

Pedestrian Dynamics to Detect Automatic Crowd Formation

By

Shovan Sarkar

ID: 10101004

And

Md. Refaat Ali

ID: 11101028

Supervisor

Dr. Hanif Bin Azhar

Assistant Professor

Dept. of Computer Science and Engineering



Department of Computer Science & Engineering

April 2014

BRAC University, Dhaka, Bangladesh

Declaration

We do hereby declare that the thesis titled “Pedestrian Dynamics to Detect Automatic Crowd Formation” submitted to the Department of CSE of BRAC University in partial fulfillment of the Bachelor of Science in Computer Science and Engineering. This is our original work and was not submitted elsewhere for the award of any other degree or any other publication.

Date:

Supervisor Dr. Hanif Azhar

Shovan Sarkar

Student ID: 10101004

Md. Refaat Ali

Student ID: 11101028

ACKNOWLEDGEMENT

We wish to acknowledge the efforts of our supervisor Dr.Hanif Azharfor his contribution, guidance and support in conducting the research and preparation of the report. We also wish to acknowledge our chairperson, Professor Mohammad Zahidur Rahman for his direct or indirect support and inspiration. we are very thankful to our parents, family members and friends for their support in completing the work. Finally, we are very thankful to BRAC University, Bangladesh for giving us a chance to complete our B.Sc. degree in Computer Science and Engineering.

ABSTRACT

In our work, we are trying to implement such a system that itself can predict whether there is a chance of forming a crowd in a public place or not, which will enrich the entire security system in such a way that the authority will be able to take precautions and avoid unexpected incidents. This system will work by using security cameras to take live streaming, identify each moving elements and analyze their movements. First we will take live streaming from security cameras and then we split the video into frames. Here we can split the video into 25-35 frames per second of playing time. Then we analyzed each frames and identify each moving object and assign them with a unique object identification number. We also highlight the object with a rectangular box so in order to make it distinctive. We collect the data of each frame and save them in a data structure. Our algorithm does calculations on those data and predicts the next possible location of each object and therefore we can predict the possible location for a crowd. Apart from crowd detection, our research also extracts features from the movement of each objects (in this case pedestrian) and also predicts what will be the position of the object in future , wheather a crowd has already formed or not , if someone is suspiciously standing in a busy way etc.

Table of Contents

Sections	Topic	Page
1	Introduction	8
1.1	Objective	10
1.2	Contribution	10
1.3	Thesis outline	11
2	Background Theories	12
2.1	Kalman Filter	12
2.2	Gaussian Mixture Model	13
2.3	Object detection and tracking	15
3	Literature review	18
3.1	Data Driven crowd Analysis in videos	18
3.2	Flow based methods	19
3.3	Overlapping Motion Patterns	19
3.4	Counting crowded moving objects	20
4	Technical overview	21
4.1	Proposed System	21
4.2	Previous work	22
4.3	Flow Chart	23
4.4	Procedure	24
4.5	Detection and Tracking	24
4.6	Motion Analysis	25
4.7	Linear motion predicting algorithm	25
4.8	Moving average predicting algorithm	27
4.9	Extrapolation	28
4.10	Extrapolation with Linear Regression	29
4.11	Raising Threat Signals	30
5	Result Analysis	33
5.1	How Our System is Unique	33
5.2	Data Set Analysis & Report	33
5.2.1	Prediction	33
5.2.2	Raising alarm	38
5.3	Limitations And Challenges	42
5.3.1	Moving Camera	42
5.3.2	Overlapping Objects	42
5.3.3	Camera Position	42
6	Conclusion and Future Works	43
7	References	44

Table of Figures

Sections	Topic	Page
Figure 1	Crowd formation in public place	8
Figure 2	Crowd gathering a person	9
Figure 3	Basic concept of Kalman filtering	12
Figure 4	Gaussian Mixture Model	14
Figure 5	Foreground detection and blob analysis	16
Figure 6	Flow chart	23
Figure 7	Linear motion prediction algorithm	26
Figure 8	Moving Average Prediction Algorithm	27
Figure 9	Extrapolation example	29
Figure 10	Extrapolation with Linear Regression	30
Figure 11	detecting the green level risk using prediction	31
Figure 12	Prediction graph	38
Figure 13	Accuracy graph of crowd formed	39
Figure 14	Accuracy graph of Predicting crowd	40
Figure 15	Accuracy graph of suspicious behavior	41

List of tables

Sections	Topic	Page
Table 1	Accuracy of linear motion analysis.	34
Table 2	Accuracy of Linear Motion Prediction Algorithm	35
Table 3	Accuracy of Extrapolation.	36
Table 4	Accuracy of Linear Regression.	37
Table 5	Accuracy in detecting crowd already formed	39
Table 6	Accuracy in predicting Suspicious behavior	40
Table 7	Accuracy in predicting crowd formation	41

1. Introduction

Computer vision algorithms have become very popular in the field of surveillance and security system. This is due to the growing concerns for public safety and security in places like airports, train stations, malls, stadiums, streets, etc. In such scenes, conventional computer vision techniques for video surveillance cannot predict the formation of a crowd.

In our modern city-life people are more concerned about safety and security. Especially in the developing country like ours, the people are always anxious in public places. Due to political instability streets are becoming hostile and danger can come from anywhere any time. There are traditional CC cameras here and there which are very slow and needs a set of eye monitoring them to detect any crime or risk.



Fig 1: Crowd formation in public place



Fig 2: Crowd gathering a person

Now suppose an unavoidable circumstance has occurred and people need immediate attention but due to the inefficiency of conventional computer vision techniques for video surveillance by the time the authority detects any danger it's already too late. So the authorities can't take proper steps to stop crimes leaving us helpless. Our system voluntarily predicts whether there will be any crowd and also pin points that exact location so that authority can prepare for it and prevent it from happening.

1.1. Objectives

- Predicting next position of object
- Predicting possible location for crowd formation
- Detecting a crowd that being formed
- Detecting suspicious activity

Overall the main objective of our project is to solely devolve the security system over the traditional current security system which is now being used in the surveillance system.

1.2. Contribution

We have implemented our work in Matlab version 13b and in our work we have used many functions that are already done in Mathlab which are very useful. The most important functions were video reader foreground detector and blob analysis. We have used the output given by these functions to complete our work. First of all we stored the data of each moving object and for that we had to design a new data structure. After storing the information we predicted the position of each moving object in certain frames and for that we developed a few motion analysis algorithms. Here we also used some of the conventional mathematical formulas like extrapolation, regression etc. Later on we made our decisions based on fulfilling certain conditions and displayed threat signals accordingly. In our work we also studied different theories like Gaussian mixture model and Kalman filtering which were very helpful. So we can say that our overall contribution in the entire project was to extract and store the data, to analyze the motion behavior of each object and to develop the necessary algorithm for our final decisions. Besides there were some lacking in the fore ground detection function we used as sometimes it fails to detect the object. We also came up with a solution to this problem. All of this is discussed in the Technical overview section.

1.3. Thesis Outline

The structure of the paper is as follow. Section 2 presents background theories, and previous works related to ours. In section 3, the literal review describes the review of some recent work done in this field is mentioned. In section 4, technical overview of our system i.e. the details of the algorithms, methods used are explained and the detailed procedures of how we applied the algorithms and designed our system are mentioned. In section 5, the experiments and result analysis is given. Finally the conclusion and future work is mentioned in section 6.

2. Background Theories

For our work we went through many theories which helped us to understand the concept of object detection and crowd dynamics more clearly.

2.1. Kalman filter

The Kalman filter^[1] is designed for tracking. You can use it to predict a physical object's future location, to reduce noise in the detected location, or to help associate multiple physical objects with their corresponding tracks. A Kalman filter object can be configured for each physical object for multiple object tracking. To use the Kalman filter, the object must be moving at constant velocity or constant acceleration.

The Kalman filter algorithm involves two steps, prediction and correction (also known as the update step). The first step uses previous states to predict the current state. The second step uses the current measurement, such as object location, to correct the state. The Kalman filter implements a discrete time, linear State-Space System.

