In contrast, Johnson and Hogg's model is primarily designed to work with complete trajectories represented in STM.

4. Trajectory Encoding

The information recorded by the object tracker is a sequence of centroid points for each tracked object, as illustrated in figures 2 and 3. A number of video sequences were recorded and divided into two sub-sets. The training sequences were captured during normal use at the test site. The test sequences were a deliberately produced mixture of normal and abnormal sequences. Figure 2 shows an example of a normal trajectory, characterised by relatively smooth, uni-directional motion. Figure 3 gives an example of the kind of erratic, unusual behaviour that we would like the system to classify as abnormal. The trajectory has the unusual characteristics of highly non-uniform velocity, with large direction changes.

For real-time use, the system needs to monitor trajectories as they are generated, rather than waiting until a complete path is created. We therefore seek a scheme that encodes any partial or complete trajectory as a fixed length vector, suitable for novelty detection.

Starting with the flow vector, $\mathbf{f} = [x, y, dx. dy]$, we added second order information, intuitively motivated by the concept that the rate of velocity change could provide additional discrimination between normal and unusual behaviour. For example, relatively low rates of velocity change are seen in normal pedestrian motion, while a

0-7695-0698-04/00 $10.00 © 2000 IEEE

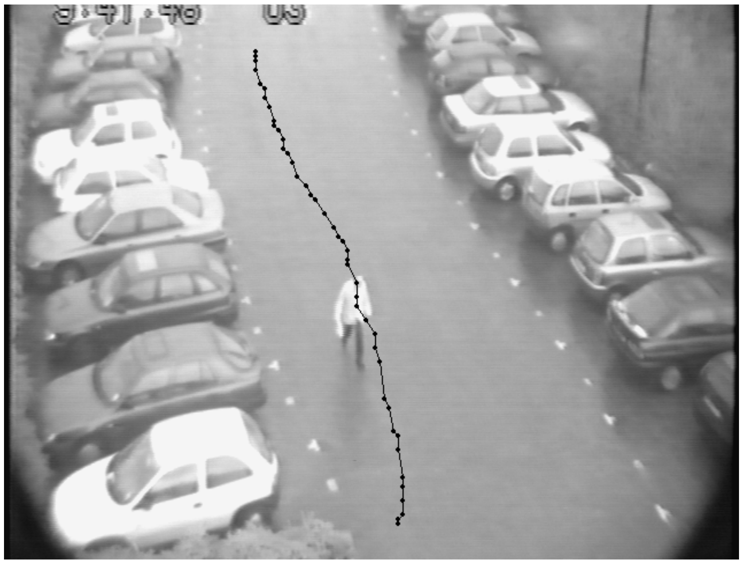

Figure 2. An example of a normal trajectory, overlaid on a frame from the sequence.

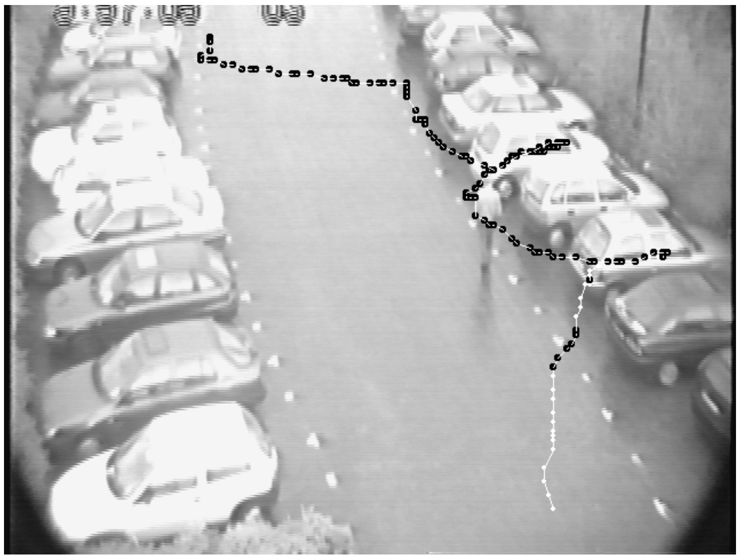

Figure 3. (a) An unusual trajectory. The unusual points are shown in black (see section 6).

