ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 4, June 2012

Intelligent System for detecting, Modeling, Classification of human behavior using image processing, machine vision and OpenCV

Niraj .B. Gadhe, Prof: Dr. B .K . Lande, Prof: Dr. B.B.Meshram

Abstract: **Surveillance Cameras has proven to be a key factor in enhancing the public security in many countries around the world . In spite of advancements in image processing and machine vision techniques very less is applied to actual implementation of such surveillance systems. Traditional methods wherein stand alone cameras provide just live feed to the observer are still deployed at most of the places but the need of today's world is to add some intelligence to this systems. In this paper we will be focusing on important features to build an intelligent surveillance system and its architecture using computer vision libraries of opencv. Background Subtraction , segmentation, tracking and human behavior analysis are step by step implemented to build such a system. A detailed analysis of background subtraction methods and their comparison is provided. Lastly we discuss the future work which can lead to human behavior analysis like crowd counting, intrusion detection, loitering , crowd movement , vehicle tracking and all.**

Index Terms: **surveillance systems, opencv, background subtraction, segmentation, machine vision, computer vision.**

I. INTRODUCTION

In traditional systems[1] a network of passive cameras able of only recording the moving images and converting it to video files are deployed at the site of prime importance. In case of accidents, terrorist attacks , theft, malicious activities such video data comes to be very important for analysis. But if traditional system is used we require to watch the video data over and over for hours together because collected data is very large and we don't know the exact time when the malicious activity took place. Such manual method is prone to human failure as human may miss small details like intrusion[2] or may neglect some activity. So its necessary to build an intelligent system that would not only provide data as per traditional system but also assist us to detect malicious activities. It would be so easy if system on its own recognizes some events as malicious and extract only those frames which are providing suspicious activity. The endless hours of watching the video can be avoided. Moreover assistance provided by system during live feed of surveillance may prove beneficial as it can alert the observer of a possible danger and drag his attention to a particular activity which he might have neglected in absence of intelligent system.

In the light of such weak security infrastructure we propose a method to develop an intelligent system which not only will capture and feedback the video at real time but also model human behavior.[2] We think of detecting , modeling and classifying human behavior using image processing to detect malicious behavior. Further we may maintain a database of suspects found over time and use face detection [24] and recognition to identify the probable matches for the suspects.

There is lot of work done in Image processing as well as machine vision techniques. A much of it is local and developed for particular domain. The object recognition frame works consist of algorithms like Pattern matching, feature matching , boundary detection, knowledge based repository systems.[18] The behavioral model uses heuristics and classification models like Motion based classification , Shape based classification, color based classification.[19][18] [17][14] Once the object is classified and detected it can be tracked using algorithms like mean shift algorithm.[10][22] Human behavior is studied under Bayesian Model and Markov Models.[12] [8] Face recognition and detection done via Eigen faces, SIFT based, Edge based, Binary pattern face recognitions, viola jones face detection are famous.[21][24]. Much of the work today is focused on detecting occluded object.[4]

The rest of the paper is organized as follows: Section 2 discusses our architecture of system and features we decided to incorporate in it. Section 3 presents a detailed analysis of famous background subtraction methods. . A detailed review of present image processing algorithms and their advantages are discussed . Section 4 we study segmentation of

Niraj Babasaheb Gadhe, Dept of computers- NIMS, VJTI, (e-mail: nirajgadhe@gmail.com). Mumbai.India,
Prof: Dr. B.K.Lande, Computer engg dept, Shah and anchor college of engg, University of Mumbai, Mumbai, India ,
Prof: Dr. B.B.Meshram, Computer engg dept, VJTI, Mumbai, India ,

object and motion tracking of object here. Section 5 discusses the behavior which can be detected using the system. Lastly we conclude with the future work.

II. BUILDING AN INTELLIGENT SURVEILLANCE SYSTEM

We are to develop a system based on image and video processing algorithms that will help us detect and classify human behavior. As detecting a human behavior from a scene we need to follow some sequential steps in our processing of the video. These sequential steps can be generalized and explained in brief as follows . [1]

A) Frame detection module.

