An Efficient Background Estimation Algorithm for Embedded Smart Cameras

Vikas Reddy, Conrad Sanderson, Brian C. Lovell, Abbas Bigdeli
NICTA, PO Box 6020, St Lucia, QLD 4067, Australia
The University of Queensland, School of ITEE, QLD 4072, Australia

***Abstract*—Segmentation of foreground objects of interest from an image sequence is an important task in most smart cameras. Background subtraction is a popular and efficient technique used for segmentation. The method assumes that a background model of the scene under analysis is known. However, in many practical circumstances it is unavailable and needs to be estimated from cluttered image sequences. With embedded systems as the target platform, in this paper we propose a sequential technique for background estimation in such conditions, with low computational and memory requirements. The first stage is somewhat similar to that of the recently proposed agglomerative clustering background estimation method, where image sequences are analysed on a block by block basis. For each block location a representative set is maintained which contains distinct blocks obtained along its temporal line. The novelties lie in iteratively filling in background areas by selecting the most appropriate candidate blocks according to the combined frequency responses of extended versions of the candidate block and its neighbourhood. It is assumed that the most appropriate block results in the smoothest response, indirectly enforcing the spatial continuity of structures within a scene. Experiments on real-life surveillance videos demonstrate the advantages of the proposed method.**

I. Introduction

Smart cameras are becoming increasingly popular with the advancements in the field of computer vision and semiconductor technology. They are widely being used in intelligent video surveillance systems for monitoring critical infrastructures such as banks, airports, ports, etc. [1], [2]. However, designing effective vision algorithms on these cameras is a challenge, owing to their limitations in terms of low computational power and memory footprint. One of the primary tasks of these cameras is real-time segmentation, tracking and analysis of foreground objects of interest. Many approaches for detecting and tracking objects are based on background subtraction techniques, where each frame is compared against a background model for foreground object detection.

The majority of background subtraction methods described in the literature adaptively model and update the background for every new input frame [3], [4], [5]. However, most of them presume the training image sequence used to model the background is free from foreground objects. This assumption is often not true in case of uncontrolled environments such as train stations and motorways, where directly obtaining a clear background is almost impossible. Furthermore, in outdoor video surveillance a strong illumination change can render the existing background model ineffective. In such circumstances, it becomes inevitable to estimate the background using cluttered sequences (i.e where parts of the background are occluded). A good background estimate will complement the succeeding background subtraction process, which can result in improved detection of foreground objects.

(i) **(ii)**

Fig. 1: Typical example of estimating the background from a cluttered image sequence: **(i)** input frames cluttered with foreground objects, where only parts of the background are visible; **(ii)** estimated background.

The problem can be paraphrased as follows: given a short image sequence captured from a stationary camera in which the background is occluded by moving foreground objects in every frame of the sequence, the aim is to estimate its background, as illustrated in Fig. 1. This problem is also known in the literature as background initialisation or bootstrapping [4].

Existing background estimation techniques, such as simple median filtering, typically require the storage of all the input frames in memory before estimating the background. This increases memory requirements immensely. In this paper we propose a robust background estimation algorithm that has been specifically designed for efficient implementation on embedded systems. It operates on the input frames sequentially, avoiding the need to store all the frames. It is also computationally less intensive, enabling the system to achieve real-time performance – this aspect is critical in video surveillance applications.

We continue as follows. Section II gives an overview of the existing methods for background estimation. Section III describes the proposed algorithm in detail. Results from experiments on real-life surveillance videos are summarised in Section IV, followed by the main findings in Section V.

II. Previous work

Existing methods to address the cluttered background estimation problem can be broadly classified into three categories: **(i)** pixel-level processing, **(ii)** region-level processing, and **(iii)** a hybrid of the first two. In the first category the simplest techniques are based on applying a median filter on pixels at each location across all the frames. The background is estimated correctly if it is exposed for more than 50% of the time. In [6], the algorithm finds pixel intervals of stable intensity in the image sequence, then heuristically chooses the value of the longest stable interval to most likely represent the background. In [7], an algorithm based on a Bayes' theorem is proposed. For every new pixel it estimates the intensity value to which that pixel has the maximum posterior probability. In [8], the first stage is similar to that of [6], followed by choosing background pixel values whose interval maximises an objective function. All these pixel based techniques perform well when the foreground objects are moving but fail at regions where the time interval of exposure of background is less than that of foreground.

