International Journal of Advances in Image Processing Techniques– IJIPT
Volume 2: Issue 1 [ISSN : 2372-3998]
Publication Date : 30 April, 2015

Kalman Filter and Projection Histogram based People Counting

Sathish Kumar Shivagurunathan, Aditya Piratla, Jayalakshmi Surendran, Monotosh Das

***Abstract*— This paper presents an algorithm for real time people counting using a stationary camera for outdoor and indoor scenarios. The objective of the algorithm is to count the number of people entering or leaving a particular area by checking for the crossing of a user defined line. The algorithm implemented consists of four modules. The first module deals with image segmentation where in the information obtained from the camera frames are converted into foreground objects using a temporal difference based algorithm, followed by a computation of an undirected optimized graph to establish the relationship between the foreground objects of two consecutive frames. The second module is related to object tracking using a Kalman based tracking mechanism. The third module is the object crossing detection module which discusses three techniques namely block matching, optimized block matching and a simple region crossover. The last module is related to counting based on a projection histogram based method. The advantage of proposed method is in its applicability to bi-directional people counting in outdoor unregulated environments. This is achieved by using a projection histogram based approach on foreground blobs and avoiding the camera calibration step.**

***Keywords*—Video surveillance, People counting, Image processing, Kalman filter application**

I. Introduction

Automated algorithms for video surveillance are gaining significance because of the inefficiency of human monitored systems in terms of cost and effectiveness. This paper discusses an algorithm for real time people counting which can be used in indoor as well as outdoor scenarios. The method discussed borrows largely from [1] and [2]. Problems related to counting in general involve three steps, namely foreground segmentation, tracking, and finally counting.

Foreground segmentation is the extraction of change with respect to a reference model. Foreground segmentation is a very commonly used technique and there are a wide variety of techniques available for extracting foreground. There are two famous approaches for foreground estimation. One is based on computing a model for each and every pixel based on a pixels statistical property while the other one is based on a non-parametric model incorporating the pixel as well as its neighborhood. [3] proposed using a Gaussian model for each and every pixel. [4] proposed a mixture of Gaussians as a single Gaussian could not model outdoor scenarios effectively.

Sathish Kumar Shivagurunathan, Aditya Piratla, Jayalakshmi Surendran, Monotosh Das

Crompton Greaves Global R and D Centre
Mumbai, India

Codebook based method is used in [5] for foreground extraction which follows the former model but has a lower computational cost compared to the Gaussian methods. [1] proposed a non-parametric kernel density estimation technique to model the per pixel background. In [1] a pixel is matched across a region rather than being compared only with the corresponding pixel in the reference.

In this paper, we use the pixel based method discussed in [2]. The method discussed in [2] is simple and computationally efficient in comparison to the other methods mentioned here. Subsequently to establish relationship between foregrounds of two consecutive frames, we use a unidirectional bipartite graph discussed in [1].

Tracking over time is finding corresponding objects in consecutive frames. The difficulties of this task are related to the complexity of the scene and the complexity of the tracked objects. The latter is again related both to the degrees of freedom of individual objects and to their representation. In addition objects may split, merge or disappear altogether because of reasons like improper foreground segmentation and occlusion.

Generally tracking involves prediction based on past data. Prediction is helpful in reducing the search radius for correspondence match. Prediction may be based on a model involving position, velocity and acceleration as proposed in [7]. An alternative approach is to learn probabilistic motion models for prediction prior to operation. A commonly used prediction technique uses Kalman Filtering. [1] proposes such a method using Kalman Filter. The Kalman filter is unfortunately restricted to data which is unimodal in nature, nevertheless since objects in predefined environments move with a certain profile, a unimodal approach may suffice. In this paper, a Kalman based approach is used for tracking.

Counting can be performed after establishing the trajectory of an individual and finding whether a certain criterion like crossing a tripwire is satisfied. The methods for people counting can be classified into two major categories, namely those based on people detection and those based on feature detection. As per the first category, once people have been detected, they can also be counted. In W4 system proposed in [8] shape information is used to identify individuals. Further [9] employs boosted cascades of simple classifiers to detect pedestrians using appearance and motion cues. The main problem with these approaches is related to their limited applicability especially in situations where people walk next to each other and/or occlude each other. The second category of approaches avoids the people detection step and aims at mapping the people-counting problem to robust, proven techniques of statistical computer vision and image processing. These methods typically extract various features based on motion of objects, such as edge density [10], edge

SEEK DIGITAL LIBRARY

orientation [7], amount of moving pixels [12], [13] and blob size [7]. In this paper, we have used a feature based method involving projection histograms which will be discussed in the following sections.

