A Highly Efficient Block-based Dynamic Background Model

David Russell

Department of Computer Science
Queen Mary, University of London
London E1 4NS, UK

Shaogang Gong

Department of Computer Science
Queen Mary, University of London
London E1 4NS, UK

Abstract

A block-based dynamic background modelling technique featuring incremental update is presented, with the aim of increasing the sensitivity of background detection by considering the local connectivity of pixels. An accumulative model is maintained on-line by a heuristic algorithm to build up a statistical representation of a dynamic scene from a fixed camera view for foreground-background segmentation. The model is compared with a block-based technique employing cooccurrence [3], and shown to be favourable in terms of both performance and conceptual simplicity. Using predominantly simple integer operations, the algorithm is highly amenable to direct implementation in hardware.

1 Introduction

The vast majority of computer vision applications employ some type of background model as a first stage in order to facilitate segmentation of an image sequence into *interesting* foreground objects and *typical* background areas. The distinction between the two is not absolute, but statistical scene modelling permits a compromise which is evidently useful in a variety of circumstances [2] [5] [4]. A noteworthy application example is surveillance [11], in which human operator attention may be directed towards unusual events highlighting potentially threatening or otherwise important situations.

Models which support *dynamic backgrounds* are increasingly prevalent, whereby stochastic or cyclic motion of objects is seamlessly assimilated into the background in order to prevent needless *clutter* from burdening subsequent higher levels of image interpretation.

By far the most popular techniques for achieving this are based on *per pixel* models, which maintain a separate statistical model for each pixel in the image. In this case, such a model is learned entirely from the history of its associated pixel, and a measure of the probability of a new pixel value from the same location being background is derived solely from that model. The literature describing pixel-based models is extensive [11] [1] [8], and it is immediately apparent that this is currently the technique of choice as the front-end of a large proportion of computer vision applications. Typically a mixture of Gaussians (~ 10) is used to represent multiple hypotheses regarding the value of a pixel [1], although useful results have also been obtained using linear prediction [10], and kernel estimators derived directly from training data [13]. Conceptual simplicity and effectiveness of the method are deemed to outweigh the heavy computational load associated with applying these techniques to an image of realistic size.

At the other extreme, subspace analysis [4] considers the connectivity of every image pixel with every other with a view to building a constrained model representing the principle axes of variation in the training data. Use of the low-dimensional method [12] permits efficient eigendecomposition of a covariance matrix related linearly to the size of the input vector (the number of image pixels) rather than to its square. Subspace analysis for background modelling is less widely covered in the literature, but has been used with success [2]. An incremental version is detailed in [4], featuring robust update whereby new image vectors are validated using the model itself as a prototype.

However, the erroneous detection rate can potentially be quite high due to the intensity of a foreground pixel falling by chance within the range of feature space defined to be background. Similarly, a background pixel might sporadically assume a value which is not normally associated with the background. In order to improve detection reliability by exploiting the *spatial correlation* between pixels, one could consider that supporting evidence from nearby pixels be used to corroborate the foreground-background classification for any given pixel. This may also be viewed as the process of considering *blocks* of pixels simultaneously when deciding whether they belong *en masse* to an area of feature space designated as background.

Comparatively little is to be found in the literature regarding block-based background models, although reasonable results have been achieved in a few notable cases. A block-based method relying on *texture* is described in [6],

