VFAST Transactions on Software Engineering
http://vfast.org/index.php/VTSE@ 2014 ISSN: 2309-3978
Volume 4, Number 2, July-August 2014 *pp-09-20*

REAL-TIME OBJECT DETECTION AND TRACKING FOR VIDEO SURVEILLANCE

AKINTOLA K.G
Computer Science Department, Federal University of Technology, Ondo State, Akure, Nigeria
Akintola2087@yahoo.com

ABSTRACT *In view of application in smart visual surveillance systems this paper presents a new method of object detection and tracking in a surveillance scenario. The paper proposes robust object detection and tracking mechanism in which the background subtraction uses parallel processed Kernel Density Estimation (KDE) and the object tracking uses spatial color models of the detected object for tracking. The models of newly detected objects are stored in a reference list. Models of object detected in the next frame are compared with the reference models to track the object in the new frame. The spatial dimension introduced make the algorithm performs very well and fast. The system has been implemented both in indoor and outdoor environments and was found not only functionally okay but very computationally efficient in terms of processing time. The proposed system also is capable of detecting collision and object merging. One major areas the system contributes is the fast processing which is required by surveillance systems.*
Keywords: KDE, surveillance, tracking, models

1. Introduction. Object detection and tracking is an important task in computer vision applications including surveillance, smart buildings, augmented reality, video compression and medical imaging processing. Before we can perform any advanced video image processing, we need to identify the particular object from the scene. Therefore detecting regions that corresponds to moving objects such as people, vehicles, animals is the first basic step of almost every vision system Akintola et al., (2011). Due to different nature of different backgrounds, some static some dynamic and some quasi-stationary, motion detection posed a difficult problem, and different algorithms have been proposed to work in different environments. Frequently used techniques for moving object detection are background subtraction, background statistical modeling, temporal differencing and optical flow. In this research we adopted the kernel density estimation method (KDE) which is modeled in parallel fashion. After object detection, the next thing is how to keep on tracking such objects in subsequent frames. Object tracking can be defined as the problem of estimating the trajectory of an object in the image plane as the object moves around the scene. In other words, a tracker assigns consistent labels to the tracked objects in different frames of a video. A visual surveillance system must be able to detect objects within the scene, track the objects for eventual activity recognition.

The tracking of real world objects is a challenging task due to the presence of noise, occlusion and clutter of objects of interest. Tracking schemes usually include two major components (Comaniciu etal., 2003). The first component deals with generating and maintaining a model for the objects within the scene while the second component searches for potential locations for these objects in the current frames. A variety of algorithms have been proposed in the literature to tackle these problems of tracking. Tracking algorithms can broadly be classified into two categories viz: deterministic methods and stochastic methods. Deterministic methods typically track by performing an iterative search for the local maxima of a similarity cost function between the template image and the current image. The cost function widely used is the sum of squared difference (SSD) between the template and the current image. In this regards, the mean-shift algorithm or other optimization techniques have been applied to find the optimal solution (Yang etal 2005). Mean Shift is a powerful and versatile non parametric iterative algorithm that can be used for lot

of purposes like finding modes, clustering etc. Mean Shift was introduced in Fukunaga and Hostetler (Fukunaga *et al.,* 1975) and has been extended to be applicable in other fields like Computer Vision.

Meanshift considers feature space as a empirical probability density function. If the input is a set of points then Mean shift considers them as sampled from the underlying probability density function. If dense regions (or clusters) are present in the feature space , then they correspond to the mode (or local maxima) of the probability density function. We can also identify clusters associated with the given mode using Meanshift. The mean shift algorithm is a simple iterative procedure that shifts the center of the region of interest to the center of mass (i.e. average of the data points). Bradski (1998) proposed an enhanced mean-shift algorithm named Continuously Adaptive Mean Shift (CamShift) that can track objects that changes shape dynamically.

Model based tracking algorithms incorporate a priori information about the objects to develop representations such as skin color, body blobs, silhouettes, kinematic skeleton, while Appearance based approaches apply recognition algorithms to learn the objects in some basis such as the eigenspace, or in kernel space Yang etal (2005).

The second category which is the stoichastic method, uses the state space to model the underlying dynamics of the tracking system. An example of this method is the Kalman filter. The basic idea of Kalman filtering based tracker is to recursively estimate the state vector given the last estimate and a new measurement. The Kalman filter is a tool that can estimate the variables of a wide range of processes. In mathematical terms we would say that a Kalman filter estimates the states of a linear system. The Kalman filter not only works well in practice, but it is theoretically attractive because it can be shown that of all possible filters, it is the one that minimizes the variance of the estimation error (Simon 2010). The Kalman filter is composed of two steps, the prediction step and the correction step. The prediction step predicts the approximate location of the object while the correction step computes the exact location of the object and the process contnues.While Kalman filter method require a Gaussian and linear process, the particle filtered method is used to model a non-linear and non-Gaussians process. Particle filters are sequential Monte Carlo methods based on point mass representations of probability densities, which are applied to any state model. Particle Filter is concerned with the problem of tracking single and multiple objects. It is an hypothesis tracker, that approximates the filtered posterior distribution by a set of weighted particles. It weights particles based on a likelihood score and then propagates these particles according to a motion model.

