Chapter 3

3 Proposed Approach

The proposed Visual Tracking algorithm identifies moving object and also tracks the moving objects. Estimation of the motion parameters such as location, direction and speed of the moving objects are derived from the image sequences with the help of CCD Camera. Two different approaches are proposed for single object tracking and multiple object tracking. To overcome the problems in Mean Shift Tracking Algorithm, a novel Block Matching Tracking Algorithm using Predictive Motion Vector based on 3D color histogram has been proposed and implemented efficiently for single object tracking. To improve the efficiency of the multiple objects tracking over the conventional methods like particle filtering and Kalman filtering, improved blob tracking method has been proposed. Multiple trackers have been used for region tracking of the objects. Feature based on Contourlet transform and color features with invariant moments are used for region tracking. Object Identification has been carried out by edge feature extraction and Contourlet transform with Principle Component Analysis (PCA) and compared with Curvelet Transform with PCA.

Effective techniques have been implemented for object Identification, background subtraction, motion segmentation; color descriptor and feature extraction for region tracking algorithms.

The proposed techniques involve mainly two tasks:

- Designing of an Object Classifier algorithm for Object Identification that has been used for Visual Tracking Process.
- Designing of Visual Tracking Algorithms for Estimating the Motion Parameters.

3.1 Object Classifier

Generalized Classifier has been designed that can be used in many applications. Classifier has been designed using two approaches: (1) Fixed sizes of the Objects in the dataset like Face dataset and Fingerprint dataset. (2) Varying size of the objects in the dataset like Vehicle dataset. It has different sizes according to the viewing angle for which three Class structures have been designed according to the length and width ratio. Designing of object classifier involves mainly two tasks: (A) Training of Classifier and (B) Object Identification of Query Image.

(A) Training of Classifier

As shown in the Figure 3.1 following steps are performed during Training of classifier:

(1) Resize all the images of dataset.

(2) Pre-processing to get the sharp images from the given dataset and perform the feature extraction of enhanced images. Flowchart including the steps for feature extraction is explained in the Figure 3.2.

(3) Generate Eigen matrix for dimensionality reduction for fast retrieval.

(B) Object Identification of Query Image

Object Identification task involves the steps as shown in the Figure 3.3:

(1) Resize the unidentified image to the same size as training dataset.

(2) Do Pre-processing to get the sharp images and perform the feature extraction of enhanced images.

(3) Project the image into Eigenspace using the Eigen matrix of trained classifier.

(4) Compute Similarity measure using the Euclidean distance classifier or neural network classifier for best match feature vector from Eigen matrix of trained dataset.

(5) Identify and label the objects using best match feature vector.

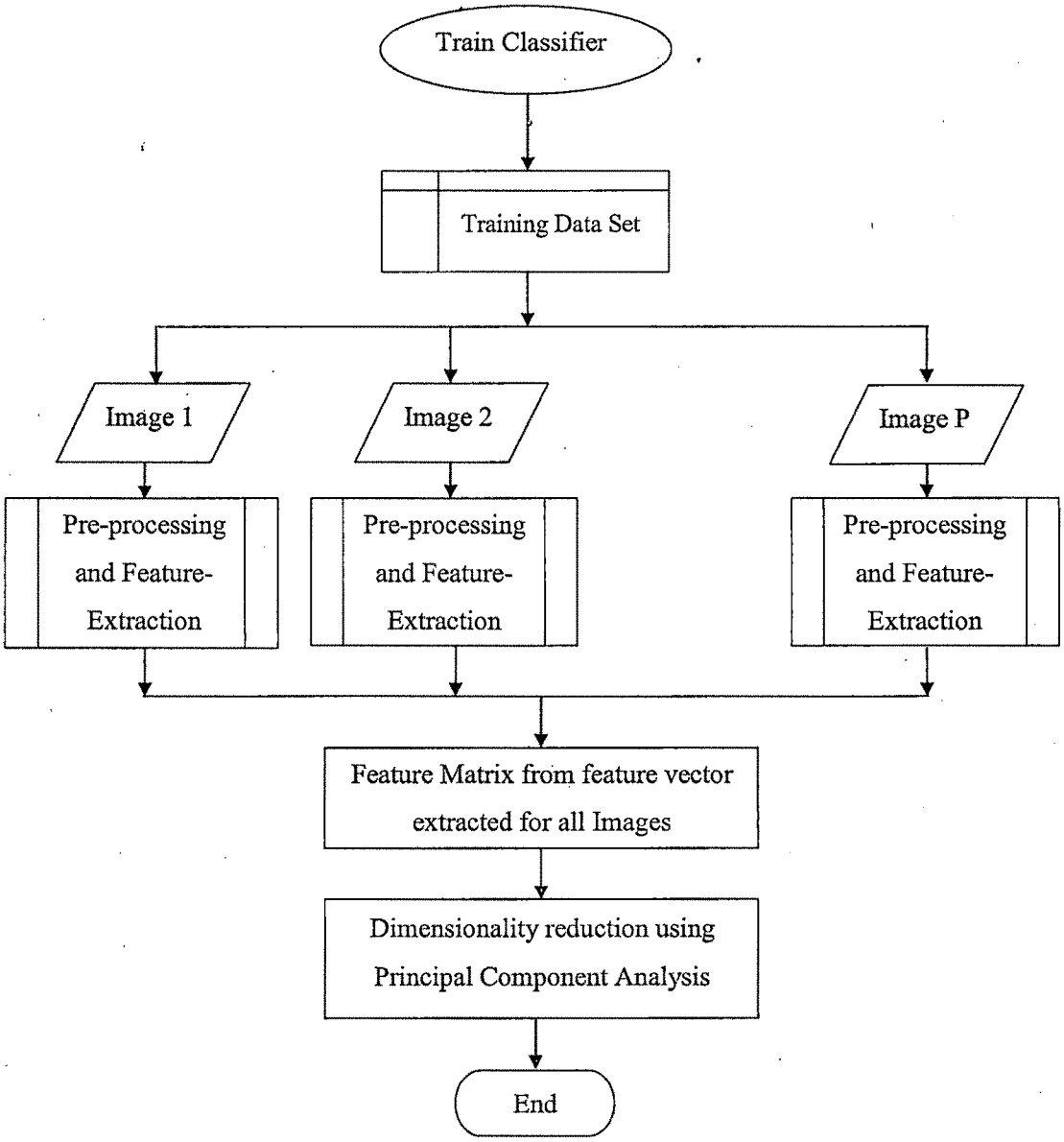


Figure 3.1 : Eigen Matrix Generation for Training Dataset

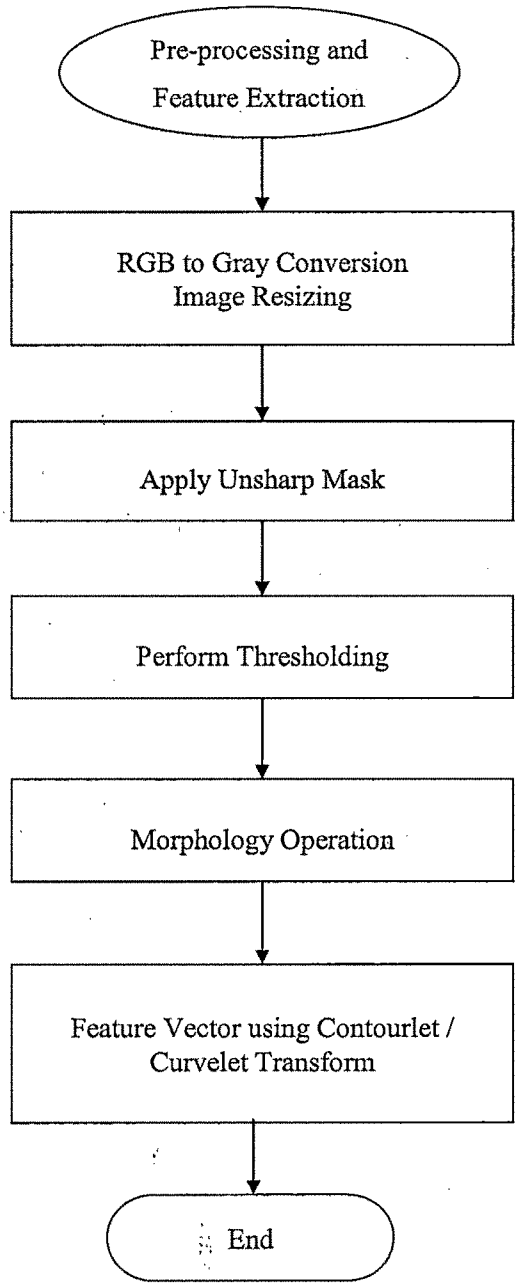


Figure 3.2 : Pre-processing and Feature Extraction

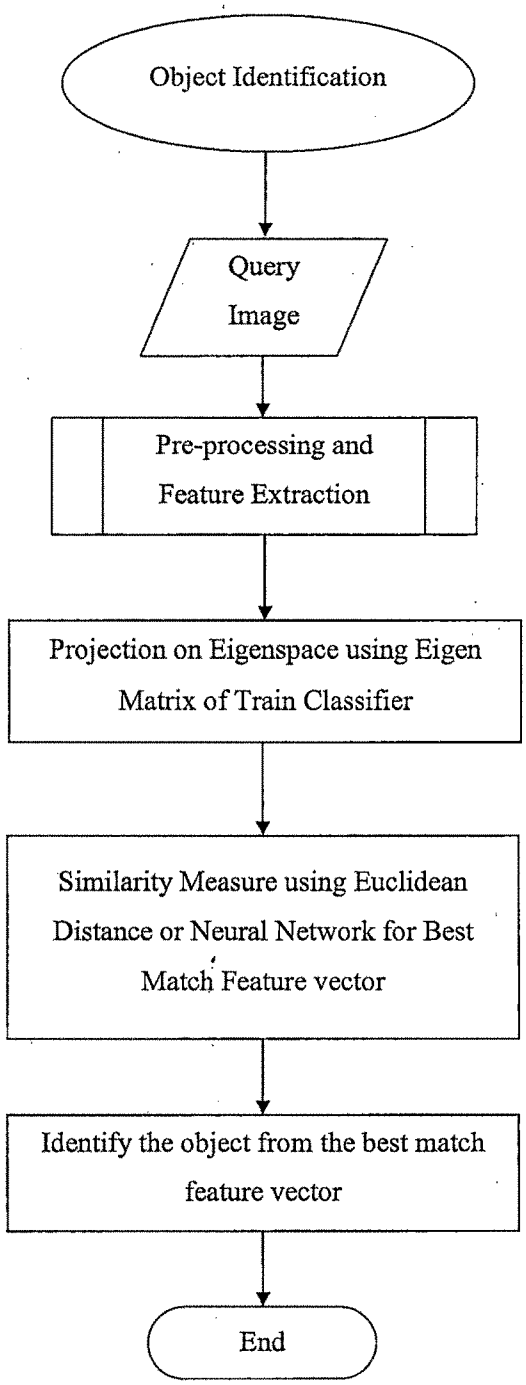


Figure 3.3 : Object Identification of Query Image

The following subsections include the mathematical models involved in each stage which are explained briefly. The subsections describe Pre-processing, Feature Extraction, and Principal Component Analysis for Eigen matrix generation and similarity measure for feature matching.

3.1.1 Pre-processing

3.1.1.1 Unsharp Filter

The Unsharp filter is a simple sharpening operator that enhances edges and amplifies high frequency components in an image via a procedure which subtracts an un-sharped or smoothed version of an image from the original image [87]. Let S be the dataset having P images for training and q images for testing. Color image $f1(m,n)$ of size $m \times n$ is converted into the gray scale image. Unsharp masking produces an edge image $g(m,n)$ from an input image $f1(m,n)$ by performing negative of Laplacian filter $f_{smooth}(m,n)$ as shown in the Figure 3.4.

$$g(m,n) = f1(m,n) - f_{smooth}(m,n) \quad \textbf{(3.1)}$$

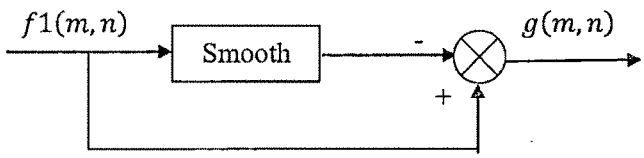


Figure 3.4 : Spatial Sharpening

Convolution has been performed with unsharp mask U and the image $f1(m,n)$ to get the edge image $g(m,n)$.

$$U = \frac{1}{(\alpha+1)}\begin{bmatrix} -\alpha & \alpha-1 & -\alpha \\ \alpha-1 & \alpha+5 & \alpha-1 \\ -\alpha & \alpha-1 & -\alpha \end{bmatrix} \quad (3.2)$$

The value of α controls the shape of Laplacian function. The range of α is from 0 to 1.