In the second category, the method proposed in [9] does a rough segmentation of input frames into foreground and background regions by working on blocks. To achieve this, each frame is divided into blocks and temporal sum of absolute differences (SAD) of the blocks of successive frames is calculated and a block similarity matrix is formed. The matrix elements that correspond to small SAD values are considered as stationary elements and high SAD values correspond to non-stationary elements. The algorithm works well in most scenarios, however, the spatial correlation of a given block with its neighbouring blocks already filled by the background is not exploited, which can result in the blending of the foreground and background if the objects move slowly or are quasi-stationary for extended periods.

In [10], given an image sequence of T frames, each frame is divided into blocks of size $N \times N$ overlapping by half of their size in both dimensions. These blocks are clustered using single linkage agglomerative clustering along their time-line. In the following step the background is built iteratively by selecting the best continuation block for the current background using principles of visual grouping. The algorithm can have problems with blending of the foreground and background due to slow moving or are quasi-stationary objects. Furthermore, the algorithm is not likely to achieve real-time performance due to its complexity.

In the third category, the algorithm presented in [11] has two stages. The first stage is similar to that of [6], with the second stage estimating the likelihood of background visibility by computing the optical flow of blocks between successive frames. The motion information aids to classify an intensity transition as background to foreground or vice versa. The results are typically good but the usage of optical flow for each pixel makes it computationally intensive. In [12] the problem of estimating the background is viewed as an optimal labelling problem. The method defines an energy function which is minimised to achieve an optimal solution at each pixel location. It consists of *data* and *smoothness* terms. The data term accounts for pixel stationarity and motion boundary consistency while the smoothness term looks for spatial consistency in the neighbourhood. The function is minimised using the α*–expansion* algorithm [13] with suitable modifications. Another similar approach with a different energy function is proposed in [14]. The function is minimised using loopy belief propagation algorithm. Both solutions provide robust estimates, however, their main drawback is large computational complexity to process a small number of input frames. For instance, in [14] the authors report a prototype of the algorithm on Matlab takes about 2.5 minutes to estimate the background from a set of 10 images of QVGA (320×240) resolution.

III. Proposed Algorithm

We propose an algorithm that addresses the problems described in the previous section and has several further advantages. It falls into the region-level processing category, with its first stage being similar to that of [10]. Specifically, image sequences are analysed on a block by block basis. For each block location a representative set is maintained which contains distinct blocks obtained along its temporal line.

The novelties of the proposed algorithm are as follows. **(i)** Unlike the abovementioned techniques, it does not expect all frames of the sequence to be stored in memory simultaneously – instead, it processes frames sequentially which is an ideal design for implementation on embedded systems. **(ii)** Robustness against high frequency image noise. **(iii)** Background areas are iteratively filled by selecting the most appropriate candidate blocks according to the combined frequency responses of extended versions of the candidate block and its neighbourhood. It is assumed that the most appropriate block results in the smoothest response, indirectly exploiting the spatial correlations within small regions of a scene.

A. Overview

In the text below we first provide an overview of the proposed algorithm, followed by a detailed description of its components (Sections III-B to III-D). It is assumed that at each block location the background is revealed at some point in the training sequence for a short interval (at least for two consecutive frames) and the camera is stationary. The algorithm has three stages.

Let the resolution of the greyscale image sequence I be $W \times H$. In the first stage, each frame is divided into blocks of size $N \times N$ pixels. Let I_f be the f-th frame of the training image sequence and let its blocks be denoted by $B_f(i,j)$, for $i = 0, 1, 2, \cdots, (W/N) - 1$, $j = 0, 1, 2, \cdots, (H/N) - 1$ and $f = 1, 2, \cdots, F$, where F is the total number of frames. For convenience, each block $B_f(i,j)$ is vectorised into an N^2 dimensional vector $\mathbf{b}_f(i,j)$. For each block location (i,j), a representative set $\mathbf{R}(i,j)$ is maintained. It contains only unique representative blocks, $\mathbf{r}_k(i,j)$ for $k = 1, 2, \ldots, S$ (with $S \leq F$)

(i) **(ii)** **(iii)** **(iv)** **(v)**

Fig. 2: **(i)** Example frame from an image sequence, **(ii)** initial background (after *Stage 2*, see Listing 1 below), **(iii)** iteration 60, **(iv)** iteration 120, **(v)** reconstructed background.

that were obtained along its temporal line. To determine uniqueness, the similarity of blocks is calculated as described in Section III-B. Let W_k denote the number of occurrences of $\mathbf{r}_k$ in the sequence, i.e., the number of blocks at location (i, j) which are deemed to be the same as $\mathbf{r}_k(i, j)$.