This routine will detect each frame at real time from a video sequence and present its status in form of matrix, its frame no, pixels, color content and sample rate, etc.[23] Now we can treat each captured frame as a image data type and apply image processing on it.

B) Background Removal and segmentation module:

This will detect and remove background features to detect the object in the scene . Algorithms like frame differencing, averaging background method and GMM based will be a tool for such removal.[22] We are interested in analyzing human behavior.[2] . Different types of background subtracting algorithms[5] like Gaussian mixture model[6][11], adaptive Gaussian mixture model[4][20] and many more are available. Then we apply segmentation algorithms to separate moving objects from background. Segmentation can be carried out by Mean shift method [10], Bayesian method [8], wrapper based approach [7], etc.

C) Detecting human and their behavior module:

Once we get a human object in frame we can track the motion of such object in subsequent frames. We achieved this by drawing contour around the foreground object using connected component technique [13] of opencv.[22][23] Then we draw a bounding rectangle around moving object. Then we can use this rectangle parameters to detect the position of object in the scene. Ot her techniques like event detection using rejection based approach[3] , kalman filters[9], optical flow[22], object recognition methods[12][13][14], detection of salient object[15], shape filters[16][17][19], repository based object recognition[18] can be used.
We provide some parameters to system to detect some human behaviors of malicious intent. We will take a simple real world problem like Trespassing or intrusion, loitering, bag being left by a human and flee, crowd analysis. [2]

D) Capturing frames containing suspicious activities module:

Capturing frames containing probable suspicious behavior, classifying them and maintaining it for further use and alarming the system . Face detection and storing face data can be done.[24][21] Further such data can be used during investigation of crimes.
According to our architecture(FIG 1) the working of our system can be summarized as follows

1. We first acquire frames from a camera or video file.
2. We first acquire frames from a camera or video file.
3. Now we pass on these frames to background subtraction module. For convenience we are using the frame differencing for background subtraction.
4. We first acquire frames from a camera or video file.
5. Now we pass on these frames to background subtraction module. For convenience we are using the frame differencing for background subtraction.

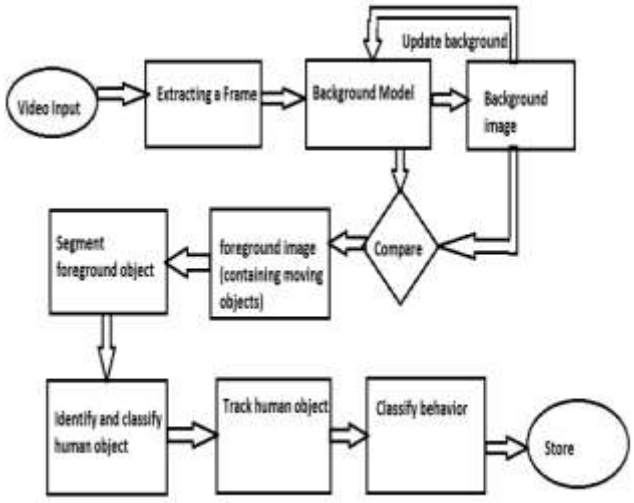

FIG: 1 Architecture of intelligent surveillance system.

6. We first acquire frames from a camera or video file.
7. Now we pass on this frames to background subtraction module. For convenience we are using the frame differencing for background subtraction.

8. Background subtraction separates background and gives us the frame with foreground object.
9. After background subtraction we detect object and draw a counter using connected point concept and opencv modules.
10. After this we draw a bounding rectangle around the counter and save its center .
11. Now we track motion of moving object using the coordinates of the centre.
12. Once we track the motion we can analyse the behavior of the person in the scene and study his interaction with the environment.

III. BACKGROUND SUBTRACTION ANALYSIS

This section focuses on how to isolate objects or parts of objects from the rest of the image.[22] We want to isolate those events and to be able to ignore the endless hours when nothing is changing. We start with separating foreground objects from learned background scenes. We will be dealing with three types of background subtraction methods for our analysis. We will implement frame differencing, averaging background method and mixture of Gaussian(MOG) model and study the results for different cases.