3.1.1.2 Thresholding using Otsu's Method

Thresholding has been applied on the Image after applying the unsharp filter. Global Thresholding has been applied using Otsu's method [87]. Otsu's method is one of the better threshold selection methods with respect to uniformity and shape measures. The Otsu method is optimal for thresholding large objects from the background [86].

The Otsu's algorithm performs the following steps:

1. Computes the normalized histogram of the input image. Denotes the components of the histogram by p_i where $i = 0,1,2,\ldots..,L-1$, where L denotes distinct intensity levels in a digital image of size $m \times n$, and h_i denotes the total number of pixels with intensity i. The normalized histogram p_i is calculated as

$$p_i = \frac{h_i}{\mathrm{m} \times \mathrm{n}} \quad (3.3)$$

2. Calculates the cumulative sums, $P1(k)$, for $k = 0, 1, 2 \ldots L-1$ using

$$P1(k) = \sum_{i=0}^{k} p_i \quad (3.4)$$

3. Computes the cumulative means, $m(k)$, for k = 0, 1, 2 ...L-1 using

$$m(k) = \sum_{i=0}^{k} ip_i \tag{3.5}$$

4. Computes the global intensity mean m_G using

$$m_G = \sum_{i=0}^{L-1} ip_i \tag{3.6}$$

5. Calculates the class variance $\sigma_B^2(k)$ for k = 0, 1, 2 ... L-1 using

$$\sigma_B^2(k) = \frac{[m_G P_1(k) - m(k)]^2}{P_1(k)[1 - P_1(k)]} \tag{3.7}$$

6. Obtain the Otsu's threshold, k^* as the value of k for which $\sigma_B^2(k)$ is maximum. If the Maximum is not unique, obtain k^* by averaging the values of k corresponding to the various maxima detected.

7. Obtain the separable measure, η^*, by evaluating the equation (3.8) at k=k*.

$$\eta(k) = \frac{\sigma_B^2(k)}{\sigma_G^2} \tag{3.8}$$

Where σ_G^2 is the global variance and can be derived by

$$\sigma_G^2(k) = \sum_{i=0}^{L-1} (i - m_G)^2 p_i \tag{3.9}$$

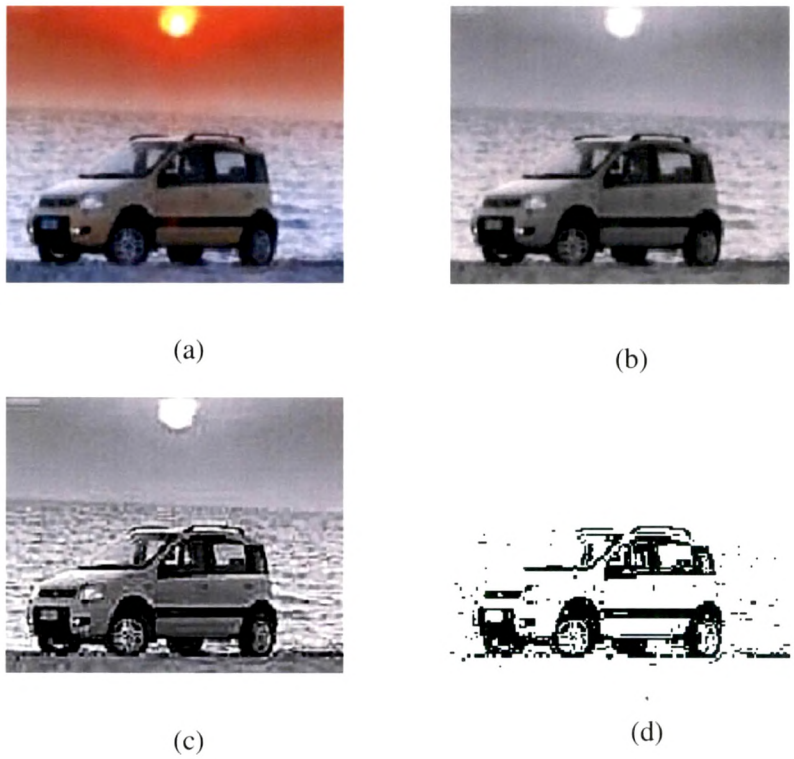
3.1.1.3 Removing Border Objects

If the original image is considered as a mask, the marker image $f_{mask}(m, n)$ can be obtained [87] using equation (3.10).

$$f_{mask}(m,n) = \begin{cases} g(m,n) \text{ if g(m, n) is on the border of image} \\ 0 \text{ otherwise} \end{cases} \quad \textbf{(3.10)}$$

The clear border image can be constructed by

$$f\,(m,n) = g(m,n) - f_{mask}\,(m,n) \quad \textbf{(3.11)}$$



(a) (b)

(c) (d)

Figure 3.5 : Pre-processing (a) Image of Car (b) Gray-scale Image (c) Image after applying Unsharp filter (d) Image after applying Threshold

Figure 3.5 shows the pre-processing steps performed on each image of dataset. Figure 3.5 (a) is the original color image. Figure 3.5 (b) is the resultant gray scale image converted into 0-255 gray levels. Figure 3.5 (c) is the resultant image after performing Unsharp filter mask. Thresholding is applied on unsharp filtered image which is followed by clearing border objects. Clearing boarder removes the border

point pixels to avoid the false classification of the object. Figure 3.5 (d) shows the resultant image after performing the thresholding and clear border operations.

(a) (b)

Figure 3.6 : (a) Bicycle Image (b) Pre-processed Image

Figure 3.6 (a) shows the original image on which the pre-processing has been applied. Figure 3.6 (b) shows the resultant image after performing the pre processing. The preprocessed image has been used for the feature extraction purpose.

3.1.2 Feature Extraction

Feature extraction is an essential pre-processing step in pattern recognition and machine learning problems. Feature extraction maps a larger information data space into a smaller feature space. The fundamental idea of feature extraction is to perform all computations in a smaller, simpler space.
Feature extraction pattern involves three design steps:

Feature Construction: This is the most challenging part of the pattern recognition system.

Feature Selection: This decision determines the balance between the search time and the post-processing time. For fast retrieval of dataset from feature matrix, Eigenvalues are constructed using the feature matrix in proposed methodology.

Feature Matching: This determines how fast the system can search the feature space. Euclidean distance classifier and Neural network classifier have been selected and compared. After comparison, Euclidean distance is found more efficient method while comparing recognition rate. So, finally for visual tracking Euclidean distance classifier has been implemented.

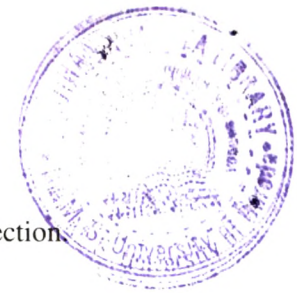
3.1.2.1 Feature Construction

After literature review, Discrete Contourlet Transform and Fast Discrete Curvelet Transform using wrapping have been found the efficient transforms for extracting the feature points. So both these transforms are implemented for feature construction and the results have been compared.

3.1.2.1.1 Discrete Contourlet Transform

Multiscale and time-frequency localization of an image is offered by wavelets. Wavelet transforms are not effective in representing the images with smooth contours in different directions. Contourlet Transform (CT) eliminates this problem by providing two additional properties known as directionality and anisotropy [23], [24].

Contourlet transform are divided into two main steps that are (1) Laplacian Pyramid (LP) decomposing and (2) Directional Filter Banks (DFB). Laplacian Pyramid decomposes the original image into a low-pass image and a band-pass image. Each band-pass image is further decomposed by DFB. The Multiscale and multidirectional decomposition of the image will be obtained by repeating the same steps upon the low-pass image [23]. Contourlet transform is a multi scale and multi directional image representation that uses a wavelet like structure for edge detection in the first



stage, and then a local directional transform for contour segment detection.

A double filter bank structure of the Contourlet is shown in Figure 3.7 .The Contourlet transforms obtains sparse expansions for images having smooth contours. In the double filter bank structure, Laplacian Pyramid (LP) [23] is used to capture the point discontinuities. Directional Filter Bank (DFB), followed by Laplacian Pyramid is used to link these point discontinuities into linear structures. The Contourlet have elongated supports at various scales, directions, and aspect ratios. These allow Contourlet to efficiently approximate a smooth contour at multiple resolutions. In the frequency domain, the Contourlet transform provides a Multiscale and directional decomposition.

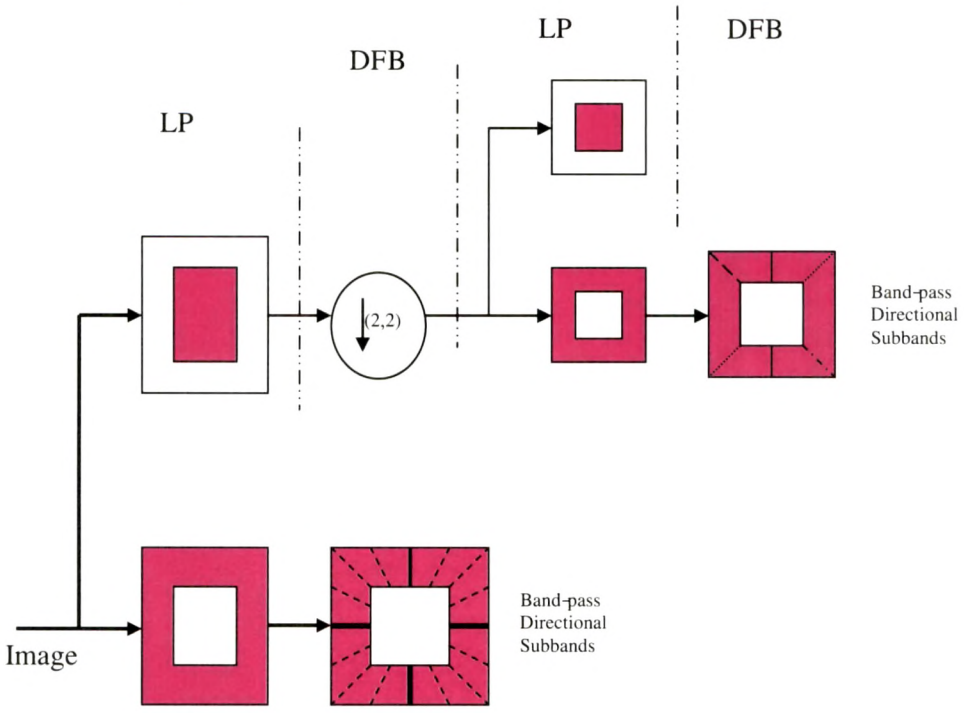


Figure 3.7 : Double Filter Bank Decomposition of Discrete Contourlet Transform

A. Pyramid frames

Multiscale decomposition is obtained by using the Laplacian Pyramid (LP) introduced by Burt and Adelson [70]. Band-pass image is obtained by first generating

the down sampled low-pass version using LP decomposition and then taking the difference between the original image and the prediction. This image is then processed by DFB stage. LP with orthogonal filters provides a tight frame with frame bounds equal to 1.

B. Directional filter banks

DFB is applied to capture the high frequency content like smooth contours and directional edges. The DFB is implemented by using a k - level binary tree decomposition that leads to 2^k directional subbands with wedge shaped frequency partitioning. DFB is constructed from two building blocks that are two channel quincunx filter bank with fan filters and shearing operator. Quincunx filter bank divides a 2D spectrum into two directions, horizontal and vertical. Shearing operators are used to the reordering of image pixels. Due to these two operations, directional information is preserved. This is the desirable characteristic in classifier system to improve retrieval efficiency.

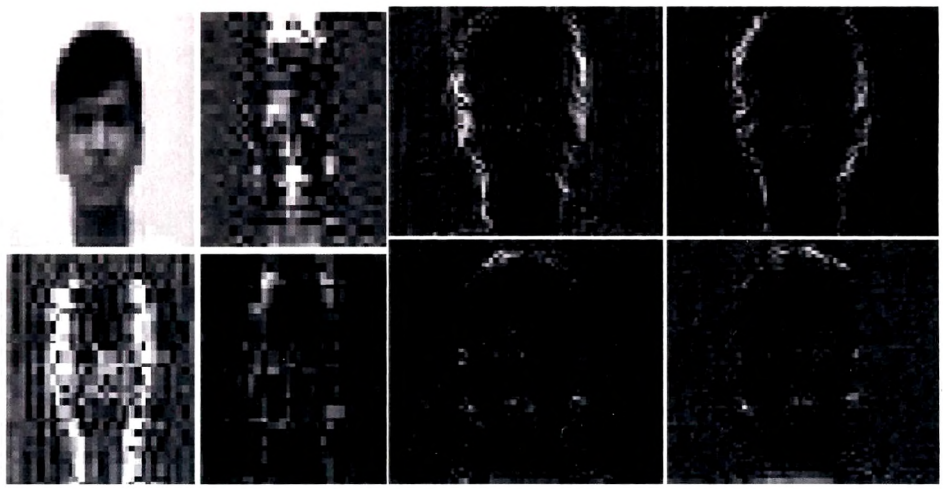


Figure 3.8 : Decomposition of Image using Contourlet Transform (2-Level and 'pkva' Filter for Pyramid and Directional Filter)

Band-pass images from the LP are applied to DFB so that directional information can be captured. The algorithm is applied on the coarse image. This combination of LP and DFB stages results in a double iterated filter bank structure known as Contourlet

filter bank .The Contourlet filter bank decomposes the given image into directional subbands at multiple scales. Figure 3.8 shows the decomposition of image using Contourlet Transform for level-2 using 'pkva' filter for both low-pass filter and direction filter bank.