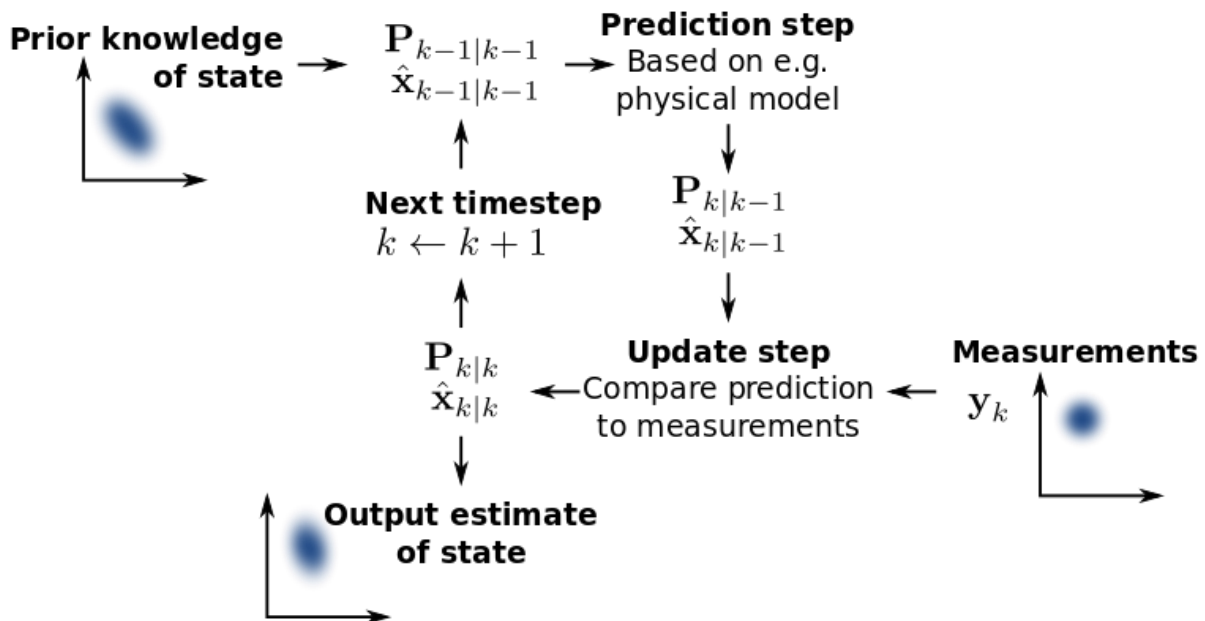


Fig 3: Basic concept of Kalman filtering

The Kalman filter keeps track of the estimated state ^[2] of the system and the variance or uncertainty of the estimate. The estimate is updated using a state transition model and measurements. $\hat{x}_{k|k-1}$ denotes the estimate of the system's state at time step k before the k -th measurement y_k has been taken into account; $P_{k|k-1}$ is the corresponding uncertainty.

One of the important lacking of Kalman filter is that the object must be moving at constant velocity or constant acceleration, but the movement of the pedestrian in public places like streets, malls etc. are not constant in fact the movement is very much random. On the other hand our algorithm does not constant movement of the objects (pedestrians) rather it is functional any kind of random movement, but in this case the accuracy is slightly compromised.

2.2 Gaussian Mixture Models

In our work we have used the built-in Matlab function foreground detection which uses the Gaussian mixture model ^[3]. Gaussian mixture models are a form of General Mixture Model. In statistics, a mixture model is a probabilistic model for representing the presence of subpopulations within an overall population, without requiring that an observed data set should identify the sub-population to which an individual observation belongs. Formally a mixture model corresponds to the mixture distribution that represents the probability distribution of observations in the overall population. However, while problems associated with "mixture distributions" relate to deriving the properties of the overall population from those of the sub-populations, "mixture models" are used to make statistical inferences about the properties of the sub-populations given only observations on the pooled population, without sub-population identity information. Gaussian mixture model is formed by combining multivariate normal density components.

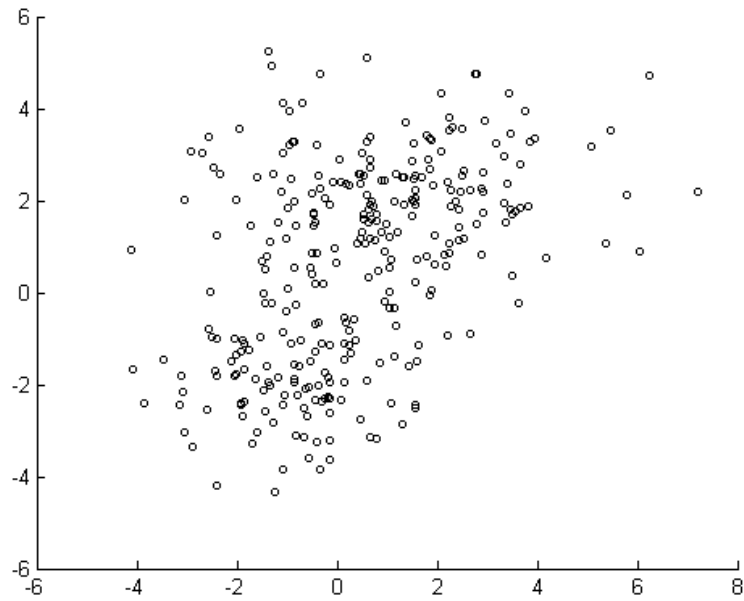


Fig 4: Gaussian mixture model

In Matlab's Statistics Toolbox ^[4] software, `gmdistribution` class is used to fit data using expectation maximization (EM) algorithm, which assigns posterior probabilities to each component density with respect to each observation. Gaussian mixture models are often used for data clustering. Clusters are assigned by selecting the component that maximizes the posterior probability. Like k-means clustering, Gaussian mixture modeling uses an iterative algorithm that converges to a local optimum. Gaussian mixture modeling may be more appropriate than k-means clustering when clusters have different sizes and correlation within them. Clustering using Gaussian mixture models is sometimes considered a soft clustering method. The posterior probabilities for each point indicate that each data point has some probability of belonging to each cluster.

2.3. Object detection and tracking

Object detecting and Tracking are the first steps of our work. For detecting any object one must know about foreground detection and blob analysis. Foreground detection is one of the major tasks in the field of Computer Vision whose aim is to detect changes in image sequences. Our applications do not need to know everything about the evolution of movement in a video sequence, but only require the information of changes in the scene. There are many techniques for foreground detection ^[5] and we have studied and implemented Dynamic Foreground detection with Gaussian Mixture Model (GMM) technique which is a built in Matlab functions.

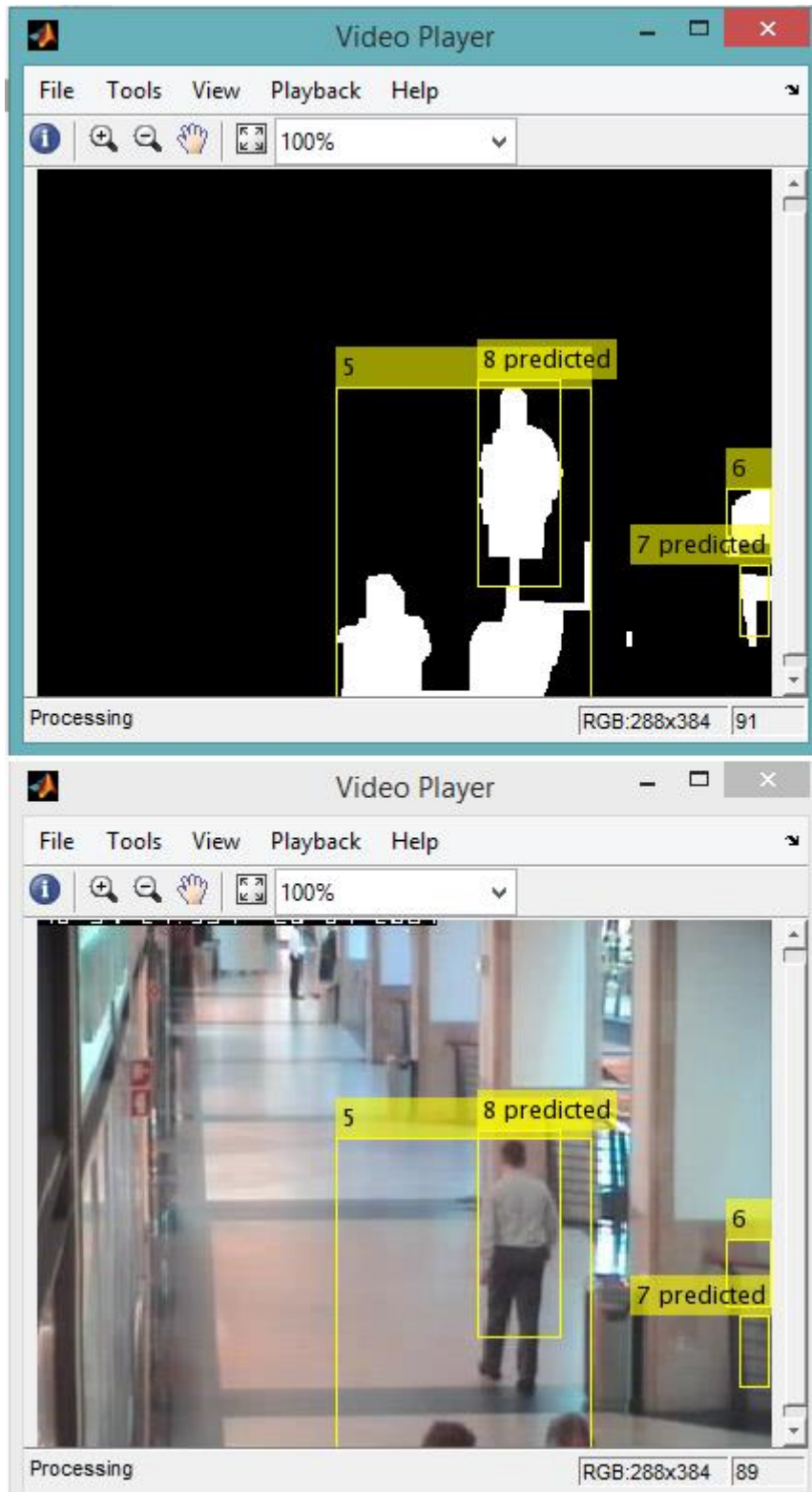


Fig 5: Foreground detection and blob analysis

The foreground detector is used to segment moving objects from the background. It outputs a binary mask. Connected groups of foreground pixels are likely to correspond to moving objects. The blob ^[6] analysis which is another built in matlab function is used to find such groups and compute their characteristics, such as area, centroid, and the bounding box.