0-7803-9385-6/05/$20.00 ©2005 IEEE

whereby an image block is characterized as a weighted sum of histograms derived from the Local Binary Pattern (LBP) texture measure. One of the advantages of LBP is its invariance to absolute intensity and contrast. An application using a block-based background is described in [5] in which persons in an outdoor swimming pool are monitored for safety reasons. Here, the refraction of randomly disturbed water provides an excellent challenge for statistical modelling. Operating in CIE L*a*b colourspace, an online algorithm maintains a Moving Average (MA) value for each block against which new images are measured using the L_1 norm. Validation by comparison with its 8 nearest neighbours improves classification reliability in a way not dissimilar to the cooccurrence method described below and referenced in [3]. Relative intensity distribution among pixels in a block is characterized in [7] by introducing the Normalized Vector Distance (NVD), essentially a measure of angle between image vectors. A spatially modulated variant of this is claimed to overcome instability of the NVD at low intensity levels by considering the relationship between sub-blocks within a block. Introduction of the Temporal NVD Cooccurrence Matrix (TNVDCM) permits classification of all blocks into one of five categories according to changes in the NVD throughout the training data, leading to five different types of background subtraction algorithms. In this way, periodic, static and random temporal behaviour patterns may be optimally dealt with on a *per block* basis.

With a view to improving background detection reliability by considering effects in adjacent image blocks, work in [3] introduces the concept of *cooccurrence*. A set of training images is split into square blocks and dimensionally reduced by Eigen Value Decomposition (EVD) to form a database. A new image is similarly split and projected into the eigenspaces on a block by block basis. The probability (P_1) of a block in the new image being background is calculated by considering its proximity to the L nearest images in the database, assuming an isotropic Gaussian distribution among the L images ($L \simeq 15$). Interpolation between the values of any neighbouring block in the *same* L database images permits an additional probability (P_2) for the neighbouring block to be calculated. Thus the overall probability for a block can be gauged as some function of its own P_1 and the eight P_2 values derived from its nearest neighbours.

The intention herein is to define an *adaptive* heuristic algorithm to perform background modelling using a block-based approach. Functionality of the model will be demonstrated utilising a challenging real-world outdoor image sequence. As a yardstick by which to gauge the algorithm's utility, it is compared against the cooccurrence algorithm detailed in [3]. It should be noted at this point that whilst the latter is concerned with the cooccurrence of adjacent blocks, the focus of this paper is consideration of cooccurrence between the pixels *within* a block.

2 Efficient Block Matching

2.1 Problem Definition

The effectiveness of a background modelling algorithm is determined by the level of reliability with which it manages to distinguish between foreground and background pixels. A crude algorithm is likely to make many misclassifications which ultimately have to be filtered or corrected by higher level algorithms in the vision process, for example the connected component algorithm [9]. Thus it is of paramount importance to maximize the accuracy of the background model.

The feature space of a pixel is generally either unidimensional in the case of grey-scale images, or three-dimensional for colour images. Although the space may be divided into regions representing foreground and background, there are almost certain to be areas which may belong to either category. Increasing the dimensionality of a pixel's feature space helps to separate these areas of confusion. However, it is difficult to imagine how to increase a pixel's dimensionality beyond 3 when constrained by conventional camera equipment, and herein lies the limitation of per-pixel background models as typified by the Gaussian Mixture technique.

A possible way forward is to consider a feature as consisting of a group of pixels, perhaps a block. In this way, the pattern of the pixel colours or intensities in a block is used to discriminate between foreground and background.

The limiting case of this is when the connectivity between all pixels in an image is considered, and this is the basis of modelling techniques relying on covariance matrix decomposition and subspace analysis. Effective though this may be, for large images and high frame rates, the computational load becomes at the very least cumbersome, and rather awkward to cast in the framework of a parallel processing solution.

2.2 Algorithm Description

In this paper, the possibility of using small blocks of pixels in colour space as features is considered. The proposed algorithm involves maintaining a fixed-size database of typical examples of each block, against which new examples of the block may be validated. If a new block is distant from all retained examples by more than a certain threshold, it is deemed to be foreground in the current image. Such new blocks may qualify to become part of the database themselves should close enough examples of them persistently reoccur over a number of frames. Acceptance of new image blocks is at the expense of examples which have not occurred sufficiently frequently over more recent frames.

Thus, overall, parts of a scene which persist over time are subsumed by the model, enabling them to be classified

as background, while conversely, non-reoccurrence of objects causes them to eventually be lost by the model's memory. Although hardly optimal in any sense, such behaviour provides at least a first step towards a simple data-driven block-based background model.

