COLOR IMAGE CLASSIFICATION

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE

OF

MASTERS OF TECHNOLOGY

IN

COMPUTER SCIENCE

BY

A. Praveen Kumar

(MTC0519)

Under the Guidance of

Prof. C.A. Murthy
(Machine Intelligence Unit)

INDIAN STATISTICAL INSTITUTE

203, Barrack pore Trunk Road

KOLKATA – 700 035

Certificate

This is to certify that the thesis entitled ***"Color Image Classification"*** *is a record of bona fide work done by* ***A. Praveen Kumar, Roll No. – MTC0519,*** *under my supervision and guidance for partial fulfillment of the requirement of the degree of* ***Master of Technology in Computer Science at Indian Statistical Institute, Kolkata, India.***

Sign:
Date: 2007
Prof. C.A. Murthy.
(Supervisor)

Sign:
Date:
External Examiner

Acknowledgements

It gives me immense pleasure and satisfaction to express my heart-felt gratitude and respect to my professor and supervisor, Prof. C.A. Murthy, for his invaluable guidance, supervision and encouragement throughout my project work. I will be grateful to him forever, for his most timely suggestions, and help regarding many aspects during this period. Working under him is a wonderful experience.

I express deep regard and thanks to all my teachers who have given me invaluable knowledge. I thank all my colleagues and my classmates for their support and maintaining an excellent work environment in the Lab.

Finally, I express my heart-felt thanks to my parents, my wife and well wishers who have unconditionally encouraged and supported me from the beginning to end.

Date:

A. Praveen Kumar.
Indian Statistical Institute,
Kolkata – 700 035.

Contents

I. Introduction…… 06

1.1 Characteristics of data……. 06

1.2 Classification……... 12

1.2.1 Feature selection…….. 12

1.2.2 Classification method ……. 15

II. Principles of classification method…… 17

2.1 Flowchart of the proposed algorithm....……. 36

III Methodologies applied……. 38

3.1 Color image segmentation …….. 38

3.2 Connected components labeling ……. 39

3.3 Edge detection …… 40

3.4 Hough Transform …… 43

3.4.1 HT for lines…….. 44

3.4.2 HT for ellipse……... 45

3.4.3 Invariant Generalized HT for arbitrary shapes…… 47

IV Experimental Results…… 54

4.1 Reasons for misclassification……... 55

V Conclusion and Scope of future work…… 56

VI References…….. 57

I. Introduction

In this dissertation, we were given a data set, and the problem is to classify each image into one of the classes. The data set is named test1.tar, and located at the web site wang.ist.psu.edu/docs/related .The data set contains ten classes of images. Each class contains hundred images. All images are color images.

The names of ten classes are:

1. Flower.
2. Dinosaur.
3. Elephant.
4. Horse.
5. Mountain.
6. Tribal.
7. Building.
8. Bus.
9. Food items.
10. Beach.

Our problem is to distinguish between the first eight classes of images in the above list, that is, if we are given one image among these eight classes then the machine would automatically classify it properly into one of the eight classes.

1.1 Characteristics of data

In this section, we describe the characteristics of the object classes in our data set. One main characteristic is, except the 'Dinosaur' class images, all other class images are 'natural' images taken from the real world. The 'Dinosaur' class images are 'artificial' images.

Experts artificially make the images of the ‘Dinosaur’ class. All the images of this class have white background and the color of the bottom portion of the dinosaur in the image is close to the white color. However, the dinosaurs have different colors. In almost all of the ‘Dinosaur’ images, less number of colors dominates an object (dinosaur) in the image. There is only one dinosaur in an image.

Figure 1. A ‘Dinosaur’ class image

The color of a flower in the ‘Flower’ class images is uniform. Usually green leaves cover the background in this class images. In most of the images, background is blurred. There are at most two flowers in most of the images. The color of all the flowers is same in an image.

Figure 2. A ‘Flower’ class image

In the ‘Bus’ class images, most of them have only one bus in an image, but in some images there are two; one bus is partially occluded by other, that is, one bus is completely visible and the other bus is partially occluded by this bus. The background is not uniform in all images. In some images it may be covered by

trees and sky and in some others, a building may be there. Important property in ‘Bus’ class images is that they contain regular shapes, buses are painted with different colors but in regular shapes, the windows of the bus are also in regular shapes (e.g. rectangular, circular, etc.).

Figure 3. A ‘Bus’ class image

We considered less number of images for the class ‘Building’. We ignored the images that do not enough features in it. In the ‘Building’ class images, some of the images contain human beings in front of the buildings, but they are small when compared to the size of the building. In some images, the background is covered with sky, whereas in some others, the building covers the whole image. ‘Building’ class images contain regular shapes (e.g. rectangular, cylindrical).

Figure 4. A ‘Building’ class image

In the ‘Elephant’ class images; there may be more than one elephant in the image. As we, all know elephants are of grey color. An elephant may be in between trees, it may be at a pond, or may be standing on clay. In most of the

images, the tusk of the elephant is visible. Trunk is also visible in almost all the images.

Figure 5. An ‘Elephant’ class image

In the ‘Horse’ class images, there are more than one horse in most of the images, but colors of horses may be different in one image. Most of the images contain trees and grasses in it. All horses have similar kind of mouth shape.

Figure 6. A ‘Horse’ class image

We considered less number of images for the classes ‘Mountain’ and ‘Tribal’ from the database. In the ‘Mountain’ class images, sky is clearly visible. There are overlapping mountains in almost all ‘Mountain’ class images. The color

of the mountain is not uniform. Because of sunlight, there may be shadow of one mountain on the other mountain in the ‘Mountain’ class images.

Figure 7. A ‘Mountain’ class image

In the ‘Tribal’ class images, we have considered only the images in which the face of the tribe is clearly visible. In this class images, only one human being is there in an image.

Figure 8. A ‘Tribal’ class image

From the above discussion, it is very clear that classifying the above classes is not easy, because all classes are having different characteristics of its own. While in some classes, the background is uniform, and in others, it is not. In some classes, if the object’s color is uniform, in other classes it is not. In some classes, if the object contains regular shapes, in some others it is not. In some classes, in an

image the number of the objects is more than one. The above classes do not share any common characteristic, by which we can identify each of the classes uniquely.

Also, in between the classes, there exists some overlapping such as, in some of the 'Bus' class images, a bus may be in front of a 'Building', in 'Building' class images, a 'human being' or an 'horse' may be in front of a building etc.

In the 'Elephant' class image, place of the elephant is not fixed, it may be in a pond, or it may be in between trees. In 'Horse' images, as the horse is in the forest, the image consists of grasses and trees, which gives rise to a lot of noise in the image.

In the 'Mountain' class images the color of the mountain is not uniform, one mountain can have different colors in it. In 'Tribal' class image, besides human being there may be so many things also in the picture like clothes of the tribe, ornaments worn by him/her, or weapons/other things held in hand by him/her. All these contribute to noise in the image.

Beside the above problems, all natural images have the effect of light. For example, if the image was taken during day, the image will be brighter due to which some portions of the object get exposed well, where as some other portions have the shadow effect, because of which we get to lose some information about the object. This effect can be observed in all natural images.

Even if lighting conditions are fine, the angle of the camera (i.e. from which side of the object the photo had been taken) will have the effect on the image. For example, for a 'Bus' class image, if the photo is taken from the side of the bus we can get much information about the bus, where as if the photo is taken from the front side we wont get much information about the image when compared to the former. Even non-homogeneous background will also affect the

image, as this tends to increase the noise in the image, which complicates the classification process.

We have to overcome all these above-mentioned difficulties in order to make an accurate good classification.

1.2 Classification:

Two main steps in the classification process are:

1. Feature Selection.
2. Classification Method.

1.2.1 Feature selection:

Feature selection, also known as *variable selection, feature reduction, attribute selection or variable subset selection*, is the technique, commonly used in machine learning, of selecting a subset of relevant features for building robust learning models. Feature selection helps people to acquire better understanding about their data by telling them that which are the important features and how they related with each other.

In a broad sense, features may include both text-based features (keywords, annotations) and visual features (color, texture, shape, faces). Within the visual feature scope, the features can further classified as general features and domain specific features. The former include color, texture, and shape features while the latter is application-dependent and may include, for example, human faces and fingerprints. The domain-specific features may involve much domain knowledge.

Color:

The color feature is one of the most widely used visual features in image. It is relatively robust to background complication and independent of image size and orientation. Some representative studies of color perception and color spaces can be found in [4].

The color histogram is the most commonly used color feature representation. To define color histograms, the color space is quantized into a finite number of discrete levels. Each of these levels becomes a bin in the histogram. The color histogram is then computed by counting the number of pixels in each of these discrete levels. Using the color histogram, we can find the images that have similar color distribution. One can think of the simplest measure of similarity by computing the distance between two histograms. Besides the color histogram, several other color feature representations have been applied in classification, including color moments, color sets, Color coherence vector and color correlogram, all these features are discussed in [5].

***Texture*:**

Texture is another important property of images. Various texture representations been investigated in pattern recognition and computer vision. Texture representation methods can be classified into two categories: *structural* and *statistical*. Structural methods, including *morphological operator* and *adjacency graph*, describe texture by identifying structural primitives and their placement rules. They tend to be most effective when applied to very regular textures. Statistical methods, including *Fourier power spectra*, *co-occurrence matrices*, *shift-invariant principal component analysis* (*SPCA*), *Tamura feature*, *Wold decomposition*, *Markov random field*, *fractal model*, and *multi-resolution filtering* techniques such as *Gabor and wavelet transform*, characterize texture by

the statistical distribution of the image intensity. All these features discussed in [5].