Section 2 describes kernel density estimation, section 3 describes the Histogram computation of the distribution formed by pixel observations at a particular spatial location. Section 4.0 describes an experiment on object detection and tracking algorithms. Section 5.0 presents performance analysis of the object tracking algorithm.

2. Related Research. Comaniciu etal., (2000) presents a new approach to real-time tracking of non-rigid objects based on visual features such as colour and texture, whose statistical distributions characterize the object of interest. The proposed algorithm is good for objects with different colour and texture patterns being robust to occlusion, clutter, rotation in depth and changes in position. The mean shift iterations are employed to find the target model with the similarity being expressed by a metric based on Bhattacharyya coefficient. Various experiment show that the algorithm performs excellently.

In Haritaoglu (1998), a real time system tagged W4 for detecting and tracking people is presented. W4 is designed as real time visual surveillance system for detecting and tracking people and monitoring their activities in an outdoor environment both day and night. It operates on a monocular grayscale video imagery or video imagery from an infrared camera. W4 is motivated by the desire to develop a people tracking oriented system that can locate and track people especially during the night time. The objective of developing the system is to answer questions about who is the there?, what is he doing?, where is he? and when was the action?.

The system can perform Background modeling using frame differencing, object detection, object tracking, occlusion handling using region merging and splitting. The object tracking aspect uses second-order motion model including velocity and acceleration for human tracking. It also uses human parts localization and tracking using cardboard algorithm for activity recognition.Fleck etal., (2006), presents a distributed network of smart cameras for real –time tracking and its visualization in 3D environment. In this project, a multi-object tracking system is developed. The live visualization of tracking result is embedded in a 3D model of the environment. The project is motivated by the desire to develop a multi tracking system that can be visualized in embedded 3D model environments. The system employs particle filter algorithm for the object tracking. The limitation of the tracking aspect is that particle filter is computationally expensive.

In Collins etal. (2001), the Architecture of the Video Surveillance and Monitoring (VSAM) is presented. The Objective of the system is to develop a single user monitoring system. The architecture of VSAM consists of an operator control unit, several sensor processing units, a graphical user interface (GUI), and several visualization nodes. Object detected is tracked using Kalman filter. The live visualization of tracking result is embedded in a 3D model of the environment. The project is motivated by the desire to develop a multi tracking system that can be visualized in embedded 3D model environments and can generate real time alerts. The system employs Kalman filter algorithm for the object tracking. The limitation of the tracking aspect is that Kalman is computationally expensive.

In Georis etal (2008) an IP-distributed computer-aided video-surveillance system is presented. It is a surveillance architecture that consists of three components: First are the computers connected together through a typical fast Ethernet network (100 Mbps). Secondly, we have the various cameras that are plugged either in an acquisition card in a PC or directly on the local network hub for IP cameras. Lastly we also have the Human Computer Interface (GUI) and storage space plugged into the system. This platform has the capacity for analysis tools, such as motion detection, segmentation, tracking and neural networks modules. The goal of these advanced tools is to provide help to operators by detecting events of interest in visual scenes and store them with appropriate descriptions. This indexation process allows one to rapidly browse through huge amounts of stored surveillance data and play back only interesting sequences. The system also employs Kalman filter tracking algorithm which is computationally expensive.In the aforementioned systems the non-functional requirement such as time of processing is traded off for the functional requirement such as track-ability. Here we propose a system that is both computationally non expensive and has high track-ability. This is required as the tracking system would be employed in surveillance environments.

2.0 The Proposed Object detection and Tracking Algorithm The proposed algorithm is composed of two stages. Background modeling and object tracking. For background modeling, the approach used in this research is Kernel density estimation with adaptive threshold and shadow removal algorithm. The goal in image segmentation is to separate background areas of the image from foreground regions of motion that are of interest for advance vision processing such as tracking. In this research, we make the fundamental assumption that the background will remain stationary. This assumption necessitates that the camera be fixed and that lighting does not change suddenly. Segmenting moving objects in still camera video frames is done in three stages in the proposed method. First is the Histogram computation, next is the threshold calculation and then foreground segmentation.