II. Proposed Method

In this section the proposed algorithm has been discussed in detail. The first step is to capture video frames from a camera and down-sample these to QCIF (176x144) size. Down sampling is done to reduce the computational cost, but in general with an effect on image quality. Since we do not use any recognition methods, the compromise in image quality is unlikely to affect the performance. The proposed method can be used in real life scenarios where bi-directional counting can be done across a user defined straight line which is generally aligned with an entrance or exit.

A. Foreground Estimation

The down-sampled images are accumulated over a certain period to form a background model for each and every pixel, represented by $bg_t(x,y)$ in Equation 1. From this background model, a foreground model is extracted. To extract foreground, maximum and minimum values are selected from the background model. This is done for every individual pixel forming a band around a pixel. These are represented by $bg_{max_t}(x,y)$ and $bg_{min_t}(x,y)$ respectively. The band is further widened by multiplying the maximum and minimum values by two different user defined thresholds represented by u_{th} and l_{th} respectively. If the value of a pixel in a new frame falls outside this band then the pixel is considered as foreground represented by $fg_t(x,y)$, otherwise it is considered a background pixel. The background model formed is continuously updated with time using an exponential updating function using a parameter α which denotes the time constant.

$$bg_t(x,y) = \begin{cases} I_t(x,y), & t = 0 \\ \alpha I_t(x,y) + (1-\alpha) bg_{t-1}(x,y), & t > 0 \end{cases}$$

$$bg_{max_t}(x,y) = u_{th} * bg_t(x,y)$$
$$bg_{min_t}(x,y) = l_{th} * bg_t(x,y) \quad (1)$$
$$fg_t(x,y) = \begin{cases} 0, & bg_{min_t}(x,y) < I_t(x,y) < bg_{max_t}(x,y) \\ 1, & otherwise \end{cases}$$

Prior to updating a pixel, it is checked whether a particular pixel is static or moving by using sum of absolute difference measure over two corresponding blocks in consecutive frames. A pixel is updated only when it is stationary. This step ensures that a pixel does not get updated with any random value caused by unnecessary motion like tree leaf movement, moving objects or illumination noises. The final outcome of the entire process is a foreground model representing change with respect to the background model described above. All subsequent processing is done on this binary image.

B. Graph Based Relation

The next step is establishing relationship between binary foregrounds of two consecutive images by using an undirected bipartite graph. The process involves finding correspondences between blobs of two foreground images. When a new set of blobs is computed for frame i, an association with blobs in frame $(i-1)$ is sought. With each new frame, blobs can split, merge, appear or disappear. The relation between the frames can be represented by undirected bipartite graph $G(V_i, E_i)$ (represented as G_i) where $V_i = B_i \cup B_{i-1}$. B_i and B_{i-1} are set of blobs associated with the blobs in frame i and $i-1$ respectively. Here vertices of the graph are represented by V_i and the edges connecting the vertices are represented as E_i. The process of blob relation establishment is equivalent to computing G_i. Let $N_i(u)$ denote the set of neighbors of vertex $u \in V_i$, $N_i(u) = v|(u,v) \in E_i$. In order to reduce the complexity, two constraints are implemented, namely the parent and the local constraint.

According to parent constraint, a blob may not participate in splitting and merging at the same time. Local constraint implies that vertices can be connected only if their corresponding blobs have a bounding box overlap area which is at least half the size of the bounding box of the smaller blob. To find the optimum G_i, a cost function $C(G_i)$ is defined, so that different graphs can be compared.