The Contourlet Transform of two levels with 'pkva' filter is applied on the dataset images $f(m,n)$. Resulting image gives the decomposed coefficients as C_1,$C_{2\text{-}1}$,$C_{2\text{-}2}$.....$C_{n\text{-}1}$,....$C_{n\text{-}v}$, where v is the number of directions as shown in the Figure 3.8. These Coefficients are used to reorder the column vector I_i of the images. Image Vector I_i is constructed by converting coefficients to a column vector and then concatenation of all coefficient vectors. Let I = [I_1,I_2,I_3,....I_P] be the Feature Image Matrix constructed by Discrete Contourlet Coefficient, then Eigenvalue and Eigenvectors are calculated for I.

3.1.2.1.2 Discrete Curvelet Transform via Wrapping

Candes and Donoho introduced a new multiscale transform named Curvelet transform that was designed to represent edges and other singularities along the curves much more efficiently than traditional transforms by using fewer coefficients for a given accuracy of reconstruction [28],[29]. Implementation of Curvelet transform involves Subband Decomposition, Smooth Partitioning, Renormalization and Ridgelet Analysis steps [29]. There are two separate Discrete Curvelet Transform (DCT) algorithms introduced by Candes, Donoho and Demanet [71]. The first algorithm is the UnequiSpaced FFT transform (Fast Discrete Curvelet Transform via USFFT), where the Curvelet coefficients are found by irregularly sampling of the Fourier coefficients of an image. The second algorithm is the wrapping transform, which uses a series of translation and a wrap around techniques. Fast discrete Curvelet transform based on the wrapping of Fourier samples has less computational complexity as it uses fast Fourier transform instead of complex Ridgelet transform [29]. In the fast discrete Curvelet Transform via wrapping, a tight frame has been introduced as the Curvelet support to reduce the data redundancy in the frequency domain [71]. Normally, Ridgelet have a fixed length that is equal to the image size and a variable

width, whereas Curvelet have both variable width and length and represent more anisotropy. Therefore, the wrapping based Curvelet transform is simpler, less redundant and faster in computation [30] than Ridgelet based Curvelet transform.

Curvelet transform based on wrapping of Fourier samples takes a 2D image as an input in the form of a Cartesian array f [m, n] such that $0 \leq m < M$, $0 \leq n < N$. It generates number of Curvelet coefficients indexed by scale *j*, an orientation *l* and two spatial location parameters (k_1, k_2) as output. To form the Curvelet texture descriptor, statistical operations are applied to these coefficients. Discrete Curvelet coefficients can be defined by [29].

$$c^D(j,l,k_1,k_2) = \sum_{\substack{0\leq m\leq M \\ 0\leq n\leq N}} f[m,n]\varphi^D_{j,l,k_1,k_2}[m,n] \qquad \textbf{(3.12)}$$

Here, each $\varphi^D_{j,l,k_1,k_2}[m,n]$ is a digital Curvelet waveform. This Curvelet approach implements the parabolic scaling law on the subbands in the frequency domain to capture the curved edges within an image more effectively. Curvelet exhibit an oscillating behavior in the direction perpendicular to their orientation in the frequency domain [29].

Wrapping based Curvelet transform is a Multiscale transforms with a pyramid structure consisting of many orientations at each scale. This pyramid structure consists of several subbands at different scales in the frequency domain. Subbands at high and low frequency levels have different orientations and positions. The Curvelet is non-directional at the coarsest scale and becomes fine like a needle shape element at high scale.

With increase in the resolution level the Curvelet becomes finer and smaller in the spatial domain and shows more sensitivity to curved edges which enables it to effectively capture the curves in an image.

To achieve higher level of efficiency, Curvelet transform is usually implemented in the frequency domain. In the Fourier frequency domain both the Curvelet and the

image are transformed and then multiplied. Combination of the frequency response of Curvelet at different scales and orientations gives the frequency tilting that covers whole image in Fourier frequency domain as shown in the Figure 3.9. The product of multiplication is called a wedge. The product is then inverse Fourier transformed to obtain the Curvelet coefficient.

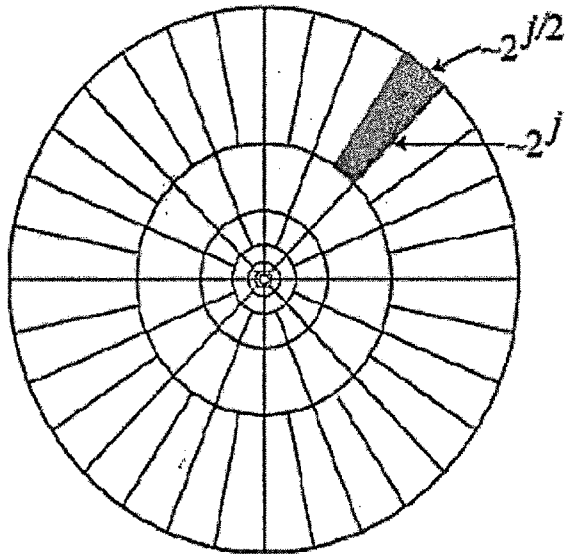


Figure 3.9 : Curvelet in the Fourier Frequency Domain [28]

For collecting Curvelet coefficients, inverse Fast Fourier transform is used. But, the trapezoidal wedge in the spectral domain is not suitable for use with the inverse Fourier transform. The wedge data cannot be accommodated directly into a rectangle of size $2^{j} \times 2^{j/2}$. To overcome this problem, Candes et al. have formulated a wedge wrapping procedure [71] where a parallelogram with sides 2^{j} and $2^{j/2}$ is chosen as a support to the wedge data as shown in the Figure 3.10. The wrapping is done by periodic tiling of the spectrum inside the wedge and then collecting the rectangular coefficient area in the center. The center rectangle of size $2^{j} \times 2^{j/2}$ collects all the information in that parallelogram. Discrete curvelet coefficients are obtain by applying 2D inverse Fourier transform to this wrapped wedge data. Wrapping based fast discrete curvelet transform is much more efficient and provides better transform results than ridgelet based curvelet transform.

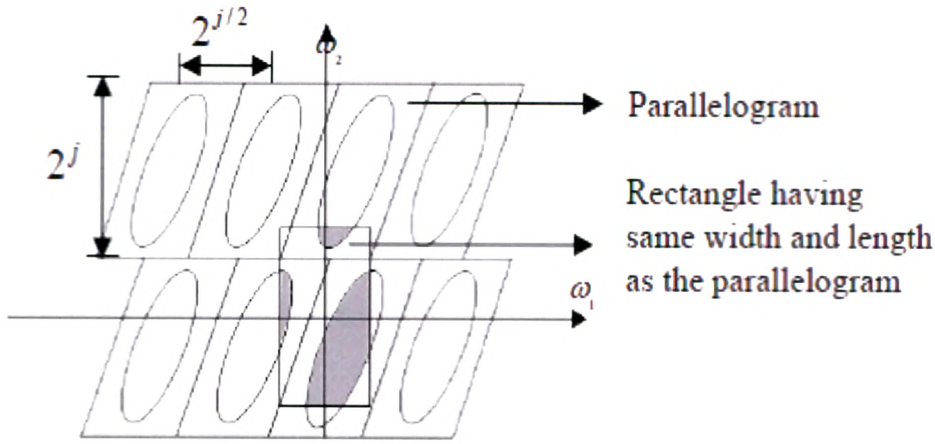
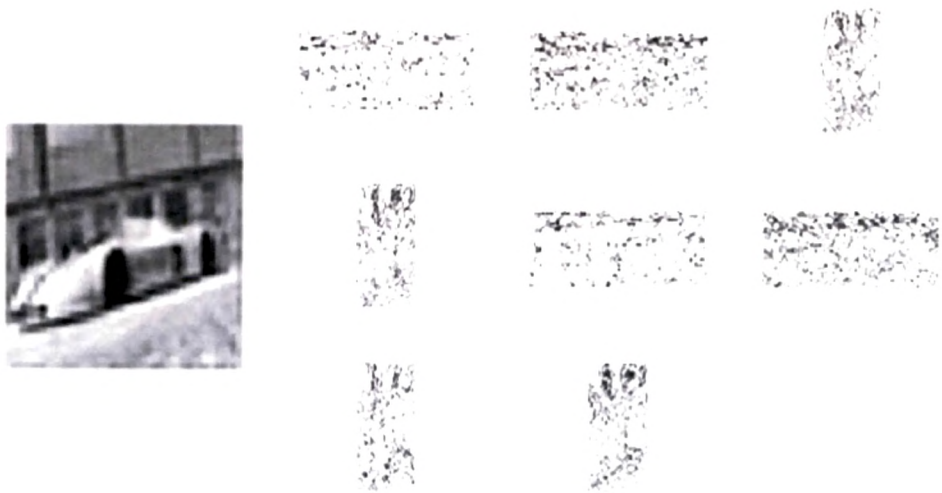


Figure 3.10 : Wrapping Wedge Around the Origin by Periodic Tilting of Wedge Data. The Angle $\boldsymbol{\theta}$ is in the Range $(\boldsymbol{\pi/4, 3\pi/4})$



(a) (b)

Figure 3.11 : Decomposition of Image using Curvelet Transform (a) First Level (b) Second Level

Considering total numbers of coefficients generated in the different levels of the Curvelet Transform and execution speed for generation of the coefficients, lower coarsest level is considered in the proposed method as it takes less execution time and it gives minor difference in the recognition rate compared to other levels.

The Curvelet transform of 1 coarsest level and 8 angles are applied on the dataset images $f(m,n)$. In the proposed method, the images are decomposed into single scales using real-valued Curvelet. The coefficients obtained using the Curvelet transform are shown in the Figure 3.11. These resultant Curvelet Coefficients are used to reorder the column vector I_i of the images. The discrete Curvelet transform via wrapping with pre-processing has been implemented in the proposed algorithm. Image Vector X_i is constructed by converting coefficients to a column vector and then performing catenation of all coefficient vectors. Let $X = [X_1, X_2, X_3....X_P]$ be the Feature Image Matrix constructed by Discrete Curvelet Coefficient, then Eigenvalue and Eigenvectors are calculated for X.

3.1.3 Feature Selection

For selecting most efficient features, Eigenvalues are calculated using Principal Component Analysis (PCA). PCA is used with two main purposes. First, it reduces the dimensions of the data to a computationally feasible size. Second, it extracts the most representative features out of the input data so that although the size is reduced, the main features remain, and still be able to represent the original data [17].

Eigenvectors and Eigenvalues are calculated for the Principal Component Analysis. Eigenvectors are derived from the covariance matrix calculated from the Feature matrix. Eigenvectors are invariant to the direction. The covariance matrix C of the input data is calculated from the equation (3.13)

$$C = \frac{1}{P}\sum_{i=1}^{P} \phi_i \phi_i^T \qquad (3.13)$$

Where the difference ϕ_i between image vector I_i and mean Ψ are calculated as equation (3.14) and (3.15)

$$\phi_i = I_i - \psi \tag{3.14}$$

$$\Psi = \frac{1}{P}\sum_{i=1}^{P} I_i \tag{3.15}$$

All Eigenvectors v_i and Eigenvalues λ_i of this covariance matrix are derived from the equation (3.16) as

$$\lambda_i = \frac{1}{P}\sum_{i=1}^{P}(v_i^T \phi_i^T)^2 \tag{3.16}$$

The set of Eigenvectors will have corresponding Eigenvalues associated with them; indicate the distribution of these Eigenvectors in representing the whole dataset. Typical references have shown that, only a small set of Eigenvectors with top Eigenvalues are enough to build up the whole image characteristic. PCA tends to find a P-dimensional subspace whose basis vectors correspond to the maximum variance direction in the original image space. New basis vectors define a subspace of images called Eigenspace.

The value of Eigenspace is represented using equation (3.17).

$$\varepsilon = \sum_{i=1}^{P} v_i \tag{3.17}$$

The weight ω_i of each input image vector I_i is calculated from the matrix multiplication of the different Φ_i with the ε Eigenspace matrix.

$$\omega_i = \phi_i \times \varepsilon \quad \textbf{(3.18)}$$

The image weight calculated from the equation (3.18) is the projection of an image on the object Eigenspace, which indicates the relative "weight" of the certainty that whether such image is an image of a training Dataset or not.

The initial training set S consists of P different Images. These images are transformed into a new set of vector T^w of all input training weight. Figure 3.12 shows the EigenImage after applying PCA to the Curvelet transform with pre-processing and without pre-processing.

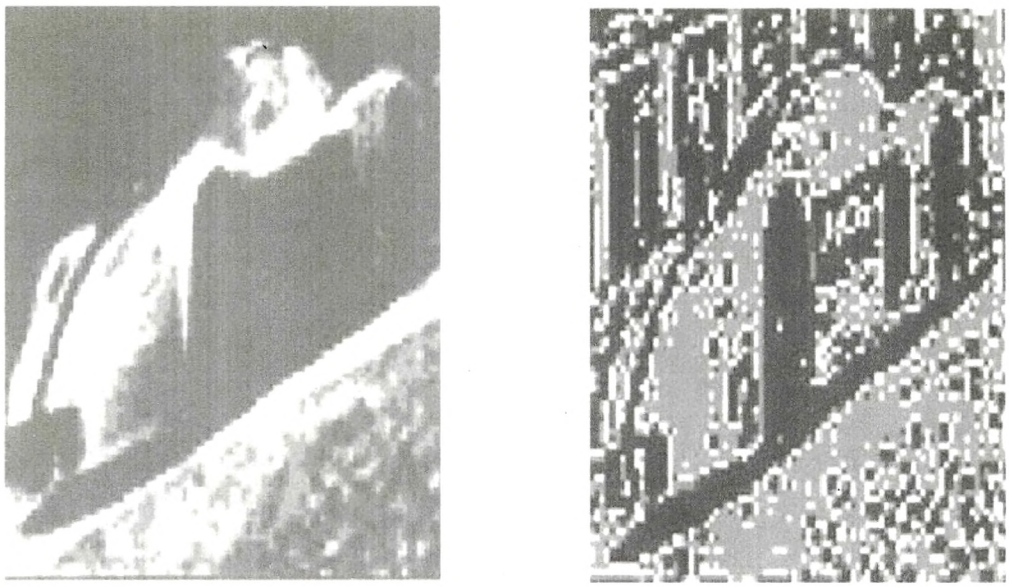


Figure 3.12 : (a) Eigenspace Image after applying Curvelet Transform without Pre-processing (b) Eigenspace Image after applying Pre-processing and Curvelet Transform

This transformation has showed how PCA has been used to reduce the original dimension of the dataset (P × m × n) to T^w (Size (P × P)) where generally P << m × n. Thus the dimensions are greatly reduced and the most representative features of the whole dataset still remain within P Eigen features only.

3.1.4 Feature Matching

For matching best feature vector from Eigen matrix, similarity measures like Euclidean distance measure and Neural network are calculated and compared.

3.1.4.1 Euclidean Distance Measure

With the coordinates (m, n) and (s, t) the Euclidean distance between coordinates p and q is defined as

$$De(p,q) = \sqrt[2]{[(m-s)^2 + (n-t)^2]} \quad (3.19)$$

3.1.4.2 Backpropagation Neural Network

Backpropagation was created by generalizing the Widrow-Hoff learning rule to multiple-layer networks and nonlinear differentiable transfer functions. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined by the Application. As shown in Figure 3.13. Networks with biases, a sigmoid level, and a linear output layer are capable of approximating any function with a finite number of discontinuities. Neuron Model (tansig, logsig, purelin) is an elementary neuron applied to the inputs. Each input is weighted with an appropriate weight matrix. The sum of the weighted inputs and the bias, form the input to the transfer function f. Neurons use any

differentiable transfer function f to generate the output. The Feedforward Neural network uses the Initialization, Activation, Weight Training and Iteration Steps to perform the Learning Phase. For training Neural Network, weight matrix T^w of Training Dataset obtained from the Contourlet-PCA/Curvelet-PCA is used as the input nodes of the Neural Network as shown in Figure 3.14.

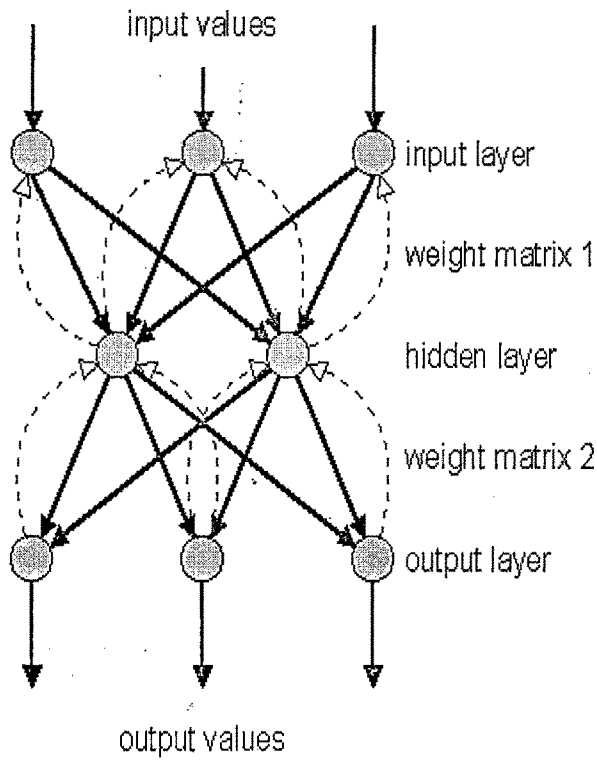


Figure 3.13 : Feed Forward Neural Network Model

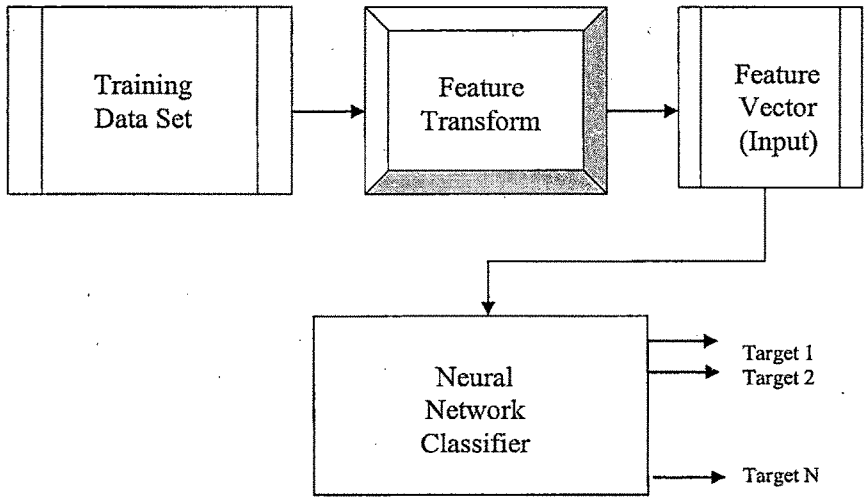


Figure 3.14 : Learning Phase of the Neural Network Classifier

When a new image from the test set is considered for recognition, the image is mapped to Contourlet-PCA / Curvelet-PCA subspace and weights are calculated for the particular image. The number of output nodes is equal to the number of total images, to be classified.

3.1.5 Proposed Methodology of Object Classifier

The objective of the proposed work is to extract the feature vectors for image Identification. Figure 3.15 illustrates overall process of calculating Contourlet transform / Curvelet transform and PCA applied to the training images and recognition of testing dataset. The first task for Feature extraction and selection and second task for Feature matching and object identification are executed as follows.

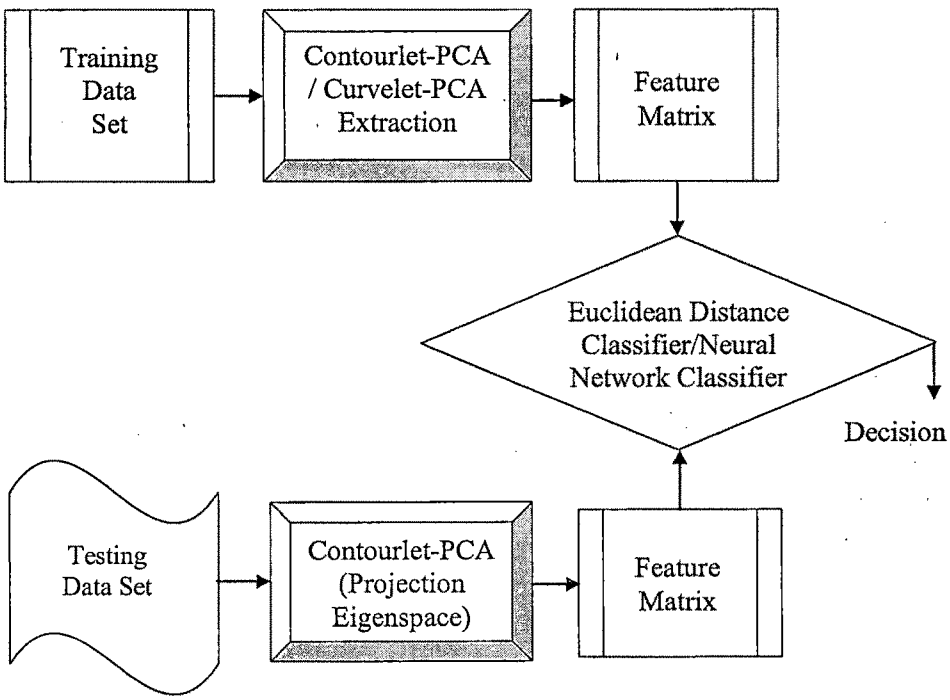


Figure 3.15 : Block Diagram of Proposed Object Classifier System

Let X_Image and Y_Image represent the training and testing datasets respectively. For gaining the best feature vector from the training dataset, at first, all the images are normalized.

Feature Extraction and Selection

The following steps are performed for feature extraction.

- RGB image is converted into grey scale image and resized to 64 × 64.

- Filtering is applied to remove noise and sharpening the image. Unsharp Contrast Enhancement filter is used for the pre-processing of face images. Thresholding has been applied for retrieving edge points.

- Feature extraction is performed using Discrete Contourlet Transform and Discrete Curvelet transform.

- **Contourlet Transform**: Decompose each image into the Contourlet transform. As a result of performing Contourlet Transform, coefficients of low frequency and high frequency in different scales and various directions will be obtained. Decomposed coefficients with the same size k × k as C_1, $C_{2\text{-}1}$, $C_{2\text{-}2}$…..$C_{n\text{-}1}$, ….$C_{n\text{-}u}$,where u is the number of directions. These Coefficients are used to reorder the column vector I_i of the images. All the coefficients are arranged to make a column vector.

  Curvelet Transform: Decompose each image into the Curvelet transform. As a result of performing Curvelet Transform, coefficients of low frequency and high frequency in different scales and angles are obtained. Decomposed coefficients of different sizes are obtained as C_1, $C_{2\text{-}1}$, $C_{2\text{-}2}$…..$C_{n\text{-}i}$….$C_{n\text{-}v}$ where v is the number of angles. These Coefficients are used to reorder the column vector Ii of the images. All the coefficients are arranged to make a column vector.

- The Feature image matrix I= [I_1, I_2, I_3 ..... I_P] is constructed from the coefficient column vector Ii, where i represent the number of images.

- Feature matrix I is transformed to lower dimension subspace Tw using PCA.

- Tw consists of Weight calculated for each image of the respective Dataset.

- Neural Network / Euclidean Classifier are used to measure the distance between the images.

Feature Matching and Identification

Feature Matching is performed by Euclidean distance classifier and neural network classifier for Object identification. The results of both methods compared and implemented in the visual tracking task.

A. Euclidean distance Classifier

In this classification method, each image transformed to a lower order subspace using Contourlet-PCA / Curvelet –PCA using the above steps. Upon observing an unknown test image X, the weights are calculated for that particular image and stored in the vector w_x. Afterwards, w_x is compared with the weights of training set T^w using the Euclidian distance using equation (3.20).

$$De(p,q) = \sqrt[2]{[T^w - w_x]} \qquad \textbf{(3.20)}$$

If average distance does not exceed some threshold value, the weight vector of the unknown image w_x is matched with the training dataset.

B. Neural Network Classifier

Weight vector is used to feed the respective neural network for obtaining the object recognition results. A threshold value near to 1 represents the classification matching

to the target and 0 represents the classification far away from the target. Logarithmic sigmoid transfer function is used for input layer and hidden layer. Back propagation training is implemented with Gradient descent with momentum.

3.2 Vehicle Identification System

Considering Visual Surveillance system and Traffic monitoring system, Vehicle Identification system has been implemented. The novel approach using three Class structures has been proposed to improve the efficiency of vehicle identification. Three classes have been classified according to the length and width ratio. For example as classification of vehicle has been categorized in bus, truck, cycle, scooter, rickshaw etc. If bus or truck is considered from side view, the length to width ratio becomes less than one. These types of vehicles which are rectangular in shapes are considered as class one. Class one vehicles are stored separately as a training set one. The second class of vehicles is considered having length to width ratio greater than one. Pedestrians are considered in this class. Vehicles having equal length and width where ratio is close to one is considered to be class 3. Different feature vectors are calculated for each class. The flow chart for training the feature matrix and Vehicle identification tasks are explained in Figures 3.16 and 3.17.

3.3 Visual Tracking

3.3.1 Single Object Tracking Algorithm

To overcome the problems arising in conventional tracking algorithms which were discussed in the literature review earlier, a novel Block Matching Tracking Algorithm using Predictive Motion Vector based on 3D color histogram has been proposed and implemented efficiently.

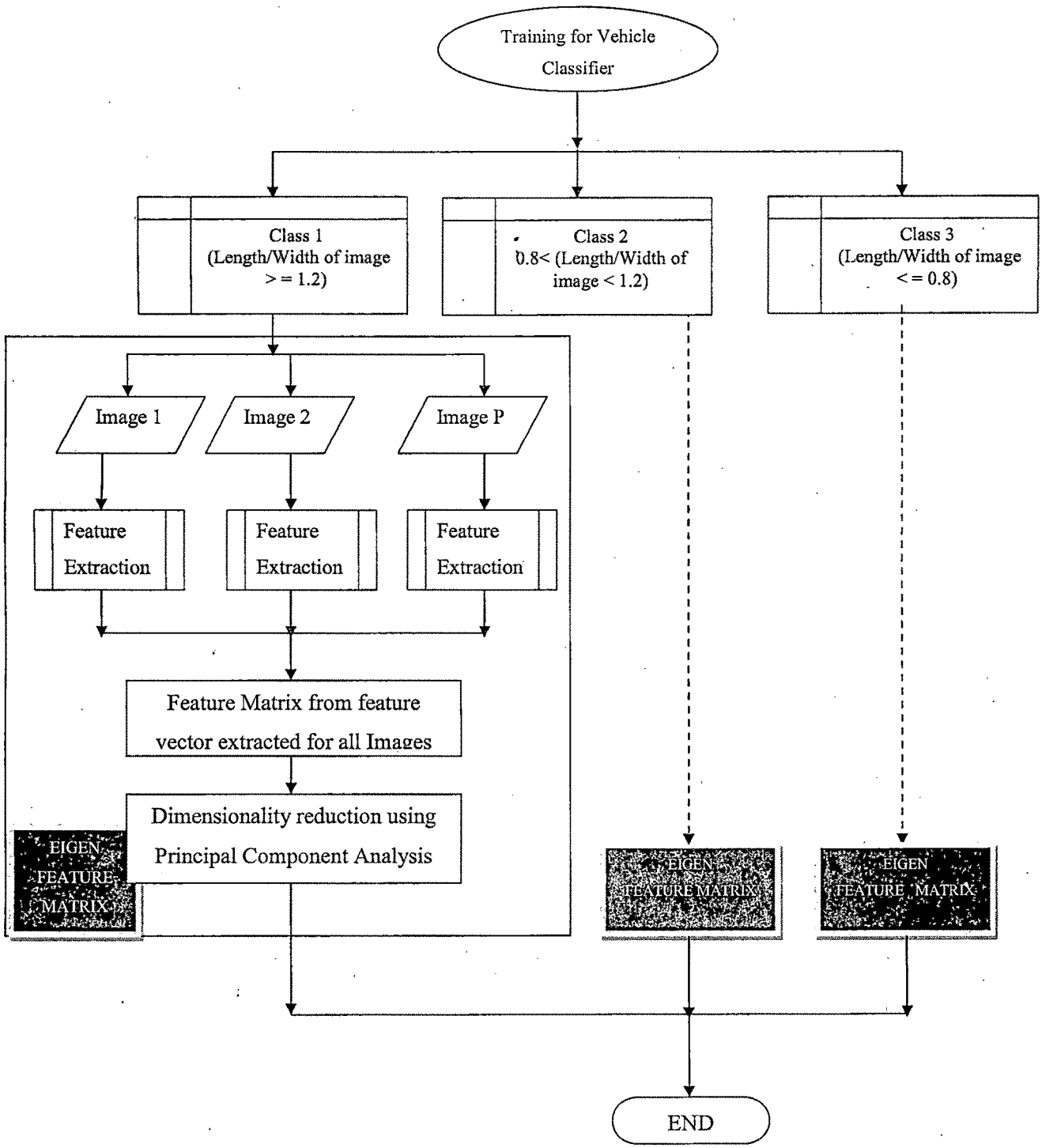


Figure 3.16 : Training of Vehicle Dataset using Three Class Structures

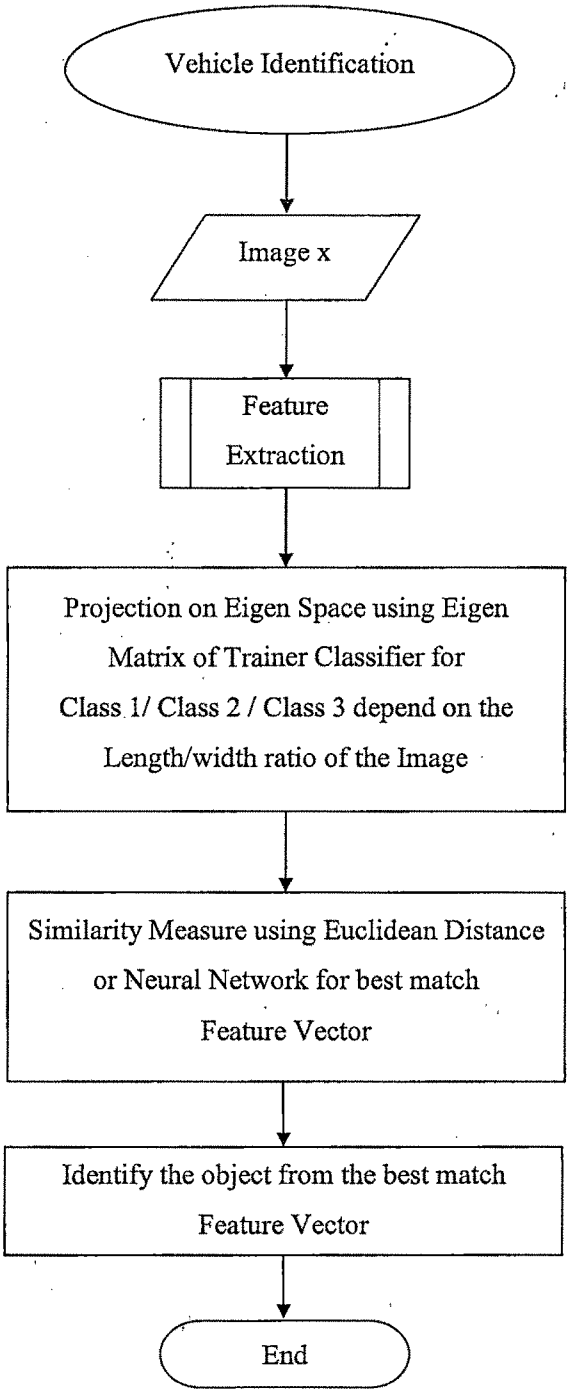


Figure 3.17 : Vehicle Identification System

To minimize the searching time of the object block in the frame, prediction based probabilistic search block matching algorithm has been implemented. The proposed algorithm is considered with a flexible size of block as well as pixel displacement. System tracks the single object selected by the user. Accuracy increased by implementing different conditions of the object like-object having similar type of background and foreground, object moving near to frame boundary, object with no motion in the frame sequence, object of boundary etc.

3.3.1.1 Object Tracking Algorithm

The main algorithmic flow chart as shown in the Figure 3.18 can be summarized as follows:

1. Define a rectangular block on the region of interest B_o in the first frame of a video sequence.

2. Compute the 3D colour histogram h1 ($m \times m \times m$) of the B_o region. Here 8 × 8 × 8 bins have been used to find colour histogram.

3. In the second frame, start from the former location and examine the surrounding windows by calculating histogram h2 ($m \times m \times m$) for each window Bj having block sizes same as B_o. Similarity measure by Histogram matching using Bhattacharya coefficient is applied across the Frame using equation (3.21).

$$\rho[p,q] = \sum_{M=1}^{m} \sqrt{p^{(u)} q^{(u)}} \qquad \textbf{(3.21)}$$

Where $p^{(u)}$ and $q^{(u)}$ are the histogram of two different images and ρ is the similarity measure. The large the value of ρ, more will be the similarities between the distributions. For two identical normalized histograms we obtain ρ =1, indicating a perfect match.

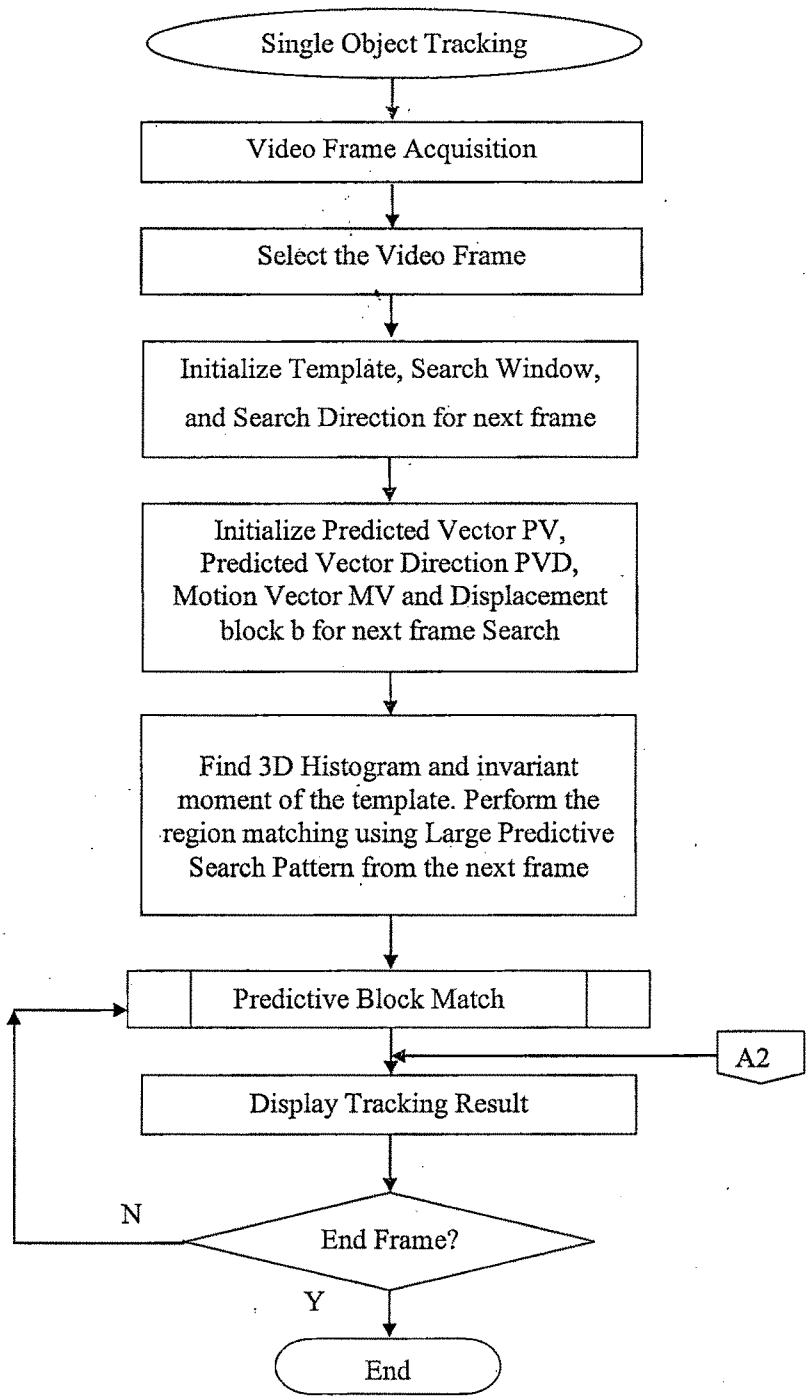


Figure 3.18 : Single Object Tracking Algorithm

4. Apply Predictive Block Matching (PBM) algorithm to find the search region and find appropriate similarity region while minimizing the distance between the detected locations using the matching criteria. The flow chart of PBM algorithm is explained in the Figure 3.20.

5. Iterate the above steps for all the frames in a sequence.

3.3.1.2 Predictive Block Matching Algorithm

Search Pattern

The proposed PBM algorithm employs pattern as illustrated in the Figure 3.19. The pattern called Large Predictive Search comprises of nine checking points from which eight points surround the centre one.