Shape:

Shape of a visual object is one of the most powerful features for human perception mechanism. It remains invariant under different transformations (rotation, scaling & translation). Compared with color and texture features, shape features are usually described after images have been segmented into regions or objects.

In general, the shape representations divided into two categories, boundary-based and region-based. The former uses only the outer boundary of the shape while the latter uses the entire shape region. The most successful representatives for these two categories are Fourier descriptor [6] and moment invariants [7].

1.2.2 Classification Method:

We have to classify a color image not a pixel in an image, and the images taken in natural environment, we need a robust algorithm to extract features from the image to classify any given image into one of the above classes.

1.2.2.1 Content Based Image Retrieval (CBIR):

CBIR [8]-[9] is image retrieval technique in which, the images are searched and retrieved based on the visual content of the image. Based on these visual contents, desirable images features can be extracted and used as index or basis of search.

There are in general three fundamental modules in CBIR system. These are

1) Feature extraction.

2) Multidimensional indexing.
3) Retrieval.

The images in the image database are indexed-based on extracted visual contents (or features) such as color, texture, shape, etc. A multidimensional vector of the extracted features from the image can represent an image. The feature vector actually acts as the *signature* of the image.

The query image can be analyzed to extract the visual features and can be compared to find matches with the indices of the images stored in the database. The extracted image features are stored as meta-data, and images are indexed based on these meta-data information. This meta-data information comprises some measures of extracted image features. The feature vectors of similar images will then be clustered in *N-dimensional* space. Retrieving similar images to a query image then boils down to finding the indices of those images in the *N-dimensional* search space whose feature vectors are in the *N-dimensional* space are within some threshold proximity to the point of the query image.

1.2.2.2 Object Recognition:

The problem in object recognition [10]-[11] is to determine which, if any, of a given set of objects appear in a given image or image sequence. Thus, object recognition is a problem of *matching* models from a database with representations of those models extracted from the image luminance data. The *representation* of the object model is extremely important. Clearly, it is impossible to keep a database that has examples of every view of an object under every possible lighting condition. Thus, object views will be subject to certain transformations; certainly *perspective* transformations depending on the viewpoint, but also transformations related to the lighting conditions and other possible factors.

There are two stages in any recognition system. The first is the *acquisition* stage, where a model library is constructed from certain descriptions of the objects. The second is *recognition*, where the system is presented with a perspective image and determines the location and identity of any library objects in the image. Generally, the most reliable type of object information that is available from an image is *geometric* information. So object recognition systems draw upon a library of geometric models, containing information about the shape of known objects. Usually, recognition is considered successful if the geometric configuration of an object can be explained as a perspective projection of a geometric model of the object.

Our method of classification is different from Content Based Image Retrieval (CBIR) and Object Recognition.

In 'CBIR', the machine will retrieve relevant images from an image database (or distributed databases) on the basis of primitive (e.g. color, texture, shape etc.) or semantic image features extracted automatically. In 'Object recognition' machine match the models from the image database with the query image. In our problem, we extract the features from the query image, using certain properties of this features we classify the image into its corresponding class.

In 'CBIR' meta-data base, many images correspond to only one object, which took from different views, at different lighting conditions, in front of different backgrounds, with different scale and orientations. In 'Object recognition' model database, for each object, models of the objects, which are results of some transformation on the object, are stored. Our database contains only one image corresponds to one object.

II Principles of classification method

In this section, we are going to discuss about the principles of our classification method. Our main aim is to overcome the problems stated in section (1.1) to make a good classifier. We proceed in a way that, we observe some common features in one set of classes to separate it from the other set of classes, then with each set of classes we proceed individually.

From the discussion of section (1.1), we can observe that, in the 'Dinosaur' and 'Flower' class images; there is some sort of separation in the color of object and background. In 'Dinosaur' class images, the background is white, the bottom portion of the dinosaur is close to white, and the color of the object contained very less number of dark colors. In 'Flower' class images, the background is either green or blurred (black shade), whereas the object is pink, or yellow, or red, clearly different from the background color. This kind of behavior we cannot observe in all other classes, because of non-homogeneous background, and in some classes even the color of the object is non-homogeneous (e.g. 'Bus', 'Mountain', etc.).

To get hold of the above property, we require an algorithm, which groups the pixels of similar color. After applying the algorithm on the color image, we expect that object pixels and background pixels group separately.

Color image segmentation is a technique that extracts the homogeneous regions from color image. There are various segmentation algorithms in the

literature. We need one such segmentation algorithm, which can get hold of color intensity values of the pixels (R,G,B), in the color image, and do the segmentation.

Modified ‘DB scan algorithm’ [12] is used for this purpose. It takes the ‘color image’ as the input, and gives the ‘segmented image’ as the output. Outline of the algorithm has been discussed later. Basic idea of the algorithm is, by fixing a radius value ‘r’, for every color pixel of the form (R,G,B), we compute a group of similar pixels (with some threshold ‘k’ on the number of pixels in a group) using Euclidean distance (similarity measure). Then all the groups of the neighboring pixels are merged to make one group until no merges are possible. In this way, we can have the groupings of the pixels done. The final groups are the segments in the color image.

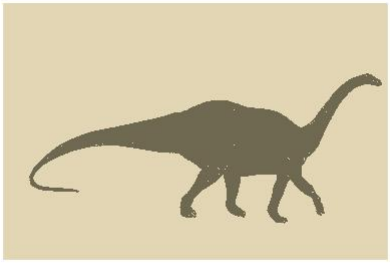


Figure 9. Segmented images

After getting the segmented image, we can observe that there are only two segments in the ‘Flower’ and ‘Dinosaur’ class images, but this is also the case with other classes, i.e. for some other class images, also we are getting only two segments. Therefore, only with segmentation we cannot separate ‘Flower’ and ‘Dinosaur’ classes from other classes.

From the segmented image, we can observe that in ‘Flower’ and ‘Dinosaur’ class images the object(s) and the background clearly gets separated as two segments. In other class images (even if only two segments are there in the segmented image), this is not the case, because of non-homogeneity in both the background and in the color of the object. This phenomenon is stated in terms of connectedness as, “In the ‘Flower’, ‘Dinosaur’ class images, the number of

connected components is less, where as in other class images the number of connected components is more because of noise in the background, etc."

Hence, on the segmented image we can apply a connected component-labeling algorithm to find out the number of connected components. First, from the segmented image, we make binary images; each one corresponds to one segment in the image. In the binary image, the pixels corresponding to a particular segment will have the value '1', the other pixels having the value '0'. Then on each of these binary images, we apply the connected component-labeling algorithm. This algorithm takes a binary image as its input and, gives an output with connected component labeling. Outline of the connected component-labeling algorithm is discussed later. Basic idea of the algorithm is, a pixel is labeled with a label based on its upper and left side neighbor, whose labels are already known. After applying the algorithm, we can get the number of connected components in the whole image, by summing up the number of connected components in all segments.

As there are at most two flowers in an image, we expect in the 'Flower' class image at most 'three' connected components. In 'Dinosaur' class images, the number of connected components in the image is two. Where as this number is more than three for all other class images. By using a threshold (4) on the number of connected components, we can separate 'Flower' and 'Dinosaur' class images from other class images.

Now we try to classify the 'Flower' class images from the 'Dinosaur' class images. The distinct feature that the 'Flower' class object (compared to the 'Dinosaur' class object) has is that the shape of the flower in the 'Flower' class image is almost convex (convexity fails at the edges of two intersecting leaves on the boundary of the flower). We use this property to separate the 'Flower' class

images from the ‘Dinosaur’ class images. This property can be simply stated as: “In the segmented image, the number of ‘sequences of object pixels in a row’ is more for ‘Dinosaur’ class image when compared to ‘Flower’ class image”. To use the above property we need to know, in the image which segment corresponds to object and which segment corresponds to background.

To separate flower images from dinosaur images, at first we do object background separation, i.e. we convert the color images into binary images where the object pixels are black (R, G, B values of each pixel is 0) and background pixels are white (R, G, B value of each pixel is 255). Object background separation is done in this way: “The majority of the four corner pixels correspond to background. So, by looking at the four corner pixels we can decide on which segment corresponds to the object”.

If we count those rows where the number of sequences of black pixels is more than three, we find that the number of such rows in dinosaur image is more than flower image. So we can fix a threshold and if we find that in an image, number of rows containing, number of sequences of black pixels is greater than or equal to three is above that threshold then we can put the image into the ‘Dinosaur’ class otherwise in the ‘Flower’ class. In this way, any image belongs to ‘Dinosaur’ and ‘Flower’ classes can be classified properly.

Now we need to separate the remaining six classes. Segmentation alone is not much useful for classifying any image in between these six classes. We have to go for some other features like shape. If we recall the discussion in section (1.1) about the characteristics of all these six classes, we can observe that the ‘Bus’ and ‘Building’ class images contain regular shapes like rectangle, etc., where as in the ‘Tribal’ class image we can go for face recognition. Not all these shape features can be extracted directly from the segmented image, so we go for edge detection.

We convert the color images into binary images by edge detection, where edge pixels are white and remaining pixels are black.

Various types of edge detection algorithms are available in the literature. All of them have their advantages and disadvantages. The early approaches to color edge detectors, which are extensions of achromatic color edge detectors failed to extract certain crucial information conveyed by color. The edge detection approaches, which simply add the gradient magnitudes of all color components, may fail to detect some crucial edge information in certain cases.

As our problem deals with hundreds of color images, we need an edge detection algorithm that can give uniformly acceptable results with a single set of parameter values. One such algorithm has been discussed in Sarif Kumar Naik *et al* [1]. We followed the algorithm given in this paper for edge detection.

Basic idea of the proposed method in the above paper:

The proposed method follows the philosophy stated in Rakesh *et al.* and uses a different technique for finding the initial edge magnitude and the direction. Rakesh *et al* [2] have found the estimated image surface using Priestly-Chao kernel smoother. Initial edge response and the direction of maximum contrast found using the directional derivatives along x-axis and y-axis of the estimated image surface. Then non-maxima suppression performed on the initial edge response. The standardized edge magnitudes at the pixels that not suppressed by non-maxima suppression are found using the variability of the estimated image surface and the directional derivatives. In the end a two level thresholding performed on the standardized edge magnitudes.

The above algorithm takes color image as its input, and gives edge-detected image as the output, where edge pixels are white and remaining pixels are black.

Figure 10. Edge detected images

In the ‘Bus’ class images, windows of the bus, and paintings on the bus are in rectangular. In the ‘Building’ class images, the pillars of the building are in cylindrical shape and windows of the buildings are rectangular. In the ‘Elephant’ class images, legs of the elephant are in cylindrical shapes. We do not have these kinds of shapes in other images. We can think of these shapes as a set of straight lines bounded with some geometrical property. Therefore, we search for straight lines in the edge image.

Various template-matching techniques exist in the literature to locate shapes in an image. As we have to deal with hundreds of images, we need a fast technique to extract the shapes from the edge images, which gives acceptable results. ‘Hough transform’ (HT) [14] is a well-known technique to extract straight lines, circles and ellipses in the image. Its prime advantage is that it can deliver the *same* result as that for template matching, but *faster*. This achieved by a reformulation of the template matching process, based on an *evidence gathering* approach where the evidence is the *votes* cast in an accumulator array.

The HT implementation defines a *mapping* from the image points into an accumulator space (Hough space). The mapping achieved in a computationally efficient manner, based on the function that describes the target shape. This mapping requires much less computational resources than template matching.

We use HT to extract straight line from the image. In this technique, we define a mapping from the Cartesian space to (ρ,θ) space, where the mapping is defined as,

$$x\cos\theta + y\sin\theta = \rho \qquad (2.1)$$

For each edge pixel in the edge image, we count the votes cast in the accumulator array, for all possible θ, ρ values.

We applied HT on the edge-detected image using a window-based approach to localize the evidence-gathering process. We used a 32x32-size window with fixed overlapping of 16 pixels.

Figure 11. Results of HT for lines on the edge detected images.

From the Hough transformed edge images, we can observe that in the ‘Building’ and ‘Bus’ class edge images we have more number of straight lines. In the ‘Elephant’ class edge images; we get less number of lines. Even in some ‘Elephant’ class images, as the legs of the elephant are covered by grass; we do not get much straight lines. Therefore, the elephant classification was not did using the number of straight lines. In other class images also there is very less possibility to get more number of straight lines, because the objects of the other classes do not have some such regular shapes in them. If the straight lines occurred in the edge image, they might have occurred due to noise in the image.

If there are noisy straight lines in the image, we expect them to be in the whole image. Hence, to get rid of the noisy lines in the Hough Transformed edge image, we do clustering of the lines using DB scan algorithm, with ‘r’ (distance), ‘k’ (minimum number of lines) as two parameters. Here, the similarity between the two lines is defined as, “Two lines are said to be within the distance ‘r’ if any of the end points of one line is within the Euclidean distance ‘r’ with any of the end points of other line”. In this way, not only we remove the noisy lines in other class edge images but also we remove the noisy lines from the edge images of the ‘Building’ and ‘Bus’ classes. After the above process we can fix a threshold on the number of lines detected in a Hough Transformed edge image to classify between {‘Bus’, ‘Building’} and {‘Elephant’, ‘Horse’, ‘Mountain’, ‘Tribe’}.

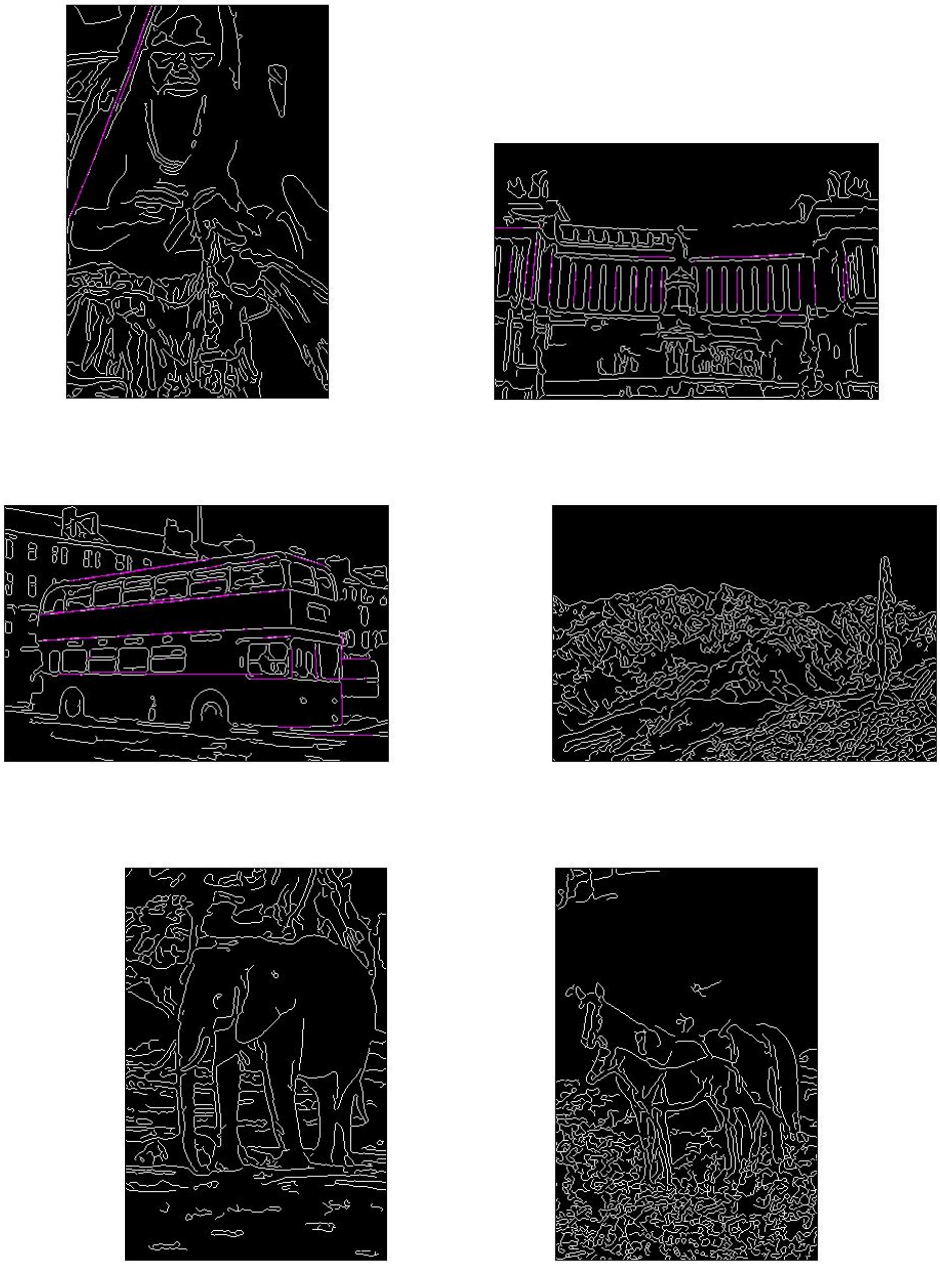


Figure 12. Hough Transformed (for lines) images after noisy lines removal.

Now, we need to classify an image between ‘Bus’ and ‘Building’ classes. In ‘Bus’ images we expect more number of horizontal lines, where as in ‘Building’ images we expect more number of vertical lines. Here we consider all lines which are having θ in between $[-\pi/18,\pi/18]$, as horizontal lines and in similar

way we consider all lines having θ in $[4\pi/9,\pi/2]$ and $[-\pi/2, -4\pi/9]$ as vertical lines. To classify between 'Bus' and 'Building' image classes, we count the number of vertical and horizontal lines in the Hough Transformed edge image. If the number vertical lines is more then we classify the image into 'Building' class and if the number of horizontal lines is more we classify the image into 'Bus' class.

Remaining four classes are 'Elephant', 'Horse', 'Mountain' and 'Tribal'. In the 'Tribal' class image, the face of the 'Tribal' is a good feature. Instead, we can try to extract eyes of the 'Tribal' and mouth of the 'Tribal' individually. In the 'Elephant' and 'Horse' images, lower part of the stomach portion, some times the upper part of the stomach portion (if detected as edge in the edge image) can be extracted. One common thing about the above shape features is that they are in elliptical shapes. Eyes and mouth of the 'Tribal' can be thought as an ellipse , where as the lower part of the stomach of 'Elephant' and 'Horse' can be thought of as 'an arc of a lower semi ellipse'. Even in 'Mountain' images, this shape (arc of a lower semi ellipse shape) observed on the edges of the mountain, also the place at which any two mountains overlap. Therefore, we extract this (arc of a lower semi ellipse) shape from the edge-detected image. A point to be noted here is, the size of this shape is different for different class images and the number of occurrences of this shape in an image may be different for different class images.

As we already discussed, HT is fast and it can be used to extract ellipse, we apply a modified version of 'Hough Transform for ellipse' to detect an arc of a lower semi ellipse. Here the mapping from the Cartesian space to polar space is derived as:

$$\left.\begin{array}{lll} a_0 = t_x & a_x = S_x \cos(\rho) & b_x = S_y \sin(\rho) \\ b_0 = t_y & a_y = -S_x \sin(\rho) & b_y = S_y \cos(\rho) \end{array}\right| \quad (2.3)$$

In the above equations, ρ represents the orientation, (S_x, S_y) a scale factor and (t_x, t_y) a translation.

$$x = a_0 + a_x \cos(\theta) + b_x \sin(\theta);$$
$$y = b_0 + a_y \cos(\theta) + b_y \sin(\theta); \quad (2.4)$$

In the above equations, (x,y) define the locus of the points on ellipse, where (a_0,b_0) defines the centre of the ellipse, θ is not a free parameter and it only addresses a particular point in the locus of the ellipse. Above equations are derived from the polar form representation of the circle. As ellipse is perspective projection of circle, using the perspective projection equations we can get the above representations.

In our problem, we do not need any rotation in the ellipse shape, which implies ρ = 0. (S_x, S_y) corresponds to lengths of the major and minor axes. There fore, the equations (2.4) are modified as,

$$x = a_0 + S_x \cos(\theta);$$
$$y = b_0 + S_y \sin(\theta); \quad (2.5)$$

We used above equations to detect an arc of lower semi ellipse shape in the edge image. Here also we have applied window-based method to localize the effect. Window size is 64x64. The implementation details and outline of the algorithm are discussed later.

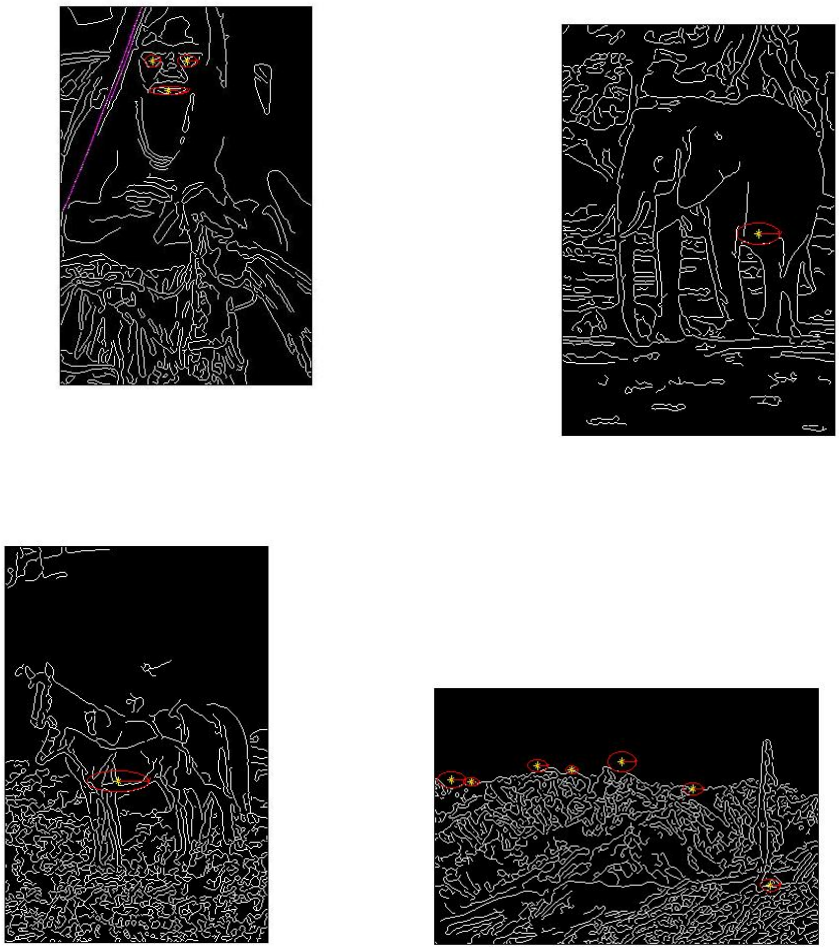


Figure 13. Results of HT for ellipse on the edge images after noisy line removal

We apply modified ‘HT for ellipse’ to detect the shape ‘arc of lower semi ellipse’. From the results, we can observe that, these shapes occur more in the ‘Mountain’ class images, where as less in all other three classes of images. There are some noisy shapes occurring in Hough Transformed images, which were not expected.

In the Hough transformed image we need to remove noisy shapes. One characteristic of the expected shapes is that the lower semi ellipse is fully contained in a connected component. This characteristic will be clarified when we

go through the segmented image of the ‘Horse’, ‘Elephant’ and other two classes of images. Hence, we define, an arc of lower semi ellipse shape is a noisy shape if, the lower semi ellipse is not fully contained in one connected component. Using this condition, we can remove all those noisy shapes that evolved from edges because of grass and other unimportant things in the edge image.

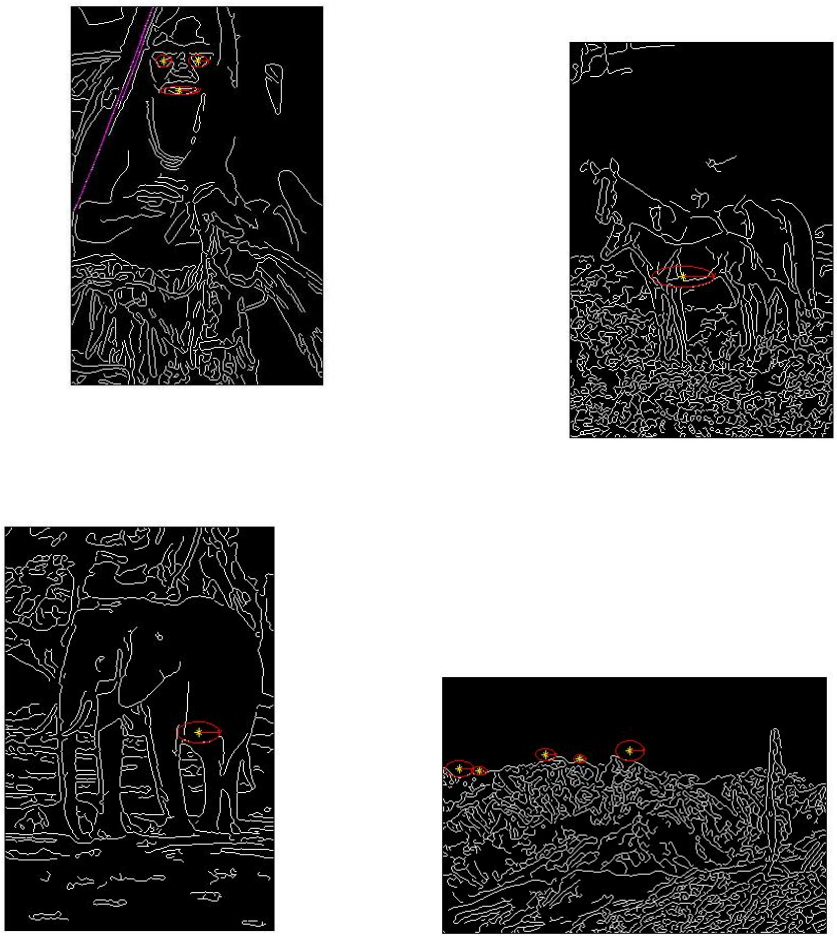


Figure 14. Results after noisy shape removal.

After removing such noisy shapes we can observe that arc of lower semi ellipse shape is more in the case of ‘Mountain’ image, where as this number is very less in the case of other three classes. In the ‘Tribe’ class images; we get only two eyes and mouth (lips) portion, as the chin shape removed, because it contains

the connected component related to mouth in the lower semi ellipse. After removing noisy shapes in the 'Elephant' and 'Horse' images, lower part of the stomach portion will be extracted, but upper part of the stomach portion may not be extracted because the background in these two class images is a collection of green leaves (forest), sky, rock, etc., which is non uniform in nature.

In the 'Tribal' class images, at most three such shapes gets extracted, where as in the 'Elephant' and 'Tribal' class images, at most four such shapes are extracted, because in an image there are at most two elephants or two horses. In the 'Mountain' class images, number of such shapes is more than four.

We can fix a threshold value five to separate the 'Mountain' class from the remaining four classes. In the mountain image, we can observe that, all detected shapes of the form arc of a lower semi ellipse are expected to lie on the upper edge of the mountain in the edge-detected image, i.e. all such shapes lie in one connected component that cover the sky portion of the image. Therefore, if an edge-detected image contain more than five such (arc of a lower semi ellipse) shapes and all such shapes lie in one connected component, then we can say that, image belongs to the 'Mountain' class and if all such shapes do not lie in one connected component then the image is classified as an 'error image'.

If an image does not contain more than four such shapes, the image may belong to any one of the three classes {'Elephant', 'Horse' and 'Tribe'}. If number of such (arc of a lower semi ellipse) shapes is three it is possible that the image belongs to 'Tribe' class.

In the 'Tribe' image, we can observe that, the reference points of three such shapes form an acute triangle with a line between two eyes as the base of the triangle. Let the lines joined in between the reference points of two eyes and the

mouth. Then, the difference between the angles made by other two lines with the base line is very small. Therefore, if an edge-detected image contains three such shapes and if the above property holds for that image, then that image belongs to the 'Tribe' class. If the above property does not hold, it may be possible that the image belongs to any one of the remaining classes.

In the elephant image, a common feature we can observe is 'tusk of the elephant'; therefore, we can use this feature. A tusk of the elephant shape is more complex than a line or ellipse. It is often possible to partition a complex shape into several geometric primitives, but this can lead to a highly complex data structure. In general, it is more convenient to extract the whole shape. 'Generalized Hough Transform' (GHT) is one such technique that can find *arbitrary* shapes using the evidence-gathering procedure of the HT. This technique again gives results equivalent to those delivered by matched template filtering, but with the computational advantage of the evidence gathering approach. Basic idea of this technique discussed later.

The problem with GHT is it is not invariant to orientation and scale, which causes the use of 4D-accumulator array to check for all possible scales and orientations. Invariant GHT is a technique, which takes care of above-mentioned problem. IGHT is invariant to scale and orientation; therefore a 2D accumulator array is enough for evidence gathering. Basic idea of this technique is discussed later.

To check whether an image belongs to the 'Elephant' class, we apply modified version of the 'Invariant Generalized Hough Transform for arbitrary shape' to extract 'tusk of the elephant' from the edge-detected image.

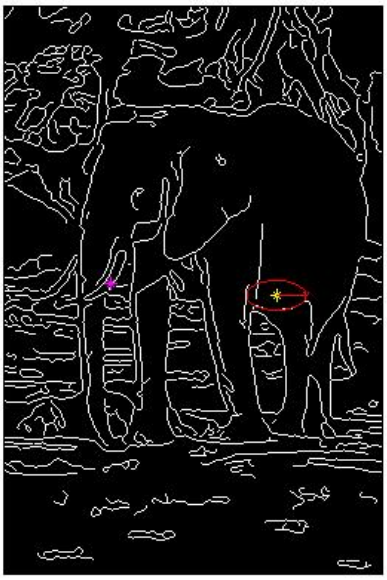


Figure 15. Result of IGHT for tusk on the image.

After applying the modified version of the 'Invariant Generalized Hough Transform' technique on the edge image, if we get at least one such (elephant tusk) shape, it may be possible that, image belongs to 'Elephant' class. We can check this by the property: "Whole elephant tusk will be detected as a connected component and the color of the connected component is close to white color". Therefore, if there is a shape that follows the above property in the edge-detected image, then that image belongs to the 'Elephant' class.

If an image that have less than five 'arc of lower semi ellipse' shapes and if it does not belong to the classes 'Tribe' and 'Elephant', it may be possible that the image belongs to the 'Horse' class.

In the 'Horse' class image, a common feature that we can observe is 'Mouth of the horse'. We can use this shape for classification. As the 'Mouth of the horse' shape is also complex, we apply modified IGHT to extract this shape

from the edge-detected image. As we already discussed, IGHT takes care of different sizes and orientations of the mouth in the edge-detected image.

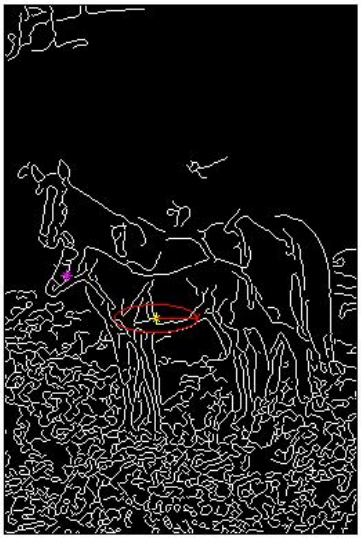


Figure 16. Result of IGHT for horse mouth on the edge image

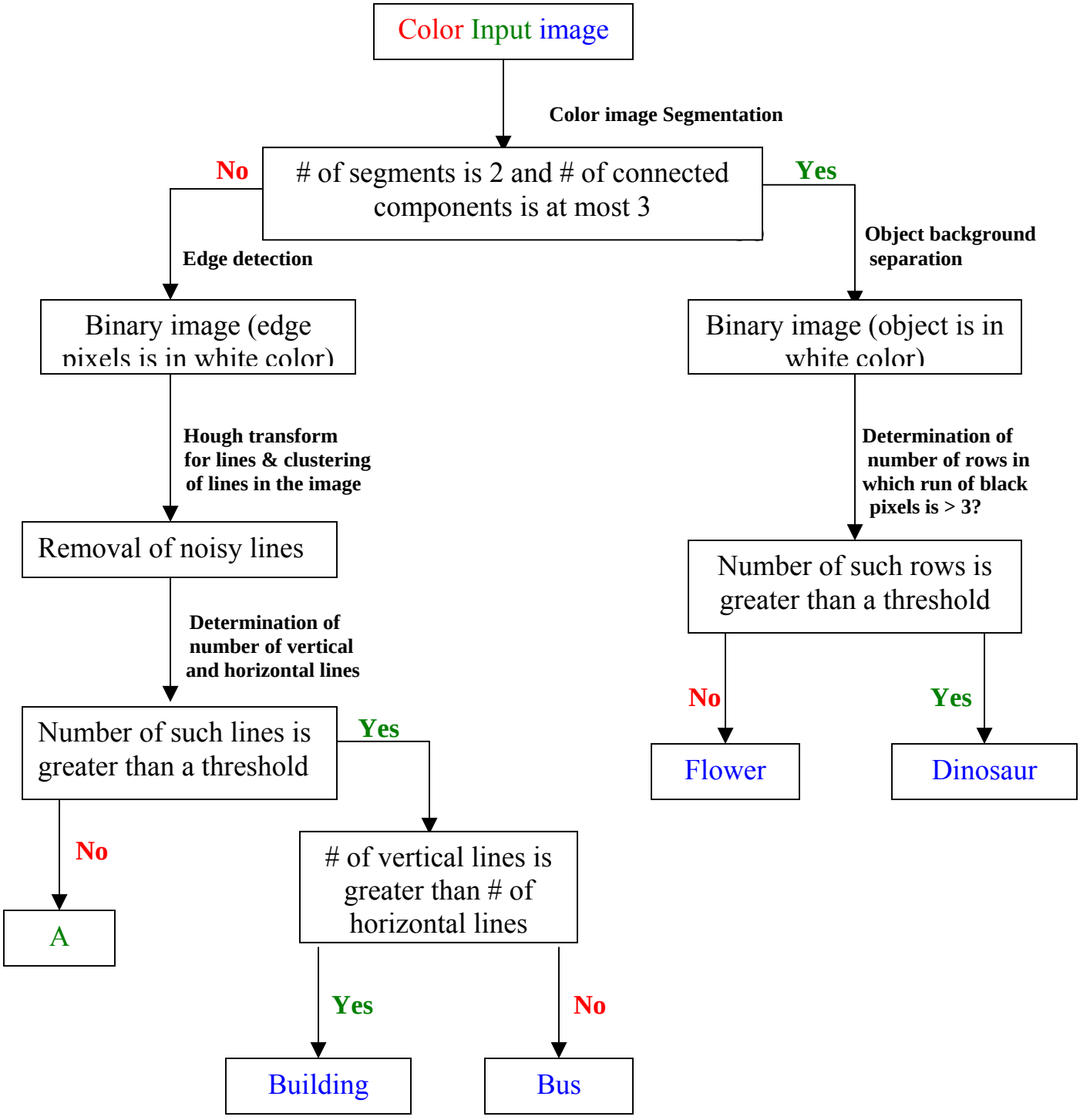
After applying the modified version of the 'Invariant Generalized Hough Transform' technique on the edge image, if we get at such (horse mouth) shape, the image belongs 'Horse' class.

If an image that have less than five 'arc of lower semi ellipse' shapes and if it does not belong to the classes 'Tribe', 'Elephant' and 'Horse', then the image is classified as an 'error image'.

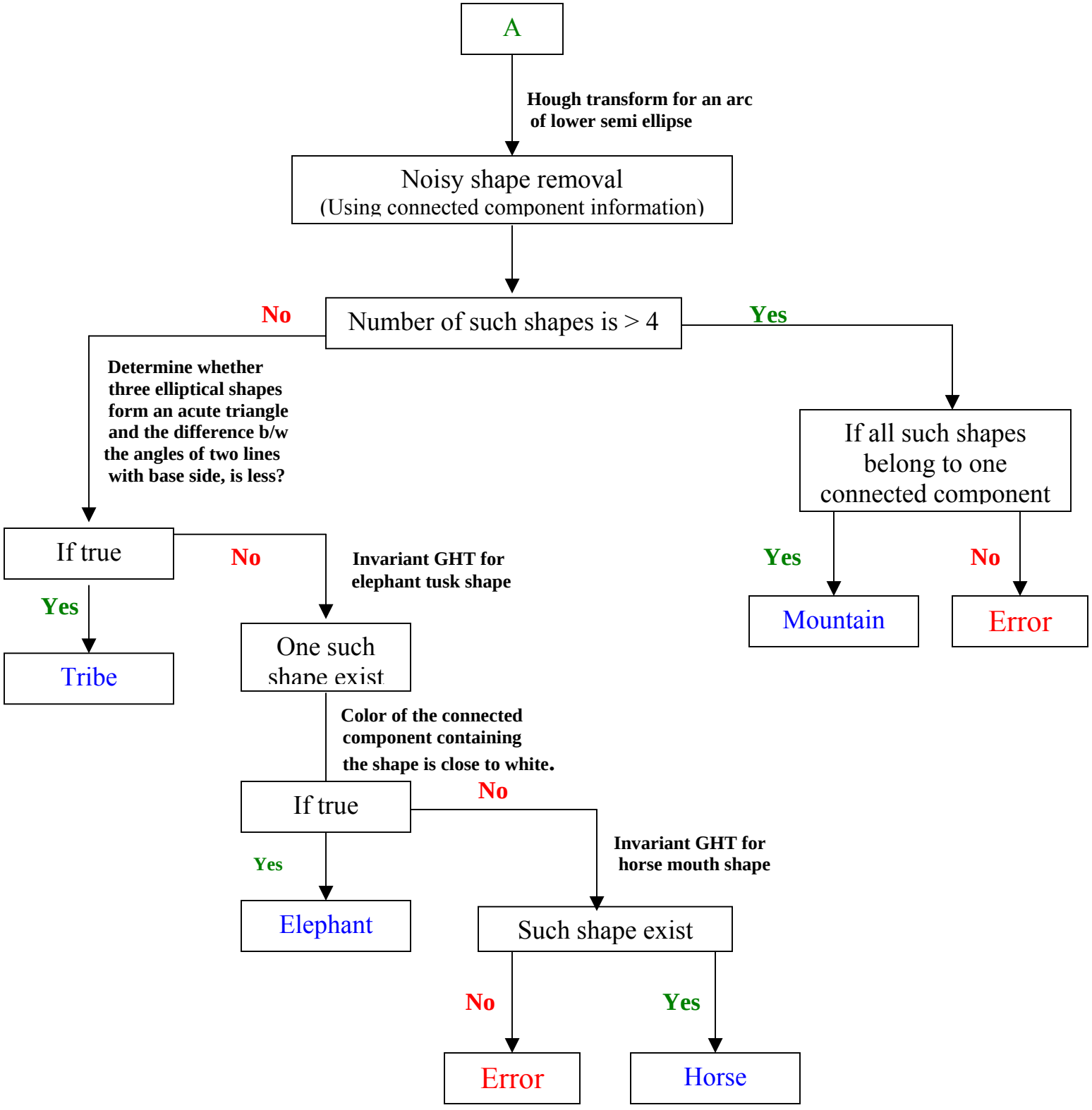
In this way, any given image from the above-mentioned eight classes can be classified into its corresponding class.