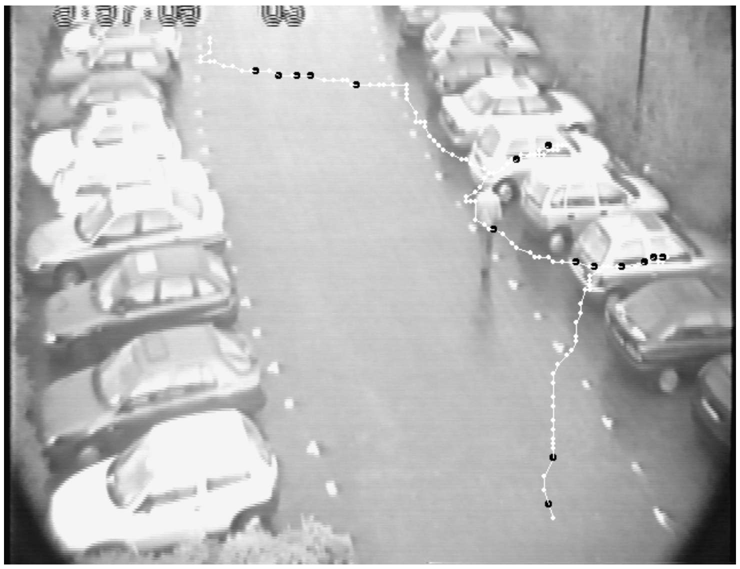

Figure 3. (b) The same trajectory as in (a) with novel points detected using flow vectors only.

pedestrian moving in between parked vehicles shows high rates of velocity change as the path reverses direction. The position, first and second order elements in the vector give the instantaneous characteristics of the object motion, so in an effort to build a vector that contains information regarding previous behaviour of the object, a time-smoothing function was applied to the elements in the vector.

It is possible to construct the vector using all instantaneous elements with a smoothed value for each. However, the feature vector used in the following experiments was made up of a subset of these elements,

$$\mathbf{F} = [\ x,\ y,\ s(x),\ s(y),\ s(dx),\ s(dy),\ s(|d^2x|),\ s(|d^2y|)\ ]$$

where the function s(.) indicates a time smoothed average of the quantity, and the first and second order differences, dx and d^2x, are given by the following:

$$dx = x_t - x_{t-1}, \text{ and}$$
$$d^2x = x_t - 2x_{t-1} + x_{t-2}.$$

The smoothing function s(.) implements a moving average window, and is defined as

$$s_t(x) = (\mu)(s_{t-1}(x)) + (1-\mu)(x)\ ,$$

where μ controls the update rate of the smoothed variable, and is between 0 and 1. The feature vector **F** contains the (x, y) position together with a "short-term memory" of the recent position, and the first and second order motion of the object. Therefore, even if an unusual trajectory falls in a well-traversed location, the feature vector should differ substantially to normal vectors along the dimensions that have a memory of the recent motion of the object (for example, the 'doubling-back' that is seen as the pedestrian in figure 3 moves in between parked vehicles will rarely be seen in normal pedestrian motion). Each individual point in the trajectory is translated into a feature vector, and as the feature vector is of fixed length, it is suitable for classification by the neural network.

5. The Self-Organising Feature Map

The self-organising feature map (SOFM), or Kohonen network, is a two layer neural network that is able to "learn" to represent distributions of the data presented to it.

A schematic of the network is shown in figure 4.

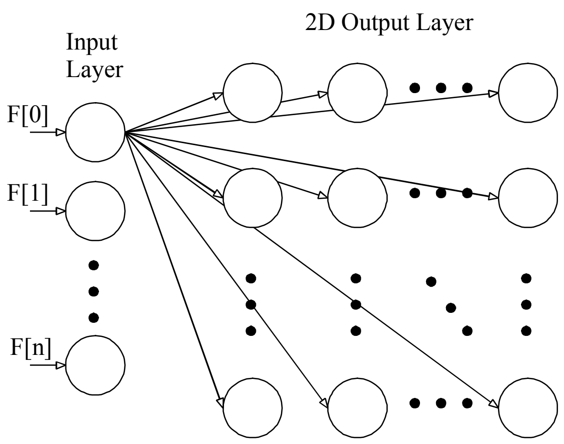

Figure 4.. The basic self-organising map (for clarity, only connections from the first unit in the input layer are shown).