In order to proceed with the formation of the cost function, we define two disjoint sets, namely parents, P_i and descendant D_i whose union is $V_i$such that $D_i = \cup N_i(u)$. $E_i$can be easily constructed by selecting from V_i all vertices of degree more than one, all vertices of degree zero, and all vertices of degree one which are only in B_i. Furthermore let $S_i(u) = \sum_{v_i} A(v)$ be the total area occupied by the neighbors of u. The cost function used penalizes graphs in which blobs change significantly in size. A perfect match would be one in which blob size remains constant (e.g., the size of a blob that splits equals to the sum of the sizes of blobs it split into). The formula for the cost function is:

$$\boldsymbol{C(G_i) = \sum_{u \in P_i} |A(u) - S_i(u) / \max(A(u), S_i(u)|} \quad (2)$$

This function is a summation of ratio of size change over all parent blobs and the aim is to minimize the function. Using this cost function, the optimum graph is calculated. This optimum graph establishes the relationship between the previous and the current frame.

C. Kalman Based Tracking

The Kalman Filter addresses the general problem of estimating the state of a discrete-time controlled process that is governed by the linear stochastic difference equation:

$$x_k = A * x_{k-1} + w_{k-1} \quad (3)$$

with a measurement $y \in R^m$ that is

$$y_k = M * x_k + v_k \quad (4)$$

where x denotes the state vector, A denotes the transition matrix, M denotes the scaling matrix and k denotes the time index. The random variables w_k and v_k represent the process noise and measurement noise respectively. They are assumed to be independent of each other with Gaussian probability distributions.

SEEK DIGITAL LIBRARY

In [3] tracking is done on an entity called pedestrian, which is nothing but the projection of an average sized human on the image plane. We avoid this step since it involves calibration of the cameras. Final counting in [3] is based on pedestrians. In this paper, we use projection histograms to do the same. Kalman filter is employed in three stages which are measurement, prediction and updation. Updation of the predicted values are done using the measurement. All these stages are discussed in the following sections.

1) Measurement Stage

The measurement process is important because it will give us the real time feedback. In this paper, measurement refers to the measured position of a blob. The current setup has two methods for measurements. The methods are selected based on the split-merge pattern of the blobs. The relation between blobs in the current and previous frame can be one to one, many to one (Merging) and one to many (Splitting). In case of splitting and one-to-one correspondence, a heuristic measure is used as the measurement mechanism. In case of merging scenario, mean shift algorithm is used both of which are explained underneath.

2) Heuristic Measure

In this step, we use the predicted Kalman locations as starting positions and employ a 2D search to locate the match in the current frame. The search attempts to find the best overlap between the blobs in the previous and the current frame. To perform the search, we employ a heuristic solution. At first, search is conducted by moving the Kalman predicted position by a large, 64 pixel step in all directions and the location with maximum overlap area is recorded. With the recorded position as starting point and half the step size of 32 pixels, the process is repeated till the step size becomes one. The resultant location is fed as measurement value to the Kalman filter.

3) Mean Shift Algorithm

In case of merging of blobs we use the mean shift algorithm, as the heuristic method employed above will not give a proper match. Mean shift tracking works by finding clusters in a particular feature space. In this setup feature space is formed by calculating a back projection image on blob models. If two blobs in the current frame i merge into a single blob in the next frame $i+1$, a model histogram is calculated for each of the blobs in i^{th} frame and a back projection image is formed with the Kalman predictions as input for the $(i+1)^{th}$ frame. Mean shift tracking will result in convergence at the densest cluster present in the back projected image. The densest clusters will correspond to the regions having features identical to the blob models.

4) Prediction and Updating Stage

The Kalman filter estimates a process by using a form of feedback control. The filter starts with some initial estimate of a process state along with measurement and process noise profile. At each and every stage current estimate is calculated based on the past estimates $\hat{x}_{k-1|k-1}$ using the following equation:

$$\hat{x}_{k|k-1} = A * \hat{x}_{k-1|k-1} \quad (5)$$

In our system, this corresponds to the current estimated position of blob. It is necessary to have an initial estimate of the blob position. If a new blob is formed then the position of the blob is recorded and is used as the initial estimate. If position is not dependent on any other state like velocity and acceleration, the above equation will be of single dimension. But generally this is not the case. In our system, position of a blob is dependent on the velocity of a person and in many cases it might also be dependent on acceleration and hence the state space equations are multidimensional. Once an estimate has been made it has to be updated by taking into account the present measurements as per the following set of equations.

$$\begin{aligned} \tilde{e}_k &= y_k - M * \hat{x}_{k|k-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k * \tilde{e}_k \\ K_k &= P_{k|k-1} * M^T * \left(M * P_{k|k-1} * M^T + R\right)^{-1} \\ P_{k|k-1} &= A * P_{k-1|k-1} A^T + Q \\ P_{k|k} &= (I - K_k * M) * P_{k|k-1} \end{aligned} \quad (6)$$