Flow chart pf the above algorithm given in the next page.

2.1 Flow chart of the proposed algorithm:



Flow chart of the proposed algorithm: (contd...)



III. Methodologies Applied

We applied some algorithms from the literature. Basic idea and outline of the algorithms are discussed here.

3.1 Color image segmentation:

Image segmentation, i.e., identification of homogeneous regions in the image, has been the subject of considerable research activity over the last three decades. Many algorithms been elaborated for gray scale images. However, the problem of segmentation for color images, which convey much more information about objects in scenes, has received much less attention of scientific community. In our problem, we applied modified version of 'DB scan' algorithm for segmentation.

Outline of the algorithm:

1. Let S be the set of distinct pixel vectors (R,G,B).
2. Let 'r' (radius) and 'k' (minimum number of points) be fixed.
3. For each element x_i in S,
    a) Define $A_i = \{x \mid d(x,x_i) < r \text{ and } x \text{ belongs to } S\}$.
    b) Let a_i be number of elements in A_i.
4. Define $B = \{i \mid a_i > k\}$.
5. Merge two sets A_i and A_j and name it A_i (if $i<j$), if
    a) A_i and A_j both have at least one common element.
    b) i, j both belong to set B.
6. Repeat the above step until no more merges possible.
7. Resultant sets $A_1 \ldots A_m$ are the segments in the image.

3.2 Connected components labeling:

After getting the segments, we need to find connected components in the segmented image. For each segment, we find connected components individually, for that, we make a binary image where the pixel value of a particular segment in the binary image is 1, all other pixel values are 0. In this way, we make a binary image corresponding to every segment in the image.

On this binary image, we apply connected components labeling algorithm to determine the connected components. Connected components labeling scans an image and groups its pixels into components based on pixel connectivity, *i.e.* all pixels in a connected component share similar pixel intensity values and are in some way connected with each other. Once all groups have been determined, each pixel is labeled with a color (color labeling) according to the component it was assigned to, for a group of pixels, mean of all the pixel vales from the original image is computed, and all the pixels in that group are labeled with this mean value . In this process, we removed those connected components whose size is less than a threshold.

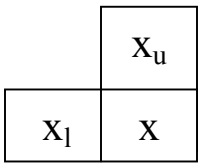


Fig 2.1 mask for detecting connected components.

Outline of the algorithm:

1) Let M denote the binary image corresponding to a particular segment.
2) Let compCt denote the counter for component.
3) Let N be a matrix of image size, denote the resultant groupings of the connected components after the algorithm complete.
4) Initialize N with a zero matrix.

5) For each nonzero pixel in M i.e. M[x] != 0, from the mask in Fig 2.1,
    a) If, $M[x_u] = 0$ and $M[x_l] = 0$, then increase compCt by one and assign it to N[x].
    b) If $M[x_u] = 0$ and $M[x_l] != 0$ then $N[x] = N[x_l]$.
       Else if $M[x_u] != 0$ and $M[x_l] = 0$ then $N[x] = N[x_u]$.
    c) If $M[x_u] != 0$ and $M[x_l] != 0$, then
       $N[x] = \min(N[x_l], N[x_u])$, say $N[x_l] < N[x_u]$, replace $N[x_u]$ with $N[x_l]$ in the whole N.
6) After completion of above step we have connected components with similar numbers.
7) We can compute the mean of each connected component and label each pixel in that connected component with that label.

3.3 Edge detection:

Various types of edge detection algorithm are available. All of them have their advantages and disadvantages. The early approaches to color edge detectors, which are extensions of achromatic color edge detectors failed to extract certain crucial information conveyed by color. The edge detection approaches, which simply add the gradient magnitudes of all color components, may fail to detect some crucial edge information in certain cases.

As our problem deals with hundreds of color images, we need an edge detection algorithm, which can give uniformly acceptable results with a single set of parameter values. One such algorithm discussed in Sarif Kumar Naik *et al* [1]. We followed the algorithm given in this paper for edge detection.
Method proposed in the above paper:

The proposed method follows the philosophy stated in Rakesh *et al.* and uses a different technique for finding the initial edge magnitude and the direction. Rakesh *et al.* have found the estimated image surface using Priestly-Chao kernel smoother. Initial edge response and the direction of maximum contrast are found using the directional derivatives along x-axis and y-axis of the estimated image surface. Then non-maxima suppression is performed on the initial edge response. The standardized edge magnitude at the pixels which are not suppressed by non-maxima suppression are found using the variability of the estimated image surface and the directional derivatives. In the end a two level thresholding is performed on the standardized edge magnitudes.

In the proposed method Priestly-Chao kernel smoother is used to estimated the images surface in each component of the of the color image separately i.e.

$f^c(x,y)=\Sigma_{i=1}^{m}\Sigma_{j=1}^{n} K(x,i).K(y,j)I^c(i,j)/\xi$

where $\xi=2\Pi mnh^2$, $K(a,b)=\Psi((a-b)/h)$, $\Psi(x)=e^{-x2}$, c = R,G,B.

Where I^c (i, j) is the gray value at (i, j)th pixel of the original image for the component c, h is the stiffness parameter, m and n are the number of rows and columns respectively.

The directional derivatives along direction of the estimated image surface are:

$f^c_x(x,y)=\Sigma_{j=1}^{n} K(y,j)\{\Sigma_{j=1}^{m} K^{/}(x,i)I^c(i,j)\}/\xi$

$f^c_y(x,y)=\Sigma_{i=1}^{m} K(x,i)\{\Sigma_{j=1}^{n} K^{/}(y,j)I^c(i,j)\}/\xi$

Where $K^{/}$ (x,i) and $K^{/}$(y,j) are the derivatives of K(x,i) and K(y,j) respectively. The estimate of data variability of the estimated image fc (i, j) is given by

$(\sigma^c)^2=\Sigma_{i=1}^{m}\Sigma_{j=1}^{n} (I^c(i,j)-f^c(i,j))^2$

The edge magnitude λ at a pixel (x,y) is given by the largest Eigen value of the 2x2 matrix,

[E F ; F G]

and the direction of maximum contrast is given by the corresponding eigenvector. In above expression

$E=f_x^R f_x^R + f_x^G f_x^G + f_x^B f_x^B$

$F=f_x^G f_y^G + f_x^G f_y^G + f_x^B f_y^B$

$G=f_y^R f_y^R + f_y^G f_y^G + f_y^B f_y^B$

Having the information about edge magnitude λ and the direction of maximum contrast at each pixel edge location can be found precisely by using non-maxima suppression. The problem occurs in thresholding at the time of hysteresis .The edge magnitude found is generally not uniform for both low intensity and high intensity regions in an image. Thus, the edge detectors are unable to extract, simultaneously, all the edges in an image having edges of different intensities. They can't all the edges in different images with a fixed set of threshold values. To get those edges, threshold values have to be tuned carefully for individual images. To overcome this problem this article proposed a technique to obtain standardized edge magnitudes. The standardization of edge magnitude is done using the variability of the partial derivatives and estimated image surface.

Let us define the derivative vector for each components of the smooth image f at a pixel (x, y) to be $v^c=(f_x^c, f_y^c)^T$ and the statistic S(x,y) as

$S(x, y)=\Sigma_{c=R,G,B} (v^c)^T (\Sigma^c)^{-1} (v^c)$

$\Sigma^c=(\sigma_{11}^c(x, y),\sigma_{12}^c(x, y);\sigma_{12}^c(x, y),\sigma_{22}^c(x, y))$

Where $\sigma_{11}^c(x, y)=(\sigma^c/\xi)^2 \Sigma_{i=1}^{m}\Sigma_{j=1}^{n} K^2(y, j)K^{/2}(x, i)$

$\sigma_{22}^c(x, y)=(\sigma^c/\xi)^2 \Sigma_{i=1}^{m}\Sigma_{j=1}^{n} K^2(x, i)K^{/2}(y, j)$

$\sigma_{12}^{c}(x, y)=(\sigma^{c}/\xi)^{2} \Sigma_{i=1}^{m} \Sigma_{j=1}^{n} K(y, j)K^{/}(x, i)K(x,i)K^{/}(y, j)$

In the above expression c takes value of R, G, B and S (x, y) is the standardized edge magnitude at pixel (x, y). The value of S is used as the standardized edge response and is thresholded using two threshold values.

Outline of the algorithm:

1) Let h=1.25(stiffness of Gaussian), T_u =30(upper threshold) and T_l =5(lower threshold).
2) Find the directional derivatives f_x^R, f_y^R, f_x^G, f_y^G, f_x^B, f_y^B for each pixel.
3) Find initial magnitude (λ) and direction of maximum contrast (eigen vector corresponding to λ).
4) For each pixels (i, j),
   a) Apply non-maxima suppression
   b) If it is not suppressed find S
   c) If $S>T_u$ Edge class(i, j) = 2 (edge pixel).
      Else if $S>T_l$ Edge class (i, j)=1(edge pixel if any of its 8-neighbor is an edge pixel).
      Else Edge class (i ,j)=0 (not an edge pixel).
5) Initialize an array Edge (i, j) of the size of the input image to zero.
6) For each pixel (i, j) if Edge class (i, j) =2 perform Track_edge (i,j).

3.4 Hough Transform:

The *Hough Transform* (HT) (Hough, 1962) is a technique that locates shapes in images. In particular, it has been used to extract *lines*, *circles* and *ellipses* (or conic sections).

Its prime advantage is that it can deliver the *same* result as that for template matching, but *faster.* This is achieved by a reformulation of the template matching process, based on an *evidence gathering* approach where the evidence is the *votes* cast in an accumulator array. The HT implementation defines a *mapping* from the image points into an accumulator space (Hough space). The mapping is achieved in a computationally efficient manner, based on the function that describes the target shape. This mapping requires much less computational resources than template matching.

In our problem, as we have to deal with so many images, and HT technique is faster than other template matching techniques we have chosen HT for template matching. We have used HT with some modifications to find line, and lower semi ellipse shapes.

3.4.1 HT for lines:

We have used *polar HT for lines*. In this method a line is represented in form:

$$x\cos\theta + y\sin\theta = \rho \qquad (1)$$

We have used a window based approach to localize the effect. In our problem, we have used a window of size 32x32 with fixed overlapping. We applied HT on these windows of edge-detected image, to find the local maximum in the accumulator array. If this local maximum is greater than a threshold, we consider that line for that particular θ, ρ.

Implementation:

1) Hough transform algorithm for lines uses a 2D array called accumulator 'acc' to detect the existence of a line. The two dimensions of the accumulator array would correspond to quantized values for θ and ρ.
2) Let θ take values of $\theta_1, \theta_2, \theta_3, \ldots, \theta_n$, and let ρ takes values of $\rho_1, \rho_2, \rho_3, \ldots, \rho_n$.
3) We consider a window of size 32x32 from the edge image.
4) For each edge pixel M(x,y) (M is the edge image) in the above window, for each possible θ_i we compute ρ_i value using the equation (1), and increase the entry $acc(\theta_i,\rho_i)$.
5) After doing the above process for all edge pixels in the window, we find in accumulator array for which θ and ρ values maximum occurs.
6) If the above, maximum is greater than a threshold we consider the θ and ρ to form a line.
7) From step (4) to step (6) we repeat the process for all possible windows with fixed overlapping.

The number of subdivision in the (ρ, θ) plane determines the accuracy of the co-linearity of the points. Let T be a threshold value.

3.4.2 HT for ellipse:

We have applied modified version of HT for ellipse to get shape of the form arc of a lower semi ellipse.

Here also we have applied window-based approach to localize the effect. In our problem, we have used a window of size 64x64 with fixed overlapping. But

the problem here is with the size of the accumulator array. This will be understood from the parameterization ellipse in terms of edge pixel.

An ellipse at the origin and parallel to the axes is characterized by an equation

$$(x/a)^2 + (y/b)^2 = 1$$

In general, we need to consider the rotated ellipse:

$$((x\,\mathrm{Cos}\phi - y\,\mathrm{Sin}\phi)/a)^2 + ((x\,\mathrm{Sin}\phi + y\,\mathrm{Cos}\phi)/b)^2 = 1$$

and we need the ellipse to be at any position in the image [xc, yc]:

$$(((x-xc)\mathrm{Cos}\phi - (y-yc)\mathrm{Sin}\phi)/a)^2 + (((x-xc)\mathrm{Sin}\phi + (y-yc)\mathrm{Cos}\phi)/b)^2 = 1$$

So we can deduce that our parameter space requires five parameters to be identified $\{xc,yc,a,b,\phi\}$.

As we can observe from the above discussion we need 4-D accumulator array to solve the problem. Here ϕ is not a free parameter, it is used to trace the boundary.

As we are searching for an arc of a lower semi ellipse shape, domain for ϕ will reduced half. Even with some quantization we reduce the number values that ϕ takes. In our problem, ‘a’ (major axis) can take values from 5 to 50; where as ‘b’ (minor axis) can take values from 5 to 25; Even if the dimension of the accumulator array is more, for our problem the technique performed well.

3.4.3 Invariant Generalized Hough Transform for arbitrary shapes:

Many shapes are far more complex than lines, circles or ellipses. It is often possible to partition a complex shape into several geometric primitives, but this can lead to a highly complex data structure. In general, it is more convenient to extract the whole shape. This has motivated the development of techniques that can find *arbitrary* shapes using the evidence-gathering procedure of the HT.

In our problem, elephant tusk shape and horse mouth shape are not easy to find. We cannot easily parameterize them as we done for line, ellipse. Hence, we have chosen Invariant Generalized HT (IGHT) to detect these shapes. Before going to discuss about IGHT, we discuss here some preliminaries about GHT (Generalized Hough Transform).

Generalized HT:

The GHT can be formally defined by considering the duality of a curve. One possible implementation can be based on the discrete representation given by tabular functions. GHT can be used to locate arbitrary shapes with unknown *position*, *size* and *orientation*.

Formal definition:

Any model shape can be represented by following a more complex definition of $x(\theta)$ and $y(\theta)$.

$$v(\theta) = x(\theta)\begin{bmatrix} 1 \\ 0 \end{bmatrix} + y(\theta)\begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad (1)$$

The shape in the image can be defined as

$$\omega(\theta, b, \lambda, \rho) = b + \lambda \mathbf{R}(\rho)\upsilon(\theta) \quad (2)$$

where $b = (x0, y0)$ is the translation vector λ is a scale factor and $\mathbf{R}(\rho)$ is a rotation matrix. The shape of $\omega(\theta, b, \lambda, \rho)$ depends on four parameters. Two parameters define the location b, plus the rotation and scale parameters. It is important to notice that θ does not define a free parameter, but only traces the curve.

In order to define a mapping for the HT, the location of the shape is given by

$$b = \omega(\theta) - \lambda \mathbf{R}(\rho)\upsilon(\theta) \quad (3)$$

Given a shape $\omega(\theta)$ and a set of parameters b, λ and ρ , this equation defines the location of the shape.

In the GHT the gathering process is performed by adding an extra constraint to the system that allows us to match points in the image with points in the model shape. This constraint is based on gradient direction. Constraint: "The Equation 3 is true only if the gradient direction at a point in the image matches the (rotated) gradient direction at a point in the (rotated) model".

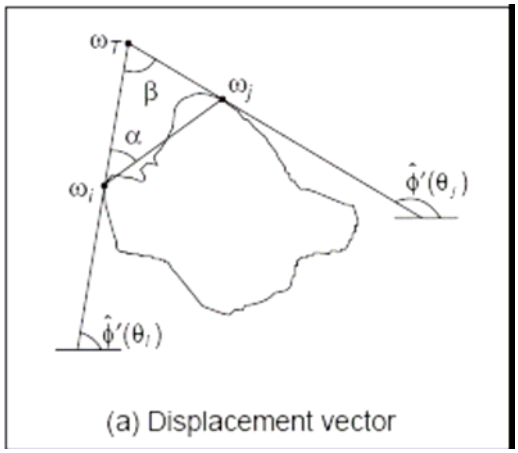
GHT technique uses a model of the shape to be detected. Using this model, first it computes the gradient direction at each of the boundary (edge) point. Using the gradient direction, it will compute $v(\theta)$, using that we get the displacement vector. GHT pre-computes and stores these values in an array called *R-table*. The R-table stores displacement vector for each value of gradient angle.

The problem with GHT is computationally it is very complex, as we are considering all possible orientations and size.

Invariant GHT:

In order to reduce computational complexity of the GHT, we can consider replacing the gradient direction by another feature. That is, by a feature that is not affected by *rotation*, hence instead of searching for a point with the same gradient direction, we will search for the point with the same invariant feature. The advantage is that this feature will not change with rotation or scale, so we only require a 2D space to locate the shape.

One such feature is: "An angle (β) is defined by three points and its value remains unchanged when it is rotated and scaled". Thus, if we associate to each edge point ω_i a set of other two points
$\{\omega_j, \omega_T\}$ then we can compute a geometric feature that is invariant to similarity transformations. Above relation is shown in the below figure.



(a) Displacement vector

An alternative geometric arrangement is: Given the point ω_i and a fixed angle α, then we determine the point ω_j such that the angle between the tangent line at ω_j and the line that joins the points is α. The third point is defined by the intersection of the tangent lines at ω_i and ω_j.