The input layer has one unit for each element of the feature vector. Each unit in the output layer is joined to the input units by a set of connections, which store an exemplar vector. During training, the 'winning' neuron is the one whose weight vector is the smallest Euclidean distance to the input vector.

The weight vector for the winning neuron is updated using the following rule,

$$\mathbf{m}(t+1) = \mathbf{m}(t) + \eta\ [\ \mathbf{x}(t) - \mathbf{m}(t)\ ]\ ,$$

where **x**(t) is the training example, **m**(t) is the current weight vector, **m**(t+1) is the new weight vector and η is a learning rate. In addition to the winning neuron, a number of units in its neighbourhood (i.e. close in position on the two dimensional topological grid) are also updated using the above rule.

Training progresses through a number of epochs, during which the learning rate and neighbourhood are progressively reduced. Training falls roughly into two phases: a rough ordering phase, during which neurons in discrete areas of the network learn to correspond to coarse clusters in the data, and a fine tuning phase during which individual neurons adjust to reflect fine distinctions.

Self-organising features maps are widely used for novelty detection [10]. Advantages over clustering techniques include: robustness, the ability to be used in on-line mode (i.e. learning additional new normal cases when they arise), and the ability to use the topological map for visualisation purposes (cluster discovery and identification).

During surveillance, the feature vector is submitted to the SOFM, the winning neuron is identified, and the Euclidean distance between the feature vector and the winning neuron's exemplar vector is calculated. If this distance exceeds a threshold value then the feature vector is considered novel, and the trajectory is identified as suspicious.

The threshold value is calculated at the end of the training process, by determining the maximum distance of the training vectors from winning neurons. We set the threshold to half this value. Adjusting the threshold alters the balance between false positive and false negative errors. Maintaining a threshold which would classify *all* training data correctly may lead to genuinely unusual points being classified as normal. This occurs because, after training, there may be normal points that have not become well represented by the network and are a significant distance from the closest neuron centre. Enlarging the accept threshold to encompass these outlying points would expand the boundary of normality into the regions that we would typically like to classify as unusual. Setting the threshold to a fraction of the maximum value will ensure that the network maintains sensitivity to new novel data.

6. Experimental Results

The feature vector **F** was generated from the trajectory data with the following μ values for the smoothing functions:

- Position, $\mu = 0.9$
- First order difference, $\mu = 0.96$
- Second order difference, $\mu = 0.96$

The higher update rate for the positional information was used to eliminate the noise generated by the tracker (cf. figs. 2 & 3) whilst retaining the essential shape of the trajectory. The training data consisted of 206 normal trajectories, containing 7,482 points. The test set included 23 unusual trajectories and 16 normal trajectories, containing 2,174 and 694 points respectively.

The self-organising map had eight input units, connected to a grid of 11x10 output units. The results can be summarised as follows:

Correct classifications:
36/39 = 92%
Unusual trajectories correctly classified:
23/23 = 100%
Normal trajectories correctly classified:
13/16 = 81%

It should be noted that a complete trajectory was classified as unusual if it contained two or more points classified as unusual.

This classification rate of 92% is good given that the training set was rather small. The effect of the small training set can be illustrated by examining more closely one of the misclassified normal trajectories.

One of the three normal trajectories classified as unusual, traced a pedestrian moving from the car in the bottom left of the image to the car-park exit at the bottom of the image (cf. fig. 3). Upon examination of the training set, it was found that a trajectory in this location and direction was only represented once in the training set. Given the nature of SOFM training, the representation of a single instance of a trajectory can be overwhelmed by the more numerous training examples. This is indicative of the largest problem with the model-free approach to novelty detection – if the training data is not sufficiently comprehensive, then novel but actually acceptable behaviour will be classified as suspicious. This implies a requirement to update the neural network online with newly-detected normal trajectories when they occur.

What features are being used by the system when detecting novel behaviour? To examine more closely the mechanism of classification, a separate network was trained using the flow vectors, **f** = [x, y, dx, dy], with no time smoothing or second order information.

Figure 3 shows one of the unusual trajectories in the test set. Figure 3(b) shows the sixteen points (out of 158 points) detected as novel when only the flow vectors were used. As most of the trajectories in the training set typically traced vertical motion, the flow vectors describing lateral movement were detected as novel, and in particular, lateral motion of large magnitude (fig. 3(b)). In other words, novel points are instances in which a trajectory segment was either in an unfamiliar orientation, or of an unfamiliar magnitude (i.e. unusual speed). However, global features of the trajectory were not well represented using the flow vectors. For example, the erratic direction changes in between the parked vehicles of figure 3 was not detected to a significant degree.