3. Literature Review

Crowd dynamics and behavioral analysis has become an active research area and a substantial body of work has been devoted to the problem. In our work we have to implement certain ideas about object detection and object tracking and relate them with Crowd Dynamics. There are many different approaches to crowd Dynamics and a lot of systems use a combination of ideas from different methods. In this section, we briefly discuss the current state of the art in the field of crowd simulation. While there is an extensive amount of research done and many papers have been written about the subject, in this section we discuss some of the more influential work.

3.1. Data-driven Crowd Analysis in Videos

Mikel Rodriguez, Josef Sivic, Ivan Laptev, Jean-Yves Audibert and Ecole Normale Supérieure^[8] came up with this new approach in crowd dynamics. In this work they present a new crowd analysis algorithm powered by behavior priors that are learned on a large database of crowd videos gathered from the Internet. This algorithm works by first learning a set of crowd behavior priors off-line. During testing, crowd patches are matched to the database and behavior priors are transferred. Despite the fact that the entire space of possible crowd behaviors is infinite, the space of distinguishable crowd motion patterns may not be all that large. For many individuals in a crowd, it is possible to find analogous crowd patches in their database which contain similar patterns of behavior that can effectively act as priors to constrain the difficult task of tracking an individual in a crowd. Their algorithm is data-driven and, unlike some crowd characterization methods, does not require us to have seen the test video beforehand. In this crowd analysis was approached from a new direction. Instead of learning a set of collective motion patterns which are geared towards constraining the likely motions of individuals from a specific testing scene, they have demonstrated that there are several advantages to searching for similar behaviors amongst crowd motion patterns in other videos.

3.2. Flow Based Methods

Hughes^[7] implemented crowds as a flowing continuum. Using fluid dynamics, he compared crowds with fluids of "thinking fluids". While Hughes' method was not well suited for low density crowds and contained some oscillations, the idea to use fluid dynamics in crowd simulation inspired many other methods.

He used his model to simulate the Battle of Agincourt. While results from the simulated battle differed from the descriptions in literature, they did find an error in more recent writings about the battle using the simulation. Treuille et al. improved on flow based crowd simulation with the Continuum Crowds method. The method does result real crowd phenomena like lane formation and vortices and has less oscillations than the model Hughes presented. The model is well suited for large homogeneous groups of people viewed from some distance, but it does not support individual variability making it less useful for situations where crowds are viewed at small distances. Related to crowd flows is the work of Loscos et al. . They are using a 2d grid to generate a single collision map including all pedestrians in a way that is comparable to the flow calculations. The collision avoidance however, is agent based. The method includes path planning and grouping behavior as well. Like many of the other methods, other pedestrians are taken into account too late. The method does introduce some basic grouping behavior but the following and avoiding actions are rather limited.

3.3. Overlapping motion patterns

Another similar algorithm is overlapping motion patterns^[17]. Overlapping motion patterns have been studied as a means of coping with multi-modal crowd behaviors. These types of approaches operate in the off-line (or batch) mode (i.e. when the entire test sequence is available during training and testing) and are usually tied to a specific scene. Furthermore, they are not well suited for tracking rare events that do not conform to the global behavior patterns of the same video. It is both practical and effective to learn motion priors for a given scene. However, the off-line assumption of previous methods is expected to limit their application in practice.

3.4. Counting Crowded Moving Objects

Most of the algorithms have focused on the general problem of tracking, without specifically addressing the challenges of a crowded scene. Crowd detecting has been addressed in a variety of contexts including the study the movement of each objects in every frame of the streaming. Many approaches have focused on segmenting the motion patterns and learning the pedestrian behaviors. For example, Rabaud et al. and Zhou et al. ^[9] detected independent or collective motions for object counting and clustering. Lin et al. and Ali et al. segmented motion fields generated by crowds ^[10]. Both of them used a mixture of dynamic systems to learn pedestrian dynamics and applied it to crowd simulation. There are other works of crowd behavior analysis and detection on surveillance applications, such as abnormal activity detection and semantic region analysis. Kratz et al. proposed efficiency to measure the difference between the actual motion and intended motion of pedestrians for tracking and abnormality detection.

In crowded scenes, research has been done on tracking by detection methods in multiobject tracking. Such approaches involve the continuous application of a detection algorithm in individual frames and the association of detections across frames. Several tracking algorithms have centered on learning scene specific motion patterns, which are then used to constrain the tracking problem.

In this work we seek to draw upon the possibility of crowd formation by the analysis of videos and acquiring behavior of the movement of the objects priors after extracting large database which is stored in a special type of data structure. In particular we build on the recent progress in large database driven methods, which have demonstrated great promise in providing priors on object locations in complex scenes and also the possible location for crowd formation.

4. Technical Overview

In this section, we have discussed the technical details of our project and the resources we have used to complete the project and make it working.

4.1. Proposed System

The main techniques used in our work are detection of moving objects using the background subtraction algorithm and information extraction. The detection of moving objects uses a background subtraction algorithm based on Gaussian mixture models. Morphological operations are applied to the resulting foreground mask to eliminate noise. Finally, blob analysis detects groups of connected pixels, which are likely to correspond to moving objects.

After each frame is extracted it is assigned with a frame number. Then using the subtraction algorithm each object is detected along with the position of in co-ordinate form. All the information is stored in a data structure in such a way that we can access the information of any object at any particular frame. The nature of the data structure is very important for our algorithm as in this case as the streaming goes on frames are being created all the time and the size of the data structure is also increasing constantly. Our algorithm always reacquires information from the data of the last few seconds for prediction and for that after a particular time we don't need the old data and so we start replacing them with new ones. We can find the track of each object observing their previous positions and movement.

The association of detections to the same object is based solely on motion. The motion of each track is estimated by calculating the previous positions and movement of the object. Thus we predict the track's location in each frame, and determine the likelihood of each detection being assigned to each track.

Track maintenance becomes an important aspect of this algorithm. In any given frame, some detection may be assigned to tracks, while other detections and tracks may remain unassigned. The assigned tracks are updated using the corresponding detections. The unassigned tracks are marked invisible. An unassigned detection begins a new track. Each track keeps count of the number of consecutive frames, where it remained unassigned. If the count exceeds a specified threshold, the example assumes that the object left the field of view and it deletes the track.

After we have found the track of each object we then give our attention to crowd formation. If two or more objects are going for a collusion then we can see this from their track and that can be one of the possible position for crowd formation . For that we need to predict the next possible position of each object. If the predicted position of more than one object is very close

then there is a possibility of collusion which may result in crowd formation. Again as we are analyzing the motion of each object we can detect if an object is suddenly slowing down or behaving abnormally. If one or more object stops at a particular frame then we can assume that a crowd has already formed in that location. Thus we raise threat signals accordingly to identify different risk level.

Usually algorithms for object detection, tracking and activity analysis which consider an object in isolation (i.e., individual object segmentation and tracking) often face difficult situations such as the overlapping of two different objects, complex events due to interactions among objects. So sometimes in a crowd pedestrians overlap with one another and it seems like a single object. For this reason, many papers consider the crowd as a single entity and analysis its dynamics

4.2. Previous Works

In recent studies, the detection and recognition of crowd in public place have been developed in many research centers. Object detection was the first step for all these research. Detection and localize abnormal behaviors in crowd videos using Social Force model was developed as part of the research project of University of Minnesota which is Published in: Computer Vision and Pattern Recognition, 2009. CVPR 2009.IEEE Conference on. Moreover, many techniques have been developed for similar type of work such as The Kalman filter, object detection by substitution, overlapping motion patterns etc. All these algorithms and researches are very impressive and useful but our algorithm is more basic and more direct. At present crowd Dynamics and behavioral analysis is a huge field and so many people are doing research in this topic. Our work is about crowd formation and it doesn't falls directly under crowd dynamics but it have a very close relationship. Therefore the main objective of our paper is to detect crowd with minimum error and indicate regions having increased possibility of crowd formation. Apart from crowd detection and prediction, our research also extracts features from the movement of each objects (in this case pedestrian) and also predicts what will be the position of the object in future, If there is going to be any collusion or not, if an object is suddenly slowing down or stoping etc and there is very few similar work done in the past.