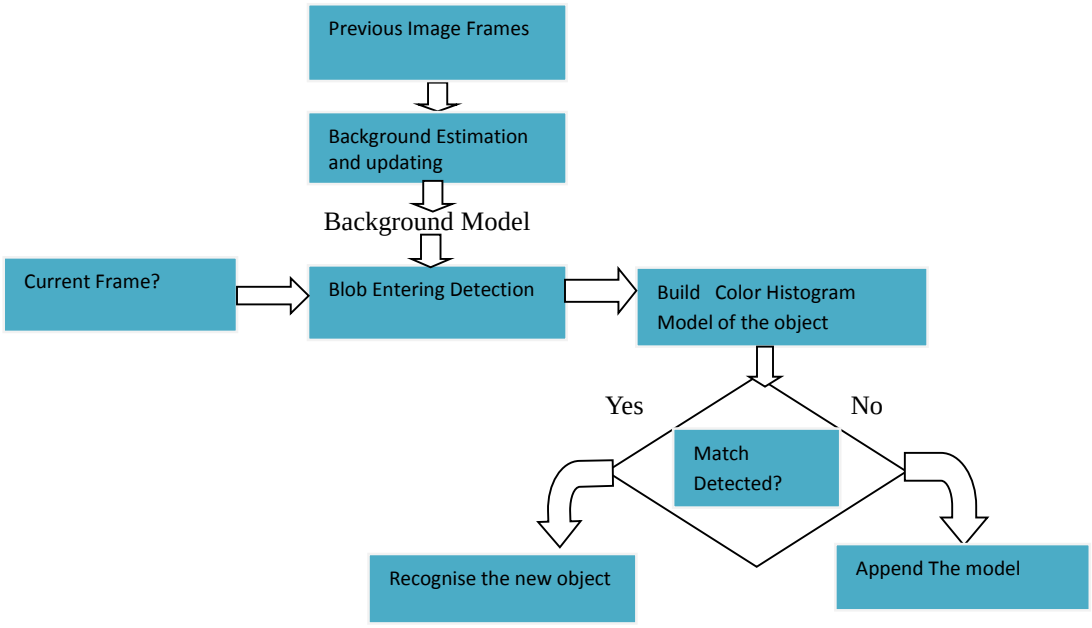


Figure 1.0 Object detection and Tracking.

2.1 Background Modeling using Kernel Density Estimation: Kernel density estimation (KDE) is the most used and studied nonparametric density estimation method. The model is the reference dataset, containing the reference points indexed natural numbered. In addition, assume a local kernel function centered upon each reference point, and its scale parameter (the bandwidth). The common choices for kernels include the Gaussian: and the Epanechnikov kernel.

The Gaussian Kernel is given by:

$$K_N = (2\pi)^{-d/2} \exp\left(-\frac{1}{2}\|\mathbf{x}\|^2\right) \quad (1)$$

The Epanechnikov kernel is given by

$$K_E = \begin{cases} \frac{1}{2}c_d^{-1}(d+2)(1-\|\mathbf{x}\|^2) & \text{if } \|\mathbf{x}\| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Let $X_1, \ldots, X_n$ be a random sample taken from a continuous, univariate density f. The kernel density estimator is given by,

$$\hat{f}(x;h) = \frac{1}{nh}\sum_{i=1}^{n} K\{(x - X_i)/h\} \quad (3)$$

K is the function satisfying $\int K(x)dx = 1$

The function K is referred to as the Kernel h is a positive number, usually called the bandwidth or window width.

2.2 Histogram computation. The first 100 initial frames in the video sequence (called learning frames) are used to build stable distributions of the pixel RGB mean. The RGB intensities of each pixel position is accumulated for the 100 frames and we calculatete the cumulative sum of the average intensities i.e (sum of (RGB)/3) is computed over 100 frames. A histogram of 256 bins is constructed using these pixel average intensities over the training frames. . The sum is then normalized to 1. That is we divide each histogram bin value with the accumulated sum to get a normalized histogram of 1.

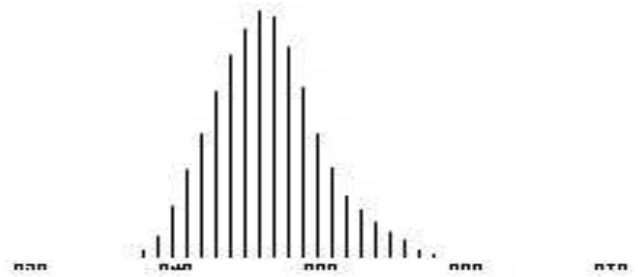


Figure 2.0 An Histogram of typical pixel location.

2.3 Threshold calculation. Threshold is a measure of the minimum portion of the data that should be accounted for by the background.

For more accuracy in our segmentation, we use different threshold for each histogram bins.

The pseudo- code for the Threshold calculation is given below

```
1  For each H[i]
2        Get sum of H[i]
3        Peak[i]=max(H[i])
4        Pth[i]=Peak[i]/2
5        Calculate sum2(H[i] > Pth[i])
6              If(sum2(H[i] > Pth[i])  is less than 0.95 of sum of Hi
7                    Pthi=Peak[i]/2
8                    go to 5
9              else
10                   threshold=Pth[i]
```

2.4 Foreground/ Background detection. For every pixel observation, classification involves determining if it belongs to the background or the foreground. The first few initial frames in the video sequence (called learning frames) are used to build histogram of distributions of the pixel means. No classification is done for these learning frames. Classification is done for subsequent frames using the process given below. Typically, in a video sequence involving moving objects, at a particular spatial pixel position a majority of the pixel observations would correspond to the background. Therefore,
background clusters would typically account for much more observations than the foreground clusters. This means that the probability of any background pixel would be higher than that of a foreground pixel. The pixel are ordered based on their corresponding value of the histogram bin. Based on the adaptive threshold calculated above, get the pixel intensity value for the subsequent frames. Locate the corresponding histogram bin Read the bin value corresponding to this intensity
If the value is < threshold
    Classify as foreground
Else
Classify as background.
The classification process is performed on all the pixel locations in the current frame.
The Algorithm
(Frames 1—N is used for training modeling the background).
Read frames 1 -- N
For each pixel
    Calculate the value of (r+g+b)/3
    Locate the corresponding bin value in the histogram of the pixel.
    Increment the bin of this value by 1
    Increment the surrounding bandwidth pixels by fraction of 1
Normalize the histogram value by dividing each bin value by the sum of bins.
Calculate the adaptive threshold as given in section 2.3.
Read the Next frame after N.
For each pixel
    Read the intensity of RGB of the pixel.
    Calculate the value of (r+g+b)/3
    Locate the corresponding bin value in the histogram of the pixel.
    Test if the value is < threshold
        Classify the pixel as foreground
    Else
        Classify the pixel as background