It is assumed that one element of $\mathbf{R}(i, j)$ corresponds to the background block. To ensure blocks from moving objects are not stored, block $\mathbf{b}_f(i, j)$ will be registered as $\mathbf{r}_{k+1}(i, j)$ only if it appears in at least two consecutive frames.

In the second stage, representative sets $\mathbf{R}(i, j)$ having just one block are considered to initialise the corresponding block locations $\mathbf{BG}(i, j)$ in the background $\mathbf{BG}$.

In the third stage, an empty background block is selected and filled with a block from the corresponding representative set $\mathbf{R}(i, j)$. The procedure for selecting the location of an empty background block is described in Section III-C. Each representative block $\mathbf{r}_k(i, j)$ is analysed (along with its already filled neighbourhood) in the frequency domain. The details of the analysis and the selection of the most appropriate representative block is described in Section III-D.

The overall pseudo-code of the algorithm is given in Listing 1 and an example of the algorithm in action is shown in Fig. 2.

Stage 1 - Collection of block representatives

1) $R \leftarrow \emptyset$ (null set)
2) **for** $f = 1$ to F **do**
   a) Split input frame I_f into blocks, each with a size of $N \times N$.
   b) **for each** block $B_f(i, j)$:
      i) Vectorise block $B_f(i, j)$ into $\mathbf{b}_f(i, j)$.
      ii) Find the representative block $\mathbf{r}_m(i, j)$ from the set $\mathbf{R}(i, j) = (\mathbf{r}_k(i, j) | 1 \leq k \leq S)$, matching to $\mathbf{b}_f(i, j)$ based on conditions in Eqns. (1) and (2).
      if ($\mathbf{R}(i, j)$ = $\{\emptyset\}$ or there is no match) **then**
      $k \leftarrow k + 1$.
      Add a new representative block $\mathbf{r}_k(i, j) \leftarrow \mathbf{b}_f(i, j)$ to set $\mathbf{R}(i, j)$ and initialise its weight, $W_k(i, j)$, to 1.
      else
      Update the matched block $\mathbf{r}_m(i, j)$ and its weight $W_m(i, j)$ as:
      $\mathbf{r}_m(i, j) \leftarrow \frac{(\mathbf{r}_m(i,j) W_m(i,j) + \mathbf{b}_f(i,j))}{W_m(i,j)+1}$
      $W_m(i, j) \leftarrow W_m(i, j) + 1$
      end if
   end for each
end for

Stage 2 - Partial background reconstruction

1) $\mathbf{BG} \leftarrow \emptyset$
2) **for each** set $\mathbf{R}(i, j)$
   if (size($\mathbf{R}(i, j)$) $= 1$) **then**
   $\mathbf{BG}(i, j) \leftarrow \mathbf{r}_1(i, j)$.
   end if
end for each

Stage 3 - Estimation of the missing background

while ($\mathbf{BG}$ not filled) **do**
  if $\mathbf{BG}(i, j) = \emptyset$ and has neighbours as specified in Section III-C **then**
  $\mathbf{BG}(i, j) \leftarrow \mathbf{r}_{min}(i, j)$, the block out of set $\mathbf{R}(i, j)$ which yields minimum value of the cost function described in Eqn. (3) (see Section III-D).
  end if
end while

Listing 1: Pseudo-code for the proposed algorithm.

B. Similarity Criteria for Blocks

We assert that two blocks $\mathbf{b}_f(i, j)$ and $\mathbf{r}_k(i, j)$ are similar if the following two constraints are satisfied:

$$\left\{(\mathbf{r}_k(i,j) - \mu_{r_k}(i,j))' \left(\mathbf{b}_f(i,j) - \mu_{b_f}(i,j)\right)\right\} \;/\; \{\sigma_{r_k}\sigma_{b_t}\} > T_1 \tag{1}$$

and

$$\frac{1}{N^2}\sum\nolimits_{n=0}^{N^2-1} |d_{k_n}(i,j)| < T_2 \tag{2}$$

Eqns. (1) and (2) respectively evaluate the correlation coefficient and the mean of absolute differences (MAD) between the two blocks, with the latter constraint ensuring that the blocks are close in N^2 dimensional space. μ_{r_k}, μ_{b_f} and $\sigma_{r_k}, \sigma_{b_f}$ are the mean and standard deviation of the elements of blocks $\mathbf{r}_k$ and $\mathbf{b}_f$ respectively, while $\mathbf{d}_k(i, j) = \mathbf{b}_f(i, j) - \mathbf{r}_k(i, j)$.