4.3. Flow Chart

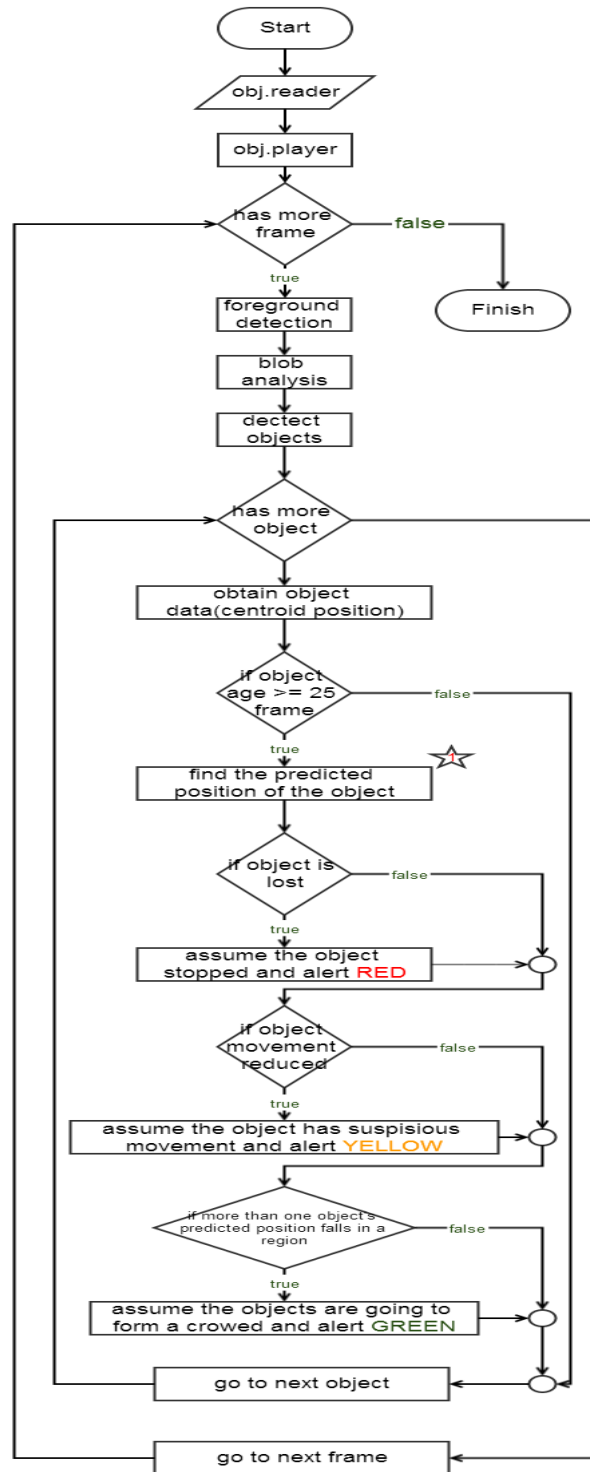


Fig 6: flow chart

4.4. Procedure

For convenience of understanding we divided the technical into 3 sections. In the first section the video file is read and the information of each object is extracted. In the second section we apply our motion analysis theories and predict the next position of each object and in the 3rd and final section we raise threat signals to indicate different risk level.

4.5. Detection and Tracking

As mentioned before the main techniques used in our work are detection of moving objects using the background subtraction algorithm and information extraction. The detection of moving objects uses a background subtraction algorithm. At first each frame is extracted and assigned with a frame number. Then using the subtraction algorithm each object is detected along with the position of in co-ordinate form. Each of the object is given an identification number called object ID. This Object ID is very important. It helps us to identify the object. All the information is stored in a modified String-based data structure. From this data structure we can access the information regarding any object at any particular frame. The nature of the data structure is very important for our algorithm as in this case as the streaming goes on frames are being created all the time and the size of the data structure is also increasing constantly. Our algorithm always reacquires information from the data of the last few seconds for prediction and for that after a particular time we don't need the old data and so we start replacing them with new ones. We can find the track of each object observing their previous positions and movement. We needed a video file reader for reading live video streaming and playing the file with our outputs. We used a few of MATLAB's built-in functions for reading video files, detecting objects and tracking the detected objects. `Vision.VideoFileReader()`^[11] was used for reading the video file which reads the video frame by frame while `vision.VideoPlayer()`^[12] plays the video in another window with our given outputs. Dynamic Foreground detection with Gaussian adaptation (GA) and blob analysis are used to detect and track each moving object. It also provides us with some information like area, centroid, age etc of each object that we use later for our calculation. One of the main problems of using dynamic foreground detection is that if any object suddenly stops moving then there is a possibility that it might not be detected. So we had to do some additional work and calculations to solve this problem.

4.6. Motion Analysis

After detecting and tracking each moving objects in the previous section here, we analyse the motion of each of the object so that we can predict the possible position of the object which is very crucial for our work. In the previous section we stored the co-ordinated of each objects in terms of pixels in each frame in our modified String based data structure. Now we extract this information to analyze the motion of each object. The process is simple and can be explained with the help of an example . Suppose a streaming is going on and we are at any particular frame say the frame number is x and there are currently 5 objects . To analyze the motion of these 5 objects we need to know where these 5 objects were in the previous frames and this information is stored in the data structure . Then we start with one we take all the objects from this frame and comparing with the object ID we can find the previous positions of this object and if the object id doesn't match then this object was not there in the previous frames e.i this object just entered the frame and its a new object and it's age is only 1 . For analysing the motion we must have the position (co-ordinates) of the object in previous frames which is not possible for a new object . So for analysing the motion the object must be a few frames old . The older the object the more information we have about it and the motion analysis is more accurate . We have selected a lower bound in the age of the object for the convenience of analysis. So if the age of the object is less than the bound we don't analyse its motion .Here the lower bound is 25 frames or 1 second.For analysing the motion and predicting the next position of the objects we used different algorithms and each of them can predict the position of the objects in the future . Our algorithms are Super naive approach , naive approach , Extrapolation and linear regression.These algorithms are suitable for different cases . In busy streets usually pedestrian usually move almost along a straight line and our Super naive approach is perfect in this case but if the movement is random then we can't rely on other approaches.

4.7. Linear Motion Prediction Algorithm

Prediction or estimation refers to finding some information based on the previous record. In this paper, we have to find the next possible position of an object depending on the record of previous positions of the objects in previous frames. The frame rate of the input video or streaming is set 25fps. Which means in 1 second, there moves 25 frames. So if we can get the history of the positions of the objects of last 25 frames, we can find possible positions of the objects in next few frames. In this particular project, we are working on human movements and predicting their future positions by analyzing their previous movements, we have to consider

some facts about human movements so that the whole project becomes simple and easy to implement. Initially, we assumed the human objects in the frame moves linearly. Depending on that theory, we came to such a decision that, if we can get the center positions of the particular object in current frame and 25th last frame we can find the distance between the two object positions and the direction of change. From those data we can find another point with same distance and on same direction. That particular position can be defined as the predicted position of the object, 25 frames later. In our case, we got to predict the future positions before 2 seconds or 50 frames; we had to multiply the distance with 2.

The equations of the Linear Motion Prediction Algorithm are described below.

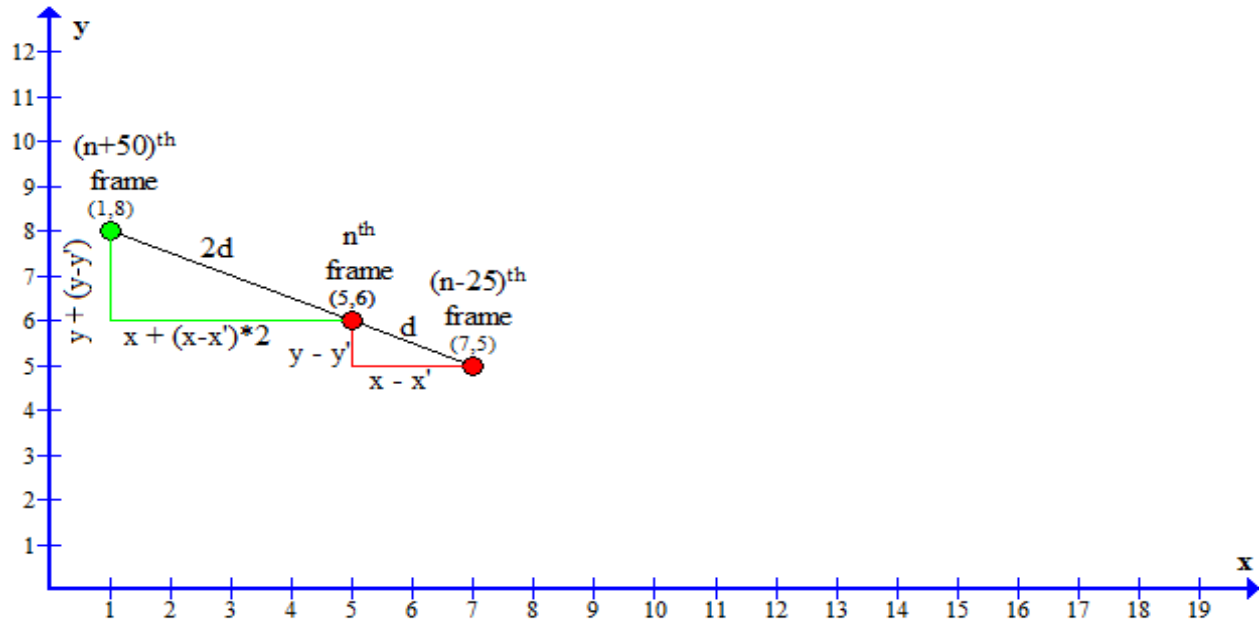


Fig 7: Linear motion prediction algorithm

If, x_{50} = predicted value of x-axis of the object at 50 frames of 2 second later and y_{50} = predicted value of y-axis of the object at 50 frames of 2 second later.