Absolute differencing method or Frame differencing:

Basic idea here is to subtract one frame from another subsequent frames and label any difference that is big enough (threshold) as Foreground .[1] This method mainly catches motion in the frames. For simplicity, let's say we have three single-channel images: frame1, frame2, and Foreground. We could then use the following code to detect the magnitude (absolute value) of foreground differences in Foreground: Opencv's cvAbsDiff( ) function can be used for such differencing.[22]. To reduce noise, we can apply threshold concept. This removes unwanted pixels which add to noise in our data. Thresholds are manually adjusted to suit our needs. cvThreshold( ) is a tool for applying thresholding. We will clean up small noise areas with cvErode( ) Function .

Algorithm:

1. Grab a random initial frame and save it in an image datatype and name it as "background frame".
2. Now for each other frame do the following
   i. Grab current frame. Save it in an image and name it "current frame" .
   ii. Take absolute difference between current and background frame and save it in a "foreground frame". Apply threshold and erosion if needed.
   iii. Display the foreground frame. Move to next frame.
3. Stop when all frames are over and release all the data types to avoid stack overflow.

Experimental results:

A. Indoor scene with minimum fluctuations:

.FIG 2 Without threshold gray level image

FIG 3 After applying threshold : binary image TH=80

FIG 4 After applying threshold and erosion technique

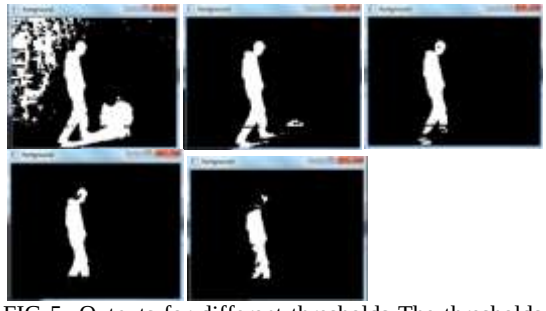

FIG 5 Outputs for different thresholds The thresholds for images from Left to right and top to bottom are Th=10,Th=30,Th=60,Th=80,Th=110.

Experimental Conclusions: [Fig 2,3,4,5]

1. Simple gray level absolute differencing causes foreground detection with shadow effects and noise.
2. Thresholding reduces noise and gives binary image which is more predictable.
3. Selection of thresholds depends upon lighting conditions.
4. Erosion technique further helps reducing noise.

Our sample video shows following results:[Fig 5]

1. Th <30 gives foreground detection with lot of noise.
2. Th>30 and Th<60 gives satisfactory results. Still has noise contents.
3. Th >60 and Th<80 gives good results without noise. But shows little loss of foreground data.
4. Th> 110 Gives only foreground data with great loss but without noise.
5. This means that 30<Th<60 is a better window for indoor scenes to achieve better foreground detection.

Advantages:

Simplest method for background subtraction. Less memory requirement as only initial frame is considered as background. Can work well for indoor lighting conditions well.

Disadvantages: Lighting fluctuations are not handled properly.Threshold depends upon lighting conditions.Cannot work well for outdoor scenes where background may change considerably.

2. Averaging background method

The averaging method makes use of four OpenCV routines: [22] [23]

1. **cvAcc(),** to accumulate images over time;
2. **cvAbsDiff(),** to accumulate frame-to-frame image diff erences over time;
3. **cvInRange(),** to segment the image (once a background model has been learned) into foreground and background regions;
4. **cvOr()**, to compile segmentations from different color channels into a single mask image.

Finding Mean , variance and covariance:

To compute a mean[22] value for each pixel across a large set of images, the easiest method is to add them all up using cvAcc() and then divide by the total number of images to obtain the mean. We can also accumulate squared images, which will allow us to compute quickly the variance of individual pixels. For finding the covariance[22] we can also see how images vary over time by selecting a specific *lag* and then multiplying the current image by the image from the past that corresponds to the given lag. Th e function cvMultiplyAcc() will perform a pixelwise multiplication of the two images and then add the result to the "running total" in acc:[1][22] [5]

Algorithm:

i. First, we create pointers for the various scratch and statistics-keeping images we will need along the way.

ii. we create a single call to allocate all the necessary intermediate images. For convenience we pass in a single image (from our video) that can be used as a reference for sizing the intermediate images. This is done in allocate images routine.