T_1 is selected empirically (typically 0.8), to ensure that two visually identical blocks are not treated as being different due to image noise. T_2 is proportional to image noise as is found automatically as follows. Using a short training video, the MAD between co-located blocks of successive frames is calculated. Let the number of frames be L and N_b be the number of blocks per frame. The total MAD points obtained will be $(L-1)N_b$. These points are sorted in ascending order and divided into quartiles. The points lying between quartiles Q_3 and Q_1 are considered. Their mean, $\mu_{\mathrm{Q}_{31}}$ and standard deviation, $\sigma_{\mathrm{Q}_{31}}$, are used to estimate T_2 as $(\mu_{\mathrm{Q}_{31}} + 2\sigma_{\mathrm{Q}_{31}})$. This ensures that low MAD values (close or equal to zero) and high MAD values (arising due to movement of objects) are ignored (i.e. treated as outliers).

We note that both constraints (1) and (2) are necessary. As an example, two vectors $[1, 2, \cdots, 16]$ and $[101, 102, \cdots, 116]$

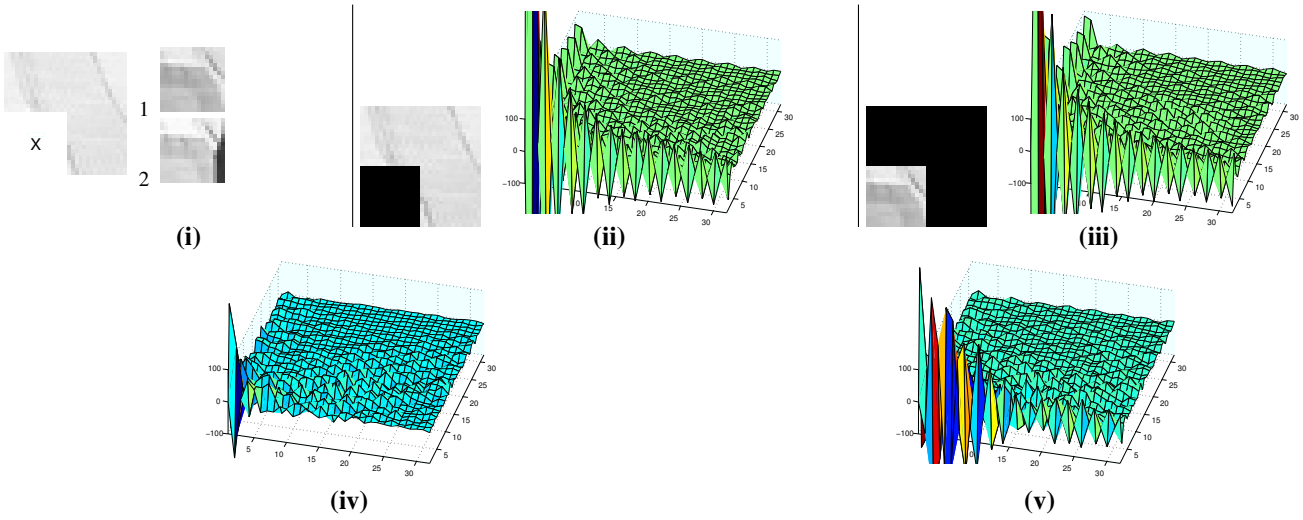


Fig. 5: An example of the processing done in Section III-D. **(i)** a superblock with two candidates for the empty block X. **(ii)** a superblock and its DCT coefficient matrix **C**, where block X is initialised to 0. **(iii)** a superblock and its DCT coefficient matrix $\mathbf{D}_1$, where block X is initialised with candidate 1 and its neighbourhood set to 0. **(iv)** combined spectral distribution $\mathbf{C}+\mathbf{D}_1$ for candidate 1. **(v)** combined spectral distribution $\mathbf{C}+\mathbf{D}_2$ for candidate 2. The smoother distribution for candidate 1 indicates it is a better fit than candidate 2 for block X.

have a perfect correlation of 1 but their MAD will be higher than T_2. On the other hand, if a thin edge of the foreground object is contained in one of the blocks, their MAD may be well within T_2. However, Eqn. (1) will be low enough to indicate the dissimilarity of the blocks. We note that in [10] the similarity criteria is just based on the sum of squared distances between the two blocks.