$$x_{+50} = x_0 + x_{-25}$$

$$y_{+50} = y_0 + y_{-25}$$

Where,

x_{+50} = predicted value of x-axis of the object at 50 frames of 2 second later.

y_{+50} = predicted value of y-axis of the object at 50 frames of 2 second later.

x_0 = value of x-axis of the object at current frame.

y_0 = value of y-axis of the object at current frame.

x_{-25} = value of x-axis of the object at 25 frames of 1 second earlier.

y_{-25} = value of y-axis of the object at 25 frames of 1 second earlier.

4.8. Moving Average Prediction Algorithm

While working with and implementing the previous algorithm (Linear Motion Prediction Algorithm) we found that, although people walk linearly, they are not straight lines at all. There are many changes in their moving path. In normal eyes, it might be look like straight, it is not. So there had to make changes in the theory of the Linear Motion Prediction Algorithm. In this case, we took the position of the center of the object of 4 more intermediate points between the current center of the object and 25 frames and calculated the movement at every point. And then sum the changes and added them with the current position. The result was identical to the previous method. So we have not used this method in our work.

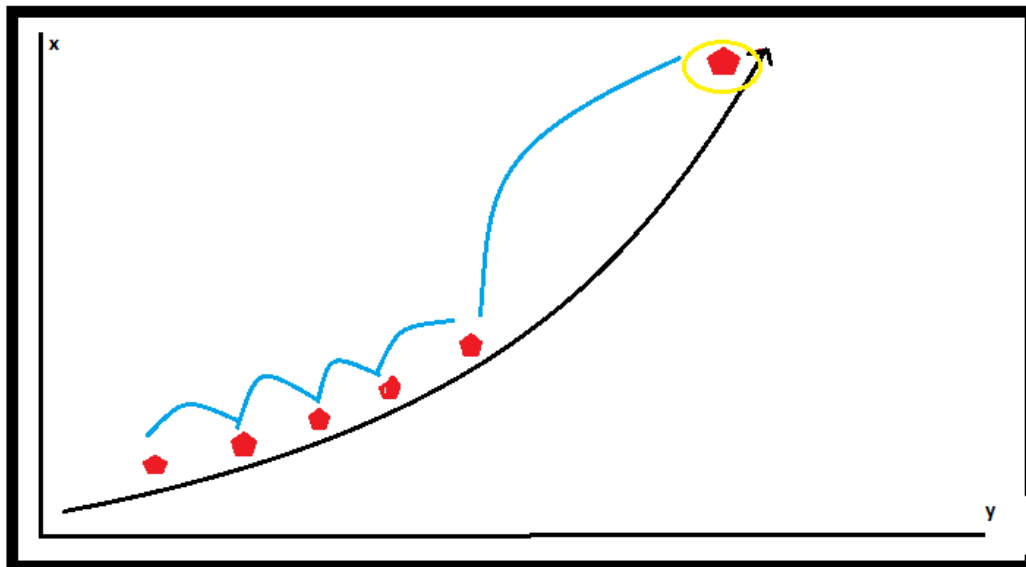


Fig 8: Moving Average Prediction Algorithm

4.9. Extrapolation

In mathematics, extrapolation ^[13] is the process of estimating, beyond the original observation range, the value of a variable on the basis of its relationship with other variables. Technically, Extrapolation means creating a tangent line at the end of the known data and extending it beyond that limit. Linear extrapolation will only provide good results when used to extend the graph of an approximately linear function or not too far beyond the known data.

If the two data points nearest the point x_* to be extrapolated are (x_{k-1}, y_{k-1}) and (x_k, y_k) , linear extrapolation gives the function:

$$y(x_*) = y_{k-1} + \frac{x_* - x_{k-1}}{x_k - x_{k-1}}(y_k - y_{k-1}).$$

We have three variables in our case. They are x-axis value, y-axis value, and frame number where we need to estimate x-axis value, y-axis value at a particular frame number which is 50 more than our current frame number which is after 2 second, as we have input streaming of 25 frame per second.

As we have two unknown values to figure out, we have to go through process twice; once for finding the value of x-axis and again for the value of y-axis. After we get both values of x-axis and y-axis, we can point that as the predicted position of the particular object after the desired frame number.

For example, in our case, the equation for predicting x-axis value will be

$$x_{+50} = x_{-25} + \frac{\text{frame number}_{+50} - \text{frame number}_{-25}}{\text{frame number}_0 - \text{frame number}_{-25}}(x_0 - x_{-25})$$

$$y_{+50} = y_{-25} + \frac{\text{frame number}_{+50} - \text{frame number}_{-25}}{\text{frame number}_0 - \text{frame number}_{-25}}(y_0 - y_{-25})$$

Where x_{+50} = predicted value of x-axis of the object at 50 frames of 2 second later.

y_{+50} = predicted value of y-axis of the object at 50 frames of 2 second later.

x_0 = value of x-axis of the object at current frame.

y_0 = value of y-axis of the object at current frame.

x_{-25} = value of x-axis of the object at 25 frames of 1 second earlier.

y_{-25} = value of y-axis of the object at 25 frames of 1 second earlier.

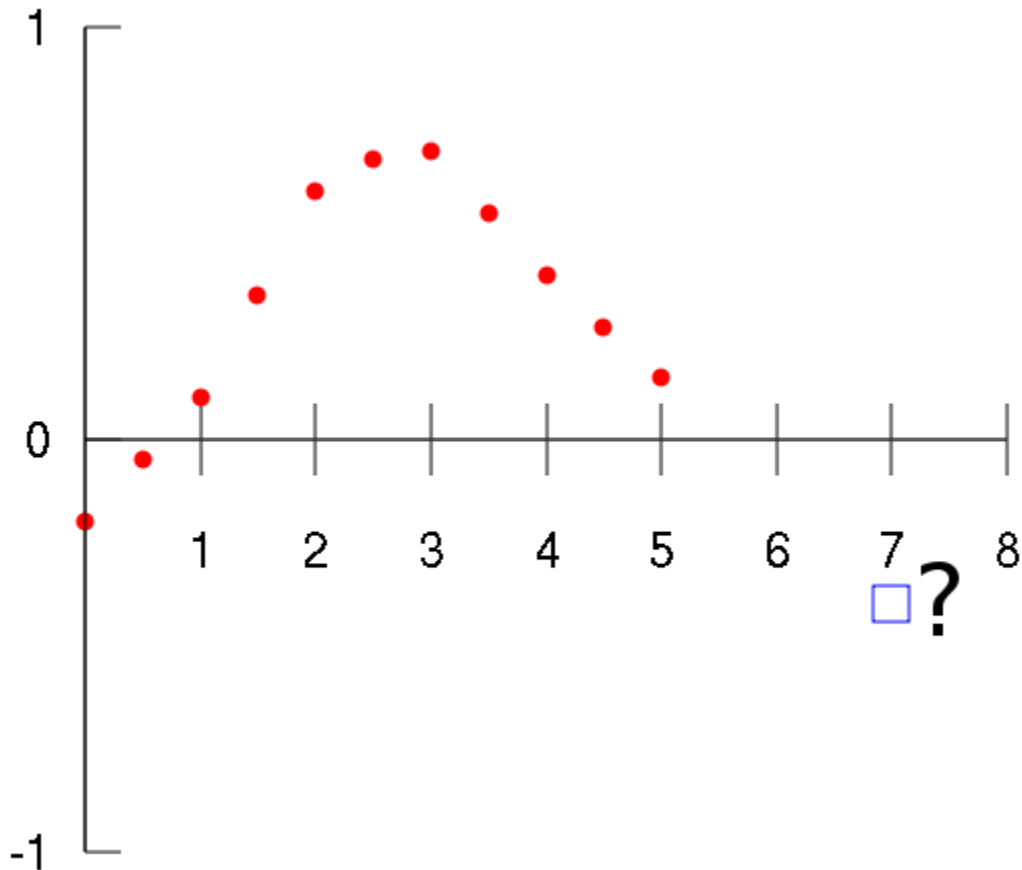


Fig 9: Extrapolation example.