iii. Next we learn the accumulated background image and the accumulated absolute value of frame-to-frame image differences. This is done in accumulatebackground routine.

iv. We first use cvCvtScale() to turn the raw background 8-bit-per-channel, three-colorchannel image into a floating-point three-channel image. We then accumulate the raw floating-point images into IavgF. Next, we calculate the frame-to-frame absolute difference image using cvAbsDiff() and accumulate that into image IdiffF. Each time we accumulate these images, we increment the image count Icount, a global, to use for averaging later

v. Once we have accumulated enough frames, we convert them into a statistical model of the background. That is, we compute the means and deviation measures (the average absolute diff erences) of each pixel: using createmodel from stats routine.

vi. setHighThreshold() and setLowThreshold() are utility functions that set a threshold based on the frame-to-frame average absolute diff erences. The call setHighThreshold(7.0) fi xes a threshold such that any value that is 7 times the average frame-to-frame absolute diff erence above the average value for that pixel is considered foreground; likewise, setLowThreshold(6.0) sets a threshold bound that is 6 times the average frame-to-frame absolute difference below the average value for that pixel.

vii. Once we have our background model, complete with high and low thresholds, we use it to segment the image into foreground (things not "explained" by the background image) and the background (anything that fits within the high and low thresholds of our background model). Segmentation is done by calling backgrounddiff routine.

viii. This function first converts the input image I (the image to be segmented) into a floating-point image by calling cvCvtScale(). We then convert the three-channel image into separate one-channel image planes using cvSplit(). These color channel planes are then checked to see if they are within the high and low range of the average background pixel via the cvInRange() function, which sets the grayscale 8-bit depth image Imaskt to max (255) when it's in range and to 0 otherwise. For each color channel we logically OR the segmentation results into a mask image Imask, since strong differences in any color channel are considered evidence of a foreground pixel here. Finally, we invert Imask using cvSubRS(), because foreground should be the values out of range, not in range. The mask image is the output result.

ix. Lastly we de-allocate all the variables and pointers.

FIG: 6 Averaging background method (Background is average of first 10 frames)

Advantages:

1. Builds a unique model of background depending upon earlier "n" frames
2. Predicts the foreground based on lighting condition changes in consecutive frames.
3. Works well if more frames are taken for averaging the background**.**

Disadvantages:

1. More complex calculations.
2. Needs more memory and time to execute than previous method.

3. Advanced background subtraction -Mixture of Gaussian:

Gaussian Function:

In mathematics, a **Gaussian function** (named after Carl Friedrich Gauss) is a function of the form:

$$f(x) = ae^{-\frac{(x-b)^2}{2c^2}}$$

for some real constants *a*, *b*, *c* > 0, and *e* ≈ 2.718281828 (Euler's number).The graph of a Gaussian is a characteristic symmetric "bell curve" shape that quickly falls off towards plus/minus infinity. (FIG 7)The parameter *a* is the height of the curve's peak, *b* is the position of the centre of the peak, and *c* controls the width of the "bell". Gaussian functions are widely used in statistics where they describe the normal distributions, in signal processing where they serve to define Gaussian filters, in image processing where two-dimensional Gaussians are used for Gaussian blurs, and in mathematics where they are used to solve heat equations and diffusion equations and to define the Weierstrass transform.

Running Gaussian average was first proposed by Wren , Azarbayejani, Darell, pentland in 1997 which fits one Gaussian distribution (μ, σ) over the histogram.[1]Generalized Gaussian model was proposed by H.Kim, R. Sakamoto et all[1][4] to cope up with the background changes and shadows using this model. In Mixture of Gaussian (MOG) model background is termed as parametric frame of values where each pixel is represented with number of Gaussian Functions as Probability Distribution function.[6]