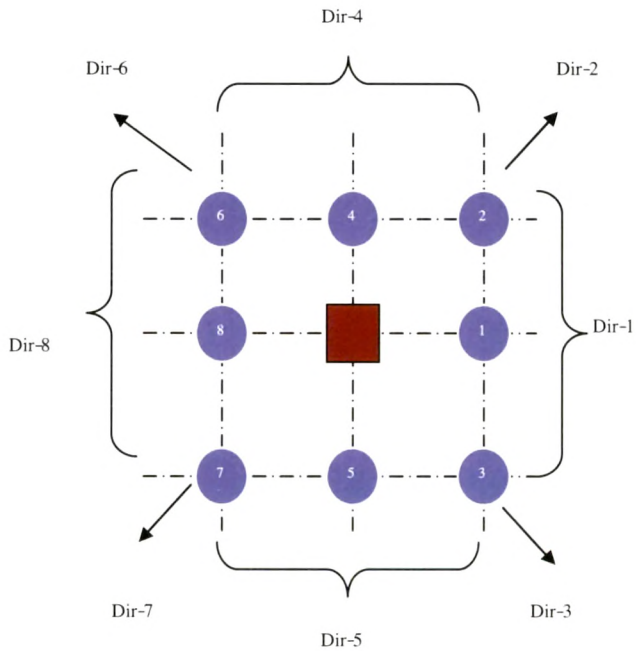


Figure 3.19 : Large Predictive Search Pattern – Nine points

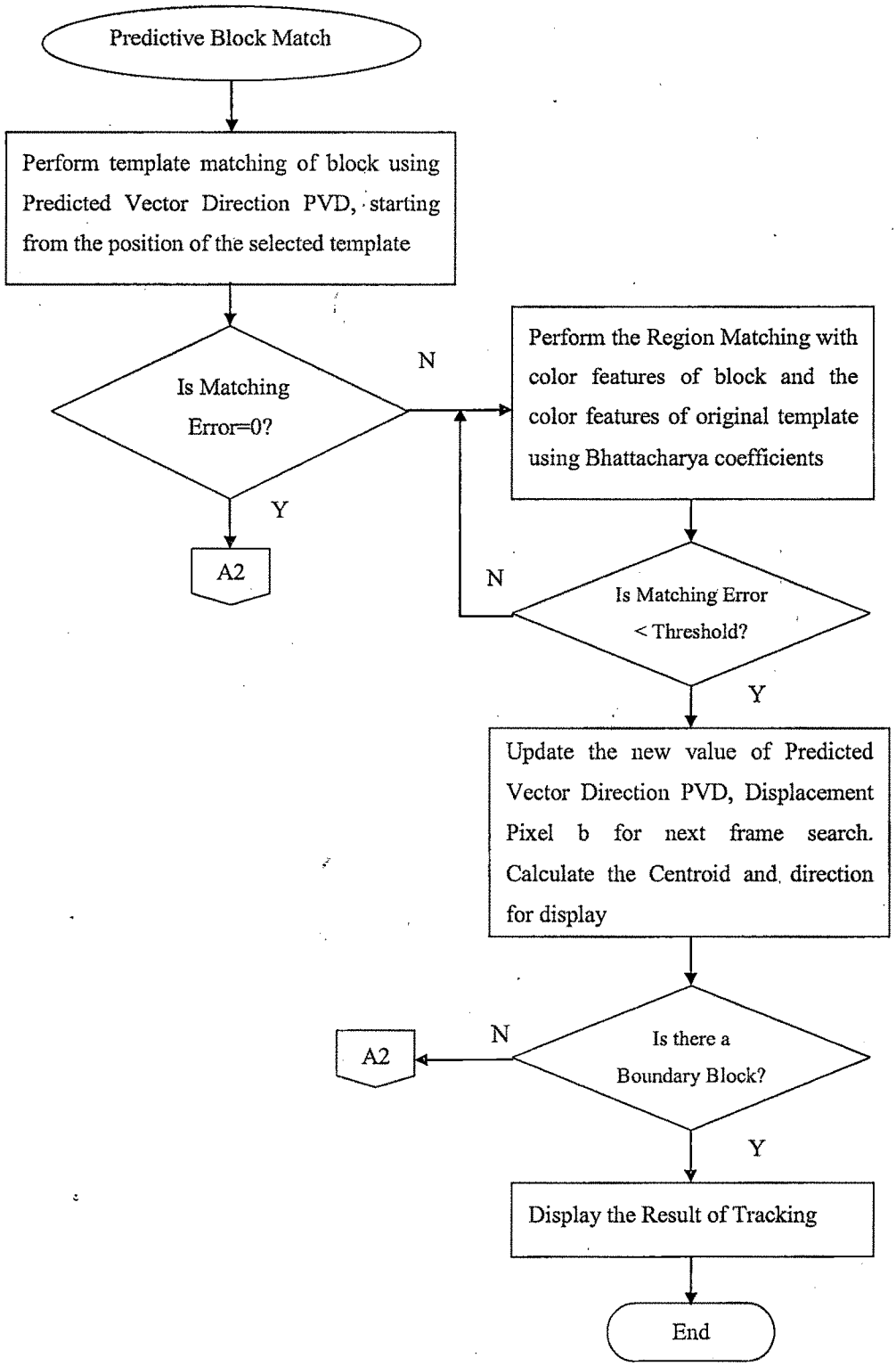


Figure 3.20 : Predictive Block Matching Algorithm

Block Matching Algorithm

The steps in the Block matching algorithm can be summarised as follows $n_1 \times m_1$ is the Bounding box size of the object, that is considered as a Block Size and the left corner pixel of the object is Bo (i, j).

a. Initialize the Predicted Vector (PV) and displacement value of pixel b. Initialize the Predicted Vector direction of search for the first frame as shown in the Figure 3.21.

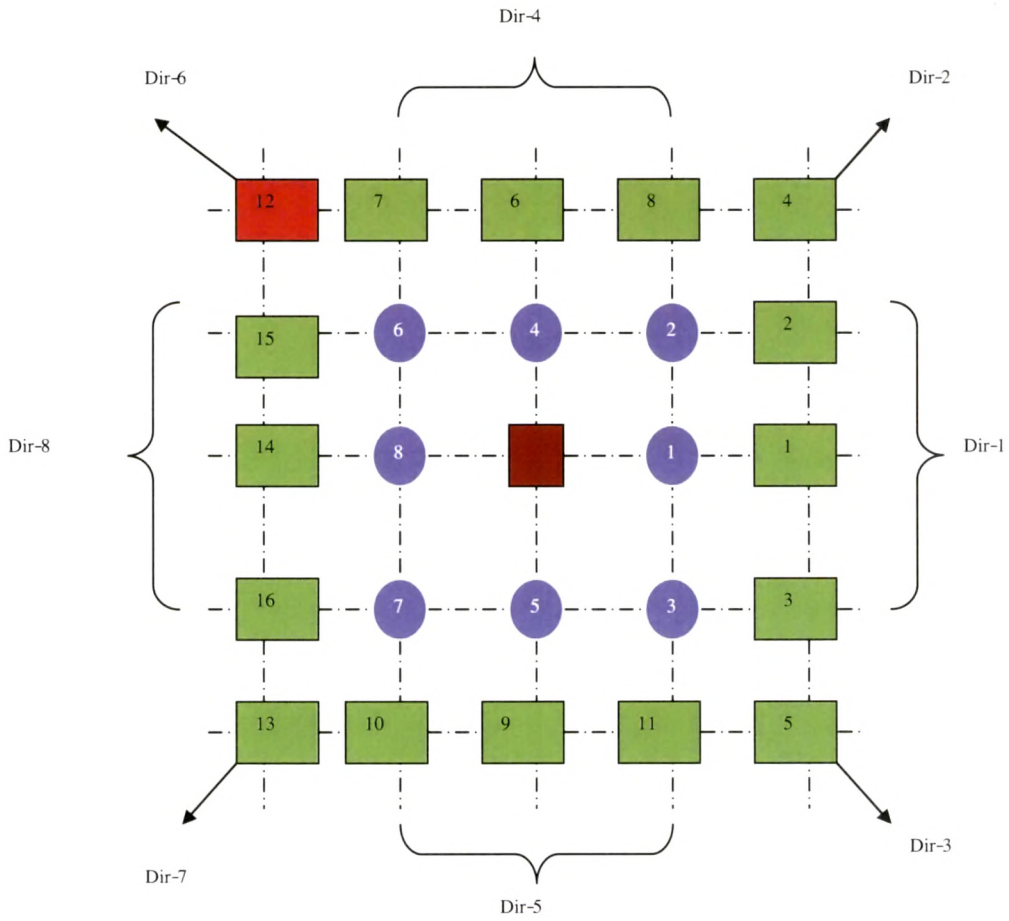


Figure 3.21 : Block Matching Direction and Search for Frame no. 1.

Best Block Match found in Direction 6

b. Compute the matching error (E) between the current block and block which appeared at the same location of the object in the reference frame.

If matching error (E) = 0

Motion vector (MV) ← 0

Exit and go to step 4;

else

Go to step 3.

c. Check nine search points of the Search window according to the direction set using predicted vector directions. Repeat the search as shown in the Figure 3.22 in all 8 directions until matching block is found.

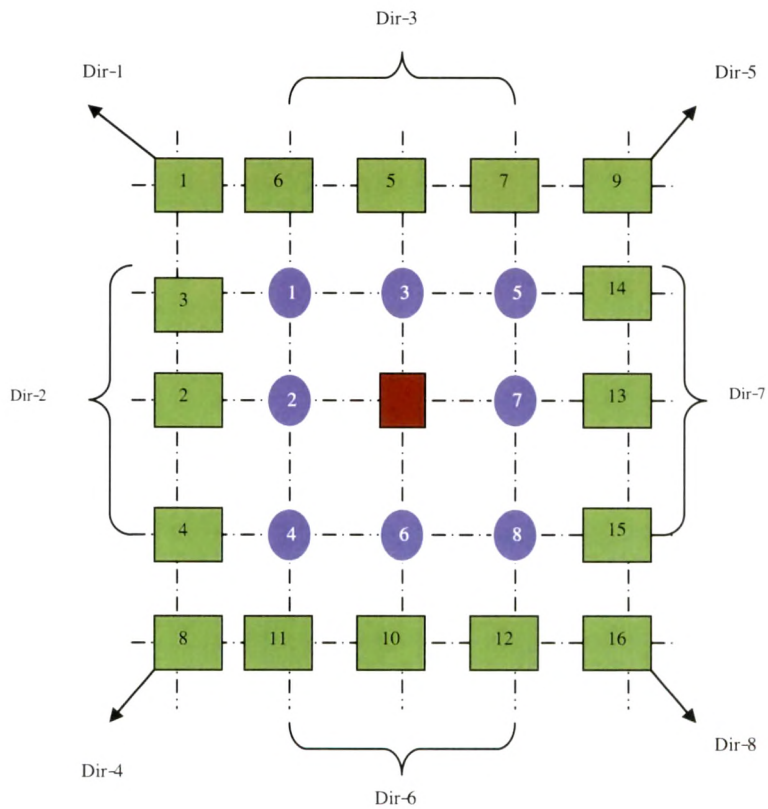


Figure 3.22 : Block Matching and Search Predicted Vector Direction for Predictor Vector found in the Direction 6

If matching Criteria > Threshold Value

Motion Vector (MV) ← Block pixel value Bj(x,y)

B11 ← x Coordinate of Match block – x Coordinate of previous Match Block

B12 ← y Coordinate of Match block – y Coordinate of previous Match Block

Predicted Vector PV ← i

Displacement pixel b ← max (B11, B12)/2

Next direction nd ← PV

Predicted Vector Direction PVD ← [nd, nd+1, nd-1, nd+2, nd-2, nd+3, nd-3, n+4]

If Match Block = Boundary Block

Exit from the main program.

else go to step3

else go to step 2.

3.3.1.3 Analysis of PBM algorithm

Zero motion prejudgment

To distinguish the static and moving blocks, a technique called Zero motion prejudgment [45] is implemented. The prejudgment is made by computing the matching error between the current block & block at the same location in the reference frame that corresponds to zero motion vector. If matching error is zero, than object is considered as a static object and algorithm jumps to the next frame without performing the remaining search.

Early termination

If the object in the frame found moving towards the boundary, the early termination of the object is considered by minimum matching difference at all the boundary points. If matching found, search process will be immediately terminated without checking next frames.

3.3.2 Multiple Objects Tracking

For multiple objects tracking, Kalman Filter tracking works well, only when an accurate model of the problem is available. Particle Filter Tracking supports multi–object tracking without requiring any modeling of the object but at the cost of higher computational speed. Mean Shift tracking is fast but not robust for extremely fast moving object and illumination changes. To overcome the above problem, Blob Tracking algorithm is used as tracking that established by temporal relationships between Blobs without the use of domain-specific information. For further improvement in the conventional Blob tracking, color segmentation was applied to retrieve color statistics of the object. To overcome the problem of same color object, features were extracted using Contourlet transform. Hybrid tracker implementation is the suggested efficient algorithm for visual tracking. The hybrid tracker has been implemented using color features and discrete Contourlet Transform.

