# Video Analysis and Anomaly Detection Using Fuzzy Logic

*Dissertation submitted in*

*partial fulfillment of the requirement*

*for the award of the degree of*

**Master of Technology**

*in*

**Computer Science and Engineering**

*by*

**Sumit Kadyan**
**University Roll No. 2K12/CSE/21**

*Under the Esteemed Guidance of*

**Dr. Rajeev Kapoor**
**H.O.D, Computer Engineering Department, DTU**

**2012-2014**

**COMPUTER ENGINEERING DEPARTMENT**

**DELHI TECHNOLOGICAL UNIVERSITY DELHI – 110042, INDIA**

# ABSTRACT

Recognition and understand of human activity in video have gained considerable research attention due to the potential application in various domains [4], such as video surveillance and monitoring, human-computer interfaces, content-based video analysis and behavioural biometrics. In video surveillance, the main objective is to be able to detect events of interest to aid security personnel. Video surveillance is done to find anomalies i.e. unwanted behaviour, in the video data.

In this dissertation we propose an approach to find the anomalies in the video data using fuzzy logic. Our work is divided in to two parts, first part tracks multiple objects in the video using kalman filter. All the moving objects in the video are detected in this step. Second part finds the trajectory of each detected object and matches these trajectories with those in the training matrix using fuzzy logic. So we find if the trajectory of the object in normal or anomalous. If the trajectory is anomalous, then an alarm is set to inform about the occurrence of anomalous event.

There is no need for any manual control by human beings. All the work is done automatically. Human operator is informed about the anomaly by the alarm which is set when some anomalous event occurs.

*Keywords*: Video anomaly detection, Trajectory Matching, Fuzzy logic

# ACKNOWLEDGEMENT

First of all, I would like to express my deep sense of respect and gratitude to my project supervisor Dr. Rajeev Kapoor for providing the opportunity of carrying out this project and being the guiding force behind this work. I am deeply indebted to him for the support, advice and encouragement he provided without which the project could not have been a success.

Secondly, I would also like to acknowledge Delhi Technological University for providing the right academic resources and environment for this work to be carried out.

Last but not the least I would like to express sincere gratitude to my parents and friends for constantly encouraging me during the completion of work.



**Sumit Kadyan**

**University Roll no: 2K12/CSE/21**

**M.Tech (Computer Science & Engineering)**

**Department of Computer Engineering**

**Delhi Technological University**

**Delhi – 110042**

**Computer Engineering Department**
**Delhi Technological University**
**Delhi-110042**
**www.dce.edu**

# CERTIFICATE

This is to certify that the dissertation titled **"Video Analysis and Anomaly Detection using Fuzzy logic"** is a bonafide record of work done by **Sumit Kadyan, Roll No. 2K12/CSE/21** at **Delhi Technological University** for partial fulfilment of the requirements for the degree of Master of Technology in Computer Science & Engineering. This project was carried out under my supervision and has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma to the best of my knowledge and belief.

**(Dr. Rajeev Kapoor)**
**Project Guide**
**Head Of Department**
**Department of Computer Engineering**
**Delhi Technological University**

Date: ____________

# Table of Contents

# List of Figures

# CHAPTER 1

# INTRODUCTION

Recognition and understand of human activity in video have gained considerable research attention due to the potential application in various domains [4], such as video surveillance and monitoring, human-computer interfaces, content-based video analysis and behavioral biometrics. In video surveillance, the main objective is to be able to detect events of interest to aid security personnel. In the literature, we can find works that perform event detection by learning patterns for specific events depending on the domain. Many of the works in this area take a traditional pattern recognition approach by explicitly modeling the events of interest based on a priori knowledge. Examples include parking in restricted areas [5], detection of abandoned objects [6], and suspicious interactions with objects [7].

More recently, there has been a paradigm shift [8] towards detection of anomalous events. Intuitively, an anomaly is a pattern that does not follow expected normal behaviour in a given context [9]. The anomaly detection problem has been studied in the literature in very diverse application domains [9]. For example, anomalous traffic patterns in a computer network may indicate an intrusion attempt which could be thwarted if measures are taken in time; in medical imaging, detecting anomalies could aid in the diagnosis of certain conditions; anomalous credit card transactions can also indicate fraudulent activity and prompt banks to block them to prevent financial loss.

In video surveillance, pattern recognition techniques are often the desired approach to perform event detection and behaviour understanding, by modelling anomalous events from training data. Detection is performed by finding patterns in new observations that conform to the previously obtained model. In contrast, in anomalous event detection, a model of normal behaviour is developed statistically and anomalies are detected by finding patterns that deviate from the model.

By specifically targeting events of interest, the more traditional approaches are able to provide high level descriptors of events occurring in the scene. These techniques construct models from training data that contain instances of the targeted anomalies, and attempt to classify as anomalies new unseen instances. Their main limitation, however, is their inability to cope with unknown behaviour, and are generally only applicable in certain controlled scenarios. Anomaly detection techniques, in contrast, are able detect arbitrary anomalies that differ from a previously obtained model of normality. While this approach broadens the amount of events that can be detected, it still poses significant challenges depending on how normality is defined. Additionally, these techniques are unable describe "what" has occurred, and would require further stages of analysis to provide higher level descriptors.

The rest of this chapter is structured as follows. Sections 1.1 and 1.2 cover the definition of an anomaly and a discussion on the different ways in which anomalies that can be defined. In section 1.3, a discussion on current challenges in anomaly detection techniques. In section 1.4, a brief discussion on different types of features is introduced. In section 1.5 a review of approaches that rely on object trajectories is presented, while alternative approaches are reviewed in section 1.6.

## 1.1 Video Surveillance

In the last two decades, the number of surveillance cameras installed to monitor private and public spaces has increased dramatically. This is mainly due to the rising fear of people about crime. To this end, cameras are installed in many public places, such as airports, train stations, city centers, or shopping malls. Other interesting applications of visual surveillance systems include the examination of crowd motion, traffic flow monitoring, biometric identification, the assessment of industrial processes or human behavior interpretation in retail spaces. From an economic perspective, the visual surveillance market is huge, and a significant part is invested in adequate software solutions.