C. Neighbourhood Selection

The background at an empty block will be estimated only if the background is available in at least 2 neighbouring blocks of its 4-connected neighbours, which are adjacent to each other and also in the diagonal block located between them. For instance, in Fig. 3, block X has blocks D, B, E and G as its 4-connected neighbours. It is assumed that blocks D, B and A have been already filled with the background. Hence, the background at block X can be estimated. Mandating that the background should be available in at least 3 neighbouring blocks located in three different directions with respect to block X ensures that the best match is obtained after evaluating the continuity of the pixels in all possible orientations.

A	***B***	C
D	X	E
F	G	H

Fig. 3: 8-connected block neighbours of block X. Blocks A, B and D have already been filled with the background.

X	***A***
C	***B***

Fig. 4: A superblock consisting of block X and its neighbouring blocks A, B and C, which have been already filled with the background.

Suppose in Fig. 3 if we had background in blocks C and E as well, then block X could have been filled by considering blocks A, B and D or B, C and E. We evaluate the cost function given by Eqn.(3) for both these combinations independently and pick the block that yields the least value. Sometimes not all the three neighbours are available. In such cases we use one of the available 4-connected neighbours which yields minimum value of the cost function defined by Eqn. (3).

D. Cost Function for Candidate Block Selection

Let us call the cluster of 4 blocks shown in Fig. 4 (block X along with its neighbours) as a *superblock*. It is assumed that block X is empty and blocks A, B, C are filled with the background. Let block X have S representatives in its set $\mathbf{R}$ for $k = 1, 2, \cdots, S$ where one of them represents the true background. Choosing the best candidate is accomplished by analysing complementary versions of the superblock in the frequency domain. For the decomposition we chose the Discrete Cosine Transform (DCT) [15] due to its decorrelation properties [16] as well as ease of implementation in hardware.

The two complementary versions of the superblock are constructed as follows. In both versions high frequency components are artificially synthesised.

1) Background data exists in blocks A, B and C. Block X is forced to zero. We take the 2D DCT of the resulting superblock. The transform coefficients are stored in matrix $\mathbf{C}$ of size $M \times M$ $(M = 2N)$ with its elements referred to as $C_{v,u}$. The DC term $C_{0,0}$ is forced to 0 since we are interested in analysing the spatial variations of pixel values.

2) A complementary operation is performed by setting blocks A, B and C to zero and initialising block X with data from $\mathbf{r}_k$, the k-th representative block from set $\mathbf{R}$. The 2D DCT of the resulting superblock is calculated and the transform coefficients are stored in matrix $\mathbf{D}_k$. The DC term $D_{k_{0,0}}$ is set to 0.

The second version of the superblock is constructed for every representative block. A graphical example of the procedure is shown in Fig. 5.

The polarity of the synthesised high frequency components in $\mathbf{C}$ and $\mathbf{D}_k$ will be opposite since the pattern of forcing blocks to 0 is complementary. When the correct pair of coefficient matrices is added (e.g. $\mathbf{C} + \mathbf{D}_1$), the synthesised high frequency components tend to get reduced to a greater extent when compared to other pairs.

It must be noted that setting blocks to 0 in the two versions of the superblocks does not always generate high frequency components. This can occur if the actual pixel values of the blocks themselves are close to 0. To ensure that high frequency components are always synthesised, we analyse the mean μ_k of $\mathbf{r}_k$: if $\mu_k \geq 128$ the blocks are set to 0, otherwise they are set to 255.

As evident in Fig. 5.(ii) and 5.(iii) the synthesised high frequency components are always significant near the origin (along first few rows and columns) and negligible as we move diagonally away from it. Typically, the significant components in those regions correspond to high frequency image noise. Hence, in our cost calculation defined below we consider only the lower 75% of the frequency components after performing a zig-zag scan from the origin.

The cost function to select the best match is given by:

$$\texttt{cost}(k) = \left(\sum_{v=0}^{P-1} \sum_{u=0}^{P-1} \left| C_{(v,u)} + D_{k_{(v,u)}} \right| \right) \lambda_k \quad (3)$$

where $P = ceil\left(\sqrt{M^2 * 0.75}\right)$ and $\lambda_k = (1 + 0.5/(\max(3, W_k)))$, with W_k being the weight of the representative block $\mathbf{r}_k$ (see Section III). The representative block which yields minimum value of the cost function is assumed to be the best continuation of the background.