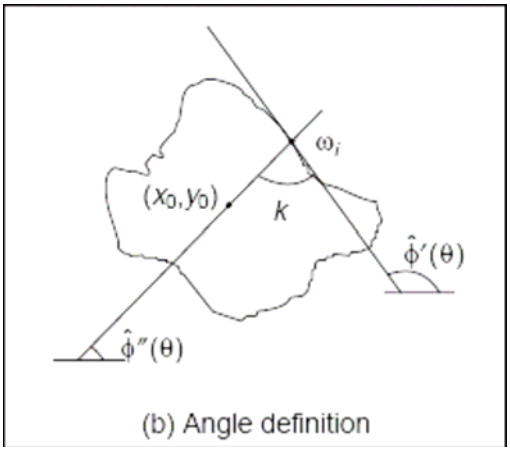
The tangent of the angle β (Q(ω_i)) can be expressed in terms of the points and its gradient directions as

$$Q(\omega_i) = \frac{\phi_i' - \phi_j'}{1 + \phi_i' \phi_j'}$$

where ϕ'_i is the tangent of the angle of the first directional derivative.

We can replace the gradient angle in the *R-table* by the angle β. Since the angle β does not change with rotation or change of scale, then we do not need to change the index for each potential rotation and scale. However, the displacement vector changes according to rotation and scale. Thus, if we want an invariant formulation, then we must also change the definition of the position vector. For this, we proceed in the following way. Above equation (3) can be developed as,

$$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} \omega_{xi} \\ \omega_{yi} \end{bmatrix} + \lambda \begin{bmatrix} \cos(\rho) & \sin(\rho) \\ -\sin(\rho) & \cos(\rho) \end{bmatrix} \begin{bmatrix} x(\theta) \\ y(\theta) \end{bmatrix}$$



(b) Angle definition

From the above figure ϕ''_i can be expressed as,

$$\phi_i'' = \frac{\omega_{yi} - y_0}{\omega_{xi} - x_0} = \frac{[-\sin(\rho) \quad \cos(\rho)]\, y(\theta)}{[\cos(\rho) \quad \sin(\rho)]\, x(\theta)} \Bigg| \qquad (4)$$

By some algebraic manipulations, we have that

$$\phi_i'' = \tan(\xi - \rho) \qquad (5)$$

where

$$\xi = \frac{y(\theta)}{x(\theta)} \Bigg|$$

In order to define ϕ''_i we can consider the tangent angle at the point ω_i. By considering the derivative of Equation (2) we have that

$$\phi_i' = \frac{[-\sin(\rho) \quad \cos(\rho)]\, y'(\theta)}{[\cos(\rho) \quad \sin(\rho)]\, x'(\theta)}$$

Thus

$$\phi_i' = \tan(\phi - \rho)$$

where

$$\phi = \frac{y'(\theta)}{x'(\theta)} \Bigg|$$

By considering equation (5), we define

$$\hat{\phi}_i'' = k + \hat{\phi}_i' \Big| \qquad (6)$$

The important point in this definition is that the value of *k* is invariant to rotation. Thus, if we use this value in combination with the tangent at a point we can have an invariant characterization. In order to see that *k* is invariant, we solve the above equation. Thus,

$$k = \phi - \rho - (\xi - \rho)$$

That is,

$$k = \phi - \xi \qquad (7)$$

That is, independent of rotation.

In order to obtain an invariant GHT, it is necessary to know for each point ω_i, the corresponding point v(θ) and then compute the value of ϕ''_i. Then evidence can be gathered by the line equation in (4). That is,

$$y_0 = \phi_i''(x_0 - \omega_{xi}) + \omega_{yi} \Big| \qquad (8)$$

In order to compute ϕ''_i we can obtain *k* and then use Equation (7). In the standard tabular form, the value of *k* can be pre computed and stored as function of the angle β.

In our problem, we designed model image for 'elephant teeth' and 'horse mouth'. Using this model, first *invariant R-table* computed and stored. Using the above *invariant R-table,* for any given image evidence is gathered by taking a 2D accumulator array. As IGHT is invariant to scale and orientation, we need only 2D accumulator array.

Invariant R-table is computed by fixing the angle α to π/4 and each element of the table stores a single value computed according to Equation (7). The more cumbersome part of the computation is to search for the point ω_j. We search in two directions from ω_i and we stop once an edge point has been located. This search is performed by tracing a line. The trace is dependent on the slope. When the slope is between –1 and +1 we then determine a value of *y* for each value of *x*, otherwise we determine a value of *x* for each value of *y*.

We have done evidence-gathering process according to equation (8). We use the value of β to index the table passed as a parameter to the function *InvGHT*.

The data k recovered from the table is used to compute the slope of the angle defined in Equation (6). This is the slope of the line of votes traced in the accumulators.

We applied, above algorithms in our problem.

IV Experimental Results:

In the chapter II, we have explained the result after applying each step of the proposed method. Here, we discuss about misclassification details.

We considered total 600 images from the database. All these images belong to eight classes. The number of images in each class is given below.

Class Name	Number of Images
Dinosaur	100
Flower	100
Bus	100
Building	70
Mountain	15
Tribal	15
Horse	100
Elephant	100

The error rates corresponding to each of the classes are given below:

Percentage of misclassification for 'Dinosaur' = 2%

Percentage of misclassification for 'Flower' = 7 %

Percentage of misclassification for 'Bus' = 3%

Percentage of misclassification for 'Building' = 5.71%

Percentage of misclassification for 'Elephants' = 8%

Percentage of misclassification for 'Horse' = 9%

Percentage of misclassification for 'Mountain' = 6.66%

Percentage of misclassification for 'Tribal' = 13.2%

Total percentage of misclassification = 6%.

5.1 Reasons for misclassification:

In the ‘Dinosaur’ class, in two images the legs are overlapped with each other, and thus only one connected component is obtained, which results in a misclassification. In the ‘Flower’ class, in some images there were more than two flowers, and three connected components resulted in some others. Some images in ‘Bus’ class are misclassified because of the presence of noise in the background. Some images in ‘Building’ class are too small for detection. Additionally, noise is observed because of the presence of human beings and animals.

In some of the ‘Elephant’ lower part of the stomach portion of the elephant is not visible, tusk of the elephant is not visible in some other. In some of the ‘Horse’ class images, lower part of the stomach portion is not visible in the image, in some other noise in the image affected the results. In one of the ‘Mountain’ class images segmentation resulted in two connected components, therefore it is misclassified. In the ‘Tribal’ class image because of ornaments that he/she wears, the misclassification has been occurred.

V Conclusion & Scope of Future work:

The system is tested with a collection of images from the database consisting images of buses, buildings, flowers, dinosaurs, mountains, horses, elephants and tribal. We have observed that the above classes can be classified with good rate of accuracy.

In our present work, shape features are given more weight against color and texture features. The color features are given more important for the 'Dinosaur' and 'Elephant' classes, shape feature are given more important for all other classes. The texture features are given more important for the 'Bus' and 'Building' class.

Scope of feature work:

1) We have considered less number of images from the 'Mountain' and 'Tribal' classes and we did not consider the 'Beach' and 'Food items' for classification.
2) In some classes, misclassification occurred because of the orientation of the object in the image, also because of the grasses and the trees. One can try to remove the noise contents from the image, by which misclassification rate will be reduced.
3) In most of the classes, number of the objects in an image is less than three. One can try for an algorithm that is invariant to number of objects in the image.

VI. References

1) Sarif Kumar Naik and C. A. Murthy, ``Standardization of edge magnitude in color images", *IEEE Trans. on Image Processing*, , vol. 15, no. 9, pp. 2588-2595, September, 2006.

2) R. R. Rakesh, P. Chaudhari and C. A. Murthy, ``Thresholding in edge detection a statistical approach", *IEEE Trans. on Image Processing*, vol. 13, no. 7, pp. 927-936, July, 2004.

3) James Z. Wang, Jia Li, Gio Wiederhold, ``SIMPLIcity: Semantics-sensitive Integrated Matching for Picture Libraries," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 23, no.9, pp. 947-963, 2001.

4) C.S. McCamy, H. Marcus, and J.G. Davidson, ``A color-rendition chart'', *Journal of Applied Photographic Engineering, 2*(3), 1976.

5) Fuhui Long, Hongjiang Zhang and David Dagan Feng, "*Multimedia Information Retrieval and Management"*, Springer – Verlag, Berlin Heidelberg, New York, 2003.

6) E. Persoon and K. Fu, "Shape discrimination using Fourier descriptors", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, pp. 170-179, 1977.

7) M.K.Hu, "Visual pattern recognition by moment invariants", in *Computer Methods in Image Analysis*, Los Angeles: IEEE Computer Society, 1977.

8) A. Pentland, R. Picard, and S. Sclaroff, "Photobook: Content-Based Image manipulation of image databases", *International Journal of Computer Vision,* vol. 18, pp. 233-254. 1996.

9) A. Gupta and R. Jain, "Visual Information Retrieval", C*ommunications of the ACM,* vol. 4, pp 12-20, 1997.

10) Logothetis, N. & Sheinberg, D., ``Visual object recognition'', *Annu. Rev.Neurosci*., vol. 19, pp. 577–621 (1996).

11) Ullman, S. *High-Level Vision: Object Recognition and Visual Cognition,* MIT Press, Cambridge, Massachusetts, 1996.

12) R.F. Ling, "Probability theory of cluster analysis", *Journal of American Statistical Association*, vol. 68, pp. 159-164, 1973.

13) Ester M., Kriegel H. P., Sander J., Xu X.: "*A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise".* KDD'96, pp. 226-231.

14) Aguado A.S., Montiel, M.E., and Nixon M.S., 1995, "Ellipse Detection via gradient direction in the Hough transform", *Proc. IEE International Conference on Image Processing and its Applications*, July, pp 375-378.

15) Aguado A.S., Montiel, M.E., and Nixon M.S., "*Feature extraction and image processing*", Newnes publications, 2002.