Here Q and R are process and measurement covariance matrices related to w_k and v_k respectively. The state vector also includes velocity of the blob. Whenever a new blob is found it is assigned zero velocity. In case of splitting operation, the corresponding descendant blobs are assigned the same velocity as that of the parent. In case of a merging operation velocity of the largest descendant blob is assigned to the parent. Finally if there is a one to one correspondence between two blobs then the velocity of current blob is measured by the following equation.

$$\boldsymbol{v} = \boldsymbol{\beta} * \frac{\boldsymbol{b_v - b_u}}{\boldsymbol{\delta t}} + (\mathbf{1} - \boldsymbol{\beta})\boldsymbol{u} \quad (7)$$

In the above equation v is the current velocity of the blob, u is the previous velocity, b_v and b_u are the blob centres and β is the weighing factor.

5) Filter Parameter Tuning

In the actual implementation of the filter, the measurement noise covariance R is usually measured prior to operation of the filter. Determining measurement error covariance R is generally easy because it is possible to measure process variables. Hence we can take some off-line samples in order to determine the variance of the measurement noise. The determination of the process noise covariance Q is generally more difficult as we typically do not have the ability to directly observe the process. Sometimes a relatively simple process model can produce acceptable results if one injects enough uncertainty into the process via the selection of Q. Certainly in this case one would hope that the process measurements are reliable. In either case, whether or not we have a rational basis for choosing the parameters, often superior filter performance (statistically speaking) can be obtained by tuning the filter parameters Q and R. The tuning is usually performed off-line, frequently with the help of another distinct Kalman filter in a process generally referred to as system identification.

As mentioned above, measurement noise is obtained by establishing uncertainty in blob position. Uncertainty in blob

SEEK DIGITAL LIBRARY

position can be established by finding the uncertainty in matching blobs of two consecutive frames. Since we are using an area based match, if corresponding blobs in two consecutive frames have equal area then the measurement has no error. In case of splitting and merging, the combined area of the smaller corresponding blobs should be equal to the area of the undivided block. So finding a ratio of these areas will give us a measure of uncertainty in measurement. Process noise can be found by establishing uncertainty in velocity measurement. Uncertainty in velocity is given by acceleration. Acceleration is dependent on the behavior of an individual blob representing a person which depends on the scenario at hand.

D. *Counting*

One of the methods to count a person or an object depends on detecting a particular feature of the object. In case of human beings, the feature can be skin, location of head, facial features, contours etc. In this paper, a histogram based technique is used. According to this technique, the extracted foreground image in a certain region is taken as the base image. The non-zero pixels along the y axis for a single value of x are calculated. This process is repeated for all values of x. The array of summed values thus obtained is plotted for different values of x. The histogram thus obtained is called a vertical projection histogram as shown in Figure1. A similar process can be done along the y axis resulting in a horizontal projection histogram. In this paper, the obtained histogram is used as an indicator for the number of persons. Based on the number of peaks in the obtained histogram, the number of people is calculated.

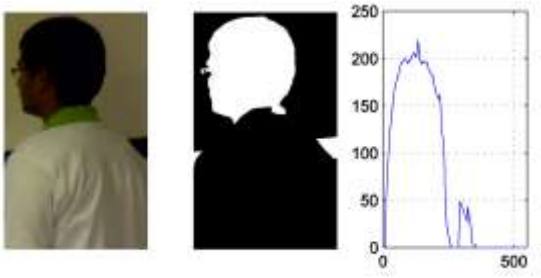

Figure 1. Example of an Image, Corresponding binary image and its Vertical Projection Histogram

III. Result

The results obtained by applying the proposed algorithm in different outdoor and indoor scenarios are shown in Figure 2. Red lines in these Figures represent a virtual fence, crossing which triggers an alarm represented by bounding rectangles. Whenever a person crosses from right to left or up from down it is shown with a green rectangle, otherwise a blue rectangle represents left to right and down from up motions. Top left corner of every image displays count in the two directions left/up (green) and right/down (blue). Parameter tuning might be essential for getting proper output for a particular scenario.

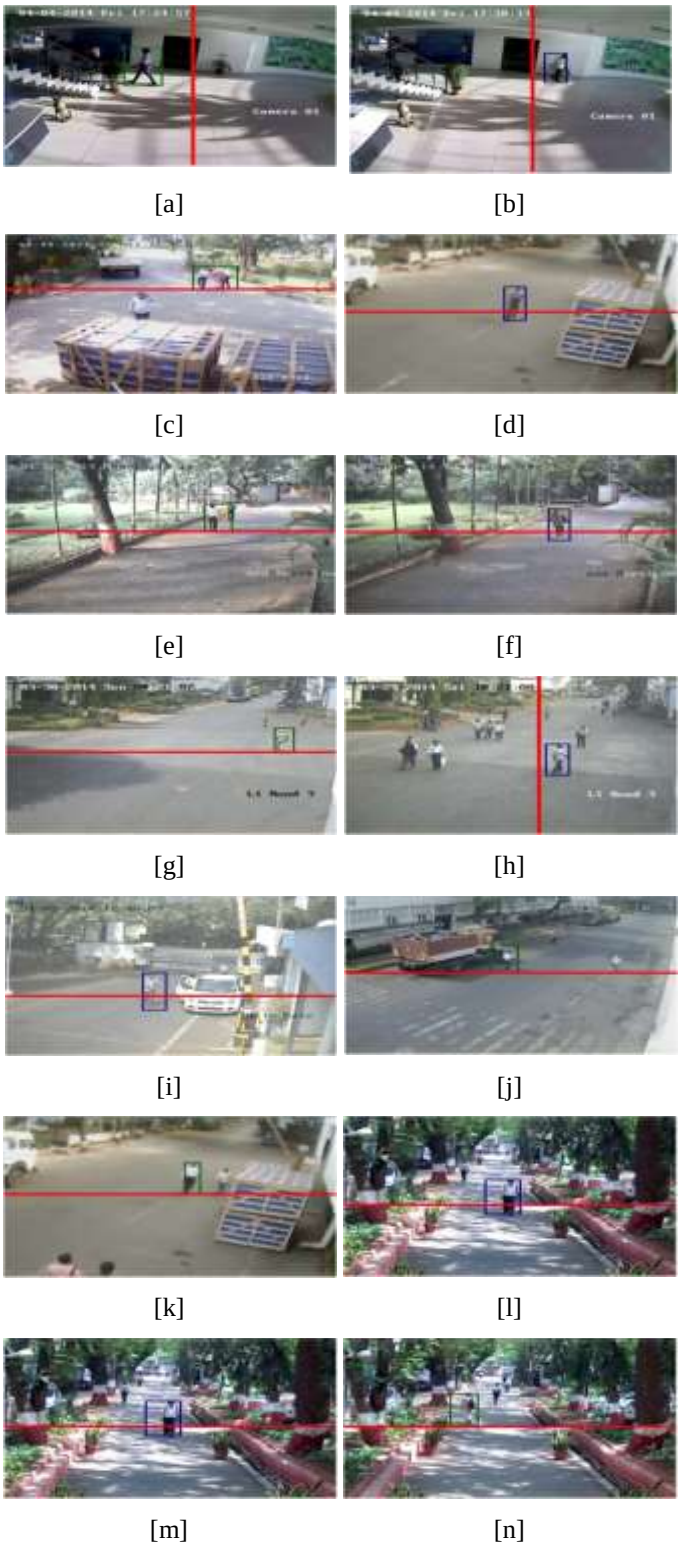

Figure 2. Output images for different scenarios

For example when the contrast of the image is low then thresholds for foreground extraction u_{th} and l_{th} should be set

SEEK DIGITAL LIBRARY

closer to one. Such a case is shown in Figure2 [a], [b]. Here u_{th} was set to 1.1 and l_{th} was set to 0.9. In Figure2 [l], [m], [n] the values were set to 1.3 and 0.7 respectively. This was done to handle noise due to tree leaf movement and its shadow. In case the objects of interest are very small in size relative to the scene dimensions then parameter for area matching module should be set accordingly. In cases when distance between objects is very small (when people walk together) relative to the scene dimensions then different objects may be merged into a single object and the algorithm will give a false count. This is illustrated in Figure2 [c] wherein two persons were counted instead of three. In Figure2 [h] to [n] it can be seen that only persons crossing the line are counted although distance from the line at which a person is detected is determined by a user defined sensitivity parameter. In Figure2 [i] it can be seen that a car is entering but the car is not counted while the person is counted. This is because the vertical projection histogram for a car is flat unlike that of a human being.