Images are considered divided into a grid of square $N \times N$ pixel blocks, as shown in Figure 1. All processing is carried out on a per-block basis, with no interaction between adjacent blocks, thus no block level cooccurrence inference is drawn.

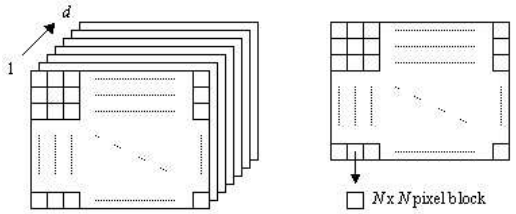

Figure 1: Database and input image divided into blocks. Note that all block stacks are manipulated independently according to the algorithm in Figure 2.

A database D is maintained, containing d samples for each block, which may be imagined as a stack numbered from 1 (highest priority) at the top, to d (lowest priority) at the bottom, as shown in Figure 2. All blocks are initialized to zero before learning commences.

When a new image arrives, each block is rasterized and then normalized to form a vector X which is subsequently compared with all blocks in the top half of the stack using Euclidean distance as the metric:

$$x_u = \min_j \sqrt{(D_{(u,j)} - X_u)^T (D_{(u,j)} - X_u)}$$

$$j = \{1 \cdots d/2\}$$

where u iterates over all blocks in the image. The block is judged background if the smallest distance is less than a threshold TS_u. This forms the block-wise binary segmentation mask M for the current new image:

$$M_u = \begin{cases} 0 & \text{if } x_u \leq TS_u \\ 1 & \text{if } x_u > TS_u \end{cases}$$

Using the same metric and mechanism, the new block is then compared with *all* images in the stack $\{1 \cdots d\}$. If the smallest distance is greater than a threshold TP_u, the new block is inserted at a point p in the stack, typically a quarter of the way from the bottom (position $d/4$). All blocks from p to d are moved down and that in the final position d is discarded, as shown in Figure 3.

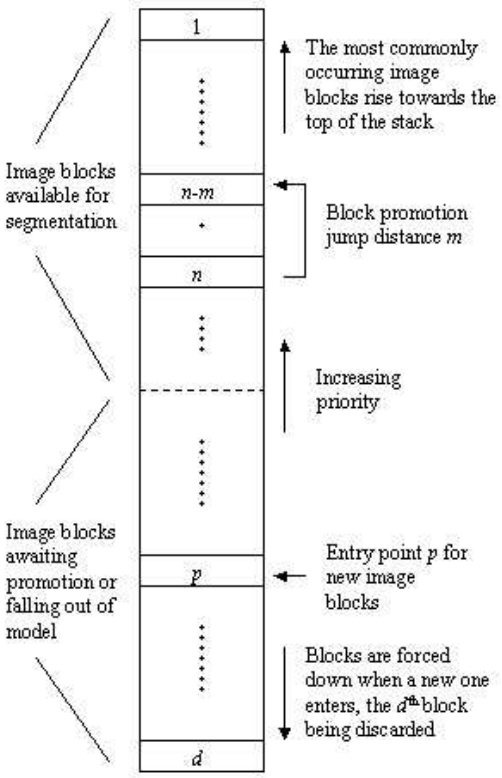

Figure 2: Operation of the stack for a single block in the block matching algorithm.

If the block is sufficiently close (i.e. less than TP_u) to its nearest match in the stack, the new block is discarded and that block to which it was closest is promoted up the stack by m positions, where $m = 1$ in current considerations. Image blocks at and below the new position in priority are shifted down to make way for it.

The thresholds $Tn = \{TS, TP\}$ are derived on a per block basis by filtering a multiple β_n of x_u with a time-constant TC:

$$Tn_{u,t} = \alpha Tn_{u,t-1} + (1 - \alpha)\beta_n x_{u,t}$$

where t represents the timestep and $\alpha = e^{\frac{-1}{TC}}$. In this way, the thresholds track the "chaos" experienced by a particular block.

