Chapter 1

INTRODUCTION

1.1. Introduction

In India 24% of the area is covered by forest. Forest industry is source of significant income to over 400 million people in India, mostly rural. Population in India is increasing at high rate to provide shelter to all of them is great challenge. So most of the people live around forest to fulfill their needs.

Since people live closer to forest they often encounter wild animals and for safety of human being often wild animals are killed or human beings are attacked by animals. Now safety of both human and animal is equally important. There is a need to implement warning system to make sure human and wild animals live safely. This has two parts: one where humans should get notification whenever wild animal comes near residential area and other part is to keep wild animal away from human without harming them.

Our project intends to solve these issues. There are number of villages which are situated near the forest area, so there is risk in front of people from wild animal. Hence animal detection plays an important role in day to day life as there is a risk for human life and the properties.

Image processing and machine vision represent an exciting and dynamic part of cognitive and computer science. Remote sensing, technical diagnostics, autonomous vehicle guidance, medical imaging (2D and 3D) and automatic surveillance are the most rapidly developing areas. In this project image processing is applied for wild animal detection and recognition. ANN is an application of machine learning. Machine learning is an artificial intelligence that learns from data without being programmed. Main objective of machine learning is to generalize from its experience.

Wild animal detection and monitoring is done using back propagation neural network. The database for training the algorithm is taken from Caltec-101. The test image is a cropped image from the input video. The motion of the animal in input video is detected using foreground detection and the animal is cropped which is the test image for the algorithm.

Chapter 2

LITERATURE SURVEY

2.1. Literature Survey

The work by Nitesh Sanklecha et al. "Motion Detection and Tracking of a Leopard in a Video" is able to detect motion and track an animal in a video. In simple words, tracking may be defined as the estimation of the trajectory of a moving object in the image plane as it moves around the scene. Many research has been done in tracking a moving object. Most of the research is been carried out on tracking either Human Beings (like, Face Recognition, Waving hand etc.) or Vehicles. In this Paper, two scenarios are considered. First problem is tracking a Leopard when no other animal is present. Second problem is tracking a Leopard in the presence of other animal. Tracking of a Leopard is implemented in Simulink using Horn – Schunck and Lucas – Kanade method of Optical Flow, and also the Background Subtraction method. A comparative analysis is being done across both Optical Flow and Background Subtraction methods.

The objective of this project to track the Leopard in a two different scenario of a video sequence was achieved. Using the Optical Flow method wherein the Horn-Schunck algorithm for motion estimation was put into effect which gave fair results. The disadvantage in using Horn – Schunck method is that it cannot eliminate the noise. Using Background Subtraction algorithm, satisfactory results were obtained. This method also could not eliminate the presence of noise. An attempt of using Lucas – Kanade algorithm gave the best results, were only the desired object was tracked and without any noise. Hence the performance parameters of this algorithm gave 100% result. The disadvantage of this method is there are some discontinuities observed while determining the Optical Flow pattern.

The work by Adedeji Olugboja et al."Detection of Moving Objects using Foreground Detector and improved morphological filter" is able to detect moving object. In this work Gaussian Mixture Model(GMM)is applied, which is established on background subtraction. Smoothing method was used for the pre-processing stage and morphological filter was applied to remove the unwanted pixels out of the background in other to resolve the background noise disruption problem.

We also demonstrated that filtering the foreground segmentation twice with the same morphological structured element but with different width was used to improve the accuracy of the result. Results shows that the proposed method can detect and track effectively the moving cars, compared to filtering the foreground segmentation just once.

From all these results we conclude that, background subtraction method gives us better and accurate result for moving vehicles detection. Filtering the foreground segmentation twice gave us a better segmentation for moving vehicles and produces good output because of the removal of the noise which now makes the blob analysis to produce accurate bounding to each of the vehicles. The simulation result shows the efficiency of the proposed method.

The work by Ankush Mittal et al. "A vision based human-elephant collision detection system" is able to avoid human-elephant collision. Elephant intrusion in areas with high human movement can prove lethal for both human beings and elephants. The proposed system seeks to identify elephants with the aid of a Video Camera. The suggested methodology was applied to zones having high intervention of human beings and elephants. Regions with higher human movements like roads were extracted from the initial video frames. This process is followed by detecting motion in the video frame. The objects in the area of motion are then identified as elephant or non-elephant with the help of Support Vector Machines (SVM) classifiers. A dataset constituting of images of elephants and other objects was used for training the proposed algorithm. An overall accuracy of 85.29% was attained when static images containing elephants and other objects were classified. The same approach for detection was applied to identify moving elephants in the video frames. The proposed system endeavors to prevent casualties that occur in areas having high human elephant interaction.

The work by Indrabayu et al."Vehicle Detection and Tracking using Gaussian Mixture Model and Kalman Filter" is able to detect and track vehicles. Intelligent Transport System (ITS) is a method used in traffic arrangements to make efficient road transport system. One of the its application is the detection and tracking of vehicle objects. In this research, Gaussian Mixture Model (GMM)method was applied for vehicle detection and Kalman Filter method was applied for object tracking. The data used are vehicles video under two different conditions.

First condition is light traffic and second condition is heavy traffic. Validation of detection system is conducted using Receiver Operating Characteristic(ROC) analysis. The result of this research shows thatthelighttrafficconditiongets100%fortheprecisionvalue,94.44% for sensitivity, 100% for specificity, and 97.22% for accuracy. While the heavy traffic condition gets 75.79% for the precision value, 88.89% for sensitivity, 70.37% for specificity, and 79.63% for accuracy. With average consistency of Kalman Filter for object tracking is 100%. The research is conducted using data of vehicle video and divided into two conditions i.e. light traffic and heavy traffic.

The detection object uses Gaussian Mixture Model method and the tracking object uses Kalman filter method. System validation for detection object is conducted with using ROC analysis the parameters of precision, sensitivity, specivisity and accuracy. Light traffic condition obtains the precision of 100%, sensitivity of 94.44% specificity of 100% and accuracy of 97.22%. While for heavy traffic condition obtains the precision of 75.79%, sensitivity of 88.89%, specificity of 70.37% and accuracy of 79.63%. The consistency of tracking object reaches 100%. The results show that GMM method working properly in light traffic.

The work by Chris Stauffer et al. “Adaptive background mixture models for real-time tracking” is able to detect real time tracking of moving region. A common method for real-time segmentation of moving regions in image sequences involves “background subtraction,” or Thersholding the error between an estimate of the image without moving objects and the current image. The numerous approaches to this problem differ in the type of background model used and the procedure used to update the model. This paper discusses modeling each pixel as a mixture of Gaussians and using an on-line approximation to update the model. The Gaussian distributions of the adaptive mixture model are then evaluated to determine which are most likely to result from a background process. Each pixel is classified based on whether the Gaussian distribution which represents it most effectively is considered part of the background model.

This results in a stable, real-time outdoor tracker which reliably deals with lighting changes, repetitive motions from clutter, and long-term scene changes.

This system has been run almost continuously for 16months, 24 hours a day, through rain and snow. This paper has shown a novel, probabilistic method for background subtraction. It involves modeling each pixel as a separate mixture model.

We implemented a real-time approximate method which is stable and robust. The method requires only two parameters, Y and T. These two parameters are robust to different cameras and different scenes. This method deals with slow lighting changes by slowly adapting the values of the Gaussians. It also deals with multi-modal distributions caused by shadows, specularities, swaying branches, computer monitors, and other troublesome features of the real world which are not often mentioned in computer vision. It recovers quickly when background reappears and has a automatic pixel-wise threshold.

The work by Deepak Kumar Panda et al. "A Gaussian Mixture Model with Gaussian Weight Learning Rate and Foreground Detection using Neighbourhood Correlation" is able to detect moving objects. Moving object detection is the first and foremost step in many computer vision applications such as automated visual surveillance, human-machine interface, tracking, traffic surveillance, etc. Background subtraction is widely used for classifying image pixels into either foreground or background in presence of stationary cameras. A Gaussian Mixture Model (GMM) model is one such popular method used for background subtraction due to a good compromise between robustness to various practical environments and real-time constraints. In this paper we assume background pixel follows Gaussian distribution spatially as well as temporally.

The proposed research uses Gaussian weight learning rate over a neighbourhood to update the parameters of GMM. The background pixel can be dynamic especially in outdoor environment, so in this paper we have exploited neighborhood correlation of pixels in foreground detection. We compare our method with other state-of-the art modeling techniques and report experimental results. The performance of the proposed algorithm is evaluated using both qualitative and quantitative measures. Quantitative accuracy measurement is obtained from *PCC*. Experimental results are demonstrated on publicly available videos sequences containing complex dynamic backgrounds. The proposed method is quiet effective enough to provide accurate silhouette of the moving object for real-time surveillance.

In this paper a new Gaussian weight learning rate for Gaussian mixture model based background subtraction is presented. The foreground detection is done by exploiting the neighborhood correlation of a pixel. The strength of the scheme lies in simple changes to the existing update equation and foreground detection of GMM model by considering the neighborhood of a pixel. A comparison has been made between the proposed algorithm and the state-of-the-art methods.

The results obtained by the proposed scheme are found to provide accurate silhouette of moving objects in complex video sequence. The algorithm fails to detect shadow. In our future research, this problem will be at the center stage.

The work by Thierry Bouwmans et al. "Background Modeling using Mixture of Gaussians for Foreground Detection" is able to detect moving object using static cameras Mixture of Gaussians(MOG) is a widely used approach for background modeling to detect moving objects from static cameras. Numerous improvements of the original method developed by Stauffer and Grimson [18] have been proposed over the recent years and the purpose of this paper is to provide a survey and an original classification of these improvements.

We also discuss relevant issues to reduce the computation time. Firstly, the original MOG are reminded and discussed following the challenges met in video sequences. Then, we categorize the different improvements found in the literature. We have classified them in term of strategies used to improve the original MOG and we have discussed them in term of the critical situations they claim to handle. After analyzing the strategies and identifying their limitations, we conclude with several promising directions for future research.

In conclusion, this paper allows the reader to survey the strategies and it can effectively guide him to select the best improvement for his specific application. Particularly, this survey paper allows: 1) Developers to choose the appropriate improvement to tackle the critical situations met in their application. 2) Researchers to have a recent state-of-the-art and so easily identify new ideas to improve the MOG. 3)Reviewers to verify quickly the originality of a paper. Our future works concern a recent state-of the-art and a repository of the most common background subtraction methods.

The work by Alexander Filonenko et al. "Detecting Illegally Parked Vehicle Based on Cumulative Dual Foreground Difference" As one of the traffic monitoring tasks, detecting an illegally parked vehicle aims to prevent car crashing between parked and other vehicles. However, developing such a task becomes more complex due to weather conditions, occlusion, illumination changing, and other factors. This work addresses a framework to detect an illegally parked vehicle using a cumulative dual foreground difference. In our framework, two background models with different learning rates are generated based on a Gaussian mixture model, defined as short- and long-term models.