IV. Conclusions and Future Scope

In this paper, we have proposed a method for people counting based on Kalman tracking of foreground blobs and projection histograms. This algorithm can be used for foot fall counting, counting number of people entering or leaving office premises, pedestrian passages etc. We have avoided calibration of camera since any change in position of camera due to external forces may need recalibration. So the proposed method enables the use of the algorithm in any outdoor unregulated environments. The use of projection histogram constrains us to use counting only along horizontal and vertical directions. If a line is drawn which is inclined to the axes then the method is not suitable. A simple remedy to this scenario is the use of principal component analysis. In addition, more human features can be incorporated into the algorithm for better robustness in human counting. The proposed paper does not address heavily crowded scenarios for which concepts of fluid dynamics might be useful.

References

[1] Osama Masoud, Nikolaos P. Papanikolopoulos. "A novel method for tracking and counting pedestrians in real-time using a single camera," Vehicular Technology, IEEE Transactions on Volume:50 , Issue: 5, 2001.

[2] P. Aditya, D. Monotosh and S. Jayalakshmi. "Left object detection with reduced false positives in real time," DPPR, Advances in Intelligent Systems and Computing Volume 177, 2013.

[3] C R Wren, A Azarbayejani, T Darell and A. P. Pentlands. "Realtime tracking of the human body," Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1997, Volume:19, Issue: 7).

[4] C. Stauffer and W. Grimson. "Adaptive background mixture models for real-time tracking," Computer Vision and Pattern Recognition. IEEE Computer Society Conference, 1999, Volume:2.

[5] Kyungnam Kim, Thanarat H. Chalidabhongse, David Harwood and Larry Davisa. "Real-time foreground background segmentation using codebook model," In: Proc. ICIP, pp. 3061–3064.

[6] Ahmed Elgammal, David Harwood, Larry Davis. "Non-parametric model for background subtraction," Computer Vision ECCV.

[7] K. Rohr. "Human movement analysis based on explicit motion models," Computational Imaging and Vision Volume 9, 1997, pp 171–198.

[8] I. Haritaoglu, D. Harwood and L. S. Davis. "W4 real-time surveillance of people and their activities," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2001, vol. 22, no. 8, pp. 809–830.

[9] P. Viola and M. Jones. "Rapid object detection using a boosted cascade of simple features," Vision and Pattern Recognition, CVPR, 2001.

[10] S. Y. Cho, T. S. Chow and C. T. Leung. "A neural-based crowd estimation by hybrid global learning algorithm," IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, 1999, vol. 29, no. 4, pp. 53–54.

[11] D. Kong, D Gray, H Tao. "Counting pedestrians in crowds using viewpoint invariant training," Proc. British Machine Vision Conference (BMVC), 2005.

[12] N. Paragios and V. Ramesh. "A MRF-based approach for realtime subway monitoring," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR, 2001, vol. 1, pp. 11034–11040.

[13] R Ma, Li Liyuan, Weimin Huang and Qi Tian. "On pixel count based crowd density estimation for visual surveillance," IEEE Conference on Cybernetics and Intelligent Systems, 2004, vol. 1, pp. 170–173.

About Author (s):

Sathish Kumar Shivagurunathan received his Master's Degree in Electrical Engineering from National Institute of Technology, Calicut in 2013. At present he is involved in development of speech and voice processing systems for automobiles.

Aditya Piratla received his Master's Degree in Electrical Engineering from National Institute of Technology, Calicut in 2012. At present he is working in the domain of analytics for video surveillance applications. His main areas of interest are computer vision and statistical image processing.

Jayalakshmi Surendran received Ph.D. in Electrical Engineering from Indian Institute of Technology, Bombay in 2008. She has worked extensively in the field of Image Processing, Video processing and various other applications of signal processing. She is currently working with Crompton Greaves Global R&D centre and contributes to video analytics for intelligent processing of video for surveillance application.

Monotosh Das received Ph.D. in Embedded real time control system from Jadavpur University, Kolkata in 2007. He has established himself as a pioneer in embedded control systems and enterprise level software design and development.

SEEK DIGITAL LIBRARY