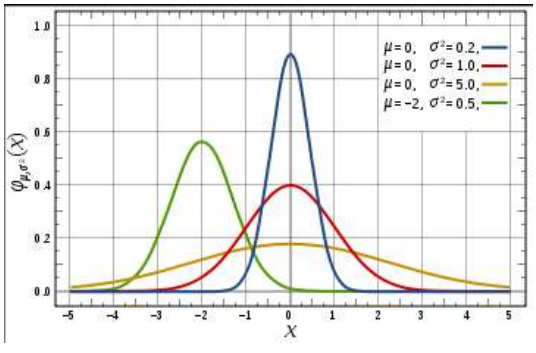

FIG: 7 Normalized Gaussian curves with expected

value μ and variance σ^2. The corresponding parameters are $a = 1/(\sigma\sqrt{(2\pi)})$, $b = \mu$, $c = \sigma$[25]

Mixture of Gaussian basics:

Sometimes different objects are likely to appear at the same pixel (i,j) location. For example, in an outdoor scene with trees partially covering a building: a same (i,j) pixel location will show values from tree leaves, tree branches, and the building itself. Other examples can be easily drawn from snowing, raining, or watching sea waves from a beach. In these cases, a single valued background is not an adequate model. Stauffer and Grimson [1] [22] [26] raised the case for a multi-valued background model able to cope with multiple background objects. Stauffer and Grimson describe the probability of observing a certain pixel value, x, at time t by means of a mixture of Gaussians. All weights wi are updated (or normalised) at every new frame. In practical cases, K is set to be between 3 and 5. Gaussians are multi-variable to describe red, green and blue values. If these values are assumed independent, the co-variance matrix, ∑i,j simplifies to diagonal.[26]

At each t frame time, two problems must be simultaneously solved a) assigning the new observed value, x,, to the best matching distribution and b) estimating the updated model parameters. These concurrent problems can he solved by AN EXPECTATION MAXIMIZATION (EM) algorithm. AN EXPECTATION MAXIMIZATION works on the buffer of the last n frames. However, as this would prove extremely costly, the matching is approximated in these terms $(X_t - \mu_{i,t}) / \sigma_{i,t} > 2.5$

The first in ranking order is accepted as a match for x_t . furthermore the parameters $(w_{i,t}\mu_{i,t}\sigma_{i,t})$ are updated only for this matching distribution and by using simple online cumulative averages . At every new frame, some of the Gaussians "match" the current value (those at a distance < 2.5 σi): for them, μi ,σi are updated by the running average. If no match is found the last ranked distribution is replaced by a new one centered in x_t , with low weight and high variance. The mixture of Gaussians actually models both the foreground and the background All distributions are ranked according to their ωi /σi and the first ones chosen as "background" .[26]

Algorithm:

i. For background subtraction ***CvGaussBGStatModelParams*** is used where it takes some parameter like ***bg_threshold*** and ***std_threshold***. This two parameters are responsible to make difference between background and foreground.
ii. There is another function called ***cvCreateGaussianBGModel*** which takes video frame to create standard background model according to parameters.
iii. After that, extracting foreground ***cvUpdateBGStatModel*** function is used where it takes video frame and background model to compute differences frame by frame.
iv. As an output ***bgModel->background*** represent background of the video and ***bgModel->foreground*** shows foreground of the video

Our chosen Gaussian parameters:

params->win_size=2; - Learning rate; alpha = 1/CV_GBG_WINDOW_SIZE .
params->n_gauss=5; = K = number of Gaussians in mixture .
params->bg_threshold=0.7; threshold sum of weights for background test .
params->std_threshold=3; lambda=2.5 is 99% params->minArea=15;
params->weight_init=0.25;
params->variance_init=30;
CV_BGFG_MOG_SIGMA_INIT*CV_BGFG_MOG_SIGMA_INIT

Results:

Fig:8 mixture of Gaussians background subtraction method

We kept all parameters constant and only changed one parameter at a time to see the probable changes in the output and best values for the parameters. We concluded following results from our observations.

Mixture of gaussian parameter analysis:

Paramter	Observed best value range
threshold sum of weights for background test (bg_threshold)	**0.1 to 0.7**
Standard threshold(std_threshold)	**2 to 5**
Number of Gaussians(n_gauss)	**3 to 6 (3 is better)**
Alpha-learning rate	**0.3 to 0.7(0 to values)**

Advantages: [Fig 8]

1. Best background prediction.
2. Model can be adapted to specific scenes by changing the parameters.
3. Least amount of noise incurred when parameters are adjusted accordingly.
4. Works well in all lighting conditions.

Disadvantages:

1. Takes largest time and memory among all the methods implemented by us.
2. Parameter adjustment is a trial and error method .
3. Needs high processing speeds for real time environments.

Summary of our analysis:

	Frame differencing	**Averaging background method**	**Mixture of Gaussians method**
Complexity	Very less	Medium	More
Memory requirements	Very less	Medium	More
Execution time	Fast	Medium	Very slow
Parameter selection	Threshold is only needed.	Few Needed .	Many parameters are needed

Sample output	 Better	 Good	 Best

IV. SEGMENTATION USING CONTOURS AND MOTION TRACKING.

A contour is a list of points that represent, a curve in an image. [22][23]A contour is represented in OpenCV by a CvSeq sequence that is, a sequence of points. The function **cvFindContours()** [22] computes contours from binary images. The return value of cvFindContours() is the total number of contours found. The first argument is the input image. The next argument, storage, indicates a place where cvFindContours() can find memory in which to record the contours. In a binary image white regions are labeled as contour and black lines as holes . One of our most basic tasks is drawing a contour on the screen. For this we have cvDrawContours( ).

Motion Tracking:[fig 9]

Algorithm:

A) Find a rectangle around counter:

1. **BoundingRect()** Calculates up-right bounding rectangle of point set. Here we use it to draw a bounding rectangle around the counter we just got in above step.
2. **BoundingRect()** takes two parameters**:**
   1. **Counter type and name=** here we have taken counter type of cvseq type.
   2. **Update flag**= update flag is zero in case of cvseq type.

B) Finding the centre of rectangle:

1. Boundingrect function will return us with an object say BNDRECT.
2. BNDRECT has following information
   X coordinate of left upper vertex
   Y coordinate of left upper vertex
   Width
   Height

3. Calculate Point 1= (X, Y)
4. Calculate point2=(X+width, Y+height)
5. Calculate Centre on X axis(here for our convenience)= avgX=(point1.x+point2.x)/2
6. This avgX can be used as a parameter to detect the position of moving object we have detected.
7. Lastly for visibility draw a rectangle using point 1 and point 2 on moving counter.

Results:

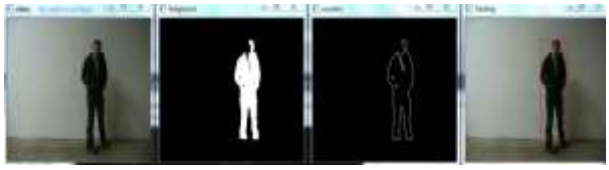

FIG:9 First frame shows raw video, second shows foreground detected, third shows contour and fourth shows a rectangle drawn around the person in the running video.

V. BEHAVIOR DETECTION – DETECTING SUSPICIOUS ACTIVITY.

Now as we have done with detection and tracking of moving object we can predict the motion and behavior of moving object.

1. Trip Wire detection:

We will take a simple scenario where one is not supposed to enter a prohibited area. [22][Fig 10]We will take this problem as Crossing the green line and or being present on prohibited area will be termed as intrusion. We will be checking avgX value and monitoring it. If the value is greater or equal to any coordinates of trip wire we will term the action as intrusion:

Result:

Fig: 10The person enters a prohibited area at a particular time and so an alarm is generated.

2. Loitering activity:

Loitering is defined as the presence of an individual in an area for a period of time longer than a given time threshold. [2][Fig 11]For example loitering can be possible at museums, hotels, shops, at car parking, hospitals, monuments and business places. Such suspicious events where a person stands at an area for a longer duration can be caught by providing such thresholds and defining the area of importance within the scene.

FIG:11 first frame shows person enters an area(marked by green rectangle). This area is area of strategic importance. Second frame shows person still inside the area. The third frame shows a "loitering" alert generated when the person stays within the area beyond a certain threshold.

As soon as we get a suspicious activity we will term the frame as suspicious and display it in a window. We will extract the intruder image and save it using bounding rectangle coordinate. Along with suspicious frames for further use we can also save the timing, date, and location data for further ease during investigations.