A distinctive feature of the proposed algorithm is that the method operates on region descriptors instead of region themselves. This means that instead of projecting the entire region into the next frame, only region descriptors need to be processed. Therefore, there is no need for computationally expensive models.

To show the validity of the algorithm, traffic monitoring system is considered at present. Different statistical conditions are incorporated for making efficient algorithm. Vehicle classifier is also incorporated with the visual tracking task which identifies and displays the class of vehicle indicating car; bus etc in the visualization of tracking. The Contourlet transform feature extraction used in the classifier and also

used to track the object of the same color. Thus Algorithm calculations serve multipurpose and also increase the efficiency.

The main algorithm for visual tracking is summarized as follows:

- Perform video pre-processing on each frame.
- Do Video Segmentation using background subtraction.
- Perform Thresholding to convert the image into binary image.
- Apply Opening and Closing function to remove unwanted small data.
- Calculate the object statistics using Blob Analysis system object.
- Do the Color Segmentation using the object statistics derived from above steps.
- Track the object based on the color histogram and set of 2D moment Invariants. Same color featured object is to be differentiated using Contourlet transform and PCA. Centroid Statistics is to be used to measure the distance and direction of the object with respect to the previous frame.
- Visualize the result in the movie frame.

The overall system flow chart for multiple objects tracking has been explained in the Figure 3.23 and Figure 3.24. Figure 3.24 explains the flow chart for visual object tracker task which has been called in the main algorithm of visual tracking explained in the Figure 3.23.

Brief description about each step implemented in the algorithms is given in the sub sections.

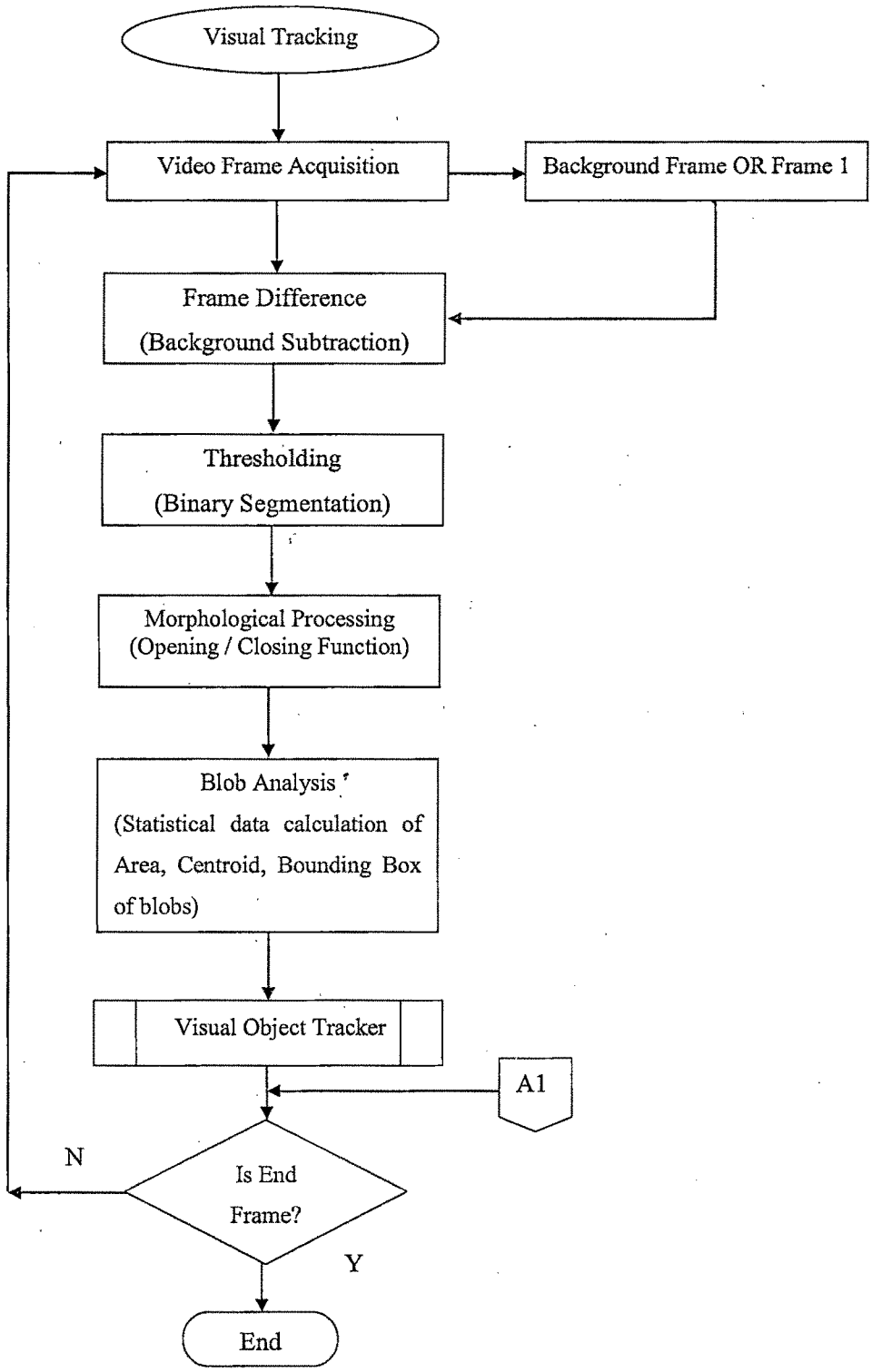


Figure 3.23 : Visual Tracking Algorithm for Multiple Objects

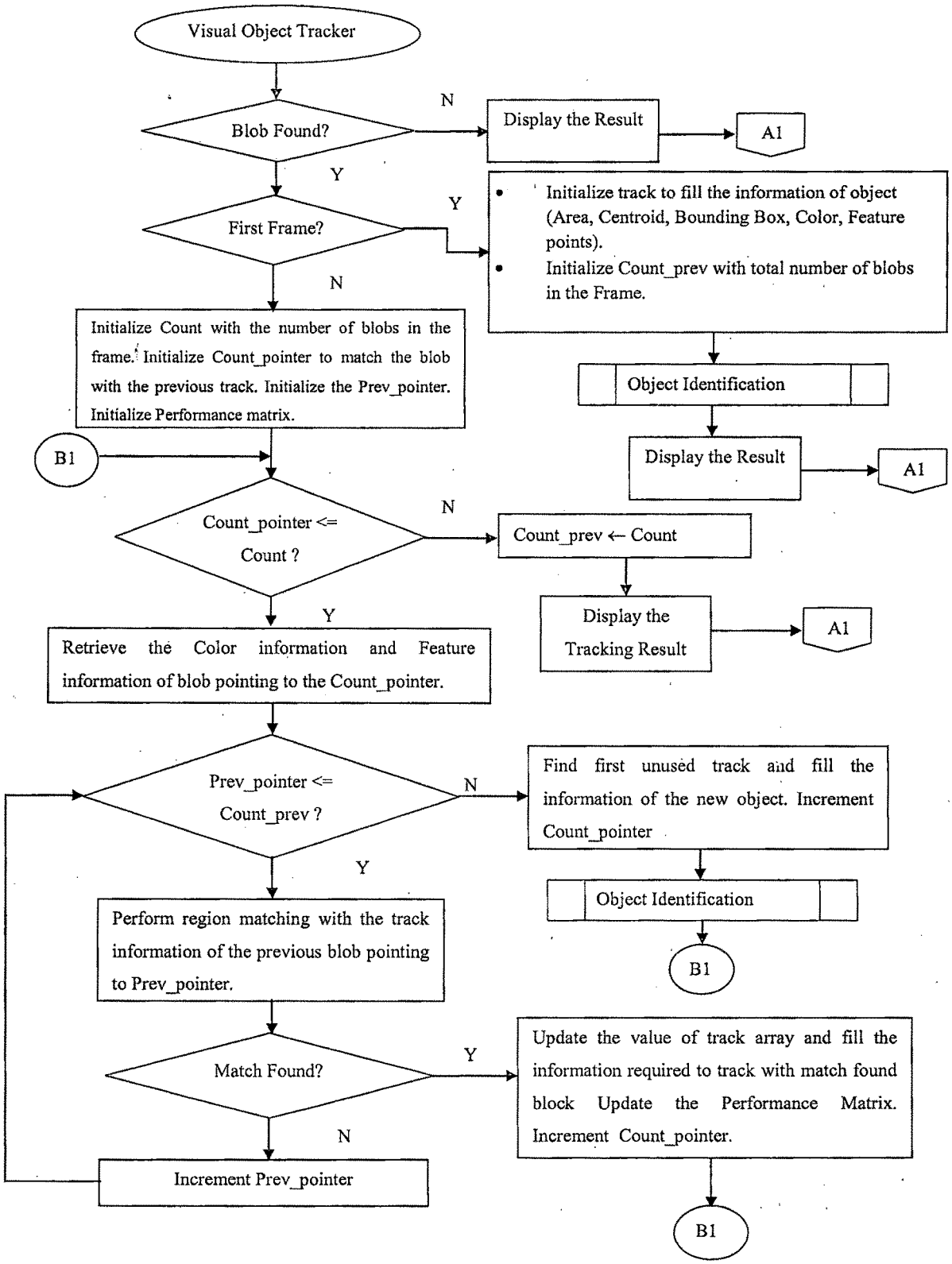


Figure 3.24 : Visual Object Tracker for performing Region Matching and Tracking

3.3.2.1 Foreground Object Extraction

For extraction of the foreground object; background subtraction, thresholding and morphological operations are performed on video frames.

- **Background Subtraction and Thresholding**

Segmentation approach to extract the foreground and background scene must be robust to shadow and changing light conditions. The method should be sensitive enough to detect actual changes in the background scene while not identifying false variations. Also the speed of execution must be at high rate which can be implemented for real time processing. To eliminate the effect of the shadow and lighting effect, all RGB frame are converted to YC_bC_r color space which is widely used for video processing [76]. In this format, luminance information is stored as a single component (y), and chrominance information is stored as two color-difference components (c_b and c_r). C_b represents the difference between the blue component and a reference value. C_r represents the difference between the red component and a reference value. They are expressed by the equations as

$$y = 0.299\,R_0 + 0.587\,G_0 + 0.114\,B_0 \quad \textbf{(3.22)}$$

$$c_b = -0.169\,R_0 - 0.331\,G_0 + 0.500\,B_0 \quad \textbf{(3.23)}$$

$$c_r = 0.500\,R_0 - 0.419\,G_0 - 0.081\,B_0 \quad \textbf{(3.24)}$$

Luminance information that is affected by the shadow and lighting conditions in the background is removed by equation (3.25) and (3.26). Foreground object is detected by the difference between current frame and image of the static background of scene which is normally considered as a first frame of the sequence or selected by the user as a background frame.

$$|Frame_i\,(y) - background\,(y)| > Th_1 \quad (3.25)$$

$$|Frame_i\,(c_b\,c_r) - background\,(c_b\,c_r)| > Th_2 \quad (3.26)$$

Where Th_1 and Th_2 are threshold values used for detection of foreground objects calculated with Otsu's threshold method.

In the proposed system, static background scenes are assumed, so a general threshold value is chosen that applies to all pixels. In addition, a number of post processing stages are used to clean up the resultant image. Images produced by background subtraction techniques have a lot of noise due to threshold selection. The second step of this algorithm is to make a morphological opening and closing in the binary image (the segmented image). The morphological operations remove small objects created by noise.

- **Morphological operation**

The morphological open and close operations, using circular structuring elements of radius 5 pixels, are applied to assist in the noise removal process. The morphological opening is composed of two basic operators with the same structure element Erosion and Dilation. Erosion and Dilation can be expressed in terms of Minkwoski addition and subtraction of two set A and B as [31]:

$$D(A,B) = A \oplus B = A + B = \cup_{b\in B}\,(A + B) \quad (3.27)$$

$$E(A,B) = A \ominus B = A - (-B) = \cap_{b\in B}\,(A - B) \quad (3.28)$$

The mathematical opening and closing can be represented as

$$O(A,B) = A \circ B = (A \ominus B) \oplus B) \tag{3.29}$$

$$C(A,B) = A \bullet B = (A \oplus B) \ominus B) \tag{3.30}$$

Where A is an input image and B is a structuring element.

3.2.2.2 Blob Analysis

Each object is processed separately and decomposed into a set of non-overlapping regions to produce the region partition. This step takes into account the spatio-temporal properties of the pixels in the computed object partition and extracts homogeneous regions. After performing morphological operations, blob analysis can be done to identify the objects in the scene and calculate their features to track the objects in the binary image frame. Typically the blob features usually calculated are area, number of pixels which compose the blob, perimeter, location and blob shape. Color segmentation is performed by extracting the color descriptor from the original frame using the blob statistics. Two different ways of connection can be defined in the blob analysis algorithm depending on the application. One consists to take the adjacent pixels along the vertical and the horizontal as touching pixels and the other by including diagonally adjacent pixels as shown in the Figure 3.25.

0	x	0
x	*P(x,y)*	x
0	x	0

(a)

x	x	x
x	*P(x,y)*	x
x	x	x

(b)

Figure 3.25 : (a) 4-Neighbourhood Pixels (b) 8-Neighbourhood Pixels

Figure 3.26 describes the overall steps for performing the blob segmentation.

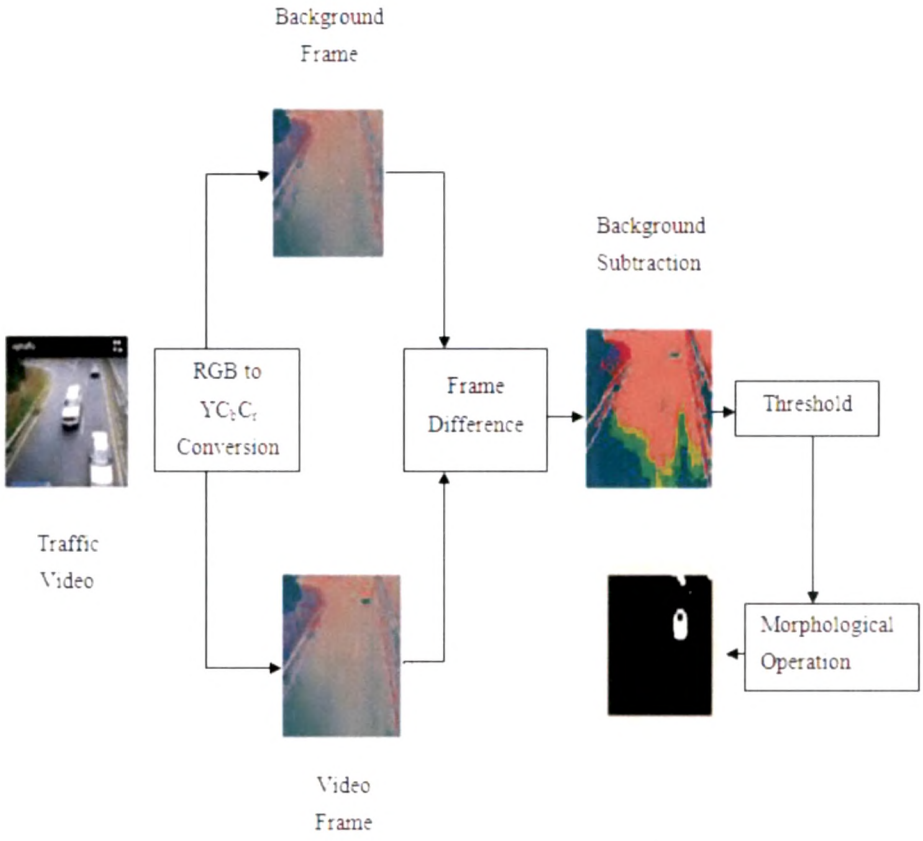


Figure 3.26 : Blob Segmentation Module

Blob statistics

For target instance called blob having the area a_t^l for l number of objects at time t for frame number n includes its Area $A(a_t^l)$. Its Centroid $c(a_t^l)$ is computed as:

$$c(a_t^l) = \frac{1}{A(a_t^l)} \sum_{i=0}^{A(a_t^l)-1} p_i \qquad \textbf{(3.31)}$$

Where p_i are the blob pixels

The similarity s and the Centroid distance $D(a_t^l, b_{t-\Delta t}^l)$ between two target instances a_t and $b_{t-\Delta t}$ in two consecutive time slices t and t-Δt are defined as:

$$s(a_t^l, b_{t-\Delta t}^l) = \left| \frac{A(a_t^l) - A(b_{t-\Delta t}^l)}{A(a_t^l) + A(b_{t-\Delta t}^l)} \right| \quad (3.32)$$

$$D(a_t^l, b_{t-\Delta t}^l) = \sqrt{\left(c(a_t^l) - c(a_{t-\Delta t}^l)\right)^2} \quad (3.33)$$

3.2.2.3 Region Descriptor and Region Matching

The performance of the blob analysis algorithm depends totally on the quality of the segmentation. With a bad segmentation, the blob analysis can detect some not interesting blobs or can merge some different blobs due to lighting condition or noise in the image. Blob analysis finds the entire bounding boxes which are created and puts a label for each one to memorize the different regions present in the scene. For each region a set of features are extracted as region descriptors. The feature space used is composed of spatial and temporal features. Color descriptor is derived for each region using blob statistics from the current frame. 3- D histogram is calculated for each region. Color descriptor is converted into seven invariant moments for dimensionality reduction.

Hu (1962) derived a set of two dimensional invariant moments in shape recognition using algebraic invariant. Two dimensional moments of a digitally sampled M × M image having gray function $f(x, y)$ for (x, y=0,.....M-1) can be expressed by

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{M-1} (x)^p . (y)^q f(x, y) \quad (3.34)$$

Where $p, q = 0,1,2,3$ .....

The moments $f(x, y)$ translated by an amount (a, b) are defined as

$$\mu_{pq} = \sum_x \sum_y (x + a)^p . (y + b)^q f(x, y) \tag{3.35}$$

The central moments m'_{pq} or μ_{pq} can be computed using equation (3.34) and (3.35) by substituting a = -$\bar{x}$ and b = -$\bar{y}$ as

$$\bar{x} = \frac{m_{10}}{m_{00}} \; and \; \bar{y} = \frac{m_{01}}{m_{00}} \tag{3.36}$$

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p . (y - \bar{y})^q f(x, y) \tag{3.37}$$

The shape of an image can be represented in terms of seven invariant moments ($\emptyset_1 - \emptyset_7$) expressed by equation (3.38) to (3.44). The first six functions ($\emptyset_1 - \emptyset_6$) are invariant under rotation and last $\emptyset_7$ is both skew and rotation invariant.

The seven moments can be defined as

$$\emptyset_1 = \eta_{20} + \eta_{02} \tag{3.38}$$

$$\emptyset_2 = (\eta_{20} + \eta_{02})^2 + 4\eta_{11}{}^2 \tag{3.39}$$

$$\emptyset_3 = (\eta_{30} - 3\eta_{31})^2 + (3\eta_{21} - \eta_{03})^2 \tag{3.40}$$

$$\emptyset_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \tag{3.41}$$

$$\begin{aligned}\emptyset_5 = (\eta_{30} - 3\eta_{31})(\eta_{30}\eta_{12}) \cdot [3(\eta_{30} + \eta_{12})^2 \\ - (\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(3\eta_{21} \\ - \eta_{03}) \cdot [3(\eta_{30} + \eta_{12})^2 \\ - 3(\eta_{21} + \eta_{03})^2]\end{aligned} \tag{3.42}$$

$$\begin{aligned}\emptyset_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})\end{aligned} \tag{3.43}$$

$$\begin{aligned}\emptyset_7 = (3\eta_{21} - \eta_{03})(\eta_{30} \\ + \eta_{12}) \cdot [(\eta_{30} + \eta_{12})^2 \\ - 3(\eta_{21} + \eta_{03})^2]) (\eta_{21} \\ + \eta_{03}) \cdot [3(\eta_{30} + \eta_{12})^2 \\ - 3(\eta_{21} + \eta_{03})^2]\end{aligned} \tag{3.44}$$

Where the normalized central moments are denoted as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \tag{3.45}$$

Where $\gamma = \frac{p+q}{2} + 1$

The $\emptyset$ values make a seven entries feature vector that is used for measuring the similarity between images.

Region Matching is the projection of the information of the current frame n into the next frame n+1. Regions of frame n and frame n+1 with most similarity are

considered as the correspondent objects and receive same labels. Seven 2D invariant moment values are used for region matching. Two way matching has been performed for accurate matching purpose. Color histograms with 2D invariant moments have been used for region matching. Color histogram handles shape variation property of the object well. If more objects having similar color features in the frame exists, than it gives more than one correspondence between the target frames. For eliminating the problem, frequency feature transform is used. Contourlet transform with Principal component analysis handles the scale deformation of the object well.

3.2.2.4 Region Tracking

After finding the corresponding regions between two successive frames, through a top-down and a bottom-up interaction with the region partition step, objects of current frame are validated and are given same labels as previous frame. Motion analysis is performed to find the direction of motion and speed of the motion using the blob statistics as shown in the Figure 3.27.

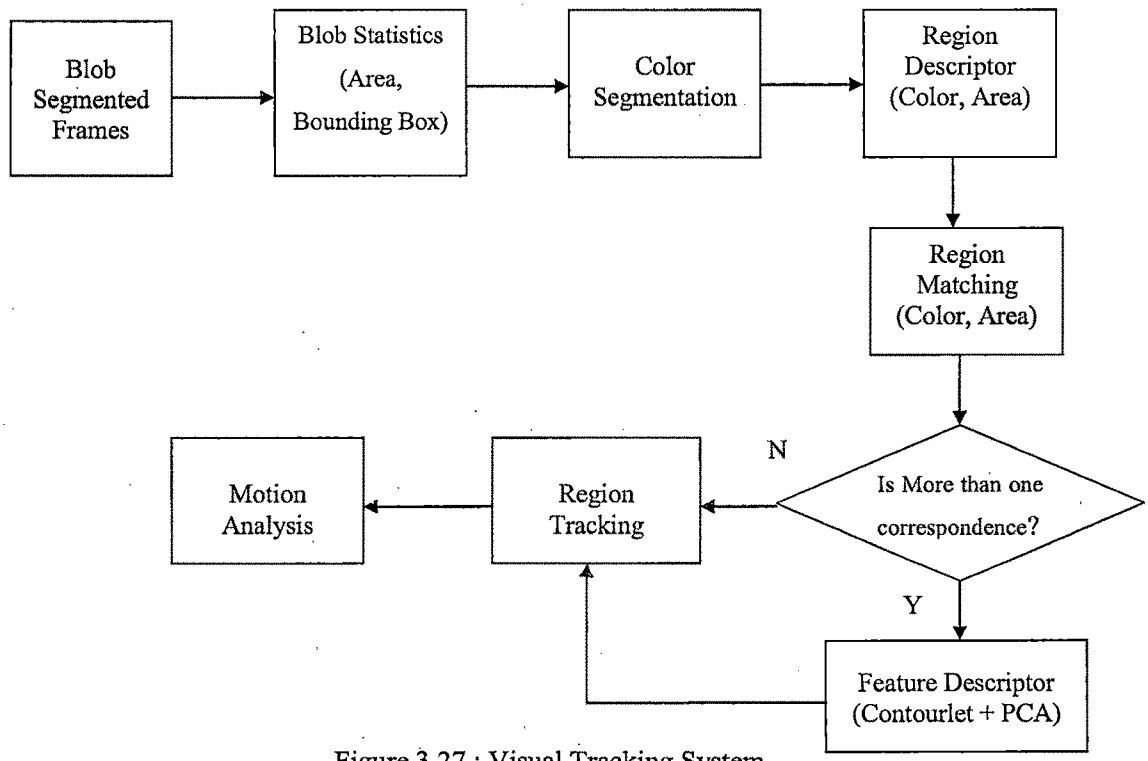


Figure 3.27 : Visual Tracking System

Summary: Object Identification and Visual Tracking algorithm for single object tracking and multiple objects tracking are discussed. For object identification, classifier is designed with the discrete Contourlet transform and fast discrete Curvelet transform via wrapping. For efficient design of Classifier, feature extraction is performed by discrete Contourlet transform and fast discrete Curvelet transform via wrapping followed by pre-processing stages used for image enhancement. For fast execution speed of feature extraction, Eigenvalues are calculated which are discriminant features of the image plays important role in the dimensionality reduction of feature matrix created for training dataset. Efficiency of the classifier is compared with the discrete Contourlet transform and discrete Curvelet transform with and without pre processing. Considering Visual Surveillance applications, Vehicle Classifier is designed with the 3-class structure for more improvement in the object identification task. Finally, discrete Contourlet transform with the pre processing is incorporated for object identification with the visual tracking task as it is more efficient and fast compared to other methods.

For single Visual tracking, novel block matching algorithm has been proposed with efficient termination conditions while tracking. Multiple objects tracking algorithm has been implemented using hybrid tracker. Hybrid tracker is more efficient for visual surveillance applications. Hybrid tracker is implemented with color features and texture features using discrete Contourlet transform, that are calculated for object identification purpose also. Visual tracking task calculates speed in terms of pixels, direction and object tracking number with object identification.

For actual speed conversion from image space to object space camera parameters are calculated. The camera parameters calculations with the experimental results of object identifications and visual tracking have been discussed in the next chapter.