Each model extracts foreground pixels and the stability of these pixels are then analyzed based on cumulative values and temporal positions over a certain period of time. Subsequently, the connected component labeling is performed on the static pixels to form stable regions. To determine whether the candidate region is vehicle, a rule-based filtering approach is performed. Finally, the detection-based tracking is applied to reduce false positives. The effectiveness of the proposed framework is evaluated using i-LIDS and IS Lab dataset.

The experiment results show that the proposed framework is efficient and robust to detect an illegally parked vehicle. Thus, it can be considered as one of the task solutions for a traffic monitoring system. In this paper, a framework for detecting illegally parked vehicle in traffic scene has been implemented. To do such a task, two background model were built using two different learning rates. A smaller learning rate produces short-term background model, while a bigger learning builds long-term background model.

The method used cumulative dual foreground difference for extracting stable regions as candidate of filtering process is performed based on geometrical properties, such as area, aspect ration, and occupation ration. For detecting parked event, detection-based tracking schema is then applied. The matching criteria is obtained according to sum of squared different of intensity value, overlapping region and the distance between two regions. Our system was evaluated using i-LIDS dataset and IS Lab dataset. In our experiments, our system can detect successfully most parked vehicles.

However, the system may fail to detect multiple occluded illegally parked vehicles which are located close each others. Thus, in our future works, part-based model is planned to be implemented for verifying the multiple occluded vehicles one the scene.

The work by Balazs Tusor et al. “Improved Back-Propagation Algorithm for Neural Network Training” is able to learn new algorithm in Neural networks. Recently Artificial Neural Networks (ANNs) have become popular because they can learn complex mappings from the input/output data and are relatively easy to implement in any application. Although, a disadvantageous aspect of their usage is that they need (usually a significant amount of) time to be trained, which scales with the structural parameters of the networks and with the quality of the input data. However, the training can be done offline; it has a non-negligible cost and further, can cause a delay in the operation.

Fuzzy Neural Networks (FNNs) are the combinations of ANNs and Fuzzy logic in order to incorporate the advantages of both methods (learning ability of ANNs and the thinking ability of Fuzzy logic).

FNNs have Fuzzy values in their weight parameters and in the output of each neuron. Circular Fuzzy neural Networks (CFNNs) are FNNs with their topology realigned to a circular topology and the connections between the input layer and hidden layer trimmed. This may result in a dramatic reduction in the training time, while the precision and accuracy of the network are not affected. To further increase the speed of the training of the ANNs, FNNs, or CFNNs used for classification, a new training procedure is introduced in this paper.

2.2. Motivation

The prime motivation for this work are as follows:

Animal human interaction creates a lot of chaos and disturbs the balance between them. Those chaos are:

- There is injury and loss of life of both humans and animals.
- Significant damage to crops.
- Damage to human property.
- Destruction of human habitat.

So this algorithm is used to effectively detect and recognize the wild animals and notify the concerned authority about it.

2.3. Objectives

- To develop an algorithm to detect the wild animal.
- Extraction of moving animal using foreground detection.
- To design the algorithm using the Artificial Neural Network/Back propagation.
- Back propagation is used for training & foreground detection algorithm is used for testing.
- Display the detected animal.

Chapter 3

DESIGN AND IMPLEMENTATION

In this project the design of the proposed method can be represented using the Figure 3.1.

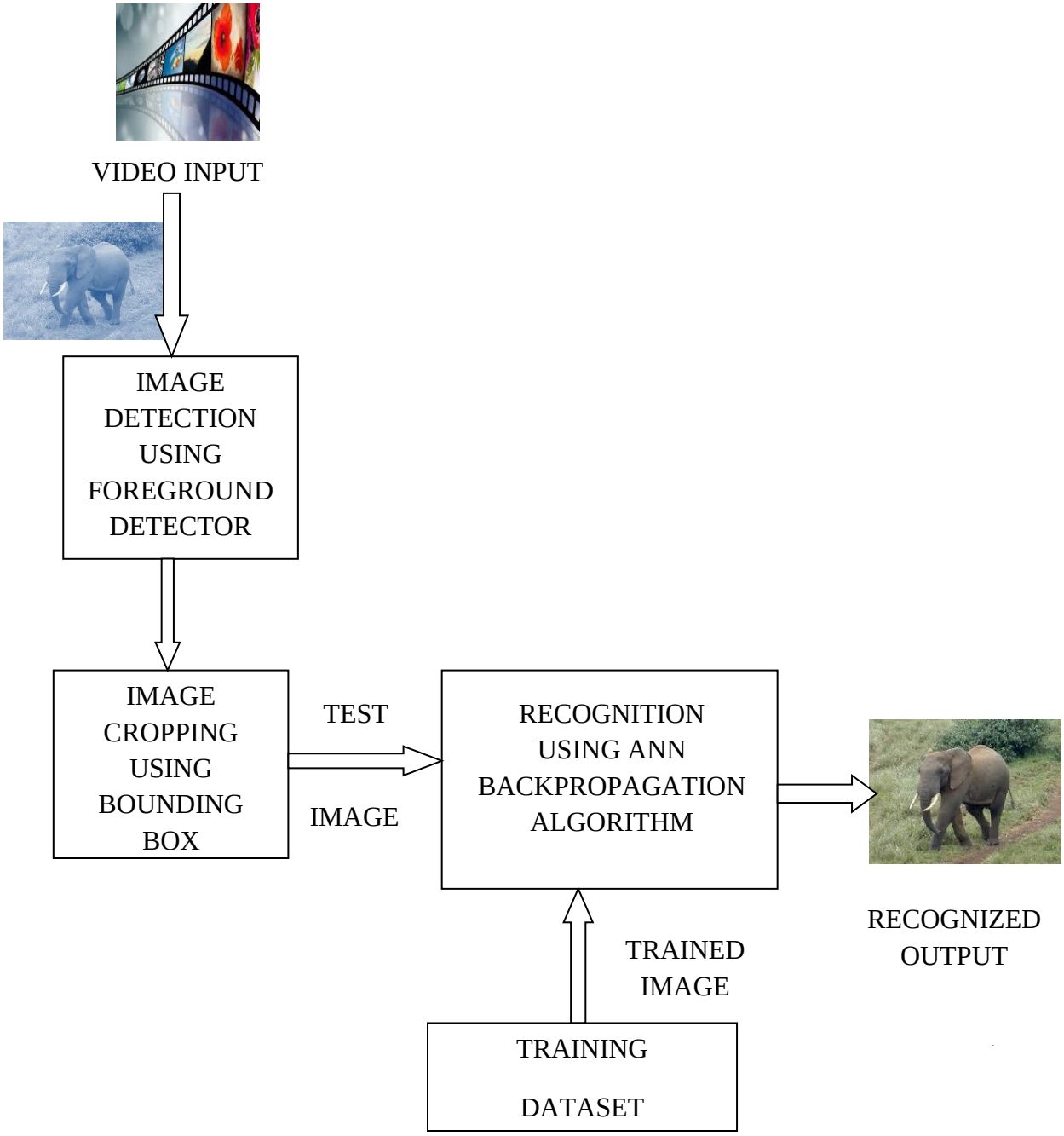

Figure 3.1. Design of proposed method

The input video is captured from a stationary camera which is in the form of .mpeg4 with the frame rate of 25frames per second. The movement of animals in the input video is detected using Foreground detector algorithm. If the binary pixel value is '0' then it is a background image which appears in black colour and if the binary pixel value is '1' then it is a foreground image which appears white.

Extraction of foreground animal is done using Back ground subtraction based Gaussian mixture model. Morphological filters are used to remove the noise from binary image obtained from background subtraction. Training and recognition is done using back propagation algorithm. If trained images are matched with the test image, then the animal is recognized.

3.1Foreground Detection

Foreground detection is one of the major tasks in the field of Computer Vision whose aim is to detect changes in image sequences. Many 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.

Detecting foreground to separate these changes taking place in the foreground of the background. It is a set of techniques that typically analyze the video sequences in real time and are recorded with a stationary camera.

All detection techniques are based on modeling the background of the image i.e. set the background and detect which changes occur. Defining the background can be very difficult when it contains shapes, shadows, and moving objects. In defining the background it is assumed that the stationary objects could vary in color and intensity over time.

Scenarios where these techniques apply tend to be very diverse. There can be highly variable sequences, such as images with very different lighting, interiors, exteriors, quality, and noise. In addition to processing in real time, systems need to be able to adapt to these changes.

A very good foreground detection system should be able to:

- Develop a background (estimate) model.

- Be robust to lighting changes, repetitive movements (leaves, waves, shadows), and long-term changes.

The Foreground Detector System object compares a color or grayscale video frame to a background model to determine whether individual pixels are part of the background or the foreground. It then computes a foreground mask. By using background subtraction, you can detect foreground objects in an image taken from a stationary camera.

3.1.1. Background Subtraction

Background subtraction, also known as foreground detection, is a technique in the fields of image processing and computer vision wherein an image's foreground is extracted for further processing (object recognition etc.).

Generally an image's regions of interest are objects (humans, cars, text etc.) in its foreground. After the stage of image preprocessing (which may include image denoising, post processing like morphology etc.) object localization is required which may make use of this technique.

Background subtraction is a widely used approach for detecting moving objects in videos from static cameras.

The rationale in the approach is that of detecting the moving objects from the difference between the current frame and a reference frame, often called "background image", or "background model". Background subtraction is mostly done if the image in question is a part of a video stream. Background subtraction provides important cues for numerous applications in computer vision, for example surveillance tracking or human poses estimation.

Background subtraction is generally based on a static background hypothesis which is often not applicable in real environments. With indoor scenes, reflections or animated images on screens lead to background changes.

Similarly, due to wind, rain or illumination changes brought by weather, static backgrounds methods have difficulties with outdoor scenes [14].

Background subtraction is the one of the crucial step in detecting the moving object. It is particularly a commonly used technique for motion segmentation in static images. It will detect moving regions by subtracting the current image pixel-by-pixel from a reference background image that is created by averaging images over time in an initialization period.

The basic idea of background subtraction method is to initialize a background firstly, and then by subtracting current frame in which the moving object present that current frame is subtracted with background frame to detect moving object.

This method is simple and easy to realize, and accurately extracts the characteristics of target data, but it is sensitive to the change of external environment, so it is applicable to the condition that the background is known.

3.1.1.1. Convectional Approaches

A robust background subtraction algorithm should be able to handle lighting changes, repetitive motions from clutter and long-term scene changes [15].

The following analyses make use of the function of V(x, y, t) as a video sequence where t is the time dimension, x and y are the pixel location variables. e.g. V(1,2,3) is the pixel intensity at (1,2) pixel location of the image at t = 3 in the video sequence.

3.1.1.2. Using Frame Differences

A motion detection algorithm begins with the segmentation part where foreground or moving objects are segmented from the background.

The simplest way to implement this is to take an image as background and take the frames obtained at the time t, denoted by I(t) to compare with the background image denoted by B.

Here using simple arithmetic calculations, we can segment out the objects simply by using image subtraction technique of computer vision meaning for each pixels in I(t), take the pixel value denoted by P[I(t)] and subtract it with the corresponding pixels at the same position on the background image denoted as P[B].

In mathematical equation, it is written as equation (1):

$$P[F(t)] = P[I(t)] - P[B] \quad (1)$$

The background is assumed to be the frame at time t. This difference image would only show some intensity for the pixel locations which have changed in the two frames. Though we have seemingly removed the background, this approach will only work for cases where all foreground pixels are moving and all background pixels are static [15][16]. A threshold "Threshold" is put on this difference image to improve the subtraction as in equation (2).

$$|P[F(t)] - P[F(t+1)]| > Threshold \quad (2)$$

This means that the difference image's pixels' intensities are 'thresholded ' or filtered on the basis of value of Threshold [17]. The accuracy of this approach is dependent on speed of movement in the scene. Faster movements may require higher thresholds.

3.1.1.3. Mean filter

For calculating the image containing only the background, a series of preceding images are averaged. For calculating the background image at the instant t is shown in equation (3),

$$B(x,y,t) = \frac{1}{N}\sum_{i=1}^{N} V(x,y,t-i) \quad (3)$$

Where, N is the number of preceding images taken for averaging. This averaging refers to averaging corresponding pixels in the given images.

Nwould depend on the video speed (number of images per second in the video) and the amount of movement in the video [17]. After calculating the background B(x,y,t) we can then subtract it from the image V(x,y, t) at time t=t and threshold it. Thus the foreground is as in equation (4)

$$|V(x,y,t) - B(x,y,t)| > Th \quad (4)$$

Where, Th is threshold. Similarly we can also use median instead of mean in the above calculation of B(x,y,t).

Usage of global and time-independent Thresholds (same Th value for all pixels in the image) may limit the accuracy of the above two approaches [15].

3.1.2 Gaussian Mixture Model

GMM is a combination of k Gaussian distribution which equates the distribution of pixel strength in the current frame. The probability of the intensity within the frame at time t can be modeled as in equation (5)

$$P(x_t) = \sum_{k=1}^{k} W_{k,t} x N(x_t, \mu_{k,t}, \sum_{k,t}) \quad (5)$$

Here k, t w,k, t μ and k, t_1 are the weight estimation, mean, and the covariance matrix of Gaussian Kth, the variance of the Gaussian Kth matrix is referred from standard deviation. ,k t k t $t_1 = \mu$ x

The intensity of the new pixel t x is evaluated regarding to each of the Gaussian element to the nearest distribution, the new parameters k ,t w , k ,t μ and k ,t_1 are updated. The set of the background pixels are identified adaptable. With this it is useful to employ GMM to remove all potential background pixels. GMM established on background subtraction is resilient against any change in illumination.

GMM is a density model that consists of several Gaussian component functions. This method can perform well when used for extraction process of background because its reliability against the changes in light and condition during repeated object detection.

Pixel in the video scene is modeled in Gaussian distribution. Each pixel in the frame was compared with model formed from GMM. Pixels with similarity values under the standard deviation and highest weight factor were considered as background, while pixels with higher standard deviation and lower weight factor considered as foreground. Pixel then categorized into one of GMM candidate model.

If the color of pixel is categorized as a background model then the pixel will be given zero(0) or black color.

While the pixel is uncategorized in background model then it will be considered as foreground and given one (1) or white color. Then, the resulting binary image will be processed further.

In the context of a traffic surveillance system, Friedman and Russel [19] proposed to model each background pixel using a mixture of three Gaussians corresponding to road, vehicle and shadows. This model is initialized using an EM algorithm [20]. Then, the Gaussians are manually labeled in a heuristic manner as follows:

The darkest component is labeled as shadow; in the remaining two components, the one with the largest variance is labeled as vehicle and the other one as road.

This remains fixed for all the process giving lack of adaptation to changes over time. For the foreground detection, each pixel is compared with each Gaussian and is classified according to it corresponding Gaussian. The maintenance is made using an incremental EM algorithm for real time consideration.

Stauffer and Grimson [18] generalized this idea by modeling the recent history of the color features of each pixel $X_1,...,X_t\{\ \}$ by a mixture of K Gaussians.

3.1.3 Morphological Filtering

Morphological filtering of a binary image is conducted by considering compound operations like opening and closing as filters. They may act as filters of shape.

For example, opening with a disc structuring element smoothes corners from the inside, and closing with a disc smoothes corners from the outside.

But also these operations can filter out from an image any details that are smaller in size than the structuring element. Figure 3.2 shows the filtering on Morphology.

Example: opening is filtering the binary image at a scale defined by the size of the structuring element.

Only those portions of the image that fit the structuring element are passed by the filter; smaller structures are blocked and excluded from the output image. The size of the structuring element is most important to eliminate noisy details but not to damage objects of interest.

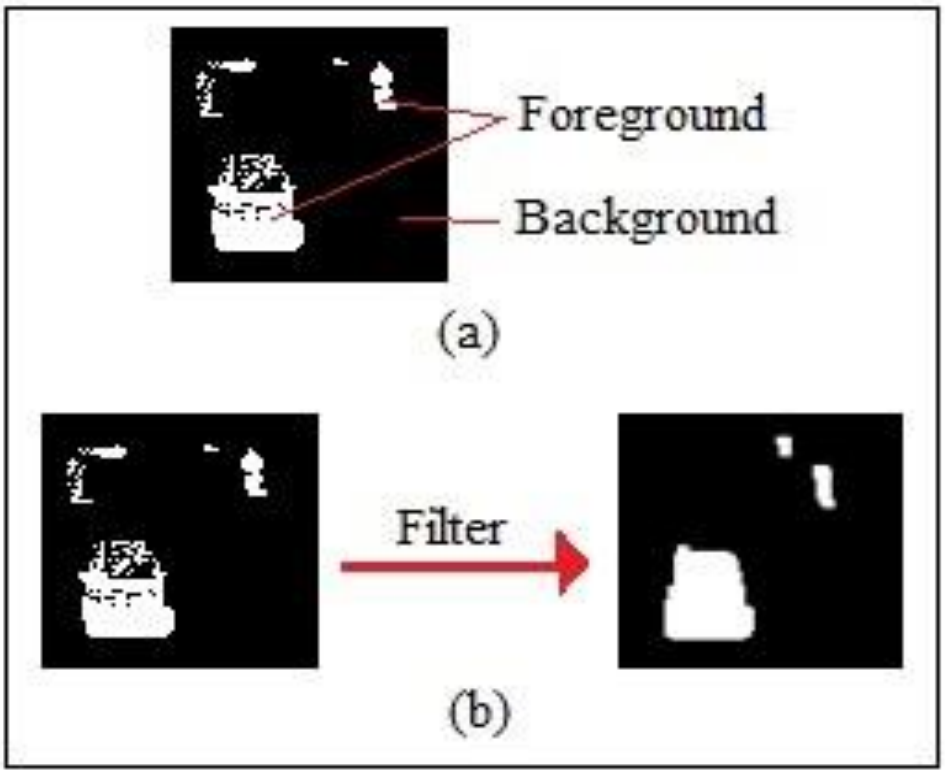

Figure 3.2. Filtering on morphology

3.1.3.1. Erosion and Dilation

Erosion: The erosion of a binary image f by a structuring element s (denoted f Θ s) produces a new binary image g = f Θ s with ones in all locations (x,y) of a structuring element's origin at which that structuring element s fits the input image f, i.e. g(x,y) = 1 is s fits f and 0 otherwise, repeating for all pixel coordinates (x,y).Figure 3.3 shows Erosion of a 3x3 square structuring element. Erosion with small (e.g. 2×2 - 5×5) square structuring elements shrinks an image by stripping away a layer of pixels from both the inner and outer boundaries of regions. The holes and gaps between different regions become larger, and small details are eliminated:

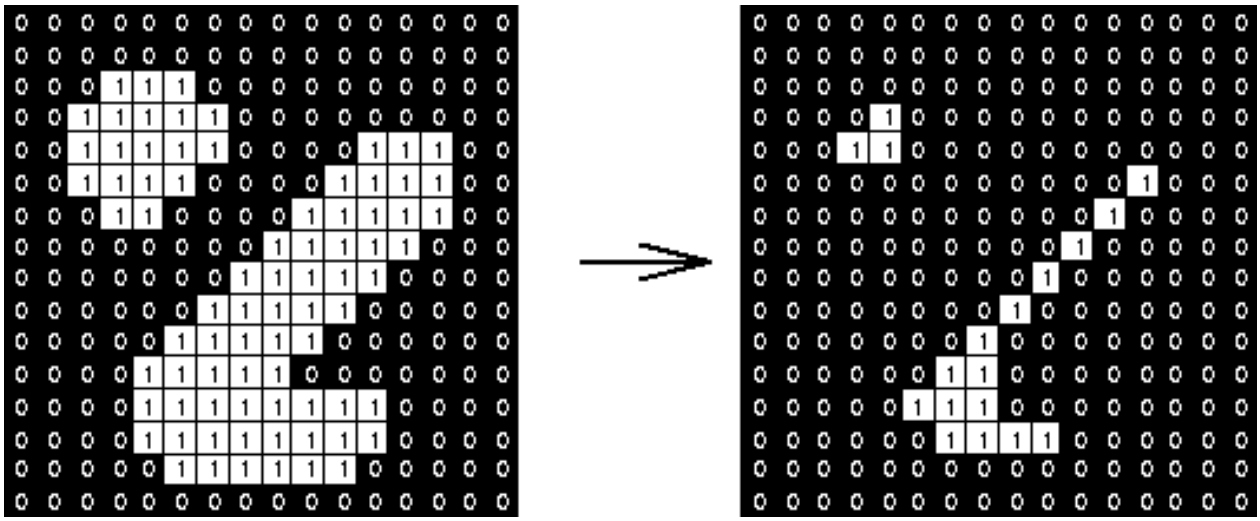

Figure 3.3. Erosion of a 3x3 square structuring element

Larger structuring elements have a more pronounced effect, the result of erosion with a large structuring element being similar to the result obtained by iterated erosion using a smaller structuring element of the same shape. If s_1 and s_2 are a pair of structuring elements identical in shape, with s_2 twice the size of s_1, then equation (6) is given as

$$f \ominus s_2 \approx (f \ominus s_1) \ominus s_1 \qquad (6)$$

Erosion removes small-scale details from a binary image but simultaneously reduces the size of regions of interest, too. By subtracting the eroded image from the original image, boundaries of each region can be found: $b = f - (f \ominus s)$ where f is an image of the regions, s is a 3×3 structuring element, and b is an image of the region boundaries.

Dilation: The dilation of an image f by a structuring element s (denoted f +s) produces a new binary image g = f + s with ones in all locations (x,y) of a structuring element's orogin at which that structuring element s hits the input image f, i.e. g(x,y) = 1 if s hits f and 0 otherwise, repeating for all pixel coordinates (x,y). Dilation has the opposite effect to erosion -- it adds a layer of pixels to both the inner and outer boundaries of regions. Figure 3.4 shows Dilation of 3x3 square structuring element

The holes enclosed by a single region and gaps between different regions become smaller, and small intrusions into boundaries of a region are filled in:

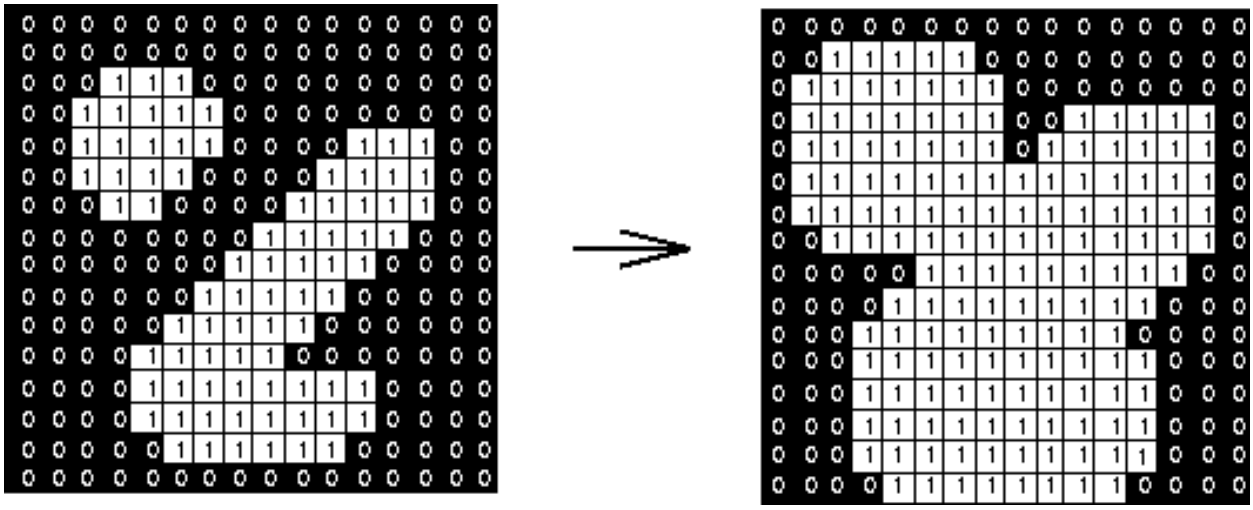

Figure 3.4. Dilation of 3x3 square structuring element

Results of dilation or erosion are influenced both by the size and shape of a structuring element. Dilation and erosion are *dual* operations in that they have opposite effects. Let f^c denote the complement of an image *f*, i.e., the image produced by replacing 1 with 0 and vice versa. Formally, the duality is written as in equation (7)

$$f + s = f^c \ominus s_{rot} \quad (7)$$

where s_{rot} is the structuring element *s* rotated by 180 . If a structuring element is symmetrical with respect to rotation, then s_{rot} does not differ from *s*.

If a binary image is considered to be a collection of connected regions of pixels set to 1 on a background of pixels set to 0, then erosion is the fitting of a structuring element to these regions and dilation is the fitting of a structuring element (rotated if necessary) into the background, followed by inversion of the result.

3.2 Blob Analysis

A fundamental image processing is called blob analysis. BLOB stands for Binary Large Object. It is the extraction of features from connected pixels that share the same logical state (blobs). The extraction of foreground object is adapted with blob area.

The object corresponding with blob area is detected as the animal object and marked by a bounding box. Using the vision blob analysis object, this found a bounding box connecting component corresponding to the moving animal.

Blob analysis mainly consists of three steps:

1. Acquire image(s)

   An image is acquired.
2. Segment image(s)

   Isolating the foreground pixels of interest from the image background using pre-processing tools and operations like thresholding and others. This is also called segmentation.
3. Extract feature

   Features like area (i.e., the number of pixels), center of gravity, or the orientation of a blob or blobs are calculated.

The purpose of BLOB extraction is to isolate the BLOBs(objects) in binary image. As mentioned above, a BLOB consists of a group of connected pixels. Whether or not two pixels are connected is defined by the connectivity, that is which pixels are neighbours and which are not. Feature extraction is a matter of converting each BLOB into a few representative numbers.

That is, keep the relevant information and ignore the rest. But before calculating any features we first want to exclude every BLOB which is connected to the border of the image. The reason is that we in general have no information about the part of the object outside the image. Bounding box of a BLOB is the minimum rectangle which contains the BLOB. Extracted image using bounding box is shown below in figure 3.5.

Figure 3.5. Image extracted using bounding box

3.3 Artificial Neural Network

Artificial neural networks (ANNs) or connectionist systems are a computational model used in machine learning, computer science and other research disciplines, which is based on a large collection of connected simple units called artificial neurons, loosely analogous to axons in a biological brain. Connections between neurons carry an activation signal of varying strength.

If the combined incoming signals are strong enough, the neuron becomes activated and the signal travels to other neurons connected to it. Such systems can be trained from examples, rather than explicitly programmed, and excel in are, as where the solution or feature detection is difficult to express in a traditional computer program.

Like other machine learning methods, neural networks have been used to solve a wide variety of tasks, like computer vision and speech recognition, that are difficult to solve using ordinary rule-based programming.

Typically, neurons are connected in layers, and signals travel from the first (input), to the last (output) layer. Modern neural network projects typically have a few thousand to a few million neural units and millions of connections; their computing power is similar to a worm brain, several orders of magnitude simpler than a human brain.

The signals and state of artificial neurons are real numbers, typically between 0 and 1. There may be a threshold function or limiting function on each connection and on the unit itself, such that the signal must surpass the limit before propagating. Back propagation is the use of forward stimulation to modify connection weights, and is sometimes done to train the network using known correct outputs.

However, the success is unpredictable: after training, some systems are good at solving problems while others are not. Training typically requires several thousand cycles of interaction. The goal of the neural network is to solve problems in the same way that a human would, although several neural network categories are more abstract. New brain research often stimulates new patterns in neural networks.

One new approach is use of connections which span further to connect processing layers rather than adjacent neurons. Other research being explored with the different types of signal over time that axons propagate, such as deep learning, interpolates greater complexity than a set of boolean variables being simply on or off. Newer types of network are free flowing in terms of stimulation and inhibition, with connections interacting in more chaotic and complex ways.

Dynamic neural networks are the most advanced, in that they dynamically can, based on rules, form new connections and even new neural units while disabling others.

Neural networks have seen an explosion of interest over the last few years and are being successfully applied across an extraordinary range of problem domains, in areas as diverse as finance, medicine, engineering, geology and physics. Neuron can have two states and that those states could be dependent on some threshold value.

There are two main reason for NN investigation, _rst is to try to get an understanding on how human brain function and second is desire to build machines that are capable for solving complex problems that sequentially operating computers were unable to solve.

If problem structure is well analyzed, traditional computers could still outperform NN but in cases where problem has not been analyzed in details, NN could be used to learn from large set of examples.

NN can handle errors better than traditional computers programs (imagine scenario where one faulty statement in program could halt everything while NN can handle errors in much better manner).

3.3.1. Neurons

Human brain has over 100 billion interconnected neurons. Most sophisticated application have only tiny fraction of that. It can only be imagined how powerful NN with this number of interconnected neurons would be. Neurons use this interconnected network to pass information's with each other using electric and chemical signals. Although it may seem that neurons are fully connected, two neurons actually do not touch each other.

They are separated by tiny gap call Synapse. Each neuron process information and then it can "connect" to as many as 50 000 other neurons to exchange information. If connection between two neuron is strong enough (will be explained later) information will be passed from one neuron to another.

On their own, each neuron is not very bright but put 100 billion of them together and let them talk to each other, then this system becomes very powerful. A typical neuron would have 4 components seen on Figure 3.6. Dendrites, Soma, Axon and Synapse. Dendrites gather inputs from other neurons and when a certain threshold is reached they generate a non-linear response (signal to other neurons via the Axon).

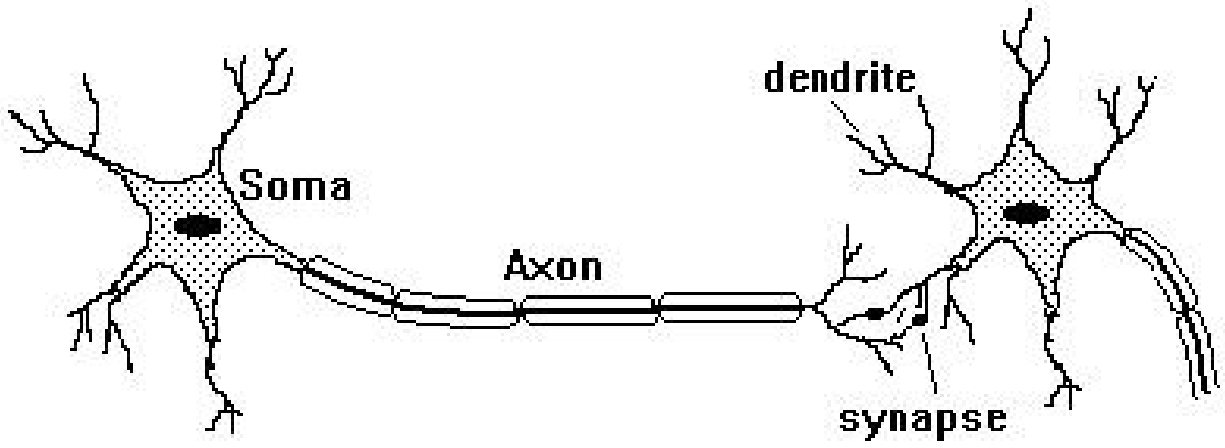

Figure 3.6Neurons

3.3.2. Back Propagation Neural Network

An Artificial Neural Network is an interconnected group of nodes, akin to the vast network of neurons in a brain. In Figure 3.7, each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

The backward propagation of errors, or back propagation, is a common method of training artificial neural networks and used in conjunction with an optimization method such as gradient descent. The algorithm repeats a two phase cycle, propagation and weight update. When an input vector is presented to the network, it is propagated forward through the network, layer by layer, until it reaches the output layer.

The output of the network is then compared to the desired output, using a loss function, and an error value is calculated for each of the neurons in the output layer. The error values are then propagated backwards, starting from the output, until each neuron has an associated error value which roughly represents its contribution to the original output.

Back propagation uses these error values to calculate the gradient of the loss function with respect to the weights in the network. In the second phase, this gradient is fed to the optimization method, which in turn uses it to update the weights, in an attempt to minimize the loss function.

The importance of this process is that, as the network is trained, the neurons in the intermediate layers organize themselves in such a way that the different neurons learn to recognize different characteristics of the total input space. After training, when an arbitrary input pattern is present which contains noise or is incomplete, neurons in the hidden layer of the network will respond with an active output if the new input contains a pattern that resembles a feature that the individual neurons have learned to recognize during their training.

Back propagation requires a known, desired output for each input value in order to calculate the loss function gradient – it is therefore usually considered to be a supervised learning method; nonetheless, it is also used in some unsupervised networks such as auto encoders. It is a generalization of the delta rule to multi-layered feed forward networks, made possible by using the chain rule to iteratively compute gradients for each layer.

Back propagation requires that the activation function used by the artificial neurons (or "nodes") be differentiable. Neural network is made up from an input, output and one or more hidden layers. Each node from input layer is connected to a node from hidden layer and every node from hidden layer is connected to a node in output layer. There is usually some weight associated with every connection.

Input layer represents an the raw information that is fed into the network. This part of network is never changing its values. Every single input to the network is duplicated and send down to the nodes in hidden layer. Hidden Layer accepts data from the input layer. It uses input values and modifies them using some weight value, this new value is than send to the output layer but it will also be modified by some weight from connection between hidden and output layer. Output layer process information received from the hidden layer and produces an output. This output is than processed by activation function.

NN is interconnected network that resembles human brain. The most important characteristic of NN is its ability to learn. When presented with training set (form of supervised learning) where input and output values are known, NN model could be created to help with classifying new data. Results that are achieved by using NN are encouraging, especially in some fields like pattern recognition. NN is getting more and more attention in last two decades. BP algorithm is most popular algorithm used in NN. It is one of the main reasons why NN are becoming so popular.

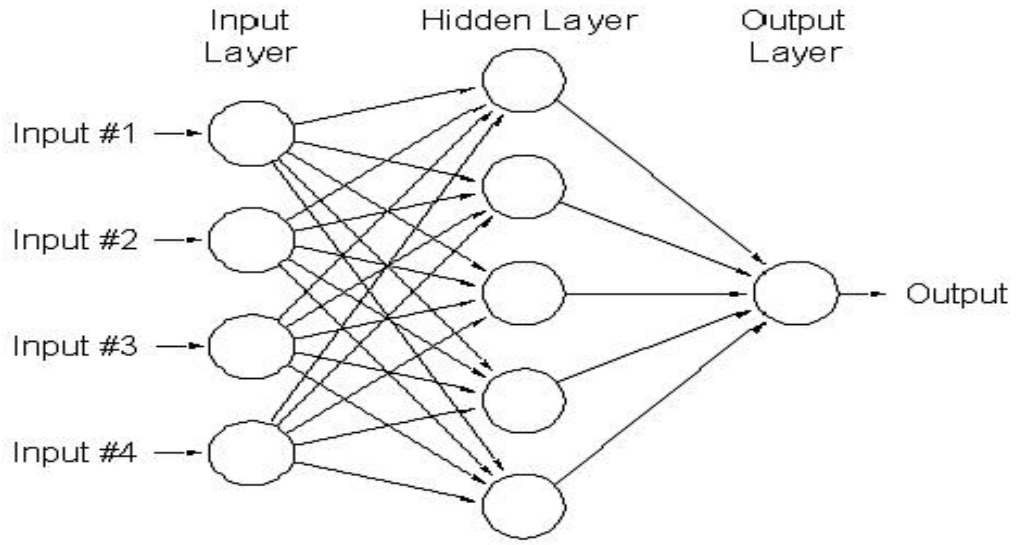

Figure 3.7. Simple Neural Network

3.3.2.1. Number of Nodes and Layers

Choosing number of nodes for each layer will depend on problem NN is trying to solve, types of data network is dealing with, quality of data and some other parameters. Number of input and output nodes depends on training set in hand. If there are too many nodes in hidden layer, number of possible computations that algorithm has to deal with increases.

Picking just few nodes in hidden layer can prevent the algorithm of its learning ability. Right balance needs to be picked. It is very important to monitor the progress of NN during its training, if results are not improving, some modification to the model might be needed.

3.3.2.2. Setting Weights

The way to control NN is by setting and adjusting weights between nodes. Initial weights are usually set at some random numbers and than they are adjusted during NN training. Some NN are dealing with thousands, even millions of nodes so changing one or two at time would not help in adjusting NN to get desired results in timely manner. Logic behind weight updates is quite simple. During the NN training weights are updated after iterations.

If results of NN after weights updates are better than previous set of weights, the new values of weights are kept and iteration goes on. Finding combination of weights that will help us minimize error should be main aim when setting weights. This will become bit more clear once the learning rate; momentum and training set are explained.

3.3.2.3. Running and Training Neural Network

Running the network consist of a forward pass and a backward pass. In the forward pass outputs are calculated and compared with desired outputs. Error from desired and actual output are calculated. In the backward pass this error is used to alter the weights in the network in order to reduce the size of the error. Forward and backward pass are repeated until the error is low enough (users usually set the value of accepted error).Training NN could be separate topic but for the purpose of this paper, training will be explained briefly. When training NN, we are feeding network with set of examples that have inputs and desired outputs.

If we have some set of 1000 samples, we could use 100 of them to train the network and 900 to test our model. Choosing the learning rate and momentum will help with weight adjustment. Setting right learning rate could be difficult task, if learning rate is too small, algorithm might take long time to converges. On the other hand, choosing large learning rate could have opposite effect, algorithm could diverge.

Sometimes in NN every weight has it's own learning rate. Learning rate of0.35 proved to be popular choice when training NN. This paper will use rate of 0.45 but this value is used because of simple architecture of NN used in example. Large values of momentum term will influence the adjustment in the current weight to move in same direction as previous adjustment.

Artificial intelligence and NN have been used more and more in recent decades. Potentials in this area are huge. Here are some NN advantages, disadvantages and industries where they are being used. NN are used in cases where rules or criteria for searching an answer is not clear (that is why NN are often called black box, they can solve the problem but at times it is hard to explain how problem was solved).

They found its way into broad spectrum of industries, from medicine to marketing and military just to name few. Financial sector has been known for using NN in classifying credit rating and market forecasts.

Marketing is another field where NN has been used for customer classification (groups that will buy some product, identifying new markets for certain products, relationships between customer and company). Many companies use direct marketing (sending its officer by mails) to attract customers.

If NN could be employed to up the percentage of the response to direct marketing, it could save companies lot's of their revenue. At the end of the day, it's all about the money. Post offices are known to use NN for sorting the post (based on postal code recognition).

Those were just few examples of where NN are being used. NN advantages are that they can adapt to new scenarios, they are fault tolerant and can deal with noisy data. Time to train NN is probably identified as biggest disadvantage. They also require very large sample sets to train model efficiently. It is hard to explain results and what is going on inside NN.

3.3.2.4. Loss Function

Loss function is also referred as the cost function or error function. The loss function is a function that maps values of one or more variables onto a real number intuitively representing some "cost" associated with the event.

For back propagation, the loss function calculates the difference between the input training example and its expected output, after the example has been propagated through the network.

3.3.2.5. Assumption about the Loss Function

For back propagation to work, two assumptions are made about the form of the error function. The first is that it can be written as an average $E = \frac{1}{N}\sum_X E_{X,}$over error functions E_X, for individual training examples, x. The reason for this assumption is that the back propagation algorithm calculates the gradient of the error function for a single training example, which needs to be generalized to the overall error function.

In practice, training examples are placed in batches, and the error is averaged at the end of the batch, which is then used to update the weights. The second assumption is that it can be written as a function of the outputs from the neural network.

3.3.3. Back Propagation Algorithm

One of the most popular NN algorithms is back propagation algorithm. BP algorithm could be broken down to four main steps. After choosing the weights of the network randomly, the back propagation algorithm is used to compute the necessary corrections.

The algorithm can be decomposed in the following four steps:

i) Feed-forward computation
ii) Back propagation to the output layer
iii) Back propagation to the hidden layer
iv) Weight updates

The algorithm is stopped when the value of the error function has become sufficiently small. This is very rough and basic formula for BP algorithm. The last step, weight updates is happening throughout the algorithm.

The back propagation learning algorithm can be divided into two phases:

1. Propagation
2. Weight update

Phase 1: Propagation

Each propagation involves the following steps:

1. Forward propagation of a training pattern's input through the neural network in order to generate the network's output value(s).
2. Backward propagation of the propagation's output activations through the neural network using the training pattern target in order to generate the deltas (the difference between the targeted and actual output values) of all output and hidden neurons.

Phase 2: Weight update

For each weight, the following steps must be followed:

1. The weight's output delta and input activation are multiplied to find the gradient of the weight.
2. A ratio (percentage) of the weight's gradient is subtracted from the weight.

This ratio (percentage) influences the speed and quality of learning; it is called the learning rate. The greater the ratio, the faster the neuron trains, but the lower the ratio, the more accurate the training is.

The sign of the gradient of a weight indicates whether the error varies directly with, or inversely to, the weight. Therefore, the weight must be updated in the opposite direction, "descending" the gradient.

Propagation and weight update phases are repeated until the performance of the network is satisfactory. Initially, before training, the weights will be set randomly.

Then the neuron learns from training examples, which in this case consists of a set of tuples (x1, x2, t) where x1 and x2 are the inputs to the network and t is the correct output (the output the network should eventually produce given the identical inputs).

The network given x1 and x2 will compute an output y which very likely differs from t (since the weights are initially random). A common method for measuring the discrepancy between the expected output t and the actual output y is using the squared error measure as in equation (8):

$$E = (t - y)^2 \tag{8}$$

Where, E is the discrepancy or error.

As an example, consider the network on a single training case: (1,1,0), thus the input x1 and x2 are 1 and 1 respectively and the correct output, t is 0. Now if the actual output y is plotted on the x-axis against the error E on the y-axis, the result is a parabola.

The minimum of the parabola corresponds to the output y which minimizes the error E . For a single training case, the minimum also touches the -axis, which means the error will be zero and the network can produce an output y that exactly matches the expected output t.

Therefore, the problem of mapping inputs to outputs can be reduced to an optimization problem of finding a function that will produce the minimal error.

However, the output of a neuron depends on the weighted sum of all its inputs as in equation (9):

$$y = x_1 w_1 + x_2 w_2 \tag{9}$$

where w1 and w2 are the weights on the connection from the input units to the output unit. Therefore, the error also depends on the incoming weights to the neuron, which is ultimately what needs to be changed in the network to enable learning.

Figure 3.8illustrates a simple neural network with two input unit and one output unit.

If each weight is plotted on a separate horizontal axis and the error on the vertical axis, the result is a parabolic bowl. For a neuron with k weights, the same plot would require an elliptic paraboloid of k+1 dimensions. Figure 3.9 below shows Error surface of a linear neuron for a single training case.

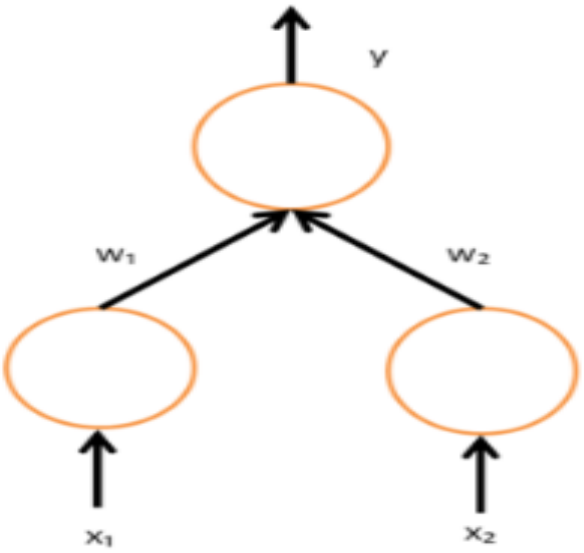

Figure 3.8. A simple neural network with two input unit and one output unit.

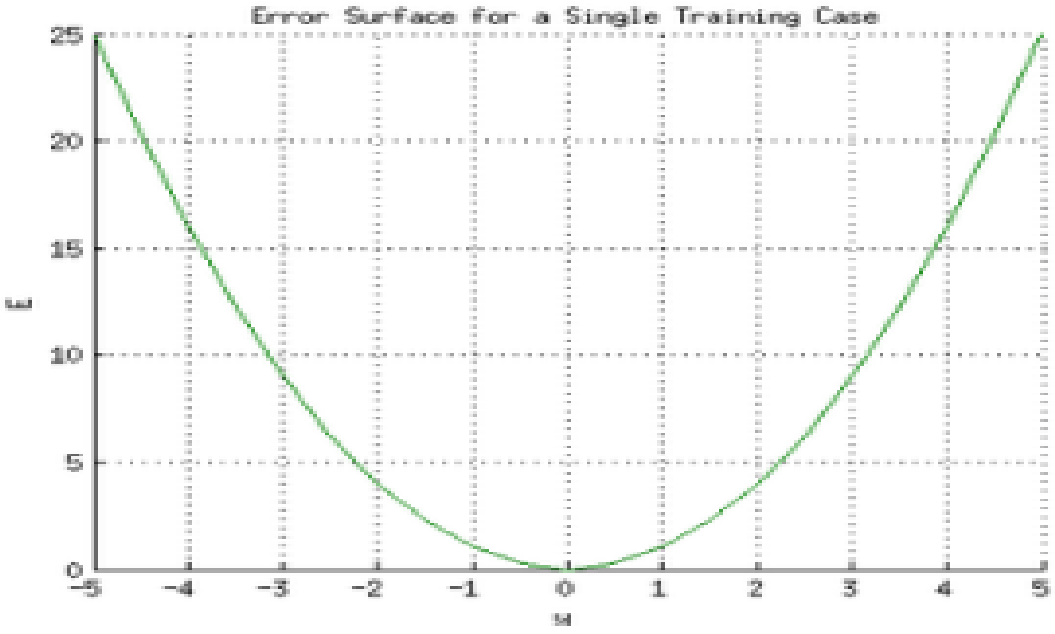

Figure 3.9. Error surface of a linear neuron for a single training case.

The back propagation algorithm aims to find the set of weights that minimizes the error. There are several methods for finding the minima of a parabola or any function in any dimension.

One way is analytically by solving systems of equations, however this relies on the network being a linear system, and the goal is to be able to also train multi-layer, non-linear networks (since a multi-layered linear network is equivalent to a single-layer network). The method used in back propagation is gradient descent. Figure 3.10 below illustrates a Error surface of a linear neuron with two input weights.

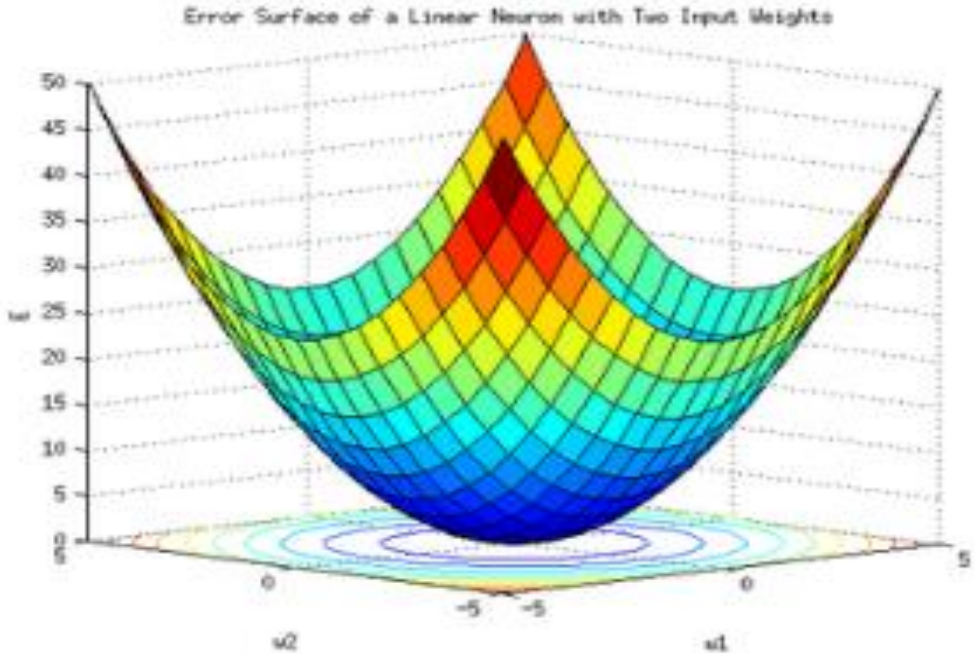

Figure 3.10. Error surface of a linear neuron with two input weights.

3.3.3.1. An Analogy for Understanding Gradient Descent

The basic intuition behind gradient descent can be illustrated by a hypothetical scenario. A person is stuck in the mountains and is trying to get down (i.e. trying to find the minima). There is heavy fog such that visibility is extremely low. Therefore, the path down the mountain is not visible, so he must use local information to find the minima.

He can use the method of gradient descent, which involves looking at the steepness of the hill at his current position, then proceeding in the direction with the steepest descent (i.e. downhill).

If he was trying to find the top of the mountain (i.e. the maxima), then he would proceed in the direction steepest ascent (i.e. uphill). Using this method, he would eventually find his way down the mountain.

However, assume also that the steepness of the hill is not immediately obvious with simple observation, but rather it requires a sophisticated instrument to measure, which the person happens to have at the moment.

It takes quite some time to measure the steepness of the hill with the instrument, thus he should minimize his use of the instrument if he wanted to get down the mountain before sunset.

The difficulty then is choosing the frequency at which he should measure the steepness of the hill so not to go off track.

In this analogy, the person represents the back propagation algorithm, and the path taken down the mountain represents the sequence of parameter settings that the algorithm will explore. The steepness of the hill represents the slope of the error surface at that point.

The instrument used to measure steepness is differentiation (the slope of the error surface can be calculated by taking the derivative of the squared error function at that point). The direction he chooses to travel in aligns with the gradient of the error surface at that point. The amount of time he travels before taking another measurement is the learning rate of the algorithm. See the limitation section for a discussion of the limitations of this type of "hill climbing" algorithm.

3.3.3.2. Back Propagation with Adaptive Learning Rate

In order to avoid oscillation inside the network, such as alternating connection weights, and to improve the rate of convergence, there are refinements of this algorithm that use an adaptive learning rate.

3.3.3.3. Modes of Learning

There are two modes of learning to choose from: stochastic and batch. In stochastic learning, each propagation is followed immediately by a weight update. In batch learning many propagations occur before updating the weights, accumulating errors over the samples within a batch. Stochastic learning introduces "noise" into the gradient descent process, using the local gradient calculated from one data point; this reduces the chance of the network getting stuck in a local minima.

Yet batch learning typically yields a faster, more stable descent to a local minima, since each update is performed in the direction of the average error of the batch samples. In modern applications a common compromise choice is to use "mini-batches", meaning batch learning but with a batch of small size and with stochastically selected samples.

3.3.3.4. Training Data Collection

Online learning is used for dynamic environments that provide a continuous stream of new training data patterns. Offline learning makes use of a training set of static patterns.

3.4 Implementation

Implementation of this project involved two phases:

A. Training phase.
B. Testing phase.

A. Training phase:

INPUT: A set of randomly selected training sample images from Caltec-101.

OUTPUT: A knowledge database(KDB).

METHOD:

- Obtain training samples.
- Feed them to back propagation neural network.
- Obtain the error value.
- Back propagate the error value through hidden layers.
- Repeat the back propagation process for certain iterations(count).
- Repeat the training process for all the images in the dataset.

B. Testing phase:

INPUT: Cropped image from the input video.

OUTPUT: Recognition of the input image.

METHOD:

- The movement of the animal is detected.
- The bounding box is applied and the corresponding image is cropped.
- The cropped image is the input to the testing phase.
- The image is then recognized as elephant, bear or leopard.

Flowchart of training process algorithm is as shown below in Figure 3.11

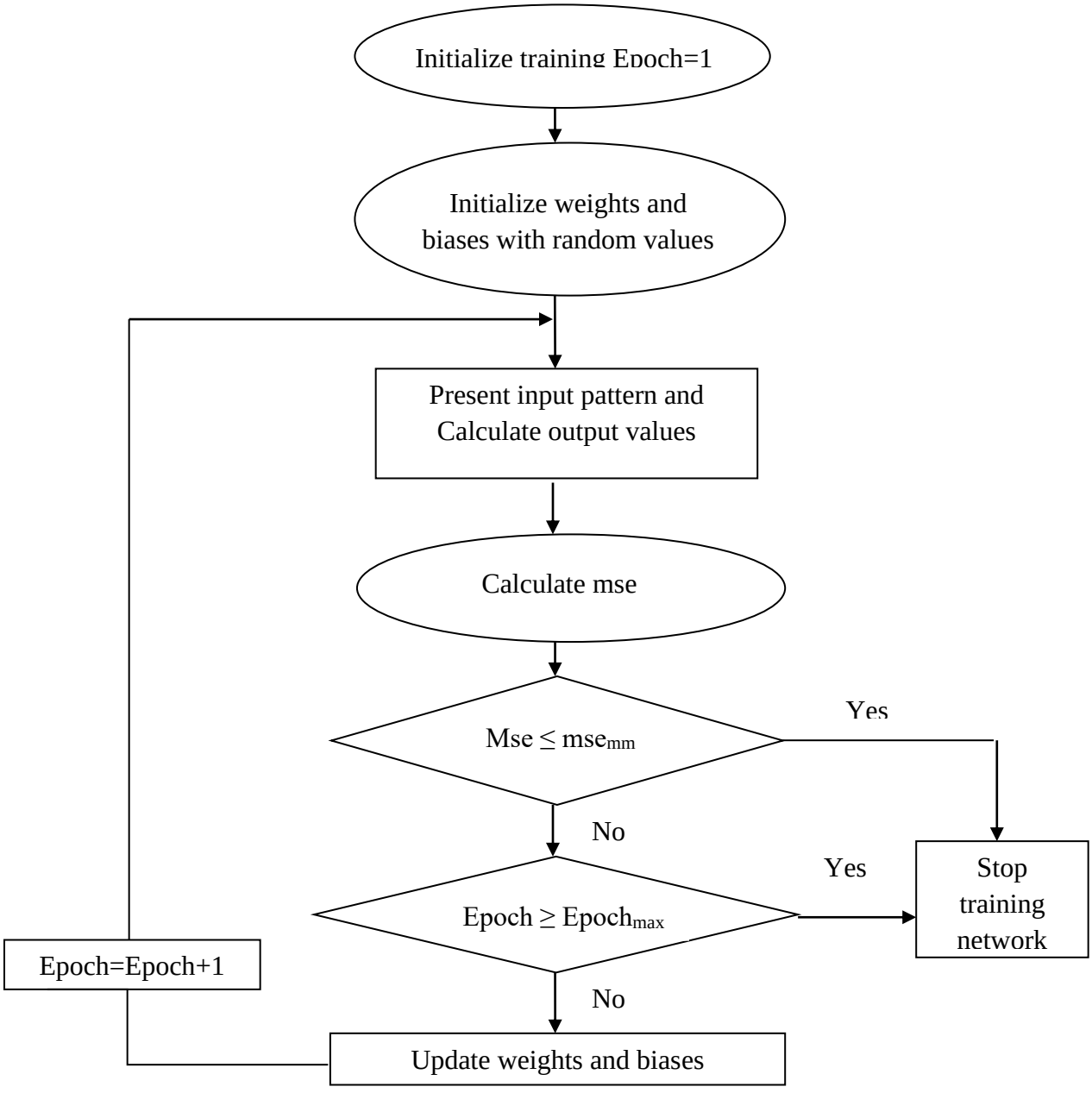

Figure 3.11. Flowchart of training process algorithm

Flow chart of proposed method is as shown below in Figure 3.1:

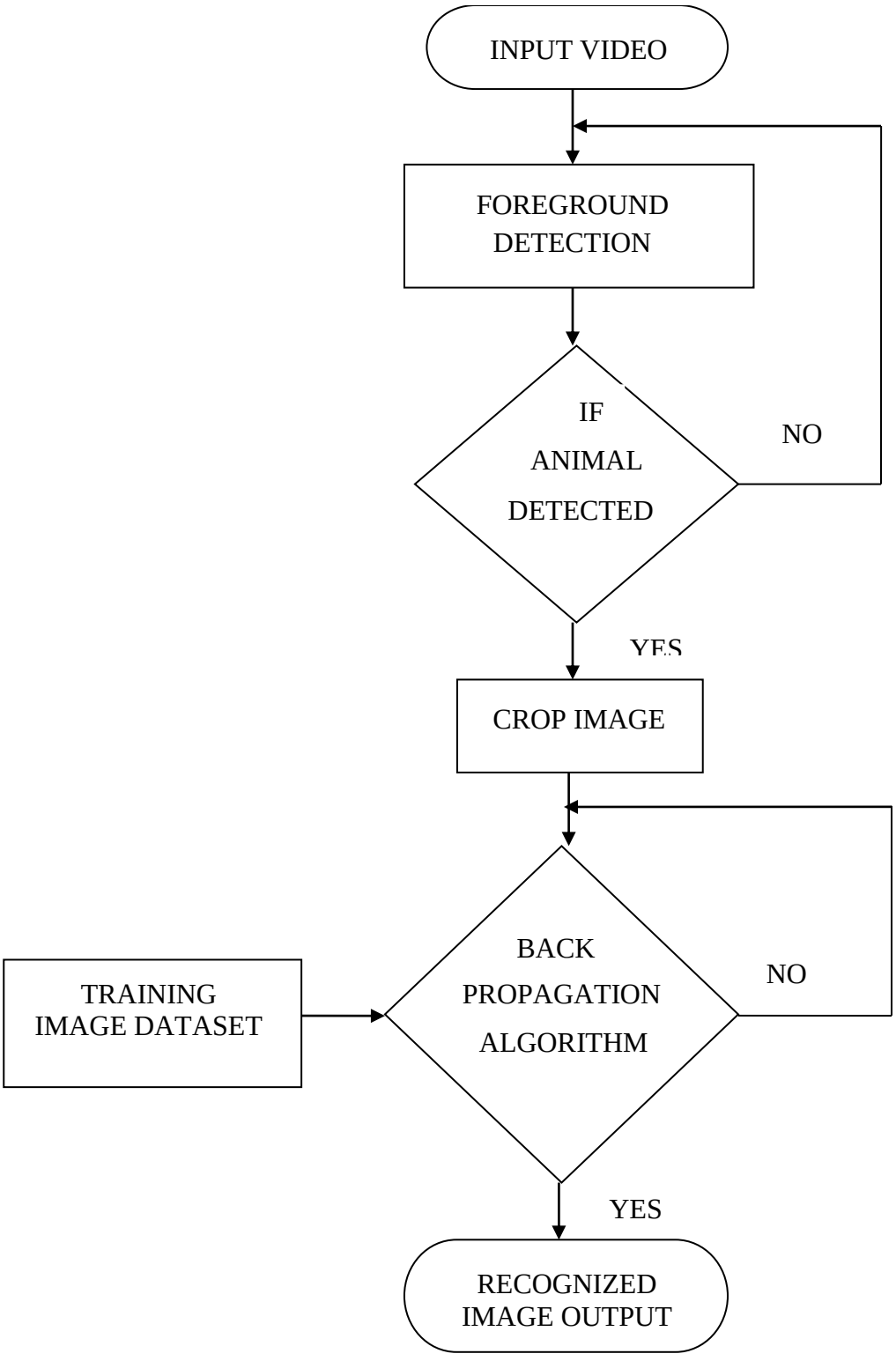

Figure 3.12. Flow chart of proposed method

The input video from the stationary cameras around the forest is the input for the algorithm. The motion in the video is detected using foreground detection. If the animal is detected then the bounding box is applied to the coresponding frame and image is croped. If the animal is not detected then next frame is fed as input to foreground detection. The back propagation algorithm is applied to croped image. The training dataset and the cropped image which is the test image is compared by the backpropagation algorithm to recognize the animal.

3.5 Database

We apply our algorithm to popularly and widely used standard datasets that are described below.

3.5.1 CALTEC-101

Caltech-101 dataset consists of 101 object categories which contains a total of 9144 images with 31 to 800 images per category.

It includes images like animals, butterfly, chandelier, gar field, cars, flowers, human face, etc which are mainly subjected to high intensity variations, occlusions and affected by corner artifacts. We have taken the pictures of elephant. Cheetah and bear.

Figure.3.13 illustrates few classes of Caltech-101 that we have used.

Most Computer Vision and Machine Learning algorithms function by training on example inputs. They require a large and varied set of training data to work effectively. For example, the real-time face detection method used by Paul Viola and Michael J. Jones was trained on 4,916 hand-labeled faces.

Cropping, re-sizing and hand-marking points of interest is tedious and time-consuming. Historically, most data sets used in computer vision research have been tailored to the specific needs of the project being worked on. A large problem in comparing computer vision techniques is the fact that most groups use their own data sets.

Each set may have different properties that make reported results from different methods harder to compare directly.

(a)

(b)

(c)

Figure 3.13.(a) Class 1: Bear, (b) Class 2: Elephant, (c) Class 3: Cheetah.

For example, differences in image size, image quality, relative location of objects within the images and level of occlusion and clutter present can lead to varying results.

The Caltech 101 data set aims at alleviating many of these common problems.

• The images are cropped and re-sized.

• Many categories are represented, which suits both single and multiple class recognition algorithms.

• Detailed object outlines are marked.

Available for general use, Caltech 101 acts as a common standard by which to compare different algorithms without bias due to different data sets.

However, a recent study demonstrates that tests based on uncontrolled natural images (like the Caltech 101 data set) can be seriously misleading, potentially guiding progress in the wrong direction.

Caltech 101 is a data set of digital images created in September, 2003, compiled by Fei-FeiLi, Andreetto, Marc 'Aurelio Ranzato and Pietro Perona at the California Institute of Technology. It is intended to facilitate Computer Vision research and techniques. It is most applicable to techniques interested in recognition, classification, and categorization.

Caltech101 contains a total of 9146 images, split between 101 distinct object(including faces, watches, ants, pianos, etc.) and a background category (for a total of 102categories).Provided with the images are a set of annotations describing the outlines of each image, along with a MATLAB script for viewing.

The Caltech 101 data set aims to alleviate many of these common problems.

• The work of collecting a large set of images, and cropping and resizing them appropriately has been taken care of.
• A large number of different categories are represented, which benefits both single, and multi class recognition algorithms.
• Detailed object outlines have been marked for each image.
• By being released for general use, the Caltech 101 acts as a common standard by which to compare different algorithms without bias due to different data sets.

3.5.1.1. Advantages

Caltech 101 has several advantages over other similar datasets:

• Uniform size and presentation.

Almost all the images within each category are uniform in image size and in the relative position of interest objects.

This means that, in general, users who wish to use the Caltech 101 dataset do not need to spend an extra time cropping and scaling the images before they can be used.

• Low level of clutter/occlusion:

Algorithms concerned with recognition usually function by storing features unique to the object that is to be recognized. However, the majority of images taken have varying degrees of background clutter. Algorithms trained on cluttered images can potentially build incorrect.

• Detailed Annotations:

The detailed annotation of object outlines is another advantage to using the dataset.

3.5.1.2. Weakness

There are several weaknesses to the Caltech 101 dataset. Some of them are conscious trade- offs for the advantages it provides, and some are simply limitations of the dataset itself. Papers who rely solely on Caltech 101 to prove their point are nowadays frequently rejected. The weaknesses are:

• The dataset is too clean:

Images are very uniform in presentation, left right aligned, and usually not occluded. As a result, the images are not always representative of practical inputs that the algorithm being trained might be expected to see. Under practical conditions, there is usually more clutter, occlusion, and variance in relative position and orientation of interest objects. In fact, by taking the average over a category one can often clearly recognize the concept, which is unrealistic.

• Limited number of categories:

The Caltech 101 dataset represents only a small fraction of the possible object categories.

• Some categories contain few images:

Certain categories are not represented as well as others, containing as few as 31 images. This means that $N_{train} \leq 30$. The number of images used for training must be less than or equal to 30, which is not sufficient for all purposes.

• Aliasing and artifacts due to manipulation:

Some images have been rotated and scaled from their original orientation, and suffer from some amount of artifacts or aliasing.

3.6 MATLAB

MATLAB is a language of technical computing where millions of engineers and scientists worldwide use this software to analyze and design the systems and products transforming our world. MATLAB is in automobile active safety systems, interplanetary spacecraft, health monitoring devices, smart power grids, and LTE cellular networks.

It is used for machine learning, signal processing, image processing, computer vision, communications, computational finance, control design, robotics and much more.

The MATLAB platform is optimized for solving engineering and scientific problems. The matrix-based MATLAB language is the world's most natural way to express computational mathematics. Built-in graphics make it easy to visualize and gain insights from data. A vast library of prebuilt toolboxes lets you get started right away with algorithms essential to your domain.

The desktop environment invites experimentation, exploration, and discovery. These MATLAB tools and capabilities are all rigorously tested and designed to work together. MATLAB code can be integrated with other languages, enabling you to deploy algorithms and applications with web, enterprise, and production systems.

MATLAB supports developing applications with graphical user interface (GUI) features.

MATLAB includes GUIDE (GUI developing environment) for graphically designing GUIs. It also has tightly integrated graph plotting features.

3.6.1. Advantages

- A very large and growing database of built-in algorithms for image processing and computer vision applications.
- MATLAB allows you to test algorithms immediately without recompilation.
- You can type something at the command line or execute a section in the editor and immediately see the results, greatly facilitating algorithm development.
- The ability to call external libraries, such as OpenCV.

- The MATLAB desktop environment, which allows you to work interactively with your data, helps you to keep tract of files and variables, and simplifies common programming/debugging tasks.
- The ability to read in a wide variety of both common and domain-specific image formats.
- Clearly written documentation with many examples, as well as online resources such as web seminars("webinars").
- Bi-annual updates with new algorithms, features and performance enhancements.
- If you are already using MATLAB for each purposes, such as simulation, optimation, statistics, or data analysis, then there is a very quick learning curve for using it in image processing.
- The ability to process both still images and video.
- Technical support from a well-staffed, professional organization(assuming your maintenance is up-to-date).
- A large user community with lots of free code and knowledge sharing.
- The ability to auto-generate C code, using MATLAB coder, for a large and growing subset of image processing and mathematical functions, which you could then use in other environments, such as embedded systems or as a component in other software.

3.7 Recognition

Foreground extracted test image and training dataset are taken for comparison.

Training data set images and test image is shown in figure 3.14 and figure 3.15 respectively.

Figure 3.14 Trained image

Figure 3.15 Test image

After comparing, the desired animal is recognized and is as shown in figure 3.16

Figure 3.16 Recognized result

Chapter 4

RESULTS AND DISCUSSION

In this section, we show experimental results of the proposed animal detection and recognition method. The proposed algorithm was implemented in MATLAB 2013a, and displays the recognized animal name in the command window.

Figure 4.1. Shows the pre-processing of input video. The input video is captured from a stationary camera which is in the form of .mpeg4 with the frame rate of 25frames per second.

Figure 4.1 Input video

The input video is now set for foreground extraction. If the binary pixel value is '0' then it is a background image which appears in black colour and if the binary pixel value is '1' then it is a foreground image which appears white as shown in Figure 4.2.

Figure 4.2 Foreground Extraction

In a video static objects are considered as background and moving objects are considered as foreground. Blobs are applied to extract this moving objects and the extracted image is considered as a test image for comparison. Blob Extracted image is as shown below in figure 4.3.

Figure 4.3 Blob Extraction

Various recognition rates are obtained. Performance analysis of proposed method is as shown below in the table 4.1 and graphical representation of comparisions of recognition rate is shown below in figure 4.4

Input Video	Total Number of Frames	Frames in which Animals are Detected	Correct Recognition of Animal	Recognized Result
Cheetah	98	44	32	72.72%
Bear	81	22	16	72.72%
Elephant	112	51	39	76.47%

Table 4.1 Performance analysis of proposed method

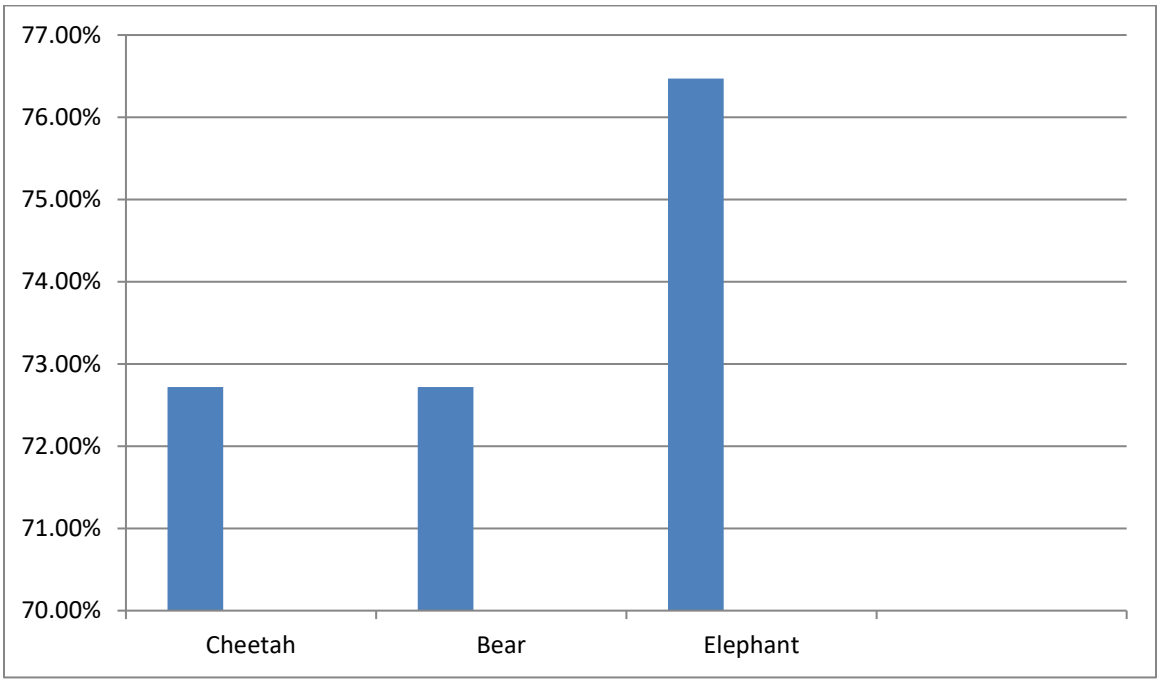

Figure 4.4Graphical representation of comparision of recognition rate

Chapter 5

CONCLUSION

Wildlife monitoring is important for effective protection of wildlife and human life, safeguard human habitat, sustainable use, and scientific management of wildlife resources. Compared to the traditional wildlife monitoring methods, wildlife monitoring based on artificial neural network image processing.

This project proposed wildlife detection and monitoring system based on artificial neural network image processing to overcome the shortcomings of traditional monitoring methods.

For this project, three classes of animals are used for training. They are Cheetah, Bear and Elephant. The total number of frames given as input to foreground detection is 98, 81 and 112 for Cheetah, Bear and Elephant respectively. The frames in which animal is detected are 44, 22 and 51 respectively.

The correct recognition of the respective animals are 32, 16 and 39. Thus, we get a recognition rate of 72.72% for Cheetah, 72.72% for Bear and 76.47% for Elephant.

From all these results we conclude that, background subtraction method gives us better and accurate result for moving animal detection. The proposed method uses Back subtraction based Gaussian mixture model because it is well suited for various lighting intensities and also in deletion of repeated frames.

Back propagation ANN is used here because it is computationally less complex and is very effective. Performance analysis shows the recognition result of the proposed method.

REFERENCES

[1] "Implementation of Back Propagation Algorithm: A Neural Network Approach for Pattern Recognition", Taranjit Kaur, International Journal of Engineering Research and Development, June 2012.

[2] Motion Detection & Tracking of a Leopard in a Video", Nitesh Sanklecha, Dr. Sudarshan Patil Kulkarni, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering ,August 2015.

[3] "Optical Flow Measurement using Lucas Kanade Method", Dhara Patel & Saurabh Upadhyay, International Journal of Computer Applications January 2013.

[4] "Moving Object Detection with Moving Background using Optic Flow ", Milin P. Patel &Shankar K. Parmar, IEEE International Conference on Recent Advances and Innovations In Engineering (ICRAIE-2014), May 09-11,2 014.

[5] "Moving Object Detection & Tracking Using Hybrid Approach in Real Time to Improve Accuracy", Dhara Trambadia, Chintan Varnagar, Prajesh Kathiriya, International Journal of Innovative Research in Computer and Communication Engineering ,April 2015.

[6] "Motion Detection Using Lucas Kanade Algorithm and Application Enhancement", Lee Yee Siong, Siti Salasiah Mokri, Aini Hussain, Norazlin Ibrahim, Mohd Marzuki Mustafa, International Conference on Electrical Engineering and Informatics, August 2009.

[7] "Image Recognition and Processing Using Artificial Neural Network", Md. Iqbal Quraishi, J Pal Choudhury & Mallika De, Recent Advances in Information Technology, 2012.

[8] "Artificial Neural Image Processing Applications: A Survey", Juan A. Ramírez-Quintana, Mario I. Chacon-Murguia and Jose F. Chacon-Hinojos.

[9] "Image Enhancement using Artificial Neural Network and Fuzzy Logic", Shweta Narnaware & Roshni Khedgaonkar.

[10] "Region Filtering and Optical Flow based Video Surveillance System", Dolley Shukla, Surabhi Biswas International Journal of Computer Applications, February 2013.

[11] "Robust Optical Flow Computation", B.H Alireza Bab-Hadiashar and S. David.

[12] "DetectionofMovingObjectsusingForegroundDetectorandimprovedMorphologicalfilter", Adedeji Olugboja & Zenghui Wang, 2016 3rdInternationalConferenceon Information Science and Control Engineering.

[13] "Identifying Moving Objects in a Video using Modified Background Subtraction and Optical Flow Method", Sumati Manchanda & Shanu Sharma, 2016 International Conference on Computing for Sustainable Global Development (INDIA Com).

[14] M. Piccardi (October 2004). Background subtraction techniques: a review (PDF). IEEE International Conference on Systems, Man and Cybernetics. 4. pp. 3099 3104.doi:10.1109/icsmc.2004.1400815.

[15] B. Tamersoy (September 29, 2009). "Background Subtraction – Lecture Notes" (PDF). University of Texas at Austin.

[16] B. Patel; N. Patel (March 2012). Motion Detection based on multi-frame video under surveillance systems. Vol. 12.

[17] N. Lu; J. Wang; Q. Wu; L. Yang (February 2012). An improved Motion Detection methodfor real time Surveillance.

[18] Stauffer C, Grimson W. Adaptive background mixture models for real-time tracking. Proc IEEE Conf on Comp Vision and Patt Recog(CVPR 1999) 1999; 246-252.

[19] Friedman N, Russell S. Image Segmentation in Video Sequences: A Probabilistic Approach. Proceedings Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI 1997), 1997; 175-181.

[20] Dempster A, Laird N, Rubin D. Maximum likelihood from incomplete data via the EM algorithm. J Royal Statistical Society, Series B (Methodological) 1977; 39(1): 1-38.

[21] M. Piccardi, "Background subtraction techniques: a review," in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, vol. 4, 2004, pp. 3099-3104.