Overall, the effect is to promote blocks which keep reoccurring into the top half of the stack where they can be used for segmentation, at the expense of those which appear less often. However, new blocks must be persistent over a number of frames before becoming available as valid choices for background, because of the chance that they represent a foreground object. Also, potentially valid background blocks should remain in the system for some time before being discarded due to lack of sufficient recent occurrence.

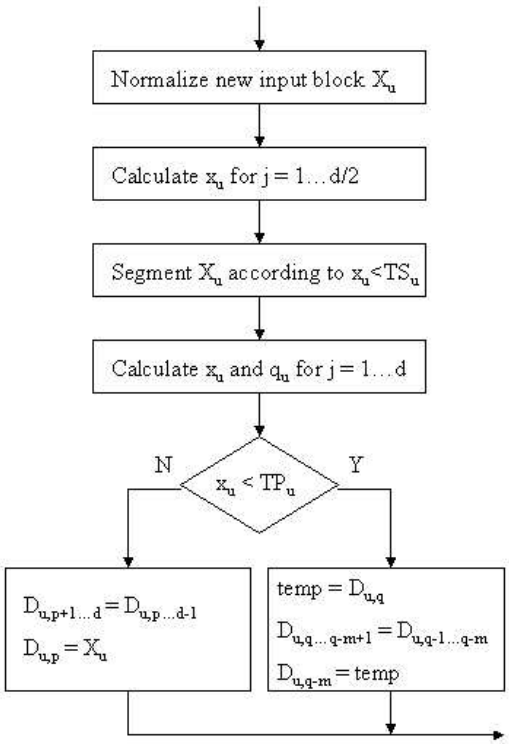

Figure 3: Flowchart for stack manipulation for one block. Note that q_u is the position in D_u at which the best match occurs. Ellipses indicate moves of a sequence of blocks.

Of course the division in the stack between blocks available for segmentation and those not is arbitrarily positioned at the halfway point for the sake of simplicity.

A consequence of block-based techniques in general is the loss of resolution in the final segmentation mask. Although this is directly dependent on block size, some resolution can be regained by defining blocks to be overlapped. As described in [3], resolution may be doubled along both axes by this technique.

3 Experimental Results

For the experiments, continuously captured video footage of a lively university courtyard scene was chosen. A challenging scene, it contains swaying trees, a periodically active fountain and various people moving across the courtyard in different directions, whilst in the distance, traffic can be seen to pass along a main road. The colour images are JPEG compressed from a resolution of 720×576, and the test sequence has 2982 frames, lasting approximately 2 minutes at 25fps.

The proposed algorithm was applied to the sequence using an overall database size of $d = 200$. The block size was 8×8 pixels, creating a grid of 0×72 square blocks over the image area. Time-constant $TC = 500$, $\beta_{TS} = 1.5$, and $\beta_{TP} = 2$.

The algorithm was evaluated against the block-based cooccurrence method presented in [3] using all the same image and block size parameters, and 50 frames of training data not subsequently used for testing. Thresholds in the cooccurrence algorithm were set empirically to yield visibly useful segmentation. In this situation, threshold values chosen are almost always a compromise between clear segmentation of wanted foreground, and accidental foreground classification of objects that ought to be considered background.

The experimental results are shown in Figures 4 and 5 as a comparison between the performance of the two algorithms. They clearly illustrate the superiority of Block Matching when it comes to incorporating the trees into the background model. Likewise the periodic fountain activity is also quite well handled. For the segmentation of moving people and traffic, the performance from both algorithms is rather similar, as depicted in Figure 5.

Although not explicitly shown, the block matching algorithm does also suffer from the effect of incorporating new objects that become stationary in the scene rather too readily. Persistence over a longer time period would be desirable, suggesting an increase in the ratio d/m, although m is already 1. A more advanced algorithm for qualifying new blocks is almost certainly required here.

In the case of the cooccurrence algorithm, the strange patch of foreground on the building entry steps in both examples of Figure 4 is due to an object present in the training data. This highlights the point that, unlike the block matching algorithm, this version of the cooccurrence algorithm does *not* learn incrementally.

Processing time was about 25 seconds per image for both block matching and cooccurrence algorithms running in MATLAB on a 1.8GHz desktop PC. The size of the image database critically affects both algorithms, since much of the computation time is spent evaluating the merit of the match between a new image and each entry. The cooccurrence algorithm is also crucially affected by the dimension of the eigenspace that each image block is projected into, and also the number of nearest images L involved in the local linear interpolation. In this case, the 64 dimensional pixel block is reduced to an 8 dimensional eigenspace, and the savings in storage and comparison time are quite significant for this reason. On the other hand, the interpolation, for which $L = 10$ here, involves matrix inversion of an $L \times L$ matrix, and consequently represents a significant speed penalty for the per block calculation.

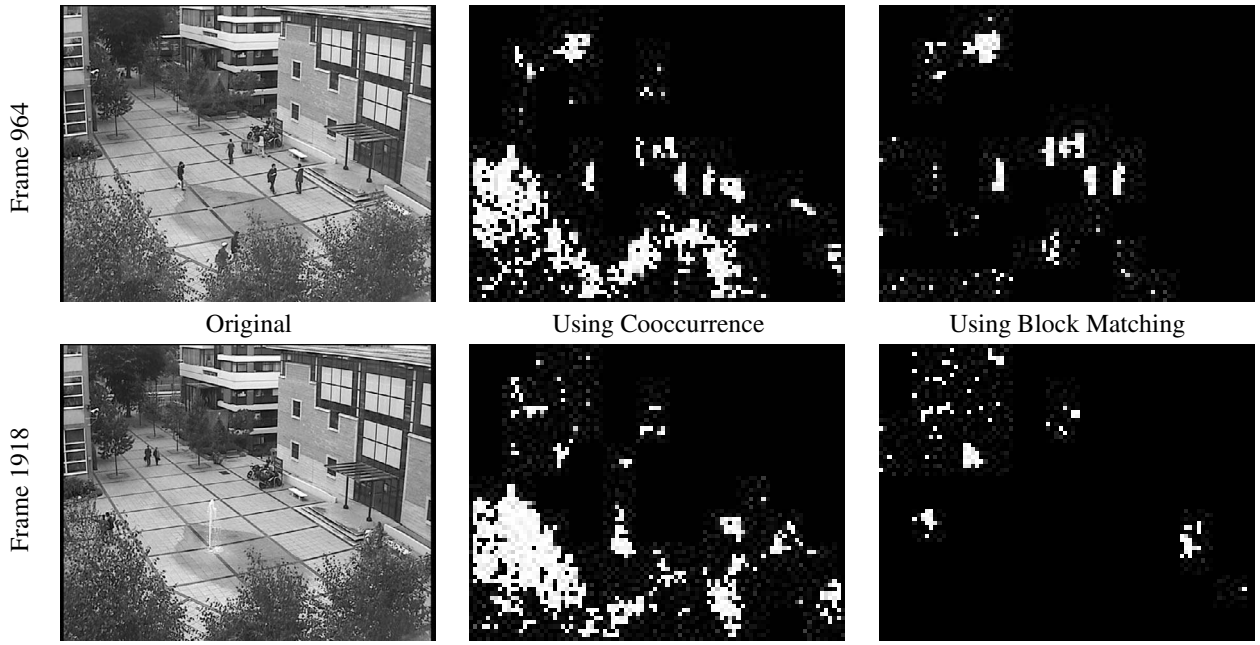

Figure 4: Comparison of Block Matching and Cooccurrence algorithms showing segmentation masks using two different example frames from a test sequence. The Block Matching algorithm models the tree movements much more effectively, leaving them mostly as background. The fountain activity was not covered in the Cooccurrence algorithm's training data, thus explaining why it is portrayed as foreground.

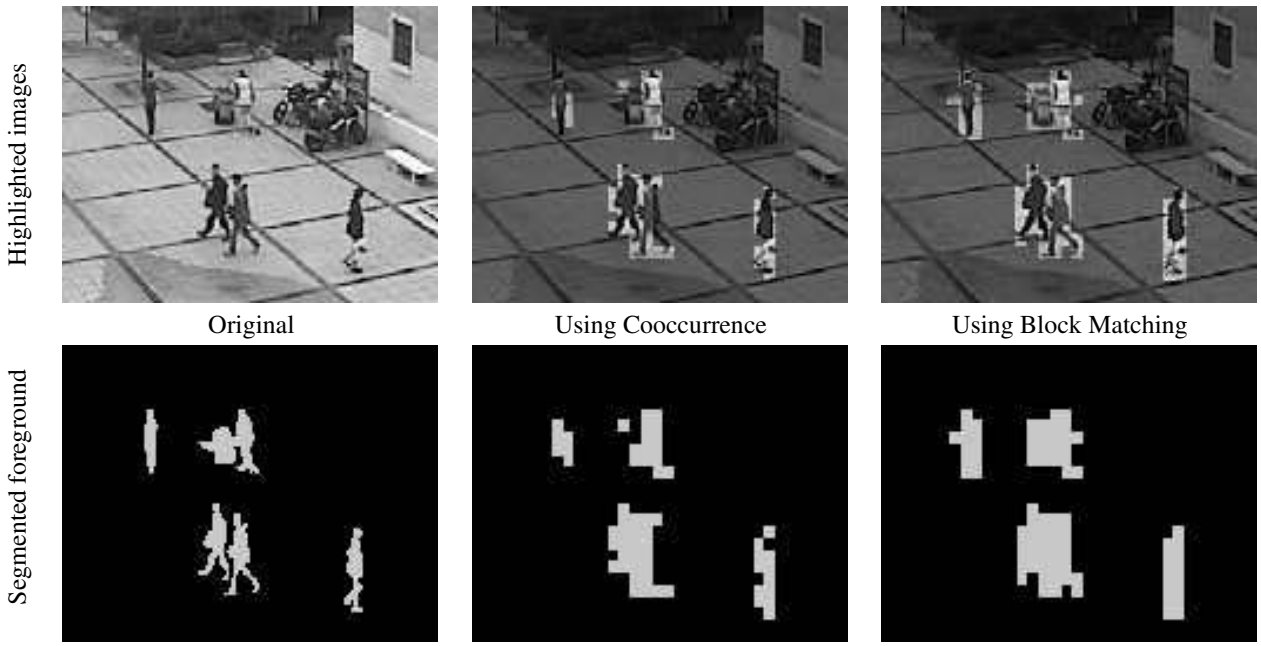

Figure 5: Close up of frame 1066 highlighting the similarity in foreground segmentation performance for the two methods for this type of content. Although the block matching algorithm does exhibit a slightly more *connected* segmentation, this may just be due to choice of threshold. Given comparable segmentation accuracy, the key advantage of the block matching algorithm is its efficiency, potentially 25 times faster than the elaborate cooccurrence model.

4 Discussion

A new block-based incremental background model has been defined and evaluated against an existing block-based technique utilising cooccurrence. The various parameters of the two systems were arranged such that processing time per frame was roughly equal. The block matching algorithm requires considerably more memory since it doesn't have the benefit of working in eigenspace. But by the same token, the cooccurrence algorithm requires considerable floating point calculation in order to maintain that eigenspace model.

The block matching algorithm, although in some sense *intensive*, uses predominantly simple integer operations, and thus lends itself well to direct implementation in hardware. Given single-cycle pipelined multiply, add, subtract, and memory access operations, it is conservatively estimated that for the 0×72 block images used for the experiment, a throughput of 25fps could be achieved with a clock frequency of 800MHz and very modest silicon resources, assuming a memory data bus width of 32 bits. Certainly the stack search and manipulation operations render the block matching approach heavily memory I/O bound. The block normalization requires a division operation, and the threshold update requires a square root, but there is only one each of these per block per frame, and thus they are not seen as limiting factors here.