The weight factor λ_k is used in order to further differentiate the true background block from other candidate blocks that may contain moving foreground objects. However, its variation as a function of the number of occurrences is significantly less as evident in Fig. 6. It is necessary to address the scenario where the true background block is visible for a short interval of time when compared to blocks containing the foreground. For example, in Fig. 2, a sequence consisting of 450 frames was used to estimate its background. The person was standing as shown in Fig. 2(i) for the first 350 frames and eventually walked off during the last 100 frames. The algorithm was able to estimate the background occluded by the standing person. It must be noted that pixel-level processing techniques fail in this case.

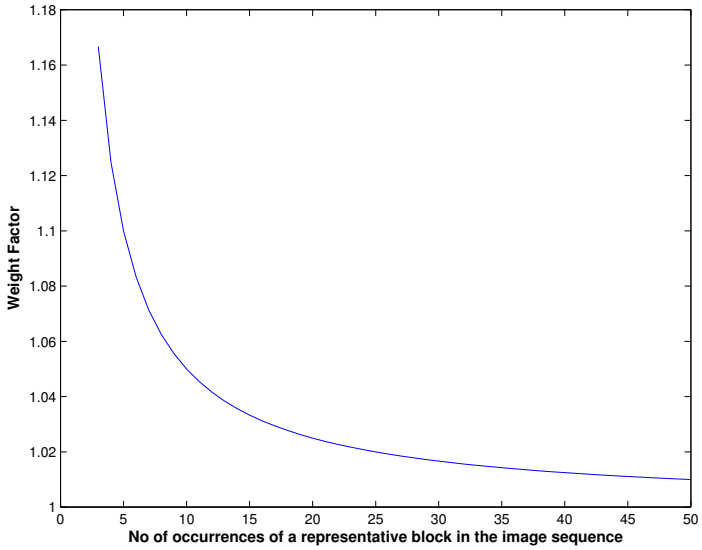


Fig. 6: The variation of the weight factor λ_k as a function of the number of occurrences W_k of a representative block $\mathbf{r}_k$. λ_k is designed such that it mitigates the influence of selecting a foreground block exposed more than the background block.

IV. Experiments

We compared the proposed algorithm with a median filter based approach as well as the agglomerative clustering method presented in [10]. We used a total of 15 surveillance videos: 10 obtained from CAVIAR dataset[1] and 5 sequences from SPEVI dataset[2].

In our experiments the testing was limited to greyscale sequences. The size of each block was set to 16×16. Based on preliminary evaluations, threshold T_1 was empirically set to 0.8 and T_2 was found to vary between 0.5 and 4 when tested on several image sequences (T_1 and T_2 are described in Section III-B). A prototype of the algorithm using Matlab on a 1.6 GHz dual core processor yielded 17 fps when processing images with a resolution of 320×240. We present both subjective and objective analysis of the results.

To evaluate objectively we considered the test criteria described in [11], where the average grey-level error (AGE), total number of error pixels (EPs) and the number of "clustered" error pixels (CEPs) are used. AGE is the average of the difference between the true and estimated backgrounds. If the difference between estimated and true background pixel is greater than a threshold, then it is classified as an EP. We set the threshold to 20, to ensure good quality backgrounds. A CEP is defined as any error pixel whose 4-connected neighbours are also error pixels. As our method is based on region-level processing we calculated only the AGE and CEP. Averaged results from the 15 image sequences are shown in Table I.

Fig. 7 shows example results on three sequences (one from SPEVI and two from CAVIAR) with differing complexities.

[1] http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/

[2] http://www.elec.qmul.ac.uk/staffinfo/andrea/spevi.html

Approach	Median filter	Agglomerative clustering [10]	Proposed method
Avg. grey-level error	4.8	2.7	1.2
Clustered error pixels	982	279	23

TABLE I: Averaged results from experiments on 15 image sequences.

In the first sequence, initially the person is motionless for a certain interval and then walks to his right. In the second sequence, two persons fight and one of them falls on the floor. The last one is complex where two people converse for most of the time while others walk slowly along the corridor. The visual results confirm the objective results in Table I, with the proposed method producing better quality backgrounds than the median filter approach and the agglomerative clustering method. It is evident that agglomerative clustering method results in blending of foreground and background, if foreground objects move slowly.

From our experiments we found the memory footprint to store the representative sets of all the blocks is on average only 5% of the memory required for storing all the frames, making it well suited for embedded systems. This is in contrast to existing algorithms, which typically require the storage of all the frames before processing can begin.

We conducted experiments on image sequences represented in other colour spaces such as RGB and YUV and evaluated the overall cost function as the sum of individual cost functions evaluated on each channel independently. The results were marginally better than those obtained using greyscale input. This is because the algorithm estimates the spatial continuity of structures within a scene which are very well represented in greyscale.

For ease of computation, the 2D-Ordered Hadamard transform [17], [18] can be utilised as an alternative to the 2D-DCT, since its computation involves no multiplication operator. However, for complex background sequences (strong texture), preliminary experiments it did not yield good results (result details not reported here).

V. Main Findings

In this paper we proposed a background estimation algorithm that is able to accurately estimate the background from cluttered surveillance videos containing foreground objects. It has several advantages, such as low memory requirements due to sequential processing of frames and computational efficiency, making the algorithm ideal for implementing on embedded systems. It is also robust against quasi-stationary or slow moving objects as well as image noise.

The performance of the algorithm is invariant to moderate illumination changes. This is because we consider only ac coefficients of the DCT in our cost evaluation defined in Eqn. (3). However, the similarity criteria defined by Eqns (1) and (2) creates multiple representatives for the same visually identical block. Tackling this problem efficiently is part of further research.

Experiments on real-life surveillance videos indicate that the algorithm obtains considerably better results (both objectively and subjectively) than methods based on median filtering and agglomerative clustering.

Acknowledgements

NICTA is funded by the Australian Government via the Department of Broadband, Communications and the Digital Economy, as well as the Australian Research Council through the ICT Centre of Excellence program.

References

[1] W. Wolf, B. Ozer, and T. Lv, "Smart cameras as embedded systems," *Computer*, vol. 35, no. 9, pp. 48–53, 2002.

[2] A. Bigdeli, B. Lovell, and T. Shan, "Smart Cameras: Enabling Technology for Proactive Intelligent CCTV," in *Recent advances in security technology: Proceedings of the 2006 RNSA Security Technology Conference Canberra*, 2006, pp. 365–371.

[3] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-Time Tracking of the Human Body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 780–785, 1997.

[4] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *International Conference on Computer Vision (ICCV)*, vol. 1, 1999, pp. 255–261.

[5] J. Yao and J. Odobez, "Multi-Layer Background Subtraction Based on Color and Texture," in *CVPR Workshop on Visual Surveillance*, Minnesota, US, 2007, pp. 1–8.

[6] W. Long and Y. Yang, "Stationary background generation: An alternative to the difference of two images," *Pattern Recognition*, vol. 23, no. 12, pp. 1351–1359, 1990.

[7] A. Bevilacqua, "A novel background initialization method in visual surveillance," in *IAPR Workshop on Machine Vision Applications*, Nara, Japan, 2002, pp. 614–617.

[8] H. Wang and D. Suter, "A Novel Robust Statistical Method for Background Initialization and Visual Surveillance," *ACCV 2006, Lecture Notes in Computer Science*, vol. 3851/2006, pp. 328–337, 2006.

[9] D. Farin, P. de With, and W. Effelsberg, "Robust background estimation for complex video sequences," in *IEEE International Conference on Image Processing (ICIP)*, vol. 1, 2003, pp. 145–148.

[10] A. Colombari, A. Fusiello, and V. Murino, "Background Initialization in Cluttered Sequences," in *CVPRW*, Washington DC, USA, 2006, pp. 197–202.

[11] D. Gutchess, M. Trajkovic, E. Cohen-Solal, D. Lyons, and A. Jain, "A background model initialization algorithm for video surveillance," in *International Conference on Computer Vision (ICCV)*, vol. 1, 2001, pp. 733–740.

[12] S. Cohen, "Background estimation as a labeling problem," in *International Conference on Computer Vision (ICCV)*, vol. 2, 2005, pp. 1034–1041.

[13] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," in *International Conference on Computer Vision (ICCV)*, vol. 1, 1999, pp. 377–384.