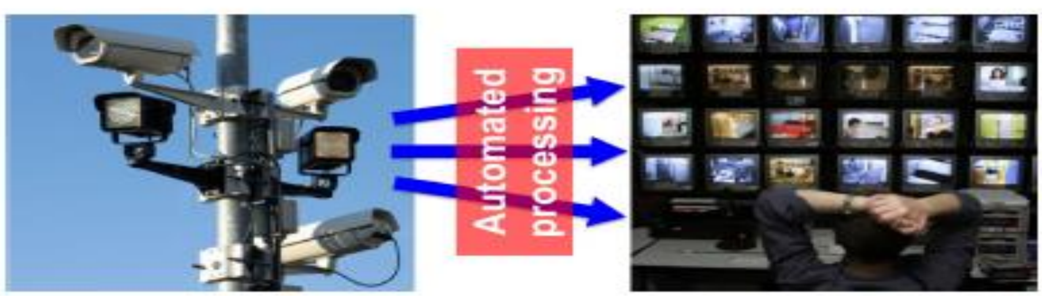


Fig. . Traditional video surveillance by human operator

Deployed by companies such as IBM, Bosch, GE, Honeywell, Siemens, ObjectVideo, or BRS labs, current surveillance systems to some extent include automatic video processing. For example, techniques for the detection of an intruder, a car driving against the traffic or an unattended piece of luggage exist currently. They apply rule-based detectors that are manually tuned to welldefined settings, in order to raise an alert in case a suspicious configuration is met. An operator then has to verify the video stream and initiate the according actions as indicated in Figure 1.1. Hence, in general, real-time monitoring installations still rely on constant verification by a knowledgeable human operator. In contrast, many closed circuit television (CCTV) systems record the video to storage devices and delete it after a certain period. This is useful to go back in time and identify the involved persons, for example if a theft or an aggression had happened. Of course, this retrospective analysis does neither prevent the crime nor detects it when it happens. Consequently, and due to the large amount of human monitoring effort involved, smart surveillance software solutions are highly desirable

## 1.2 What is an anomaly

In spite of the diversity of solutions and applications, there is a lack of agreement on how anomalies are defined. In the literature, anomalies have been referred to as “unusual events” [10], “anomalous events” [11], “abnormality” [12], “suspicious activities” or “irregularities” [13].

In broad terms, we can define an anomaly as an observation that does not follow expected normal behaviour [9]. For video sequences, anomalous events can be seen as motions or sequence of motions that stand out in their surrounding context in space and time [8]. This enables a statistical treatment of anomaly detection, by considering anomalies as events of low probability with respect to a probabilistic model of normal behaviour. [9][14].

This definition has certain implications that limit which anomalies can be detected. Firstly, it makes anomalies dependent on a given context. An event that is anomalous at a certain moment, may be perfectly normal at other times. Such is the case of traffic interactions, in which certain actions are only allowed given certain conditions like the state of traffic lights. Secondly, the anomalous events that can be detected are directly limited by the features and the scale at which normality is defined [14]. E.g., an event that is anomalous at a certain scale may be perfectly normal at a different scale.

As a result of the different ways in which anomalous events can be defined, very diverse approaches can be found in the literature.

## 1.3 Types of anomalies

Depending on the levels of spatio-temporal context and which anomalous events are modelled, we can broadly distinguish between the following different types of anomalies [9].

**Point anomalies** indicate that the values of extracted features at a specific location deviate significantly from what is considered normal. Therefore, these anomalies do not take into account past values or the information given by nearby objects or points. If one models the normal velocities of moving objects at all locations in the scene, any object that displays a velocity that does not fit the model can be considered an anomaly. This includes, for example, detecting motion of objects at unusual locations.

**Contextual anomalies** consider information from the temporal context (the sequence of events), or the spatial context (nearby objects). Anomalies that take into account the temporal context, also called sequential anomalies, analyse irregularities in the temporal sequence of a given extracted feature. For example, in traffic sequences, a car making an illegal turn at an intersection may display "normal" velocity as it passes through it, as it trajectory will partially overlap different normal traffic paths. However, the trajectory itself is anomalous as it does deviate from the predefined path from that direction. For anomalies in the spatial context, information from nearby objects is taken into account. For example, the authors in [15] analyse the avoidance strategies of nearby people to detect anomalies in walking paths. In [11], the authors consider co occurrence anomalies, by detecting pairs of events that do not usually occur simultaneously.

These distinctions highlight the fact that anomalies are heavily dependent on the given context, and can be arbitrarily complex depending on the features extracted. As a result, it becomes difficult to directly compare diverse solutions found in the literature. Depending on the capabilities of the algorithms, different sets of anomalies can be found on the same datasets.

## 1.4 Challenges

While anomalous events are easy to define intuitively, there are a number of factors that pose challenges to anomaly detection techniques:

- The definition of anomaly is heavily dependent on how normality is modelled and which features are extracted. In particular, context, features, and the scale at which features are extracted will ultimately determine which anomalies can be detected.

- A non stationary context may alter normality at different times in a given scenario. A robust detection system should be able to adapt to changing dynamics to account for these changes.

- Anomalous events are generally infrequent, sparse, and unpredictable [14]. This makes the examples found in training sequences limited in number. In particular, validation of techniques becomes a challenge if the number of anomalous events are insufficient.

## 1.5 Features

According to [8], we can distinguish between pixel based abstractions (features extracted at the pixel level) and object-based abstractions (features associated with an object or blob).

Among pixel-based abstractions, we find approaches that capture spatio-temporal features such as pixel change frequency and pixel change retainment [16], filling ratio of foreground pixels [17], histogram of pixel change frequency [18], gradient magnitude [13], accumulation of pixel differences [19]. Motion features are also common, extracted by optical flow techniques at the pixel level [20][21].

Among object-based abstractions, they can either be derived from appearance features or motion features. Among appearance features, we can include blob size[22][21] [23][17] and texture [21]. Motion features derived from object tracking are widely popular. Object tracking produces trajectories as a sequence of object location over time. From these, different features such as velocity/speed [24][11], and moving direction or orientation [22][25][20] can be extracted.

Among existing techniques, we can make a broad distinction between approaches that rely on extracted object trajectories, and those that do not [8][25]. The former imply preprocessing modules for object segmentation and tracking, whereas the later rely on other object features, or pixel-based abstractions.

# CHAPTER 2

# LITERATURE REVIEW

Detecting anomalous patterns from video sequence is useful for many applications such as surveillance, novelty extraction, automatic inspection and etc. The identification of anomalies can lead to the discovery of truly novel information from the video, e.g. suspicious object movements and/or rare trajectory patterns. With recent advances in visual tracking, it becomes feasible to obtain the trace of moving objects with reasonable accuracy from surveillance videos. This could be a very important source of features for event detection.

Previous research on abnormal activity detection can be roughly divided into two categories: parametric approaches and non-parametric approaches. For the first category [1], the explicit parametric model for normal and/or abnormal activities is constructed based on the features extracted from the observed data. Visual features such as position, speed, appearance and etc. can be extracted from object detection and tracking algorithms [2][3][4]. These models are either rule based or obtained from supervised machine learning techniques - most by probabilistic graphical models [5][6][7]. The second category, on the other hand, does not explicitly predefine the models for the normal events, instead learns the normal and abnormal patterns from the statistical properties of the observed data. In [8], joint co-occurrence statistics of object trajectories over a codebook are accumulated and hierarchical classification algorithm is applied to identify activities. [9] uses feature descriptor vectors to represent an image and detect unusual activities in a co-embedding space. In [10], a video is considered regular if it can be composed from large chunks of spatial-temporal patches, and the irregular videos are detected by probabilistic inference in graphical model.

## 2.1 Approaches based on object trajectories

Several approaches found in the literature for anomaly detection in video sequences employ information extracted from object trajectories, i.e., the temporal sequence of locations of a given object in the scene. Therefore, these techniques require a pre-processing stage in which moving object detection and object tracking are performed. Generally, background subtraction is employed for moving object detection and existing tracking techniques can be applied to extract object trajectories.

The main advantage of trajectory-based techniques is the possibility of constructing models in a fully unsupervised manner, i.e., labeling training data is not required. By considering anomalies as events with low frequency, clustering methods can be applied to discard outliers [11]. This is done by clustering trajectory paths to model "normal" trajectories. Anomalies are then detected by computing the distance of new unseen trajectories to existing "normal" cluster centroids. Those that are far enough from clusters are considered anomalous trajectories.

In [26] the authors propose a method in which trajectories are modelled as Hidden Markov Models (HMM) and grouped with hierarchical clustering. A similarity metric between HMMs is designed to determine the distance to clusters. A similar approach is taken in [27], which applies two-layers of hierarchical clustering to trajectories that are represented as a set of feature vectors that include location, velocity and size. The authors propose two similarity measures to detect point anomalies, by computing the probability of an anomaly when an object enters point k; and contextual anomalies, by computing the probability for an entire trajectory of being an anomaly.).

In [28], each trajectory is summarized as the parameters of a quadratic curve. At every spatial point, a Gaussian Mixture Model (GMM) is used to model the motion patterns of trajectories that pass through that point. For new observations, anomalies are detected as motions that display a low probability as predicted by the GMM model. The authors include appearance information by distinguishing between cars and pedestrians. However, the application of this technique is limited to constrained scenarios where the trajectories can be simplified as quadratic curves.

The authors in [23] generate a probability density function (PDF) from a Kernel Density Estimation (KDE) model for each pixel location in the image, taking object location as well as size features. New observations are detected as anomalies if they have a low probability as predicted by the pdf. A similar approach is described by [29], in which object locations and transition times are employed to estimate a probability density function using a GMM model.

In [11], the authors extract motion related features (location, moving direction and velocity) and devise different strategies for point-anomaly detection, sequential anomaly detection (temporal context) and co-occurrence anomaly detection (spatial context). The first are addressed by computing histograms of features, and consider observations with low probability as anomalous. For sequential anomalies, the authors apply the data-mining CloSpan algorithm [30] to obtain the frequency of different sequences of feature vectors. For co-occurrence anomalies, HMM models are used.

These techniques are affected by challenging scenarios in which background subtraction and object tracking do not perform well. In particular, background subtraction performs poorly in crowded scenes, as well as situations with non-stationary backgrounds and sudden illumination changes. Object techniques often find difficulties in crowded scenarios in which occlusions are frequent, resulting in inaccurate tracks that impact the subsequent anomaly detection analysis negatively. Tracking does not scale well with object clutter, as it increases computational complexity and thus it is unsuitable for real-time applications.

## 2.2 Approaches based on pixel level abstractions

To overcome the limitations of techniques that extract motion features from trajectories, a number of methods have been recently proposed that do not use tracking and work at either the pixel or the region level by dividing the image in blocks. level. Some techniques do, however, incorporate features from the objects passing through pixel locations in the image, and therefore object segmentation is also required to extract these features.

In [18], the authors first slice the video in short sequences that are assumed to contain one event. This limits the applications of this approach and also makes it unable to locate the anomaly in space. After performing background subtraction, they compute a spatial histogram by blocks depicting object motion. Applying techniques from document-keyword clustering, the authors compute the co-occurrence of extracted features in video segments. Video segments that are sufficiently dissimilar from others are classified as anomalies.

In [13], a spatiotemporal video patch descriptor is computed for patches in the image, which contains information from the spatial gradient at different spatial scales. A set of patches at different scales is extracted in the training phase to construct a database. For new observations, the authors propose to make use of a method to compose the patches of observed regions from patches in the database. If the new observations cannot be recomposed or if they can only be composed using the smallest patches, they are considered anomalous. Additionally, a strategy to progressively update the database of “normal” behavior is described.

The authors in [19] present a method to characterize the amount and location of motion inside a video segment (a collection of frames belonging to the same scene. These are described by proposed magnitudes “Total Motion” and “Average Motion”, computed from the data obtained by background subtraction to spatially locate motion in video frames. Hierarchical clustering is then used to obtain the cluster centroids of normal events. For new observations, they are detected as anomalies if the distance to the closest cluster is above a threshold.

In [16], the authors extract two different pixel-wise features: pixel change frequency (number of transitions between foreground and background in a given time) and pixel change retainment (amount of time a pixel is considered as moving foreground). For noise reduction, these features are down sampled into an 8x8 super-pixel containing the average values. Similar to other approaches, the authors attempt to compute the posterior probability of an observation given past events. This is done in a Sequential Monte Carlo framework by modeling events as HMMs. Aside from point anomalies, this technique is able to detect contextual anomalies. However, this method requires substantial supervision as labeled instances of normal behavior are required.

The work proposed in[31] extracts texture information from patches in the non-stationary parts of the video. Patches are clustered into one of two behavior categories: A and B. Patches roughly correspond to moving blobs. Contextual information for each blob is extracted by taking into account the categories of the nearest blob neighbors. This approach shows good results on test sequences, however, the number of behavior categories or clusters is arbitrarily chosen for the application domain (bi-directional pedestrian motion in [31]) and may behave differently in other application domains.

The authors in [24] describe a Point-wise Motion Image in which motion information is coded in each color component (speed, orientation and motion duration, respectively). A correspondence measure is developed to detect anomalies in a given PMI. This technique is only capable of detecting point anomalies as it does not take into account sequential information.

Authors in [20] start from low level features from moving pixels: position and motion direction (from optical-flow). Quantized position and motion direction are assigned a word from a codebook. Unsupervised learning is employed, using techniques from language processing for clustering (Hierarchical Bayesian Models). Similar to other approaches, unseen observations are considered anomalies if they have a low likelihood. Additionally, anomalies based on interactions can be detected.

In [32], a framework for detecting different types of anomalies in video sequences is described. Pixel activity is model by a binary Markov chain that associates a feature vector (size, shape, motion) with the moving state. The transitions between moving and background states, along with the associated features, provide a statistical model for normal activity. For point-based anomalies, the authors take an approximation to the probability density function of normal activity from the model and classify unseen observations as anomalies based on low probability. For spatial co-occurring anomalies, Markov Random Fields (MRFs) are incorporated into the framework. Furthermore, the authors describe a framework for multi-camera anomaly detection.

The aforementioned approaches pose significant advantages in scenarios in the presence of clutter, when compared to those that rely on tracking information. However, the proposed solutions are diverse in terms of contextual depth and the anomalies that can be detected, which makes it difficult to compare different techniques.

## 2.3 Fuzzy Logic

Fuzzy Logic was initiated in 1965 [1], [2], [3], by Lotfi A. Zadeh , professor for computer science at the University of California in Berkeley. Basically, Fuzzy Logic (FL) is a multivalued logic, that allows intermediate values to be defined between conventional evaluations like true/false, yes/no, high/low, etc. Notions like rather tall or very fast can be formulated mathematically and processed by computers, in order to apply a more human-like way of thinking in the programming of computers [4]. Fuzzy systems is an alternative to traditional notions of set membership and logic that has its origins in ancient Greek philosophy. The precision of mathematics owes its success in large part to the efforts of Aristotle and the philosophers who preceded him. In their efforts to devise a concise theory of logic, and later mathematics, the so-called ”Laws of Thought” were posited [5]. One of these, the ”Law of the

Excluded Middle," states that every proposition must either be True or False. Even when Parminedes proposed the first version of this law (around 400 B.C.) there were strong and immediate objections: for example, Heraclitus proposed that things could be simultaneously True and not True. It was Plato who laid the foundation for what would become fuzzy logic, indicating that there was a third region (beyond True and False) where these opposites "tumbled about." Other, more modern philosophers echoed his sentiments, notably Hegel, Marx, and Engels. But it was Lukasiewicz who first proposed a systematic alternative to the bi–valued logic of Aristotle [6]. Even in the present time some Greeks are still outstanding examples for fussiness and fuzziness, (note: the connection to logic got lost somewhere during the last 2 mileniums [7]). Fuzzy Logic has emerged as a a profitable tool for the controlling and steering of of systems and complex industrial processes, as well as for household and entertainment electronics, as well as for other expert systems and applications like the classification of SAR data.

### 2.3.1 Fuzzy Sets and Crisp Sets

The very basic notion of fuzzy systems is a fuzzy (sub)set. In classical mathematics we are familiar with what we call crisp sets. For example, the possible interferometric coherence g values are the set X of all real numbers between 0 and 1. From this set X a subset A can be defined, (e.g. all values $0 \le g \le 0.2$). The characteristic function of A, (i.e. this function assigns a number 1 or 0 to each element in X, depending on whether the element is in the subset A or not) is shown in Fig.1.

The elements which have been assigned the number 1 can be interpreted as the elements that are in the set A and the elements which have assigned the number 0 as the elements that are not in the set A.

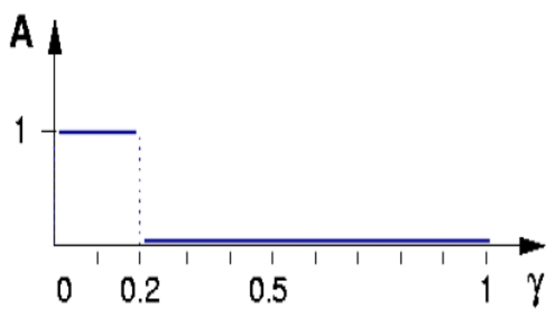

Fig. Characteristic Function of a Crisp set

This concept is sufficient for many areas of applications, but it can easily be seen, that it lacks in flexibility for some applications like classification of remotely sensed data analysis. For example it is well known that water shows low interferometric coherence g in SAR images. Since g starts at 0, the lower range of this set ought to be clear. The upper range, on the other hand, is rather hard to define. As a first attempt, we set the upper range to 0.2. Therefore we get B as a crisp interval B=[0,0.2]. But this means that a g value of 0.20 is low but a g value of 0.21 not. Obviously, this is a structural problem, for if we moved the upper boundary of the range from g =0.20 to an arbitrary point we can pose the same question. A more natural way to construct the set B would be to relax the strict separation between low and not low. This can be done by allowing not only the (crisp) decision Yes/No, but more flexible rules like " fairly low". A fuzzy set allows us to define such a notion. The aim is to use fuzzy sets in order to make computers more 'intelligent', therefore, the idea above has to be coded more formally. In the example, all the elements were coded with 0 or 1. A straight way to generalize this concept, is to allow more values between 0 and 1. In fact, infinitely many alternatives can be allowed between the boundaries 0 and 1, namely the unit interval I = [0, 1]. The interpretation of the numbers, now assigned to all elements is much more difficult. Of course, again the number 1 assigned to an element means, that the element is in the set B and 0 means that the element is definitely not in the set B. All other values mean a gradual membership to the set B. This is shown in Fig. 2. The membership function is a graphical representation of the magnitude of participation of each input. It associates a weighting with each of the inputs that are processed, define functional overlap between inputs, and ultimately determines an output response. The rules use the input membership values as weighting factors to determine their influence on the fuzzy output sets of the final output conclusion.

The membership function, operating in this case on the fuzzy set of interferometric coherence g, returns a value between 0.0 and 1.0. For example, an interferometric coherence g of 0.3 has a membership of 0.5 to the set low coherence (see Fig. 2). It is important to point out the distinction between fuzzy logic and probability. Both operate over the same numeric range, and have similar values: 0.0 representing False (or non-membership), and 1.0 representing True (or full-membership). However, there is a distinction to be made between the two statements: The probabilistic approach yields the natural-language statement, "There is an 50% chance that g is low,” while the fuzzy terminology corresponds to ”g’s degree of membership within the set of low interferometric coherence is 0.50.” The semantic difference is significant: the first view supposes that g is or is not low; it is just that we only have an 50% chance of knowing which set it is in. By contrast, fuzzy terminology supposes that g is ”more or less” low, or in some other term corresponding to the value of 0.50.

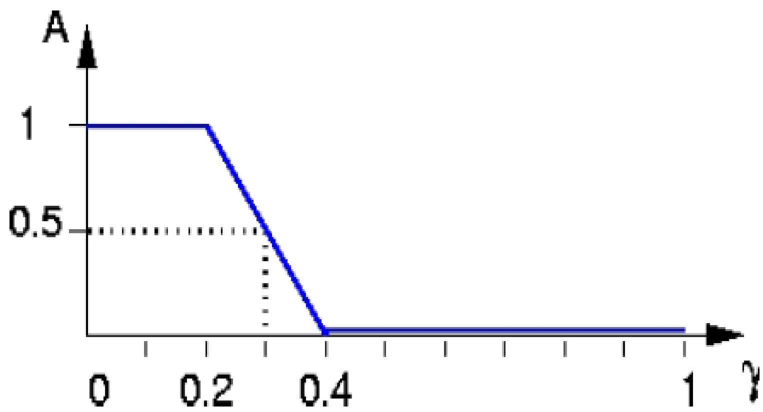


Fig. Characteristic Function of Fuzzy Set

### 2.3.2 Membership Functions

For representation of the membership functions, we can use the following functions:

• Triangular Membership Functions

A triangular MF, as shown in Figure 2.7 (a), is a function with 3 parameters defined by

$$\text{triangle}(x\ ;\ a\ ,b\ ,c) = \max(\min(\frac{x-a}{b-a},\frac{c-x}{c-b}),0)$$

• Trapezoidal Membership Functions

A Trapezoidal MF, as shown in Figure 2.7 (b), is a function with 4 parameters defined by

$$\text{trapezoid}(x\ ;\ a\ ,b\ ,c\ ,d) = \max(\min(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}),0)$$

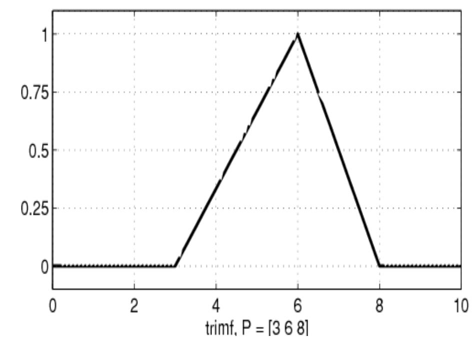


Fig. Triangular membership function

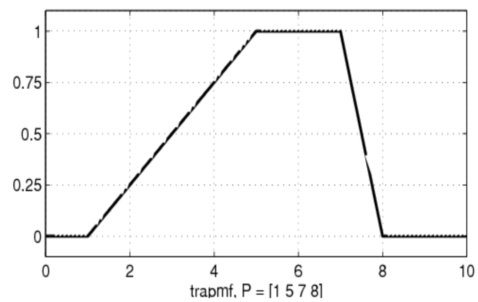


Fig. Trapezoidal membership function

• Gaussian Membership Functions

A Gaussian MF is a function with two parameters defined by

$$\text{gaussian}(x\,;\,\sigma\,,c) = e^{\frac{-(x-c)^2}{\sigma^3}}$$

where c is the center and σ is the width of membership function

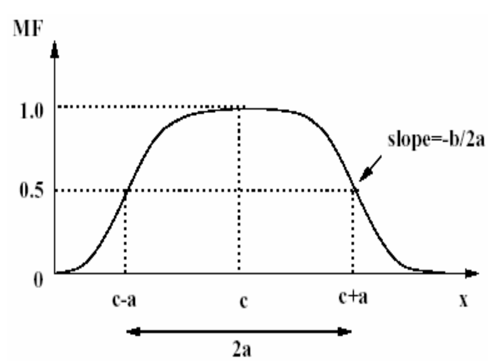


Fig. Bell membership function

• Bell Membership Functions

A bell MF, as shown in Figure 2.8, is a function with two parameters defined by

$$\text{bell}(x\ ;\ a\ ,b\ ,c) = \frac{1}{1+|\frac{(x-c)^{2b}}{a}|}$$

• Sigmoidal Membership Function

A Sigmoid MF is a function with two parameters defined by

$$\text{sigmoid}(x;\ k\ ,c)= \frac{1}{1+\ e^{-k(x-c)}}$$

where parameter k influences sharpness of function in the point where a = c. If k >0 the function is open on right site, on the other hand, if k<0 the function is open on left site and therefore this function can be use for describing conceptions like “very big” or “very small”. Sigmoid function is very often used in Neural Networks like activation function.

## 2.3.3 Fuzzy Rules and Fuzzy Reasoning

Fuzzy rules and fuzzy reasoning are the backbone of fuzzy inference systems, which are the most important modeling tool based on fuzzy set theory. They have been applied to a wide range of real-world problems, such as expert systems, pattern recognition, and data classification. A detailed discussion about fuzzy inference systems is provided in [Jang et al., 1998].

### 2.3.3.1 Fuzzy If-Then Rules

Fuzzy if-then rules (also known as fuzzy conditional statements) are expressions of the form

If x is A , then y is B

where A and B are linguistic labels defined by fuzzy sets on universe of discourse X and Y, respectively. Often “x is A” is called the antecedent or premise, while “y is B” is called the consequence or conclusion. Due to their concise form, fuzzy if-then rules are often used to capture the imprecise modes of reasoning and play an essential role in the human ability to make decisions in an environment of uncertainty and imprecision. Fuzzy if-then rules have been used extensively in both modeling and control. From another angle, due to the qualifiers on the premise parts, each fuzzy if-then rule can be viewed as a local description of the system under consideration.

#### 2.3.3.2 Fuzzy Reasoning

Fuzzy reasoning, also known as approximate reasoning, is an inference procedure that derives conclusions from a set of fuzzy if-then rules and known facts.

# CHAPTER 4

# PROPOSED WORK

This chapter gives the understanding of the proposed work along with the details of the new features introduced. Our work is divided in to two parts, first part tracks multiple objects in the video using kalman filter. The Kalman filter, also known as linear quadratic estimation (LQE), is an algorithm that uses a series of measurements observed over time, containing noise (random variations) and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone. More formally, the Kalman filter operates recursively on streams of noisy input data to produce a statistically optimal estimate of the underlying system state.

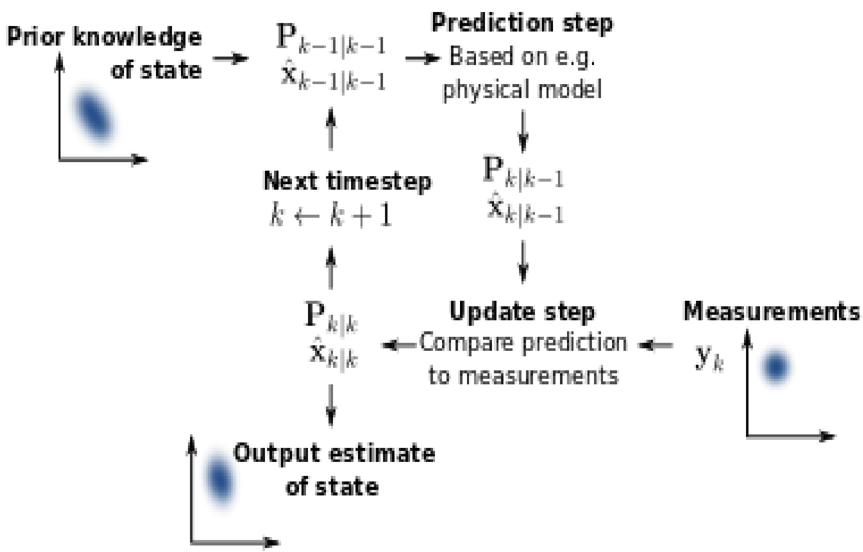


Fig. Kalman filter estimation process

The Kalman filter keeps track of the estimated state 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.

The second part extracts the trajectory of the objects moving in the video and matches this trajectory with the training dictionary using fuzzy logic. If the current trajectory matches with any trajectory in the training dictionary then it’s a normal event. But if it does not matches with any trajectory in the training dictionary then it is an anomalous event. We make use of fuzzy membership function for approximate matching of the trajectories.

## 4.1 OBJECT TRACKING USING KALMAN FILTER

### 4.1.1 Typical Kalman filter

Mathematically, Kalman filer is an estimator that predicts and corrects the states of wide range of linear processes[9]. It is not only efficient practically but attractive theoretically as well Precisely, the optimal state is found with smallest possible variance error, recursively. However, an accurate model is an essential requirement.

In Kalman filer, we consider a tracking system where k x is the state vector which represents the dynamic behaviour of the object, where subscript k indicate the discrete time. The objective is to estimate k x from the measurement k z . Following is the mathematical description of Kalman filer, which for understanding we have sectioned into four phases.

#### 4.1.1.1) Process equation

$$x(k)=Ax(k-1)+w(k-1) \tag{1}$$

Where A represents the transition matrix and x(k) the state at time k -1 to k . Vector w(k-1) is the Gaussian process noise N(.) With following normal probability distribution p(w) .

$$p(w) \sim N(0,Q) \tag{2}$$

#### 4.1.1.2) Measurement equation

$$Z(k) = Hx(k) + v(k) \tag{3}$$

Where H is the measurement matrix and k z is the measurement observed at time k -1 to k respectively. V(k) is the Gaussian measurement noise N(.) with normal probability distribution p(v) .

$$p(v) \sim N(0, R) \tag{4}$$

#### 4.1.1.3) Time update equations

Equation (1) and (3) describes a linear model at time k. As x(k) is not measured directly, therefore the information provided by measured z(k) is used to update the unknown states x(k). Apriori estimate of state ˆx(k) and covariance error ˆx(k) estimate is obtained for the next time step k .

$$\hat{x}_k^- = A\hat{x}_{k-1} + w_k$$
$$P_k^- = AP_{k-1}A^T + Q \tag{6}$$

#### 4.1.1.4) Measurement update equations

These equations are associated with the feedback of the system. The objective is to estimate aposteriori estimating ˆx(k) which is a linear combination of the apriori estimate and the new measurement z(k) .These equations are given below:

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$
$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$
$$P_k = (1 - K_k H)P_k^-$$

K is the Kalman gain which is computed by above the measurement update equations. After that aposterior state estimate ˆx(k) and aposterior error estimate P(k) is computed by the measurement z(k) .The time and measurement equations are calculated recursively with previous aposterior estimates to predict new aprior estimate. This recursive behaviour of estimating the states is one of the highlights of the Kalman filter.

### 4.1.2 Kalman Filter for Multi-object Tracking

Describing the object's geometric features can include location, shape and center of mass (centroid)[10], etc.. The camera we used in this paper sampling time is 30fps, so there is relatively little changed of moving object in two adjacent frames, so we can consider that the size of tracking window and centroid position of moving target in the adjacent two frames are little changed, that is said, moving target in the adjacent two frames of the movement is continuously. Therefore, we choose centroid and size of tracking window as the feature value to describe moving object. After the moving objects have been segmented, some process preparations for the subsequent moving object tracking is needed. First to assign a tracking window for each moving object in the scene, in order to prevent a lot of noise easy to come in, tracking window should not be too large. We make the tracking window size slightly larger than the object image size, so that it not only can reduce the noise interference, but also can reduce image processing time and increase speed of operation. Kalman filter tracking model can be divided into three sub-modules which are :motion model, feature matching, model update[11].

#### 4.1.2.1) Motion Estimation Model

Kalman filter used for tracking is defined in terms of its states, motion model, and measurement equations matrix k X is an eight-dimensional system state vector, which can be expressed as:

$$x(k) = [x(0,k), y(0,k), l(k), h(k), v(x,k), v(y,k), v(l,k), v(h,k)]^T \tag{10}$$

Where, $x(0,k)$ , $y(0,k)$ represent horizontal and vertical centroid coordinate, l(k) , h(k) represent half-width and half-height of the tracking window, v(x,k) , v(y,k) , v(l,k) , v(h,k) represents their speed respectively.

The measurement vector of the system adopts the following from:

$$z(k) = [x(0,k), y(0,k), l(k), h(k)]^T \tag{11}$$

In the following, A is the transition matrix and H is the measurement matrix of our tracking system along with the Gaussian process w(k) and measurement v(k) . These noise values are entirely dependent on the system that is being tracked and adjusted empirically.

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Observation matrix *H* can be described as:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

After the state equation and measurement equation of motion model are defined, in the next frame, Kalman filter can be used to estimate the object's location and size in a small range, and to gain trajectories of moving objects.

**Feature matching**

Each moving object is described by its centroid and tracking window, the horizontal and vertical centroid coordinates and the area of the i object in the k frame were respectively described as $x_k^i$, $y_k^i$ and $S_k^i$.

First, centroid distance function between the *ith* object in the *kth* frame and the *jth* object in the *k*+1*th* frame is defined as:

$$D(i,j) = \frac{\left|\sqrt{(x_k^i - x_{k+1}^j)^2 + (y_k^i - y_{k+1}^j)^2}\right|}{Max_n \left|\sqrt{(x_k^i - x_{k+1}^n)^2 + (y_k^i - y_{k+1}^n)^2}\right|} \tag{12}$$

Second, the area difference between the *ith* object in the *k* frame and the *jth* object in the *k*+1*th* frame is defined as:

$$A(i,j) = \frac{\left|S_k^i - S_{k+1}^i\right|}{Max_n \left|S_k^n - S_{k+1}^n\right|} \tag{13}$$

It reflects the degree of deformation of the window, the smaller the value is, the closer description of the two objects's shape is.

With these definitions we define cost function is:

$$V(i,j) = \alpha D(i,j) + \beta A(i,j) \tag{14}$$

Where, $\alpha + \beta = 1$, we can make $\alpha = 0.8$, $\beta = 0.2$ here. The smaller the cost function's value is, the two objects are more likely have correspondence.

**Model Update**

When the minimum value of cost function is found, we use the $k+1th$ frame features to update parameters of Kalman filter motion model, and use them as the input in the next frame. Repeatedly doing this to finish the model update until the moving objects disappeared.

## 4.1.3 OCCLUSION PROBLEM

When we found by detection that multiple objects's region connected together, we believe that the objects' region have merged, to make the multiple objects as a whole object to track, and to establish a new eigen value for object matching. When a object contains more than one moving object split into several independent moving objects, first to judge whether merge occurred before, if it happened, matching the split objects' with objects' feature before splitting. If not, we consider the objects are news by splitting, new eigen values will be established and new tracking windows will be assigned for moving objects tracking.

The algorithm steps are as follows:

1 If the current image is the first frame, establishing motion model and assigning tracking window for each moving object in the scene. If the current image is the kth frame, and the moving object do not fall into any of those established tracking windows, we consider it is a new object, establishing a new Kalman filter motion model, initializing the model for tracking.

2. Searching features for each object near the tracking window in the scene, calculate the value of the cost function, the minimum value is the best match.

3. To judge whether there is an occlusion happened, if it happened, go to the merge or split treatment. If not, keep tracking the object until it disappeared.

4 Turning to the handling of the next frame until the object disappeared, the tracking is complete.

An example of object tracking is shown in fig. below.

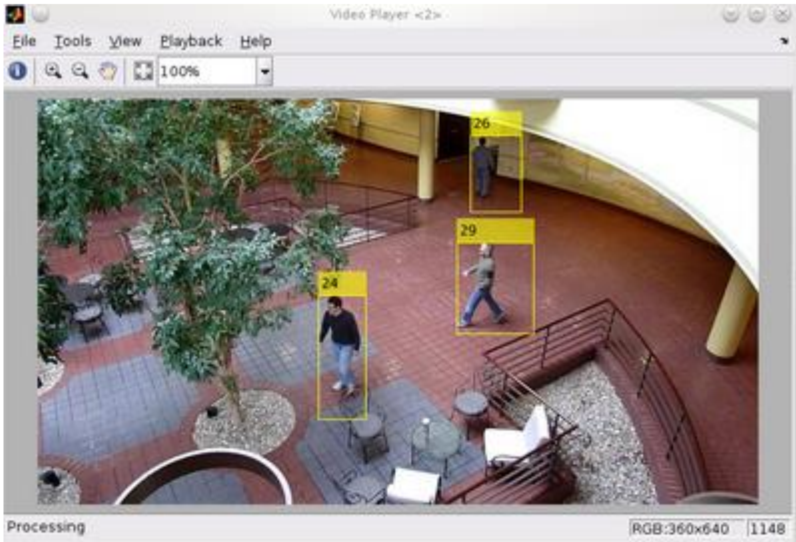


Fig. Tracking of multiple objects using Kalman filter

## 4.2 Trajectory Matching using Fuzzy Logic

Trajectories are generally represented by variable-length sequences of 2-D coordinates as

$$((X_1,y_1), (X_2,y_2),(X_3,y_3)\ldots\ldots\ldots\ldots\ldots(X_n,y_n)) \qquad (15)$$

Where $X_i$ denotes the x-coordinate of the centroid of the object and $y_i$ denotes the x-coordinate of the centroid of the object

### 4.2.1 Training

First, a training dictionary is created by extracting the centroids of the different moving objects in respective frames of the video. The trajectories are found by storing the centroids of the moving objects in a 2-dimensional matrix. Each column of the matrix stores a single trajectory.

The training matrix can be represented as :

$$A = [T_1\ T_2\ T_3\ T_4\ \ldots\ldots\ldots\ldots\ldots T_n] \qquad (16)$$

Where, n is the total number of trajectories. Each $T_i$ is a 1*2N matrix storing the trajectory of a single object.

$$T_i = \begin{pmatrix} x_{i,1} \\ y_{i,1} \\ x_{i,2} \\ y_{i,2} \\ : \\ : \\ : \\ x_{i,N} \\ y_{i,N} \end{pmatrix} \tag{17}$$

where, $x_{i,1}$ is the x-coordinate of the $i_{th}$ object detected in the first frame it is detected and $y_{i,1}$ is the y-coordinate of the $i_{th}$ object detected in the first frame it is detected. Similarly, $x_{i,N}$ is the x-coordinate of the $i_{th}$ object detected in the Nth frame it is detected and $y_{i,1}$ is the y-coordinate of the $i_{th}$ object detected in the Nth frame it is detected.

**4.2.2) Membership function**

A membership function (MF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. The input space is sometimes referred to as the universe of discourse, a fancy name for a simple concept.

There are different membership functions like:

- Triangular membership function
- Trapezoidal membership function
- Gaussian membership function
- Bell membership function
- Sigmoidal membership function, etc

These membership functions differ in their shapes. Membership function is chosen according to the input given and the output value required.

Here we use modified Gaussian function in our work. Our membership function is given by the equation:

$$\mathrm{mgauss}(x;\ \sigma, c) = e^{\frac{-(x-c)^N}{\sigma^2}} \quad (18)$$

where c is the center and σ is the width of membership function and N varies from 1 to 5 i.e.

$$1 \leq N \leq 5$$

The membership function can be given by the curve given below in fig.

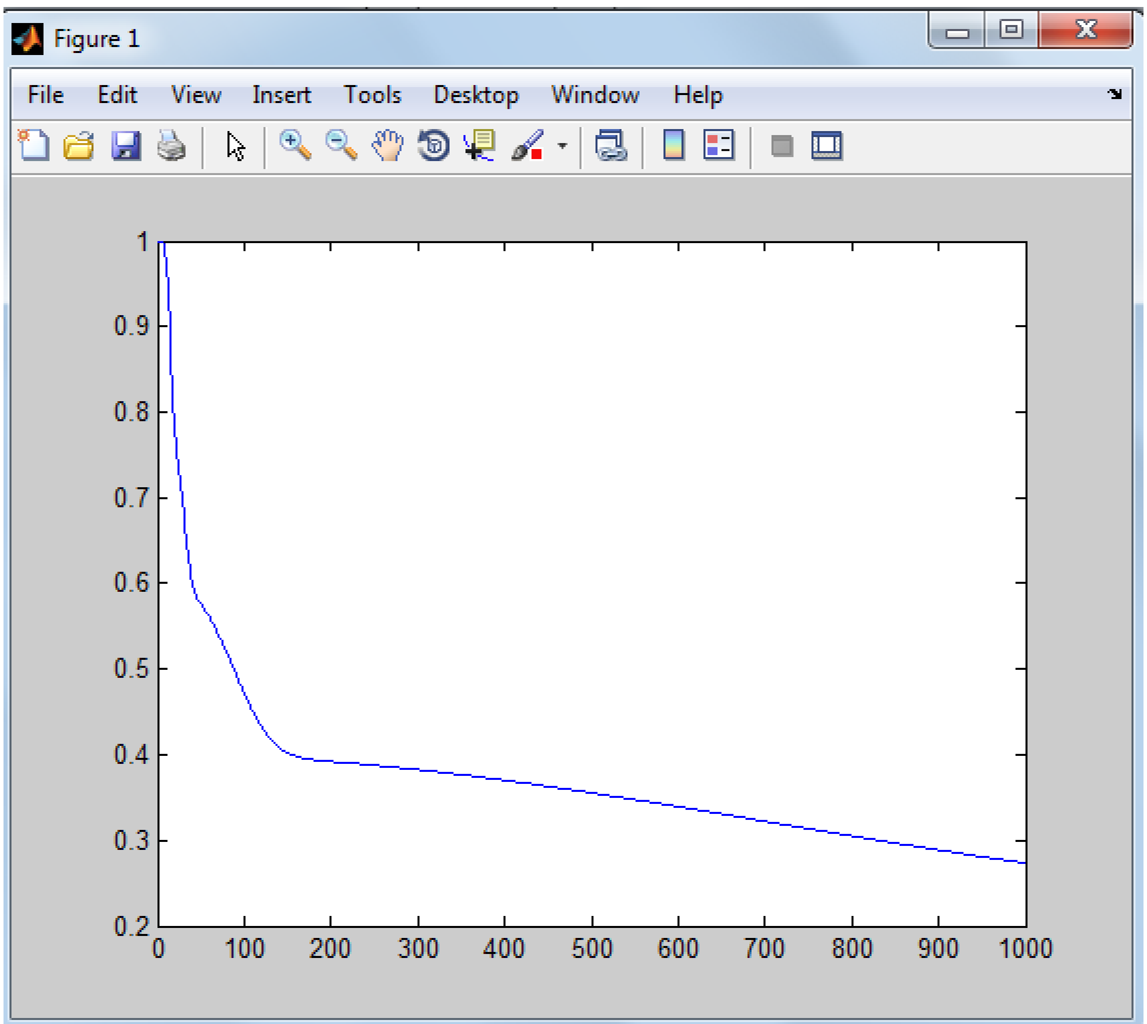


Fig. Membership Function

### 4.2.3) Trajectory Matching

The final and main step of our proposed method is trajectory matching. A video is given as input to our algorithm. Our proposed method tracks all the moving objects in the video file and extracts the trajectory of all the objects and stores this trajectory in a matrix. Let there are ‘p’ objects in video. Then the matrix can be seen as a p* 2N matrix:

$$A = [T_1\ T_2\ T_3\ T_4 \ldots\ldots\ldots\ldots T_p] \tag{19}$$

where each $T_i$ is a column matrix containing trajectory of $i_{th}$ moving object.

$$Ti = \begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ : \\ : \\ x_n \\ y_n \end{pmatrix} \tag{20}$$

where n is the number of frames for which the object is being detected.

Now the trajectory matching of each object is done individually with the training dictionary which was stored previously. Training dictionary contains different trajectories extracted from the training video set which was provided to the trajectory tracking algorithm as input. More than 1 training video is used to get the training dictionary.

The matching of each trajectory in the video is done using fuzzy logic. The distance of each trajectory is found with each trajectory already stored in the training dictionary. The distance is then given as input to the membership function and the membership value of each distance is calculated. The training trajectory having smallest distance from the current trajectory is chosen.