To capture the global properties of object motion, we added second order information and a moving average function to the feature elements, as described in section 4. Figure 3(a) shows the novel points (marked in black) as detected using the full feature vector. A total of 131 out of 158 points were classified as novel for this trajectory. The first two unusual points occur after the pedestrian deviates slightly from his original course, but at this point the trajectory is not overly suspicious. However, more points are classified as unusual when the pedestrian doubles back on his path while moving between the parked vehicles. It should be noted that the absolute value of the second order changes was used so that the smoothed value would be cumulative, avoiding the possibility that an erratic zigzag motion could cancel out over time and produce a smoothed value that tended to zero.

As shown in figure 3(b), if instantaneous motion alone is used for classification, most individual flow vectors will have been encountered in the training set and will therefore be seen as normal. However, the sequence in which the flow vectors occur, the global properties of the object motion, should be considered to properly classify the trajectory.

7. Conclusion

We have demonstrated that trajectory classification is possible using a model-free neural network approach, provided that the training data is sufficiently representative of the full range of normal behaviour. Applying a moving average window to the motion information allows the classifier to examine global information, such as the object's present motion in relation to motion in the past. This allows the system to detect globally unusual behaviour, such as erratic direction changes occurring at an otherwise normal speed. Moreover, the system is computationally simple enough to be used in real-time on a standard PC.

The major limitation of the approach is that prior knowledge is difficult to build into self-organising networks, and the system is unable to differentiate between normal paths that have not been previously seen and genuinely suspicious behaviour.

Future work will involve integration of the Trajectory Classifier with existing systems we have developed to update a neural network based novelty detector on-line, as "unusual" events are processed and labelled as normal by the user.

In addition, we may attempt to add a predictive element to the trajectory classification. This would entail using a partial trajectory to generate a pool of potential normal trajectories that would satisfactorily complete the path being traced. This set of normal paths could be refined using evidence obtained from the actual object motion. Such a scheme would allow more robust event detection. For example, the pool of normal trajectories available to the system could be subject to short-term heuristic changes; a trajectory considered normal during working hours could be removed from the normal set during the hours of darkness. In other words, "normality" is subject to the environment. The pool of normal trajectories could also evolve over the long-term, with trajectories that have not been used for some time being removed from the normality set.

8. References

1. Foresti, G. L. "A Real-Time System for Video Surveillance of Unattended Outdoor Environments," IEEE Trans. Circuits and Systems for Video Technology, vol. 8, no. 6, 1998.
2. Grimson, W. E. L., Stauffer, C., Romano, R. and Lee, L. "Using Adaptive Tracking to Classify and Monitor Activities in a Site," Proc. of CVPR, 1998.
3. Johnson, N. and Hogg, D. "Learning the Distribution of Object Trajectories for Event Recognition," Proc. BMVC, vol. 2, 1995.
4. Kohonen, T. "Self-Organising Maps," Springer-Verlag, 1995.
5. Makarov, A. "Comparison of Background Extraction Based Intrusion Detection Algorithms," IEEE Int. Conf. Image Processing, 1996.
6. Remagnino, P., Bumberg, A., Grove, T., Hogg, D., Tan, T., Worral, A. and Baker, K. "An Integrated Traffic and Pedestrian Model-Based Vision System," Proc. BMVC, vol. 2, 1997.
7. Remagnino, P., Tan, T., and Baker, K. "Agent Orientated Annotation in Model Based Visual Surveillance," Proc. of ICCV, 1998.
8. Skifstad, K. and Jain, A. "Illumination Independent Change Detection for Real World Sequences," Computer Vision, Graphics, and Image Processing, vol. 46, 1989.
9. Stauffer, C. and Grimson, W. E. L. "Adaptive Background Mixture Models for Real-Time Tracking," Proc. of CVPR, 1999.
10. Taylor, O., Tait, J. and MacIntyre, J. "Improved Classification for a Data Fusing Kohonen Self-Organising Map Using a Dynamic Thresholding Technique," Proc. of IJCAI, vol. 2, 1999.