3.0 A spatio-color Algorithm for Object Tracking The proposed algorithm is composed of two stages. First is the appearance correspondence mechanism. Once detected, Appearance models are generated for objects appearing in the scene. The model is the estimate of probability distribution of colour of pixel colours. Multiple models are developed for a single object. These models are then used in subsequent frames to match the set of currently detected models and that of target models. In the second phase occlusion resolution and object merge and separation are handled.
The foreground object detected in previous stage is passed to the object tracker. We adopt a multi-part tracking algorithm in our system. That is, we segment each silhouette into upper-body area and lower-body area and generate a histogram of colures in HSV color space. This approach is good enough at discriminating individuals because of varying intensity in identical objects with similar color and occlusion. Our approach makes use of the object color histograms of previous frame to establish a matching between objects in consecutive frame. Our method is also able to detect object occlusion, object separation and label the object appropriately during and after occlusion. The diagram of our object tracker is shown in Figure 4.12.

3.1 Building the Spatio-Color Histogram Model Color models are obtained by histogramming techniques in the Hue-Saturation-Value (HSV) color space (Perez etal. 2002) in order to decouple chromatic information from shading effects. Color information is however only reliable when both the saturation and the value are not too small. Hence, we populate an HS histogram with NhNs bins using only the pixels with saturation and value larger than two thresholds set to 0.1 and 0.2 respectively (Perez etal. 2002). The remaining “color-free” pixels can however retain

crucial information when tracked regions are mainly black and white. We thus found useful to populate Nv additional value-only bins with them. The resulting complete histogram is thus composed of $N = N_h N_s + Nv$ bins.bins (Perez etal. 2002). We shall denote $b_t(\mathbf{u}) \in \{1, \ldots, N\}$ the bin index associated with the color vector $\mathbf{y}_t(\mathbf{u})$ at pixel location **u** in frame t. Given an occurrence of the state vector $\mathbf{x}_t$, the candidate region in which color information will be gathered is defined as:

R(**x**t) ˙ =**d**t + stW. (4)

Within this region a kernel density estimate **q**t(**x**) = {qt(n; **x**)}n=1・・・N of the color distribution at time t is given by (Perez etal. 2002)

$$qt(n; x) = K \sum_{u \in N(x)} w(|u - d|)\delta[b_t(d) - n]. \quad (5)$$

where δ is the Kronecker delta function, K is a normalization constant ensuring $\sum_{n=1}^{N} q_t(n; x) = 1$, w is a weighting function, and locations u lie on the pixel grid, possibly sub-sampled for efficiency reasons. This model associates a probability to each of the N color bins. Here we set $w = 1$, which amounts to standard bin counting. This model associates a probability to each of the N color bins. In Comaniciu etal., (2000) the weight function is a smooth kernel such that the gradient computations required by the iterative optimization process can be performed. Here we set w ≡ 1, which amounts to standard bin counting.

3.2 Histogram Matching After foreground object detection has been achieved, we use a bounding box to get the area occupied by the object. For each object detected in the scene, we just build a two set reference color histograms and keep it in a list of reference models. To cater for the spatial distribution of color and to increase the ability of the tracker we divide the bounding box into two regions, called the lower and the upper regions. We then compute the upper-region histogram and the lower-region histogram separately and stored in histograms vector object. We call this list {RO-List}. The list of currently detected objects in the scene is kept in {CO-List}. The idea is to search on {CO-List} their corresponding blob (appearance model) in {RO-List} from the previous frame. This algorithm will be very fast since searching is done in the list and not in spatial window. For objects in subsequent frames used for matching, as we have done for reference histograms, we calculate the histograms of the list of objects found in the subsequent frames. For each object, we compare the computed histograms both (upper and lower) with the reference histogram both upper and lower using the Bhattacharyya distance measure. Bhattacharyya distance measure returns a range of bounded values between [0 to 1]. 0 indicates that the two objects are very similar while 1 indicates they are not similar. If BD(obj1,obj2) < 0.419 then they can still be similar. BD() function is Bhattacharyya distance measure. Since we are adoting a Bhattacharyya distance measure:

RO-List(i)=arg Min [(P(CO-List(i) | RO-List(k))] vk .. (6)

At time *t*, the color model $\boldsymbol{q}t(\boldsymbol{x})$ associated with hypothesized state $\boldsymbol{x}$ will be compared to the reference color model $\boldsymbol{q}^* = \{q^*(n)\}n=1・・・N$, with $\sum_{n=1}^{N} q*(n) = 1$. In this research work, the reference distribution is gathered at an initial time t_0 at a location/scale $\mathbf{x}^*$ t_0 automatically provided by a foreground detection module.

$$q^* = \mathbf{q}^* t_0 (\mathbf{x}\ t_0) \quad (7)$$

The data likelihood must favor candidate color histograms close to the reference histogram; we therefore need to choose a distance D on the HSV color distributions. Such a distance is used in the deterministic techniques (Comaniciu etal.,2000), as the criterion to be minimized at each time step. In (Perez etal. 2002 , Comaniciu etal.,2000), D is derived from the Bhattacharyya similarity coefficient, and is defined as

$$D(q^*, q_t(x))] = \left[\sum_{n=1}^{N} \sqrt{q^*(n) q_t(n; x)}\right]^{\frac{1}{2}} \quad (8)$$

This distance between probability distributions is a bounded within [0,1]. Note that and empty bins are not a source of concern (Perez etal. 2002 , Comaniciu etal.,2000)

3.3. Occlusion Handling To track object efficiently, we must handle the situation when an object are occluded by another object. Our tracking system uses simple heuristics to detect objects merging and separation.

3.4. **Detecting Object Merge** When the number of contours in the current frame is less than the number in the previous frame, we try to determine whether it is as a result of objects leaving the scene or due to object merging. Object presumed to merge if the distance between object in the previous frame is less than 1 and the bounding box size of the new contour is 1.5 greater than the previous size.

3.5 Detecting Object Separation When the number of contours in the current frame is more than the number in the previous frame, we try to know whether it is as a result of new object entering the video or object merge. Object merge is presumed if suddenly the bounding box size of the new contour is 1.5 less than the previous size and the distance between the two contours object in the current frame previous frame is less than 1.

4.0 Experiment The proposed object tracking technique has been tested on a variety of indoor video sequences. It has been used to track both fast and slow moving objects under different conditions and found to be very efficient. Figure 3.4 shows the example of the application of the algorithm in an indoor

(a) background

(b)Moving objects moved

(e)Detected Foreground Mask with morphological operation performed

Figure 3.0. Human Segmentation process.

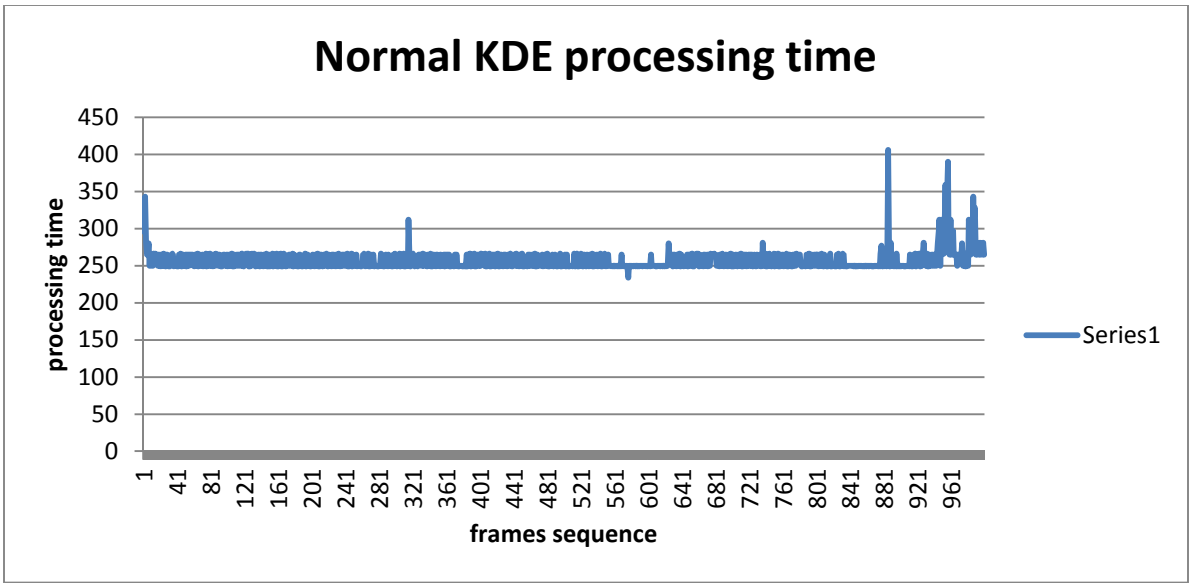


Figure 4.0 Processing time of normal KDE

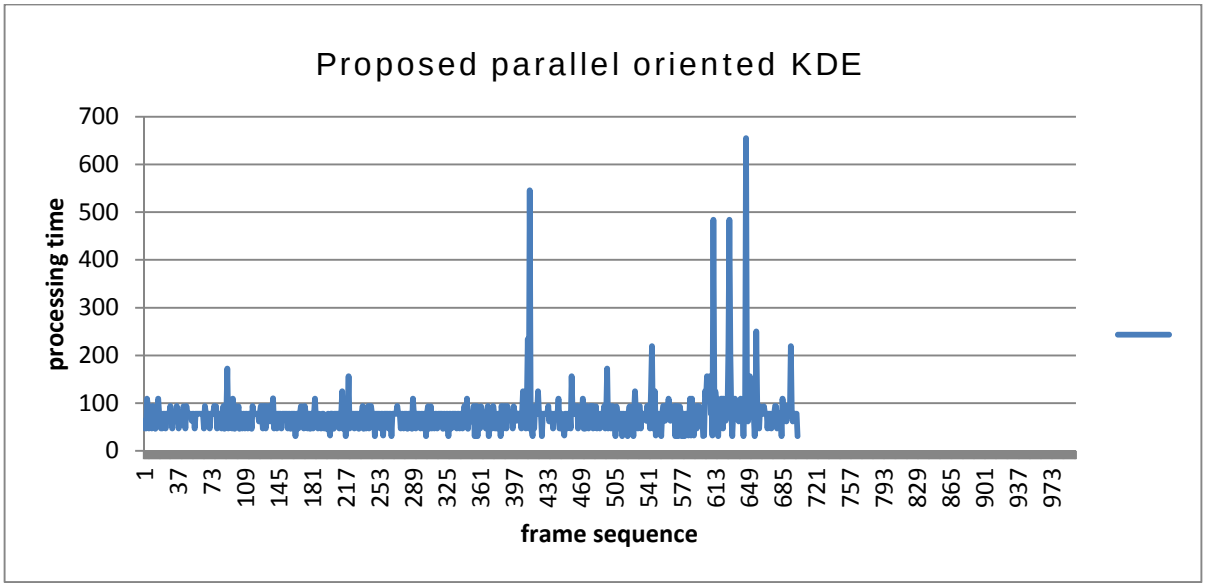


Figure 5.0 Processing time of proposed parallel KDE

Object Tracking Results


861 1.2 s
860 0.5 s
859 0.5 s
858 0.8 s
857 0.6 s
855 1.0 s
854 0.4 s
853 1.4 s
851 0.4 s
849 0.5 s
848 0.3 s
847 0.8 s
845 0.3 s
844 1.5 s
843 0.5 s

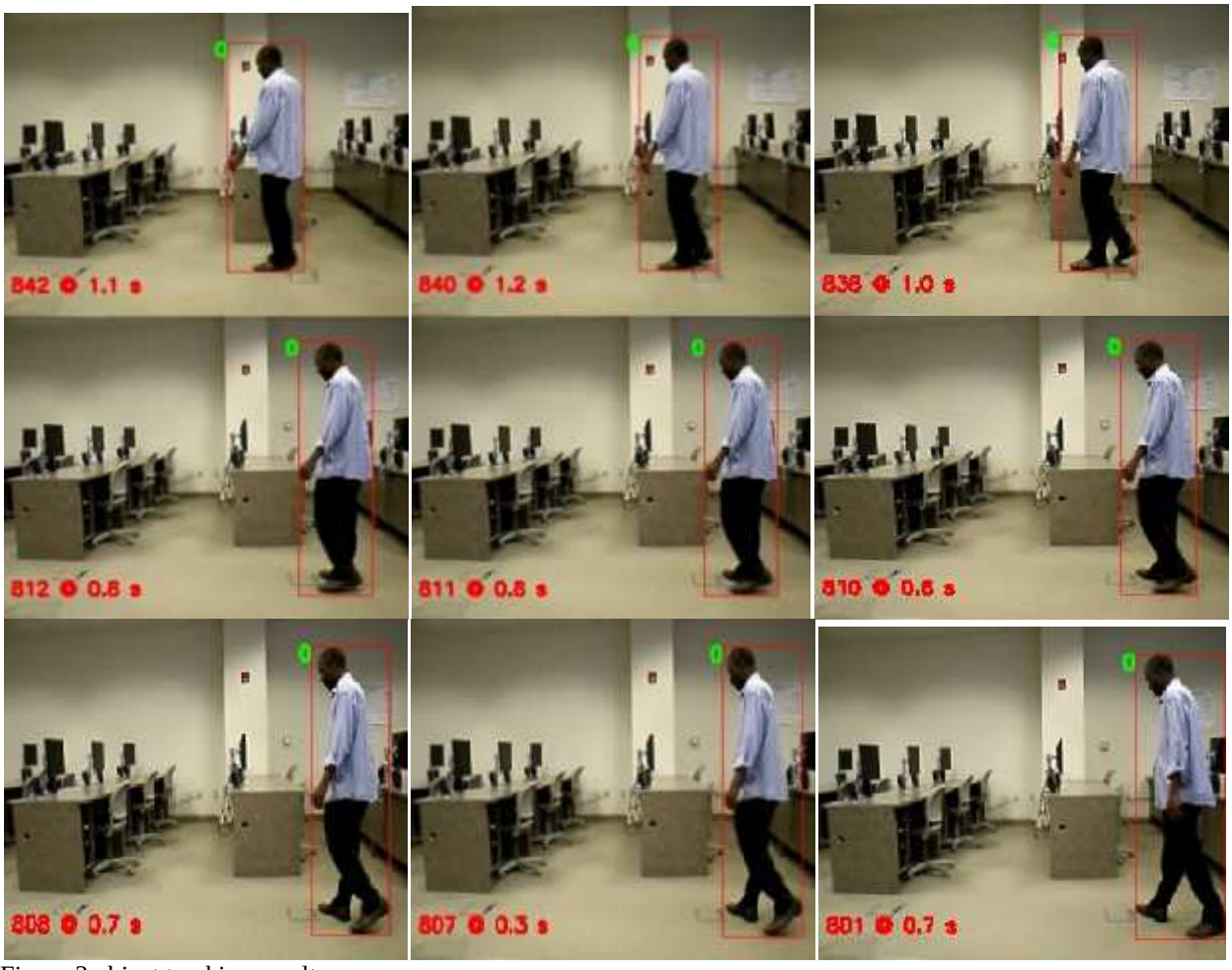


Figure 3 object tracking results

5.0 Performance Analysis of the object tracking Algorithms

The time required to process each frame was recorded over some frames. Figure 5.10 (a) presents the proposed spatio-colour histogram tracking algorithm. While (b) gives the particle filter based tracking algorithm. It is observed that the proposed algorithm takes 0.85 ms while particle filter based takes averaged of 4464 ms to process a frame. We conclude that particle filter is not too good for tracking real time objects which requires small processing time to detect and track the moving object within the scene.

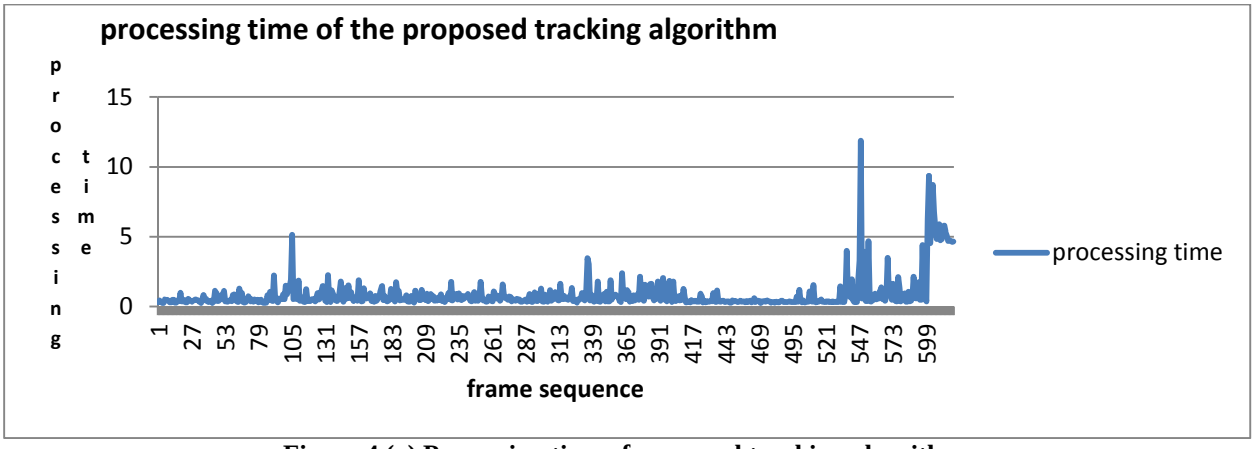


Figure 4 (a) Processing time of proposed tracking algorithm

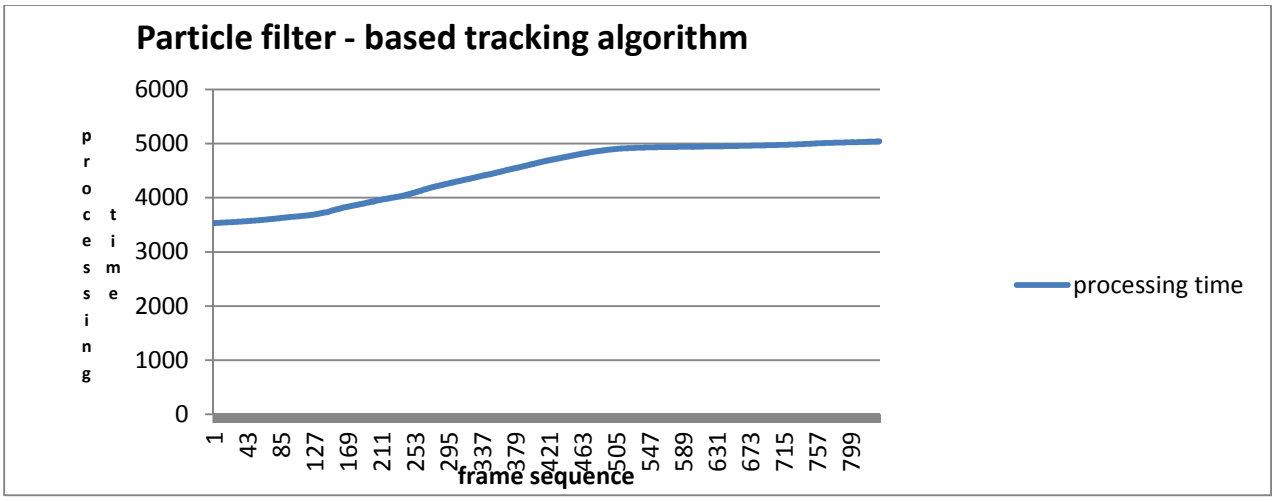
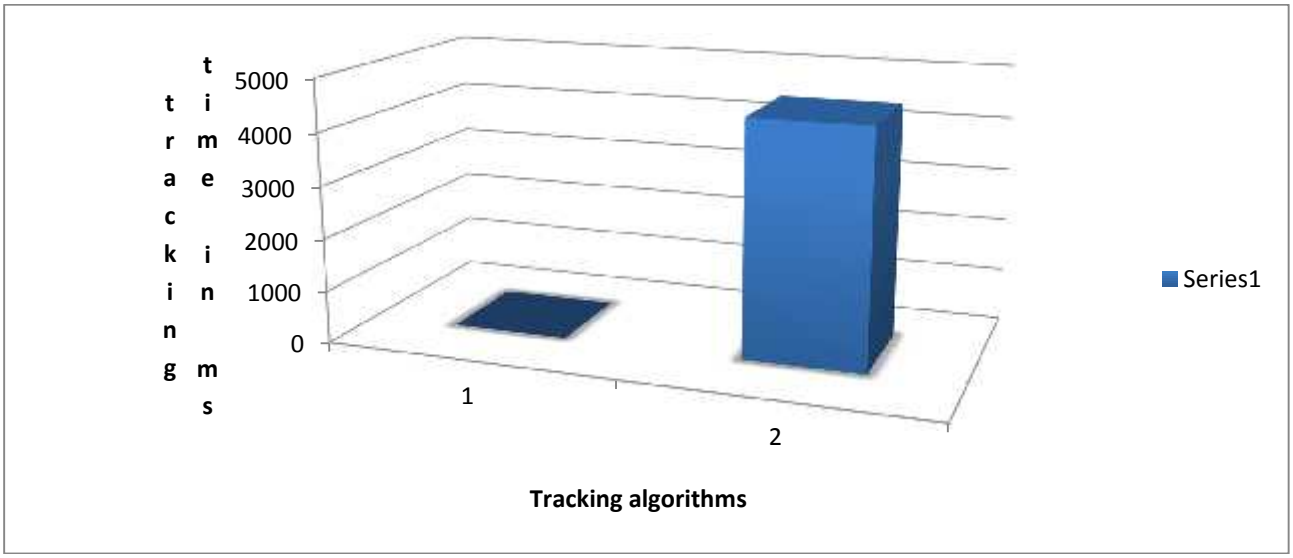


Figure 4 (b) Processing time of particle filtered –based tracking algorithm



6.0 **Conclusion** We have implemented object tracking algorithm algorithm in this paper using multi-part spatio-color algorithm together with motion detection. It is observed that the algorithm performs faster than particle filter based algorithm in terms of processing speed. This will be highly appreciated in a surveillance environment where time is a critical requirement. Future work can be done on this study by using feature based tracking algorithm.

REFERENCES

[1] Akintola K.G , Tavakollie A., (2011) , Robust Foreground detection in Videos using Adaptive Colour Histogram Thresholding and shadow removal 7th International Symposium on Visual Computing Las-Vegas, NV, September, 2011.

[2] Bradski, G.,(1998) Real time face and object tracking as a component of a perceptual user interface, in Proc. IEEE Workshop on Applications of Computer Vision, (WACV'98) Princeton ,NJ, October, 1998, pp. 214–219.

[3] Collins, R. T., Lipton, A. J., Fujiyoshi, H., & Kanade, T.(2000). A system for video surveillance and monitoring: VSAM final report. Technical report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, May 2000.

[4] Comaniciu, D., Ramesh, V. and Meer. P.(2000)Real-time tracking of non-rigid objects using mean shift. In Proc. Conf. Comp. Vision Pattern Rec., pages II:142–149, Hilton Head, SC, June 2000.

[5] Fleck S. Busch F, Biber P., Straßer W.,(2006) 3D Surveillance-A Distributed Network of smart cameras for Real–Time Tracking and its visualization in 3D proceedings of the 2006 conference on computer vision and pattern recognition workhop (CVPRW'06) Washington DC U.S.A.

[6] Fukunaga and Hostetler,(1975), The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition, IEEE Transactions on Information Theory vol 21 , pp 32-40 ,1975.

[7] Georis B., Desurmant X, Demaret D., Redureau S., Delaigle J., Moeq B (2003) Ip-Distributed computer-aided video-surveillance system Intelligent Distributed system workshop Lavon UK Feb., 26 2003.

[8] Haritaoglu I., David H., Larry S.D (1998) W^4: Who? When ? Where? What? A Real time System for detecting and tracking people International conference on face and gesture recognition April 14-16 1998 Nara Japan

[9] Perez P., Hue C., Vermaak J., and Gangnet M., (2002) "Color-Based Probabilistic Tracking," Proc. European Conf. Computer Vision, Copenhagen, Denmark, vol. I, pp. 661-675, 2002.

[10] Simon D (2010), "Kalman Filtering." *Embedded Systems Programming*, 2001. Retrieved `March 20, 2010 from the World Wide Web: tttp://www.embedded.com/9900168?_requestid=8285

[11] Yang C., Duraiswami R.,and Davis L.,(2005) Fast Multiple Tracking via Hierarchical particle Filter proceedings Of International Conference on Computer Vision – ICCV, pp. 212-219, 2005.