4.10. Extrapolation with Linear Regression

Linear regression ^[14] is a simple statistics model describes the relationship between a scalar dependent variable and other explanatory variables. If there is only one explanatory variable, it is called simple linear regression, the formula of a simple regression is $y = ax + b$, also called the line of best fit of dataset x and dataset y.

The basic of this algorithm is to find a specific point, where a set of data is available and a value for which the point will be predicted by pointing the known data, drawing a best-fit-line and putting the value on the line. In the linear regression model, the dependent variable is assumed to

be a linear function of one or more independent variables plus an error introduced to account for all other factors.

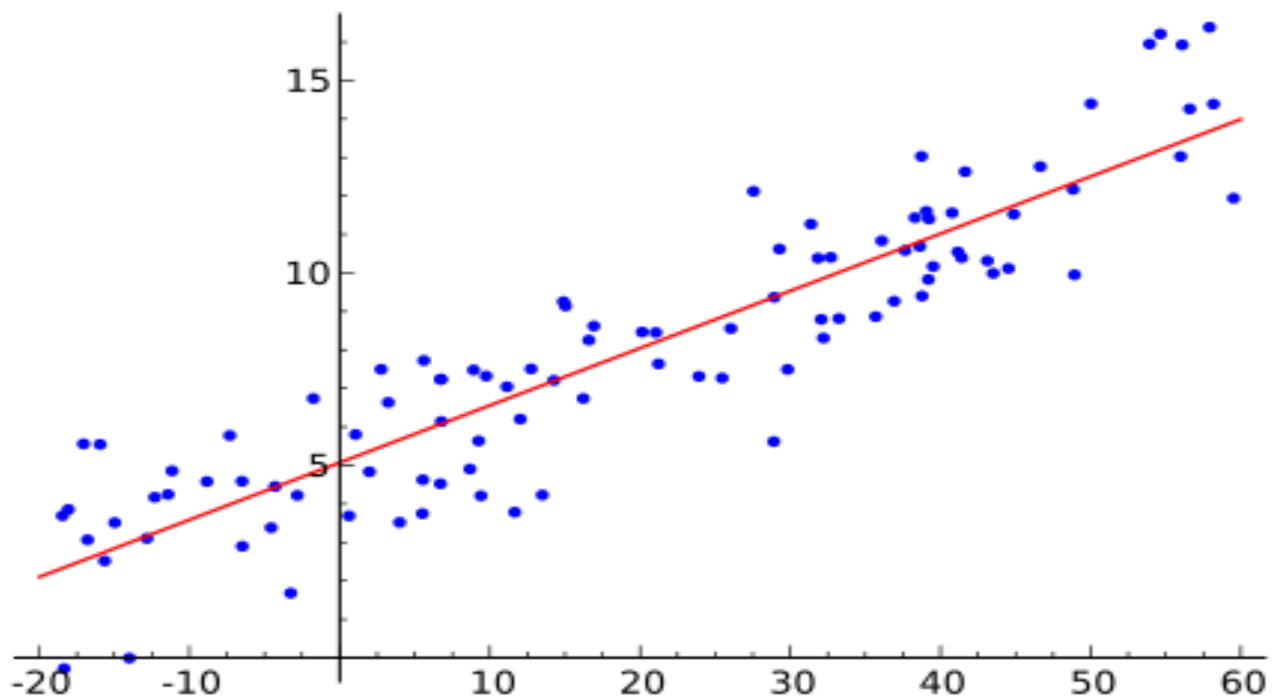


Fig 10: Extrapolation with Linear Regression

As we have two dependent variables with the known variable, frame number, we have to do this thing twice; once for the value of x-axis and then for y-axis. From this x-axis and y-axis value, we can find the center point of the predicted position of an object.

4.11. Raising Threat Signals

Our prediction system starts when the frame number is greater than 25. In other words, we can only predict when the age of each object is more than 1 second. Depending on the risk level, we have defined three category of threat. They are green level, yellow level and red level. They are described below.

Green level: Depending on the prediction algorithms, we can find the future possible positions of objects. Assume we are in now frame number n . we have three objects in the frame and we have the predicted data of the frame number $(n+25)$. Now analyzing the predicted data, if we find the predicted positions of the objects with a less distance or overlapped among those points, we can say they are going to meet or there is a possibility of gathering. Depending on this idea, we can alert ‘Green’ level of risk. It means there is a mild chance of forming crowded.

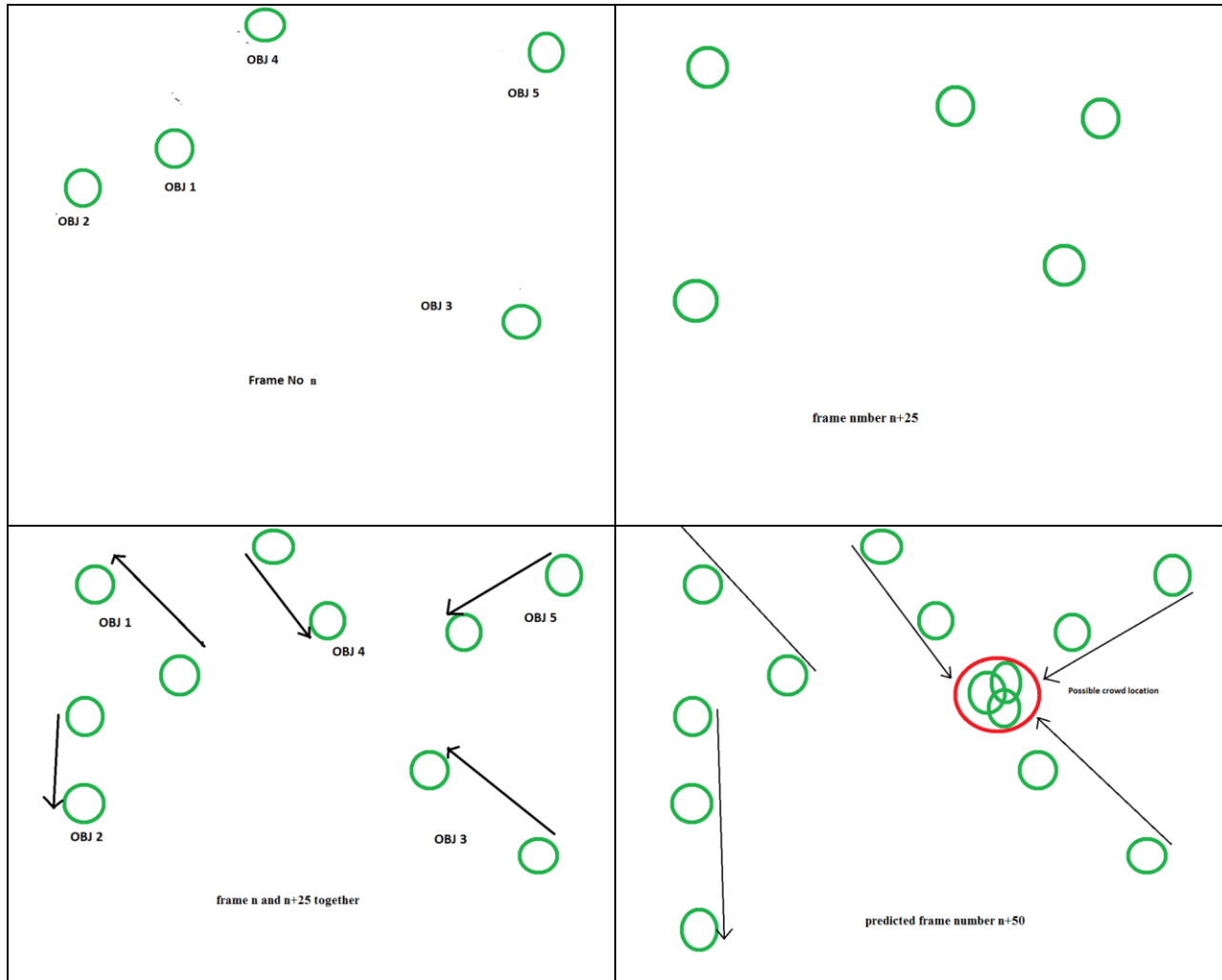


Fig 11: detecting the green level risk using prediction

Yellow Level: We have the positions of each object in each frame. Now if we find any of the objects reduced its movement speed suddenly, we identify that as a suspicious movement and we mark that place as a ‘Yellow’ level of risk.

Red Level: In a particular frame, if an object is lost, that may indicate two possibilities. Either the object is stopped its movement as we detect objects by GA model or ‘Adaptive Gaussian

mixture model' model where the detection of the object is done by dynamically. If an object does not change its position, a certain time later, it will be considered as background. So in this case, we can say the object has stopped and raise an RED signal. Or if two object marge together, the two objects will be one object and an object will be lost. In this case, there are more than two objects together. So we consider that a formation of crowd as well and mark them 'Red' as well.

5. Result Analysis

In this section, we are going to discuss about the outputs of our system and in the following tables and diagrams we have tried to explain the result graphically and numerically.