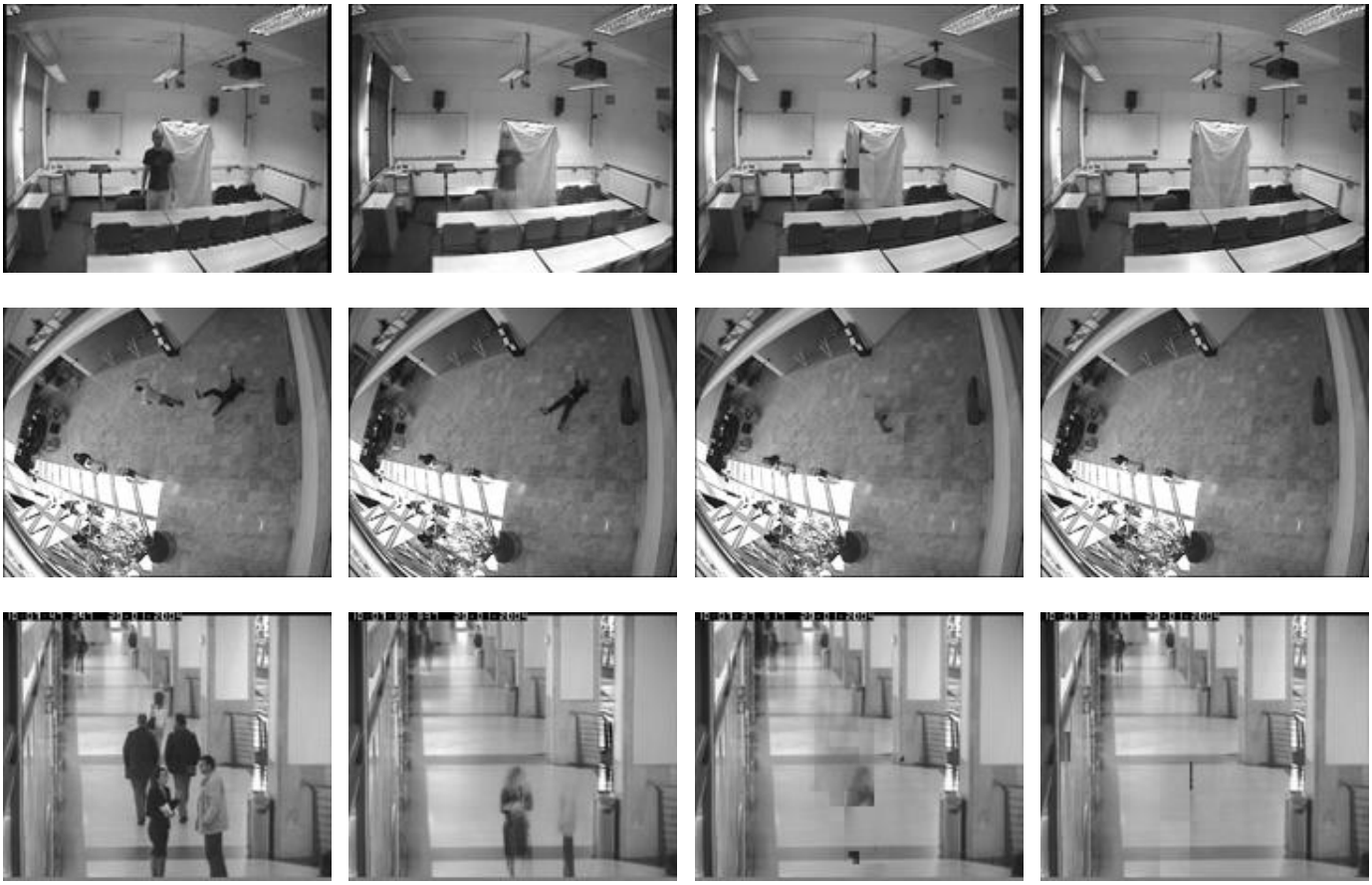
[14] X. Xu and T. Huang, "A Loopy Belief Propagation approach for robust background estimation," in *CVPR*, 2008, pp. 1–7.

[15] N. Ahmed, T. Natarajan, and K. Rao, "Discrete Cosine Transfom," *Transactions on Computers*, vol. 100, no. 23, pp. 90–93, 1974.

[16] C. Sanderson and K. K. Paliwal, "Polynomial features for robust face authentication," in *IEEE International Conference on Image Processing (ICIP)*, vol. 3, 2002, pp. 997–1000.

[17] R. Gonzalez and R. Woods, *Digital image processing*. Prentice Hall, 1992, pp. 141–143.

[18] C. Sanderson, D. Gibbins, and S. Searle, "On statistical approaches to target silhouette classification in difficult conditions," *Digital Signal Processing*, vol. 18, no. 3, pp. 375–390, 2008.



(i) **(ii)** **(iii)** **(iv)**

Fig. 7: **(i)** Example frames from three videos, and the reconstructed background using: **(ii)** median filter, **(iii)** agglomerative clustering [10], **(iv)** proposed method.