By comparison, the more elaborate cooccurrence algorithm requires extensive floating point arithmetic. If it is assumed that a twenty-five fold speed increase might be achievable by coding it efficiently in a fully compiled high-level language such as C, rather than MATLAB, still a frame rate of only one per second could be achieved on a desktop PC. Thus it seems clear that given the right hardware platform, the block matching algorithm offers a speed advantage of at least a factor of 25. A possible hardware scenario might integrate the necessary devices, a programmable gate array and synchronous DRAM, into a video capture card. In this way, the host PC could be relieved of the burden of the computationally expensive background modelling task, perhaps to concentrate on higher cognitive processes.

Future research might benefit from considering a block-based model incorporating some of the advantages of both the block matching and cooccurrence algorithms. It is believed that there is an important future for cooccurrence techniques in general, since after all, the newly described algorithm is already really just considering the cooccurrence of the pixels in a single block.

References

[1] Chris Stauffer and W.E.L. Grimson, "Adaptive background mixture models for realtime tracking", In *Proceedings of the CVPR*, pp. 2:246-252, Fort Collins, Colorado, June 1999.

[2] Nuria Oliver, Barbara Rosario, and Alex Pentland, "A Bayesian computer vision system for modeling human interactions", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831-841, August 2000.

[3] Makito Seki, Toshikazu Wada, Hideto Fujiwara, and Kazuhiko Sumi, "Background Subtraction based on co-occurrence of image variation", In *Proceedings of the CVPR*, Madison, Wisconsin, June 2003.

[4] Yongmin Li, "On incremental and robust subspace learning", *Pattern Recognition*, 37(7):1509-1518, July 2004.

[5] H. L. Eng, K. A. Toh, A. H. Kam, J. Wang and W. Y. Yau, "An automatic drowning detection surveillance system for challenging outdoor pool environments", In *Proceedings of the ICCV*, pp. 532-539, Nice, France, October 2003.

[6] M. Heikkila, M. Pietikainen, and J. Heikkila, "A Texture-based Method for Detecting Moving Objects", In *Proceedings of the 15th BMVC*, pp. 187-196, Kingston, UK, September 2004.

[7] T. Matsuyama, T. Ohya, and H. Habe, "Background subtraction for nonstationary scenes", In *Proceedings of the ACCV*, pp. 662-667, Taiwan, 2000.

[8] Ahmed Elgammal, David Harwood, and Larry Davis, "Non-parametric Model for Background Subtraction", In *Proceedings of the ECCV*, pp. 751-767, Dublin, Ireland, May 2000.

[9] Berthold K. P. Horn, "Robot Vision", The MIT Press, Cambridge, Massachusetts, 1986.

[10] Antoine Monnet, Anurag Mittal, Nikos Paragios, and Visvanathan Ramesh, "Background Modelling and Subtraction of Dynamic Scenes", In *Proceedings of the ICCV* pp. 1305-1312, Nice, France, October 2003.

[11] W.E.L. Grimson, C. Stauffer and R. Romano and L. Lee, "Using adaptive tracking to classify and monitor activities in a site", In *Proceedings of the CVPR*, pp. 22-29, Santa Barbara, California, June 1998.

[12] S. Chandrasekaran, B.S. Manjunath, Y.F. Wang, J. Winkeler and H. Zhang, "An Eigenspace Update Algorithm for Image Analysis", In *Graphical Models and Image Processing*, 59(5):321-332, September 1997.

[13] Anurag Mittal and Nikos Paragios, "Motion-Based Background Subtraction using Adaptive Kernel Density estimation", In *Proceedings of the CVPR*, pp. 270-273, Washington, D.C., June 2004.