5.1. How our System is Unique

In the normal surveillance system, it is mandatory to keep a man who always keeps an eye on the monitors. It is really tough for a person, to monitor all the screens all the time. In this case, our implemented system can help. In our system, if there is any kind of risk arising issue occurs, the system itself can alarm the operator so that he or she can look at that particular monitor and take required decisions. If there is any kind crowd already formed, our system will mark that red, if any object is rapidly slows down, we will consider that as a suspicious movement and mark them as yellow and after predicting their future position and analyze them, if it is found that there is a possibility to form crowd, we mark them as green which is not present on the tradition surveillance security system.

5.2. Data Set Analysis and Report

In order to test our project, we had to take some videos. Mainly those videos are of some people walking, talking, meeting and fighting. Most of the videos are taken from internet and some of them are made by us from BRAC University. These videos are of different frame size. We had to do that so that we can check whether our system works on different frame size or not. But all the videos are of 25fps which means, in the videos, in one second it passes through 25 frames.

5.2.1. Prediction

We have done many experiments in our work to find the accuracy of program. In our work there is a lot of work regarding prediction and analysis them. We have predicted the next possible position of all the objects and on the basis of this information later we have taken many important decisions. We have used different approaches and algorithms for this and it was necessary to find their accuracy. We compared the predicted positions with the actual positions and found best case and worst case for each of the motion analysis algorithms.

These are the following table that we get from our experiments.

Frames	Object No.	Actual		Predicted location		Distance from actual
308	3	184.7626	75.7998	200.9878	104.7437	33.18141742
309	3	183.8804	75.8094	201.1194	106.2955	35.02264145
310	3	181.9523	76.5527	201.7736	108.2929	37.42090632
311	3	179.898	76.9069	202.5673	111.2527	41.15253504
312	3	177.8105	76.8581	201.5836	108.4135	39.50827195
313	3	177.1792	77.9259	203.5733	109.959	41.50624062
314	3	176.6342	78.307	202.7759	107.1982	38.96267337
315	3	175.7173	77.4482	200.7725	105.1414	37.34536616
316	3	175.0495	77.6082	198.4711	100.2814	32.59824144
317	3	168.5708	74.7673	200.3814	98.5582	39.72305622
318	3	169.0991	74.635	199.9647	98.6634	39.11584423
319	3	167.9784	74.0283	195.0058	102.7876	39.46615876
320	3	168.137	74.01	190.8477	104.2814	37.84354043
Average						37.91129949

Table 1: Accuracy of linear motion analysis

Frames	Object No.	Actual Location		Predicted Location		Distance from actual
308	3	184.7626	75.7998	200.9878	104.7437	33.18141742
309	3	183.8804	75.8094	201.1194	106.2955	35.02264145
310	3	181.9523	76.5527	201.7736	108.2929	37.42090632
311	3	179.898	76.9069	202.5673	111.2527	41.15253504
312	3	177.8105	76.8581	201.5836	108.4135	39.50827195
313	3	177.1792	77.9259	203.5733	109.959	41.50624062
314	3	176.6342	78.307	202.7759	107.1982	38.96267337
315	3	175.7173	77.4482	200.7725	105.1414	37.34536616
316	3	175.0495	77.6082	198.4711	100.2814	32.59824144
317	3	168.5708	74.7673	200.3814	98.5582	39.72305622
318	3	169.0991	74.635	199.9647	98.6634	39.11584423
319	3	167.9784	74.0283	195.0058	102.7876	39.46615876
320	3	168.137	74.01	190.8477	104.2814	37.84354043
Average						37.91129949

Table 2: Accuracy of Linear Motion Prediction Algorithm.

Frames	Object No.	Actual Location		Predicted location		Distance from actual
308	3	184.7626	75.7998	200.9878	104.7437	33.18141742
309	3	183.8804	75.8094	201.1194	106.2955	35.02264145
310	3	181.9523	76.5527	201.7736	108.2929	37.42090632
311	3	179.898	76.9069	202.5673	111.2527	41.15253504
312	3	177.8105	76.8581	201.5836	108.4135	39.50827195
313	3	177.1792	77.9259	203.5733	109.959	41.50624062
314	3	176.6342	78.307	202.7759	107.1982	38.96267337
315	3	175.7173	77.4482	200.7725	105.1414	37.34536616
316	3	175.0495	77.6082	198.4711	100.2814	32.59824144
317	3	168.5708	74.7673	200.3814	98.5582	39.72305622
318	3	169.0991	74.635	199.9647	98.6634	39.11584423
319	3	167.9784	74.0283	195.0058	102.7876	39.46615876
320	3	168.137	74.01	190.8477	104.2814	37.84354043
Average						37.91129949

Table 3: Accuracy of Extrapolation.

Frames	Object No.	Actual Location		Predicted Location		Distance from Actual
308	3	184.7626	75.7998	205.1976	102.9151	33.95333
309	3	183.8804	75.8094	203.0228	105.6341	35.4393
310	3	181.9523	76.5527	204.787	105.9225	37.20227
311	3	179.898	76.9069	205.1743	108.4018	40.38341
312	3	177.8105	76.8581	200.5155	108.0292	38.56364
313	3	177.1792	77.9259	201.7795	109.6411	40.13762
314	3	176.6342	78.307	200.8362	108.6734	38.83111
315	3	175.7173	77.4482	199.802	105.7492	37.16207
316	3	175.0495	77.6082	198.5149	102.8782	34.48475
317	3	168.5708	74.7673	199.0749	102.6757	41.34464
318	3	169.0991	74.635	197.9643	103.5757	40.87498
319	3	167.9784	74.0283	195.1667	105.1864	41.35252
320	3	168.137	74.01	191.4771	106.3003	39.84249
Average						38.42863

Table 4: Accuracy of Linear Regression.

To describe the tables above, we have come to figure the following chart from where we can understand the result more clearly.

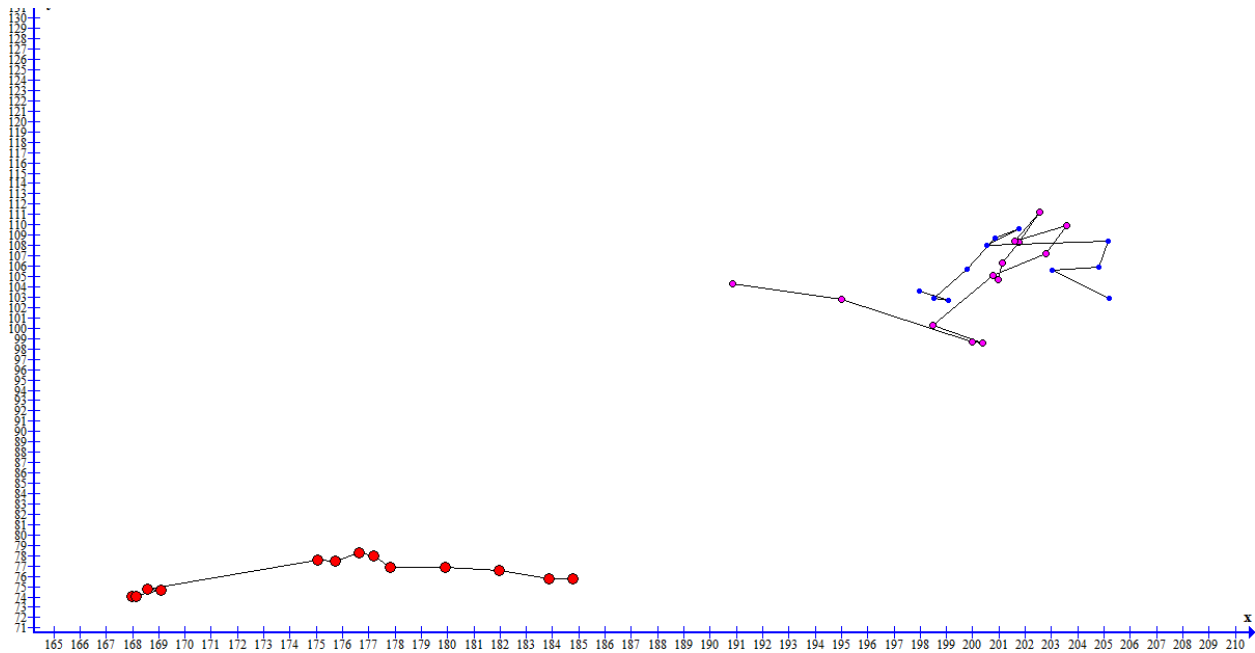


Fig 12: Prediction Graph

From the graph above where the red dots represent the current positions, purple dots represent the predicted position by using linear motion analysis and the blue dots represent the predicted positions by using regression analysis.

This table is taken from one of the videos of the dataset we have taken. This portion of this table indicates our worst result. Which means, the distance between the predicted centers of the objects is and the centers of the objects at the real time are 37 to 38 pixels depending on the prediction algorithms.

5.2.2. Raising Alarm

In the next step we raised threat signals indicating different risk level following different conditions. To analyse the accuracy of results we have done experiments with several videos and came up with the following data tables.