3. Abandoned baggage or boxes:

Here the system continuously checks for motion in the frames, as well as checks for objects that are not present before. Any stationary object not present before can act as a threat for the public. So such baggage or box can be termed as harmful and can be highlighted.[Fig 12]

Face Detection:

If face is detectable [24] we will detect and save it too. These suspicious frames, intruders and face data can be used for future use in case of emergency.

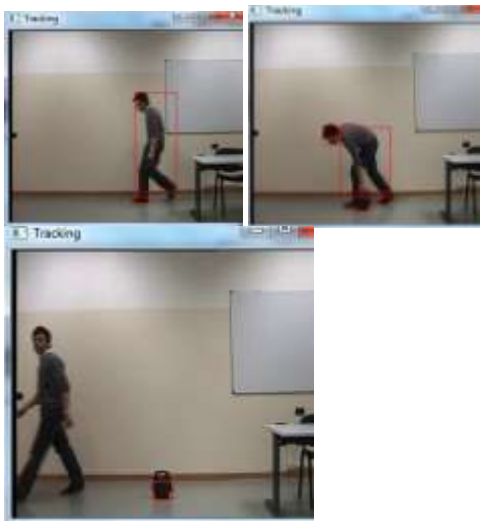

FIG: 12 The first frame shows a person carrying a bag, second shows the person keeping the bag on the floor, the third frame shows bag being detected as abandoned object by the system.

The face data is detectable using HAAR classifiers [22] and viola jones algorithms[22][24] provided by opencv library. [Fig 13]The haar classifier used is haarcascade_frontalface_alt2.xml file which can detect frontal faces in an image. Whenever a face is detected we draw a rectangle around it and save it. Future work may comprise of supplying such face data to a face matching system which will find out correct matching faces of the suspicious person. Future work also focuses mainly on creating concrete classifiers for detecting human and non human objects in a scene.

Algorithm:

1. Create a cascade of haar classifiers[22] using haarcascade_frontalface_alt2.xml file and CvHaarClassifierCascade* cascade = (CvHaarClassifierCascade*)cvLoad("haarcascade_frontalface_alt2.xml");
2. Detect objects using cvHaarDetectObjects( );
3. This will return number of objects and the objects will be stored in memory.
4. For each object detected(each face detected) draw a rectangle bounding the face.

Result:

FIG: 13 Faces are detected and rectangle are drawn bounding the faces.

CONCLUSION

Thus we have shown that use of present day machine vision algorithms along with opencv can lead us to develop an intelligent surveillance infrastructure. Yet a lot of work needs to be done in refinement of this behavior detection process. More and more behavior needs to be considered and identified.

ACKNOWLEDGMENT

I would like to thank all those people whose support and cooperation has been an invaluable asset during the course of this paper. I would also like to thank my Guide **Prof. Dr. B.K.Lande** and our dept head **Dr: B.B. MEshram** for guiding me throughout this paper and giving it the present shape. It would have been impossible to complete the paper without their support, valuable suggestions, criticism, encouragement and guidance.

I am also grateful for all other teaching and non-teaching staff members of the Computer Technology for directly or indirectly helping us for the completion of this project and the resources provided.

REFERENCES

[1] Niraj Gadhe, Dr:B.K . Lande, VJTI, IOSR Journal of Engineering (IOSRJEN),Vol. 2 Issue 2, Feb.2012

[2] Joshua Candamo, Matthew Shreve, *Member, IEEE*, Dmitry B. Goldgof, *Fellow, IEEE*, Deborah B. Sapper, and Rangachar Kasturi, *Fellow, IEEE* "Understanding Transit Scenes: A Survey on Human Behavior-Recognition Algorithms" in IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 11, NO. 1, MARCH 2010

[3] Margarita Osadchy and Daniel Keren "A Rejection-Based Method for Event Detection in Video" in IEEE TRANSACTIONS

ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, VOL. 14, NO. 4, APRIL 2004

[4] H. Kim, R. Sakamoto, I. Kitahara, T. Toriyama and K. Kogure "Background subtraction using generalized Gaussian family model" in ELECTRONICS LETTERS 31st January 2008 Vol. 44 No. 3

[5] Shahrizat Shaik Mohamed, Nooritawati Md Tahir, Ramli Adnan "Background Modelling and Background Subtraction Performance for Object Detection" in 2010 6th International Colloquium on Signal Processing & Its Applications (CSPA)

[6] Dar-Shyang Lee, Member, IEEE "Effective Gaussian Mixture Learning for Video Background Subtraction " in IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 27, NO. 5, MAY 2005

[7] Michael E. Farmer, *Member, IEEE,* and Anil K. Jain, *Fellow, IEEE "*A Wrapper-Based Approach to Image Segmentation and Classification" in IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 14, NO. 12, DECEMBER 2005

[8] A. Colmenarejo, M. Escudero-Vin˜olo and J. Besco´s "Class-driven Bayesian background modelling for video object segmentation" in ELECTRONICS LETTERS 1st September 2011 Vol. 47 No. 18

[9] Jesse Scott et all "Kalman Filter Based Video Background Estimation" in 2007IEEE

[10] Madhurima , et all "Object tracking in a video sequence using Mean-Shift Based Approach" in IJCEM International Journal of Computational Engineering & Management, Vol. 11, January 2011

[11] Sen-Ching S. Cheung and Chandrika Kamath "Robust techniques for background subtraction in urban traffic video"

[12] Padmini Jaikumar, Abhishek Singh and Suman K Mitra "Background Subtraction in Videos using
Bayesian Learning with Motion Information" in 2008.

[13] Scott Helmer and David G. Lowe "Object Class Recognition with Many Local Features" in Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops.

[14] Lun Zhang, Stan Z. Li, Xiaotong Yuan and Shiming Xiang "Real-time Object Classification in Video Surveillance Based on Appearance Learning" in 2007 IEEE

[15] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Fellow, IEEE,et all "Learning to Detect a Salient Object" in IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 33, NO. 2, FEBRUARY 2011

[16] *Zhiyong Wang, Zheru Chi and David Feng "*STRUCTURAL REPRESENTATION AND BPTS LEARNING FOR SHAPE CLASSIFICATION" in Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'O2), Vol. 1

[17] Cosmin Grigorescu, *Student Member, IEEE,* and Nicolai Petkov "Distance Sets for Shape Filters
and Shape Recognition" in IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 12, NO. 10, OCTOBER 2003

[18] Chensheng Wang, Fei Wang "A Knowledge-based Strategy for Object Recognition and Reconstruction" in 2009 International Conference on Information Technology and Computer Science

[19] Junqiu Wang, *Member, IEEE,* and Yasushi Yagi, *Member, IEEE "*Integrating Color and Shape-Texture Features for Adaptive Real-Time Object Tracking" in IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 17, NO. 2, FEBRUARY 2008

[20] Zoran Zivkovic "Improved Adaptive Gaussian Mixture Model for Background Subtraction" in Proc. ICPR, 2004

[21] Implementing the Viola-Jones Face Detection Algorithm Ole Helvig Jensen Kongens Lyngby Technical University of Denmark, 2008,

[22] Book on " Learning Open Cv – Computer vision with Open CV library" by Gary Bradski and Adrian Kaebler – o'reilly

[23]Book on "OpenCV 2 Computer Vision Application Programming Cookbook" by Robert Laganière-PACKT PUBLISHING.

[24] Image Processing Project on "Face Recognition Techniques "by Jorge Orts , Univerity of Wisconsin, Madison.

[25]www.wikipedia.org

[26] "MIXTURE OF GAUSSIANS USING OPENCV" AS IN HTTP://MMLAB.DISI.UNITN.IT/WIKI/INDEX.PHP/MIXTURE_OF_GAUSSIANS_USING_OPENCV.

[27] IMAGE AND VIDEO DATABASES OF CVONLINE: IMAGE DATABASES HTTP://HOMEPAGES.INF.ED.AC.UK/RBF/CVONLINE/IMAGEDBASE.HTM#RETRIEVE

[28] Surveillance databases – VISOR- video urviellance online repository-
http://imagelab.ing.unimore.it/visor/video_videosInCategory.asp?idcategory=14