Raising red threat signal (crowd just forms)

Video	True positive (Correctly identified)	false negative (Incorrectly rejected)	False positive (Incorrectly identified)
1	1	0	0
2	3	0	1
3	3	0	9
4	0	0	0
5	3	1	2

Table 5: Accuracy in detecting crowd already formed

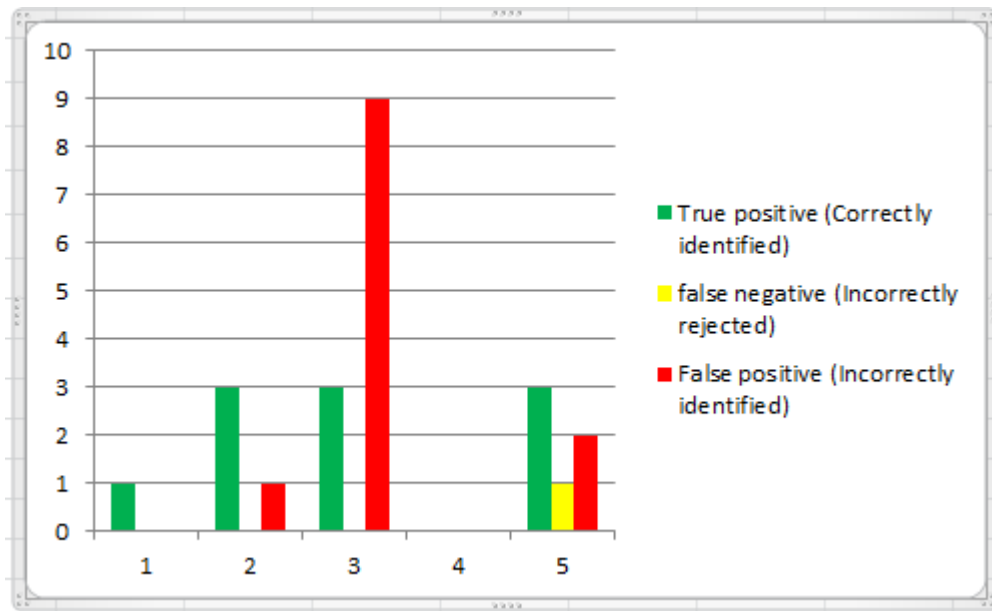


Fig 13: Accuracy Graph of crowd already formed

Raising green threat signal (crowd may form)

Video	True positive (Correctly identified)	false negative (Incorrectly rejected)	False positive (Incorrectly identified)
1	0	1	0
2	2	1	0
3	0	0	0
4	0	0	0
5	2	0	0

Table 6: Accuracy in predicting crowd formation

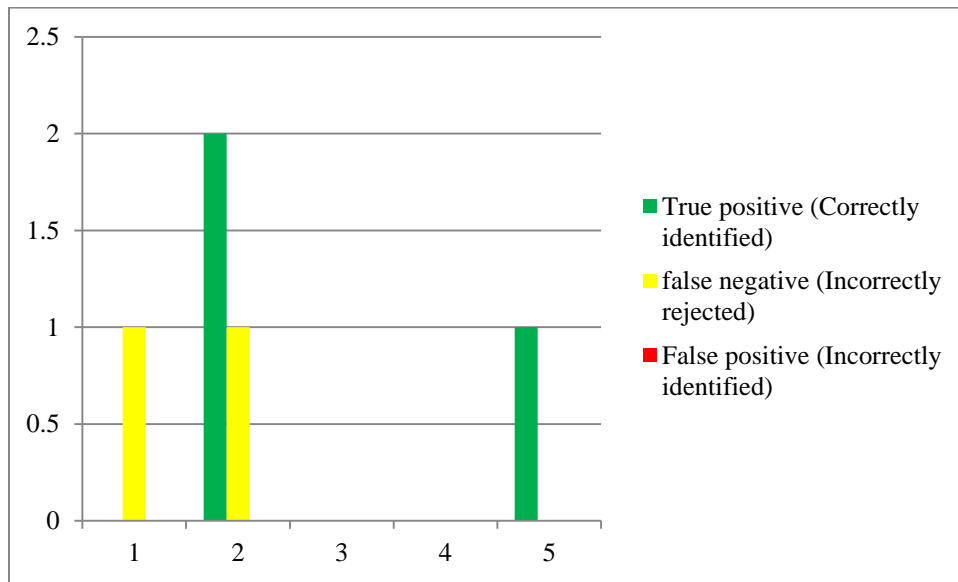


Fig 14: Accuracy Graph of predicting crowd formation

Raising Yellow Threat signal (Suspicious Behavior)

Video	True positive (Correctly identified)	false negative (Incorrectly rejected)	False positive (Incorrectly identified)
1	4	1	1
2	5	0	1
3	3	0	0
4	4	0	0
5	3	0	2

Table 7: Accuracy in predicting suspicious behavior



Fig 15: Accuracy Graph of suspicious behavior

5.3. Limitations and Challenges

While working on this project, we have faced some problems and had to go through some limitations, which are described below.

5.3.1. Moving Camera

The most significant limitation of our project is that, it does not work with moving security cameras. It is because we are segmenting and detecting object using dynamic foreground detection process and it is not possible to detect the foreground for moving cameras.

5.3.2. Overlapping objects

Sometimes when two or more objects are very close to one another then our system fails to identify all the objects separately. In that case it thinks that all the objects (very close to each other) as a single object. We can solve this problem if we decrease the threshold value then the rate of error will increase as the system will detect unwanted segments and identify them as moving objects.

5.3.3. Camera position

It has been found that, the camera position has a dramatic effect on the accuracy of our system. Vertical camera position results in more accurate results where horizontal camera position results in less accurate. Because if the video is taken from the top, the object will be on a 2d graph which is more accurate while predicting and tracking.

6. Conclusion and future work

In this work we have approached crowd analysis from a new direction. Instead of learning a set of collective motion patterns which are geared towards constraining the likely motions of individuals from a specific testing scene, we have demonstrated that there are several advantages if we analyse the optical flow from which we can later predict the motion in future in videos. In our work we have shown that by leveraging a large database of previously observed object positions we are able to track individual motion behavior patterns.

Though our work falls under crowd dynamics, but since our work is closely related to behavioral analysis. So in the future we are planning to work more on behavioral analysis. Abandoned object detection is a very popular at present. As we are using dynamic foreground detection techniques it is a bit difficult to detect abandoned objects. We are planning to implement an algorithm that can detect abandoned objects in Dynamic foreground detection. Again so far all our work is based on videos with a fixed frame i.e. the camera was fixed. In the future we are planning to implement this work on moving cameras where the frame is not fixed.

7. References

1. Kalman filter (Wikipedia)

http://en.wikipedia.org/wiki/Kalman_filter

2. Greg Welch and Gary Bishop “An Introduction to the Kalman Filter”

University of North Carolina at Chapel Hill , 2006

3. Mixture model (Wikipedia)

http://en.wikipedia.org/wiki/Gaussian_mixture_model

4 Gaussian mixture model Matlab (MathWorks)

<http://www.mathworks.com/help/stats/gaussian-mixture-models.html>

5. Foreground detection (MathWorks)

<http://www.mathworks.com/help/vision/ref/vision.foregrounddetector.step.html>

6. Blob analysis (MathWorks)

<http://www.mathworks.com/help/vision/ref/blobanalysis.html>

7. R. L. Hughes, “The flow of human crowds”

Annual Review of Fluid Mechanics, vol. 35, pp. 169{182, 2003.

8. Mikel Rodriguez, Josef Sivic, Ivan Laptev, Jean-Yves Audibert and Ecole, Normale Supérieure

“Data-driven Crowd Analysis in Videos”

Quaero, OSEO, MSR-INRIA, ANR DETECT , CROWDCHECKER , 2010.

9. V. Rabaud and S. Belongie. “Counting crowded moving objects”

Proc. CVPR, 2006.

10. B. Zhou, X. Tang, and X.Wang. “Detecting coherent motions from crowd clutters”

In *Proc. ECCV*, 2012.

11. Video Reader(MathWorks)

<http://www.mathworks.com/searchresults/?c%5B%5D=entiresite&q=Vision.VideoFileReader%28%29>

12. Video Player (MathWorks)

<http://www.mathworks.com/searchresults/?c%5B%5D=entiresite&q=vision.VideoPlayer%28%29%5B>

13. Extrapolation (Wikipedia)

<http://en.wikipedia.org/wiki/Extrapolation>

14. Regression (Wikipedia)

<http://en.wikipedia.org/wiki/Regression>

15. D. Lin, E. Grimson, and J. Fisher. “Learning visual flows: A Lie Algebraic Approach”
Proc. CVPR, 2009.

16. C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. “Sift flow”
Proc. ECCV, 2008.

17. Berkan Solmaz, Brian E. Moore, and Mubarak Shah. “Identifying Behaviors in Crowd Scenes Using Stability Analysis for Dynamical Systems”

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE