

BIRD RECOGNITION USING IMAGE PROCESSING

Submitted towards the partial fulfilment of requirement
for the award of degree of
Bachelor of Engineering
In
Computer Engineering

By

Shreyas Kumari
Varsha Shankar

2K5/COE/144
2K5/COE/154

Under the Guidance of

Dr. Rajiv Kapoor
Assistant Professor

Department of Electronics and Communication Engineering



DEPARTMENT OF COMPUTER ENGINEERING
DELHI COLLEGE OF ENGINEERING
UNIVERSITY OF DELHI
2009

ACKNOWLEDGEMENT

We wish to express our deep gratitude to Dr. Rajiv Kapoor for the guidance, endless encouragement, enthusiasm and support that he provided throughout the duration of this project.

We would also like to extend our sincere thanks to Dr. Daya Gupta; Head of Department, Computer Engineering, Mrs. Rajni Jindal; Lecturer, Department of Computer Engineering for their support and encouragement.

CERTIFICATE

It is hereby declared that the work that is being presented in this project entitled '***BIRD RECOGNITION USING IMAGE PROCESSING***' by the undersigned students of final year B.E. Computer Engineering, Delhi College of Engineering is an authentic record of their original work at the end of the academic year 2008-2009 performed to satisfaction under the guidance of Assistant Professor Dr. Rajiv Kapoor.

Shreyas Kumari
2K5/COE/144
Final Year B.E. COE
Dept. of Computer Engineering

Varsha Shankar
2K5/COE/154
Final Year B.E. COE
Dept. of Computer Engineering

Dr. Rajiv Kapoor
Assistant Professor
Department of Electronics and Communication
Delhi College of Engineering
University of Delhi
Delhi -110042

Table of Contents

Chapter 1 Introduction.....	5
Chapter 2 Literature Survey.....	6
2.1 Colour-based Indexing.....	6
2.2 Colour histogram based Indexing.....	7
2.3 Indexing in Flower databases.....	7
Chapter 3 Objective.....	8
Chapter 4 Design and Methodology.....	9
4.1 Background removal.....	9
4.2 Discrete Wavelet Transform for edge detection.....	12
4.3 Noise Removal.....	19
4.4 Convex hulling.....	20
4.5 Connected component analysis.....	26
4.6 Morphological closing operation.....	27
4.7 Picking up frames from videos.....	29
4.8 Template matching	29
4.9 Deskewing	37
4.10 Clustering and calculation of inter cluster distances.....	42
4.11 Iterative Closest Point (ICP).....	44
4.12 Dempster Shafer Theory	50
Chapter 5 Results.....	54
Chapter 6 Conclusion and Future Scope.....	57
Chapter 7 References.....	58
Chapter 8 List of Figures.....	60
Chapter 9 List of Tables.....	61

Chapter 1 Introduction

With the increasing encroachment of man into the natural habitat of birds and animals, their existence has been placed in danger; requiring conservationists to keep constant track of their movements. Maintaining such detailed records is time consuming and tedious if done manually. An easier way is to install cameras that can monitor their movement for extended periods of time and transmit the recorded data to a processing station. The flight patterns of migratory birds can be observed by installing noninvasive cameras. Seasonal variations in bird density and species can be studied as this reveals patterns in environmental changes and human influences.

The use of non-invasive camera surveys provides baseline information on distribution, activity, and habitat associations can be obtained. A wealth of knowledge regarding habits, characteristics, routines, food habits, social setup and safety of the species can also be gathered. At processing stations, the data may be analyzed to gather this knowledge. The first step in such processing is to know which species the data is relevant to. It is our belief that it is possible to automate this crucial first phase. We aim to prove this by way of implementing bird recognition in this seminal work.

With the massive amount of data being stored in the form images, it has become necessary to develop an efficient method for the retrieval of images based on the content. Particularizing the application of content based image retrieval to one specific field would help us study the same in a more efficient manner.

Chapter 2 Literature Survey

2.1 Colour-based Indexing

Research in this area mainly involved indexing the images based on the colour distribution of the object to be recognised as exemplified by [1] wherein images were indexed in domain specific databases using colours computed from the object of interest only (in this case, the bird), instead of the whole image. The challenge in the task was the segmentation of the bird from the background. Possible object colours were found by first finding background colours and eliminating them. Pixels are marked as background pixels by successively counting the maximum number of pixels with a specific colour around the margin of the image. After eliminating the marked background pixels the most prominent objects are selected by connected component analysis. This is followed by extracting the largest of all the components. This process is repeated with some other colour as background if we are not able to retrieve the bird as the most prominent object. Edges are detected based on colour gradient. The edge image is then combined with the foreground segment output by the colour-based background elimination process. The first step in the combination process is to eliminate edge points that are not in the foreground segment. This eliminates most of the edges from the background. The edge image at appropriate scale was used to eliminate parts of the image that were not in focus and did not contain significant structures. The edge information was combined with the colour based background elimination to produce regions of interest. Indexing the image is done based on the colour of the image. The colour histogram of the object that is queried is matched with the object having similar colour distribution in the database.

In the above method, the segmentation technique assumes that the bird remains in the centre region and occupies a significant region of the image. Due to the nature of the domain, there are many images of birds, however, include those in flight, very small birds, images taken from afar etc. Also, there are instances where the bird's colour is not as significant as shape and structure. In such cases, texture based indexing would be more fruitful than colour based indexing.

As the method for recognition is based on colour and texture it becomes necessary to first separate the object of interest from the rest of the image. Segmentation is a hard problem and it would be very difficult to detect objects which are as varied in color, shape, size and viewpoint as pictures of birds. It becomes difficult with birds because in most cases the birds are camouflaged with the background making it difficult to distinguish. A number of segmentation techniques have been proposed most common being background subtraction.

2.2 Colour histogram based Indexing

Retrieval based on colour histogram has been widely used in the field of content based image retrieval. Their advantages are efficiency and insensitivity to small changes in camera viewpoint. A histogram is a coarse characterization of an image, and so images with different appearances can also have similar histograms. Several improvements of this method have been proposed, histogram refinement being one [2]. Histogram refinement splits the pixels in a given bucket into several classes, based upon some local property. Within a given bucket, only pixels in the same class are compared. The method made use of till now is called global colour histogram (GCH) as colour histogram is a characteristic of the whole image. An improvement over GCH called local colour histogram (LCH) makes use of histograms of grids of cells of fixed size distributed over the image [3].

This method is efficient only in those cases in which there is a remarkable difference in the colour intensity of the bird and the background, else the bird will also be removed in background elimination. It also requires the bird to be the central object in the image containing no other object of interest which is not possible in all cases. Colour based background subtraction and indexing is not efficient as these factors can change with lighting conditions and weather.

2.3 Indexing in Flower databases

Some indexing strategies for flower databases were proposed in [4,5] which could be used for specialised databases also. They illustrate their approach by using it to solve the problem of indexing flower images when searching a flower patents database by color. The flower region is isolated from the background using an automatic iterative segmentation algorithm with domain knowledge-driven feedback. The flower's color is defined by the color names present in the flower region and their relative proportions. The database can be queried by example and by color names. The system provides a perceptually correct retrieval with natural language queries by using a natural language color classification derived from standard color names.

Since there has not been much work done in this field, we found the necessity of building this system for identifying birds. Studying the behaviour of birds with change in weather and temperature due to human activities makes it even more important. The methods used so far have made use of colour pattern in the bird, not taking into account any other feature of the bird, namely shape and texture. It becomes important to consider other features also to ensure that the bird has been recognised correctly. We have improved on the previous techniques by using the texture based features of the bird to distinguish it from others.

Chapter 3 Objective

The aim of this work is to propose an approach to identify the species of a bird based on texture characteristics obtained from a video of the bird.

Our database is built based on information collected through sample videos of birds and processing it in five stages. The user query, consisting of a video of a bird is input and processed, giving the likely hood of a match to one of the birds in our database.

The five proposed stages of the database building include:

1. Approximate median filter background subtraction [6].
2. Discrete wavelet transform on desired frames to extract edge information [21] [22].
3. Connected component analysis to obtain the largest component presumably containing the bird.
4. K-means clustering on the largest extracted component from above.
5. Feature extraction involving calculation of inter-cluster distances such as Bhattacharya distance [7], Hausdorff distance and Euclidean distance.

The user query image is first put through the five stages mentioned above and then through the following stages:

6. Iterative closest points (ICP) algorithm for texture mapping [8].
7. Data obtained from both feature extraction and ICP are served as input to a classifier based on Dempster-Shafer theory i.e. a DST decision maker [9].

Chapter 4 Design and Methodology

Several approaches were tried to achieve the objective of this work with varying degrees of complexity and success rates. This section outlines those methods and techniques.

4.1 Background removal

Identifying moving objects from a video sequence is a fundamental and critical task in many computer-vision applications. Background subtraction is the process of separating out foreground objects from the background in a sequence of video frames. Background subtraction is used in many emerging video applications, such as video surveillance, traffic monitoring, and gesture recognition for human-machine interfaces, to name a few. As computer vision begins to address the visual interpretation of action applications such as surveillance and monitoring are becoming more relevant. Similarly, recent work in intelligent environments and perceptual user interfaces involve vision systems which interpret the pose or gesture of users in a known, indoor environment. In all of these situations the first fundamental problem encountered is the extraction of the image region corresponding to the person or persons in the room.

Previous attempts at segmenting people from a known background have taken one of three approaches:

4.1.1 Background subtraction

Most common is some form of background subtraction. It identifies moving objects from the portion of a video frame that differs significantly from a background model, using statistical texture properties of the background observed over extended period of time to construct a model of the background, and use this model to decide which pixels in an input image do not fall into the background class. The fundamental assumption of the algorithm is that the background is static in all respects: geometry, reflectance, and illumination.

4.1.2 Stationary background

The second class of approach is based upon image motion only presuming that the background is stationary or at most slowly varying, but that the person is moving. In these methods no detailed model of the background is required. Of course, these methods are only appropriate for the direct interpretation of motion; if person stops moving, no signal remains to be processed. This method also requires constant or slowly varying geometry, reflectance, and illumination.

4.1.3 Geometry based

The final approach is based upon geometry. Dense depth maps in real-time are computed to provide a background disparity value that the algorithm can perform real-time depth segmentation or "z-keying" with. The only assumption of the algorithm is that the geometry of the background does not vary. However, the computational burden of computing dense, robust, real-time stereo maps, requires great computational power.

There are many challenges in developing a good background subtraction algorithm. It must be robust against changes in illumination. It should avoid detecting non-stationary background objects such as moving leaves, rain, snow, and shadows cast by moving objects. Finally, its internal background model should react quickly to changes in background such as starting and stopping of vehicles. Background removal is one of the most efficient methods used for segmenting the object in the video.

We have made use of the approximate median filter background subtraction to remove the background from the video in order to separate the bird as the region of interest.

4.1.4 Approximate Median Filter Background Subtraction

In median filtering, the previous N frames of video are buffered, and the background is calculated as the median of buffered frames. Then (as with frame difference), the background is subtracted from the current frame and thresholded to determine the foreground pixels.

Median filtering has been shown to be very robust and to have performance comparable to higher complexity methods. However, storing and processing many frames of video (as is often required to track slower moving objects) requires an often prohibitively large amount of memory. This can be alleviated somewhat by storing and processing frames at a rate lower than the frame rate thereby lowering storage and computation requirements at the expense of a slower adapting background.

A more efficient compromise was devised back in 1995 by UK researchers N.J.B. McFarlane and C.P. Schofield. While doing government funded research on piglet tracking in large commercial farms, they came up with an efficient recursive approximation of the median filter. Their 'approximate median' method, presented in their seminal paper, 'Segmentation and tracking of piglets in images' [6], has since seen wide implementation in the background subtraction literature, and been applied to a wide range of background subtraction scenarios.

The approximate median method works as such: if a pixel in the current frame has a

value larger than the corresponding background pixel, the background pixel is incremented by 1. Likewise, if the current pixel is less than the background pixel, the background is decremented by one. In this way, the background eventually converges to an estimate where half the input pixels are greater than the background, and half are less than the background—approximately the median (convergence time will vary based on frame rate and amount movement in the scene.) The more slowly adapting background incorporates a longer history of the visual scene, achieving about the same result as if we had buffered and processed N frames.



Fig 4.1
A sequence of frames obtained after applying approximate median filter background subtraction to a video



Fig 4.1 (contd.)
A sequence of frames obtained after applying approximate median filter background subtraction to a video

This method is a very good compromise. It offers performance near what you can achieve with higher-complexity methods (according to academic literature), and it costs not much more in computation and storage than frame differencing. In most cases, the background is removed to a great extent, providing a good starting point for the stages to follow.

4.2 Discrete Wavelet Transform for edge detection

An edge in an image is a contour across which the brightness of the image changes abruptly. In image processing, an edge is often interpreted as one class of singularities. In a function, singularities can be characterized easily as discontinuities where the gradient approaches infinity. However, image data is discrete, so edges in an image often are defined as the local maxima of the gradient.

4.2.1 Introduction to wavelets

Fourier analysis of a signal is a representation of it as a superposition of sines and cosines.

$$\hat{f}(\xi) := \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \xi} dx, \quad (4.1)$$

for every real number ξ .

The signal is analyzed in the time domain for its frequency content. This is done by transforming the function in time domain to frequency domain using (4.1).

The Fourier coefficients of the transformed function represent the contribution of each sine and cosine function at each frequency allowing the signal to be analyzed for its frequency content.

The signal in time domain can be recovered using the inverse Fourier transform as shown in (4.2).

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{2\pi i x \xi} d\xi, \quad (4.2)$$

for every real number x .

For non-stationary signals, we use the short term Fourier transform (STFT). The signal is divided into small enough segments, where these segments of the signal can be assumed to be stationary. For this purpose, a window function "w" is chosen. The width of this window must be equal to the segment of the signal where its stationarity is valid.

$$\text{STFT} \{x(t)\} \equiv X(\tau, \omega) = \int_{-\infty}^{\infty} x(t)w(t - \tau)e^{-j\omega t} dt \quad (4.3)$$

If we use a window of infinite length, we get the Fourier transform, which gives perfect frequency resolution, but no time information. The narrower the window, the better the time resolution and better the assumption of stationarity but poorer the frequency resolution.

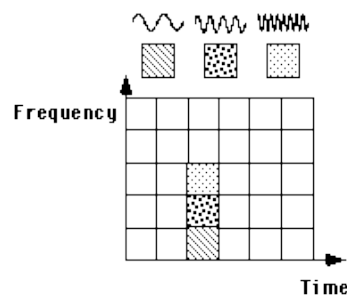


Fig 4.2

***Coverage of time-frequency plane by Fourier basis functions in STFT
where the window is a square wave
(from [21])***

As a single window is used for all frequencies in the STFT, the resolution of the analysis is the same at all locations in the time-frequency plane.

In wavelet transforms, the windows vary in size. For good frequency resolution, we use a large window at low frequencies whereas for good time resolution, we use a

narrow window at high frequencies.

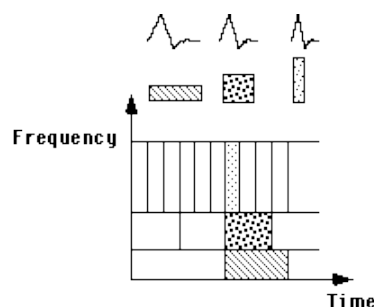


Fig 4.3
Coverage in time-frequency plane by Daubechies wavelet basis
(from [21])

As abstracted in [12], wavelets are mathematical functions that cut up data into different frequency components, and then study each component with a resolution matched to its scale. They have advantages over traditional Fourier methods in analyzing physical situations where the signal contains discontinuities and sharp spikes. Wavelets were developed independently in the fields of mathematics, quantum physics, electrical engineering, and seismic geology. Interchanges between these fields during the last ten years have led to many new wavelet applications such as image compression, turbulence, human vision, radar, and earthquake prediction.

A wavelet is a mathematical function used to divide a given function or continuous-time signal into different scale components. Usually one can assign a frequency range to each scale component. Each scale component can then be studied with a resolution that matches its scale. A wavelet transform is the representation of a function by wavelets. The wavelets are scaled and translated copies (known as "daughter wavelets") of a finite-length or fast-decaying oscillating waveform (known as the "mother wavelet"). Wavelet transforms have advantages over traditional Fourier transforms for representing functions that have discontinuities and sharp peaks, and for accurately deconstructing and reconstructing finite, non-periodic and/or non-stationary signals.

This representation is a wavelet series representation of a square-integrable function with respect to either a complete, orthonormal set of basis functions, or an over complete set of Frame of a vector space (also known as a Riesz basis), for the Hilbert space of square integrable functions.

Wavelet transforms are classified into discrete wavelet transforms (DWTs) and continuous wavelet transforms (CWTs). Both DWT and CWT are continuous-time (analog) transforms. They can be used to represent continuous-time (analog) signals. CWTs operate over every possible scale and translation whereas DWTs use a specific subset of scale and translation values or representation grid.

Unlike the Fourier transform, wavelet transform does not a single set of basis functions (sine and cosine). Instead there are an infinite number of possible sets of

basis functions.

4.2.2 Continuous Wavelet Transform (CWT)

The CWT was developed as an alternative approach to the STFT to overcome the resolution problem. The wavelet analysis is done in a similar way to the STFT analysis, in the sense that the signal is multiplied with a function, i.e. the wavelet, similar to the window function in the STFT, and the transform is computed separately for different segments of the time-domain signal. However, there are two main differences between the STFT and the CWT:

1. The Fourier transforms of the windowed signals are not taken, and therefore single peak will be seen corresponding to a sinusoid, i.e., negative frequencies are not computed.
2. The most significant difference is that the width of the window is changed as the transform is computed for every single spectral component.

$$X_w(a, b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t) \psi^* \left(\frac{t-b}{a} \right) dt \quad (4.4)$$

CWT is a function of two variables viz. b, the translation and a, the scale. $\psi(b,a)$ is similar to the window function in STFT and is called the mother wavelet as it is the general function from which the other windows are generated. Examples of mother wavelets include Daubechies, Cauchy's, Mexican hat, Coiflet and Haar wavelets.

4.2.3 Discrete wavelet transform (DWT)

CWT and STFT cannot be practically computed with analytical equations, integrals etc. A discretized CWT is calculated instead. Due to the inherent changes in scale during calculation of wavelet transform, the number of samples to be taken (the natural choice is a uniform sampling rate) can be reduced. At lower frequencies i.e. higher scales, the sampling rate can be lower (Nyquist theorem) and at higher frequencies i.e. lower scales the sampling rate is higher. Dyadic sampling is usually done.

4.2.4 DWT and Multiresolution analysis

The continuous wavelet transform was computed by changing the scale of the analysis window, shifting the window in time, multiplying by the signal, and integrating over all times. In the discrete case, filters of different cutoff frequencies are used to analyze the signal at different scales. The signal is passed through a series of high pass filters to analyze the high frequencies, and it is passed through a series of low pass filters to analyze the low frequencies.

The resolution of the signal, which is a measure of the amount of detail information in the signal, is changed by the filtering operations, and the scale is changed by upsampling and downsampling (subsampling) operations. Subsampling a signal corresponds to reducing the sampling rate, or removing some of the samples of the signal.

The procedure for calculation of DWT (Fast wavelet transform – Daubechies' and Mallat's algorithm) starts with passing a signal through a half band digital low pass filter. After passing the signal through a half band low pass filter, half of the samples can be eliminated according to the Nyquist's rule. This is called downsampling or subsampling by two and affects the scale. The resolution (the amount of information held in the signal) is affected by filtering.

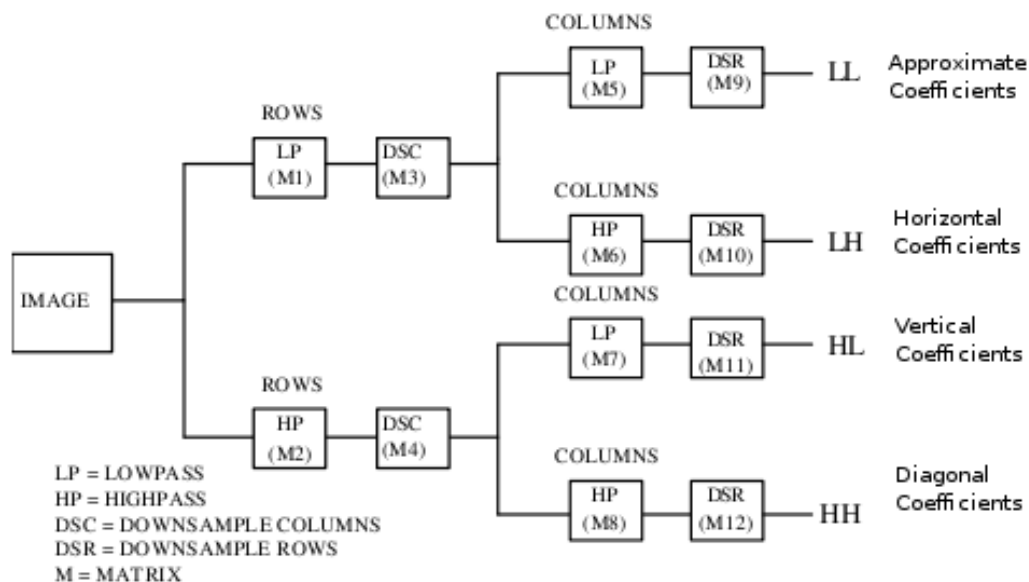


Fig 4.4
Stages of multiresolution in calculation of DWT using FWT
Daubechies and Mallat's Algorithm

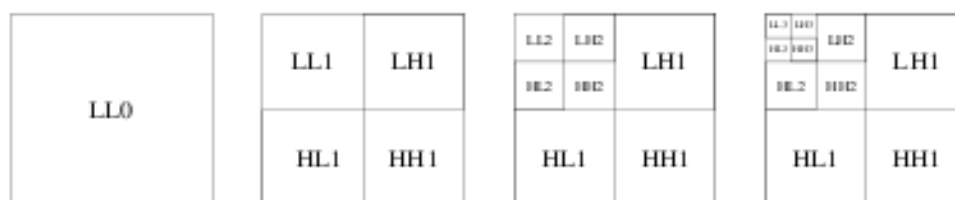


Fig 4.5
Image and its 1,2,3 level dyadic wavelet transform

4.2.5 Edge detection using wavelets

When the wavelet transform is used with a smoothing function, it is equivalent to Canny edge detection. The derivative of a Gaussian is convolved with the image, so that local maxima and minima of the image correspond to edges.

Edges are characterized mathematically by their Lipschitz regularity which is related to the wavelet transform. The wavelet transform can characterize the local regularity of functions. For an image $f(x, y)$, its edges correspond to singularities of $f(x, y)$, and thus are related to the local maxima of the wavelet transform modulus. Therefore, the wavelet transform is an effective method for edge detection.

We applied two-level discrete Daubechies 4 (db4) wavelet transform on the frames selected after background removal..

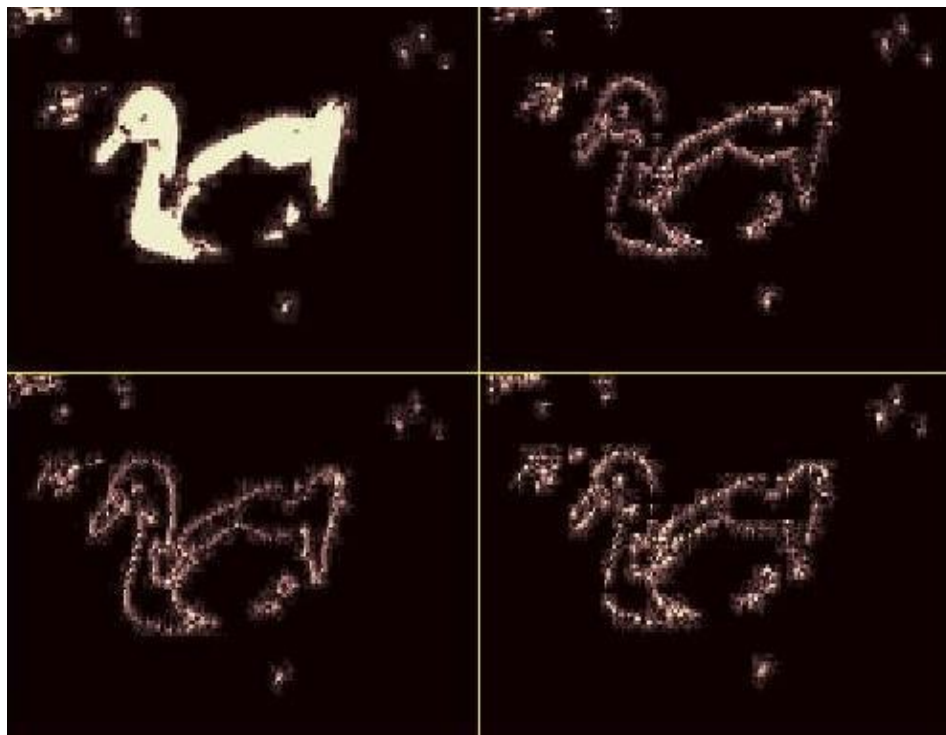


Fig 4.6

First level decomposition using db4 DWT

Upper left: Approximation coefficients

Upper right: Horizontal detail coefficients

Lower left: Vertical detail coefficients

Lower right: Diagonal detail coefficients



Fig 4.7

Second level decomposition using db4 DWT

The detail coefficients obtained at the second level were used to reconstruct the image with the horizontal, vertical and diagonal details only. The approximate coefficients (indicating low frequency information) are ignored as edge information is not

contained in them. The reconstructed images from the three detailed coefficients are added to produce an image as input to the next stage of our procedure i.e. connected component analysis.

4.3 Noise Removal

Noise is considered to be any measurement that is not part of the phenomena of interest. Digital images are prone to a variety of types of noise. There are several ways that noise can be introduced into an image, depending on how the image is created. For example: If the image is scanned from a photograph made on film, the film grain is a source of noise. Noise can also be the result of damage to the film, or be introduced by the scanner itself. If the image is acquired directly in a digital format, the mechanism for gathering the data can introduce noise. Electronic transmission of image data can introduce noise. A number of different ways are available to remove or reduce noise in an image. Different methods are better for different kinds of noise. The methods available include:

4.3.1 Linear filtering

Certain filters, such as averaging or Gaussian filters, are appropriate for this purpose. For example, an averaging filter is useful for removing grain noise from a photograph. Because each pixel gets set to the average of the pixels in its neighbourhood, local variations caused by grain are reduced.

4.3.2 Median filtering

The value of an output pixel is determined by the median of the neighbourhood pixels, rather than the mean. The median is much less sensitive than the mean to extreme values. Median filtering is therefore better able to remove these extreme values without reducing the sharpness of the image.

4.3.3 Adaptive filtering

The wiener filter applied to an image adaptively, tailors itself to the local image variance. Where the variance is large, wiener filter performs little smoothing. Where the variance is small, wiener filter performs more smoothing. This approach often produces better results than linear filtering. The adaptive filter is more selective than a comparable linear filter, preserving edges and other high-frequency parts of an image. Median filter is used to remove noise from the binarised image before convex hulling.

4.4 Convex hulling

The bird was to be segmented from the rest of the image following a convex hulling procedure.

Convex hulling [10] is a novel method for image segmentation where image contains a dominant object. The method is applicable to a large class of images including noisy and poor quality images. It is fully automatic and has low computational cost. It may be noted that the proposed segmentation technique may not produce optimal result in some cases but it gives reasonably good result for almost all images of a large class. The method is found very useful for the applications where accuracy of the segmentation is not very critical, e.g., for global shape feature extraction, second generation coding etc.

We employed the pseudo-convex hull algorithm, which is outline below.

4.4.1 Pseudo convex hull algorithm

The proposed algorithm works for a class of images. Depending on the contents, images may be grouped into three classes:

1. Class of images containing a single dominant object (Class-1). Class-1 contains images of a child, friend, relative, home, car, pet, object of our interest (e.g., ancient building, monument, sculpture and statue, biomedical image, animal, bird, etc.), famous personality, and so on. These objects, in the image, occupy the major area mostly at the centre and are sharply focused. It is also observed that the dominant object or the objects of interest in the Class-1 images are closely convex shaped.
2. Class of images containing many objects of more or less equal significance (Class-2). Images of a group of people, cluttered objects, busy area (e.g., railway station, departmental store, city street, etc.), business meeting and like belong to Class-2.
3. Class of images containing no objects of specific interest, but their combination appears very picturesque (Class-3). The class-3 is exemplified by outdoor scenery consisting mostly of sky, water body (like, sea, river, lake etc.), grass-field, beach etc. none of which is particularly important, but surely the combination is.

4.4.1.1 Definitions

1. An object A is said to be *convex* if its intersection with a line having any slope angle θ produces at most one line segment.
2. Suppose an image contains an object A . If two distinct line segments, with an angle θ between them, starting from every point on the boundary of A can reach the image frame without intersecting any of the interior point of A , then we call A is a *pseudo-convex* object with respect to θ ; It is readily evident that the objects we mostly deal with are neither strictly convex nor concave, but

are of type pseudo-convex. Hence, we classify 2D objects into three groups: Convex, Pseudo-convex and Concave. Since we work on discrete domain and it is known that digital straight line segment can uniquely be defined only for the slope 0° , 45° , 90° and 135° , we confine our definition of pseudo-convex objects in terms line segment of said orientations only.

3. A digital object A is said to be *pseudo-convex* if two line segments, with an angle θ between them and one of them is either horizontal (slope 0°) or vertical (slope 90°), starting from every point on the boundary of A can reach the image frame without intersecting any of the interior point of A .
4. A is a true convex object for $\theta \geq 180^\circ$ and it is taken as a concave object if $\theta < 45^\circ$. Otherwise, if $45^\circ \leq \theta < 180^\circ$ then the object is 'closely convex shaped' which can be further classified as follows. If $135^\circ \leq \theta < 180^\circ$ then the shape of A is called ramp-convex. It is ortho-convex if $90^\circ \leq \theta < 135^\circ$. A is wedge-convex for $45^\circ \leq \theta < 90^\circ$.

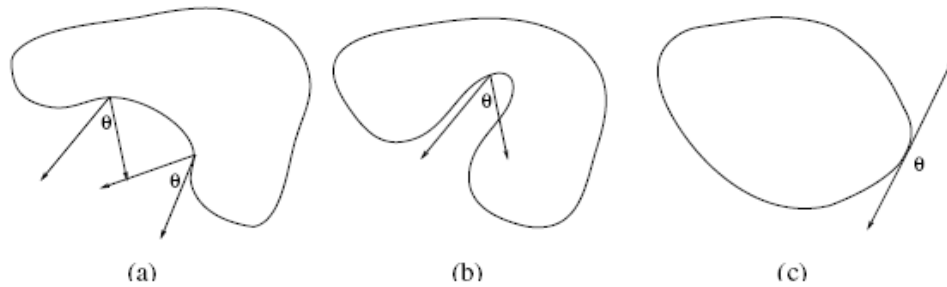


Fig 4.8
Different types of objects
(a) Pseudo-convex (b) Concave (c) Convex

4.4.1.2 The Algorithm

This segmentation algorithm is based on the idea of obtaining a closely convex region corresponding to the dominant object in an image. It may be noted that this region is nothing but the pseudo-convex hull of the dominant object.

$B(i, j)$ is the binary image containing a set of points A whose pseudo-convex hull is to be determined. That means $B(i, j)$ may be represented as

$$B(i, j) = \begin{cases} 1 & (i, j) \in A \\ 0 & \text{otherwise} \end{cases} \quad (4.5)$$

The steps of the algorithm are:

Step 1: Take four other arrays $H(i, j)$, $V(i, j)$, $D1(i, j)$ and $D2(i, j)$ of same size as that of $B(i, j)$, and initialize them with 1's.

Step 2: Now for each row of $H(i, j)$

1. Start from first column, change its pixel value to zero and move right until $B(i, j) = 1$ or the last column is reached.

2. If the last column is not reached then start from last column, change pixel values to zero, and move left ward until $B(i, j) = 1$.

Step 3: Now repeat sub-steps of 2 for $V(i, j)$, $D1(i, j)$ and $D2(i, j)$ with appropriate directions i.e., upward and downward for V and so on.

Step 4: Finally, produce a binary image $P(i, j)$ that contains the pseudo-convex hull of the given point set A as follows:

$$P(i, j) = \begin{cases} 1 & H(i, j) + V(i, j) + D1(i, j) + D2(i, j) \geq th \\ 0 & \text{otherwise} \end{cases} \quad (4.6)$$

1. $th = 1$ is equivalent to $\theta = 135^\circ$ and we have ramp-convex hull.
2. $th = 2$ is equivalent to $\theta = 90^\circ$ and if only $H(i, j)$ and $V(i, j)$ taken, we have ortho-convex hull.
3. $th = 3$ is equivalent to $\theta = 45^\circ$ and we have wedge-convex hull.

Finally, it may be noted that wedge-convex hull ($hull_w$) is the closest estimate of the object as

$$A \subseteq hull_{wedge}(A) \subseteq hull_{ortho}(A) \subseteq hull_{ramp}(A) \subseteq hull(A) \quad (4.7)$$

4.4.1.3 Steps in processing

Assuming that the image contains only one dominating object and other objects, if present, are small in comparison to the object of interest the steps of the algorithm are as follows:



Fig 4.9

Input image to pseudo convex hull algorithm

1. Noise removal.

Smoothing filters are, in general, used for noise removal and blurring. Blurring is used as a preprocessing step to remove small details from the image prior to extraction of large objects as well as bridging of small gaps in lines and curves. A 5x5 window was used over which median filtering was done to remove noise.

2. Initial Segmentation.

i. Formation of a gradient image

This was done by summing up the images as obtained by reconstructing it from the Daubechies coefficients from HL,LH and HH sub band blocks. The resultant image holds edge information of the central object.



Fig 4.10

Image (4.9) after edge detection by DWT

ii. Thresholding to obtain a binarized image.

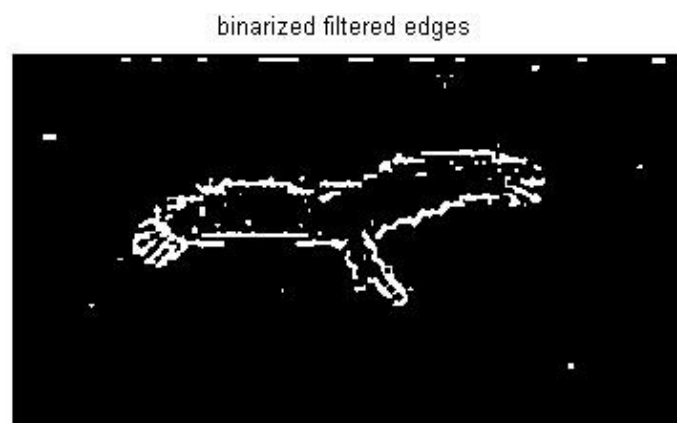


Fig 4.11
Binary (4.10) image after thresholding

3. Final Segmentation.

- i. Approximate object area determination.
To find out the object, the edge image obtained in previous step undergoes the pseudo-convex hull algorithm to determine the wedgeconvex hull of the set of edge pixels, i.e., use $th = 3$.

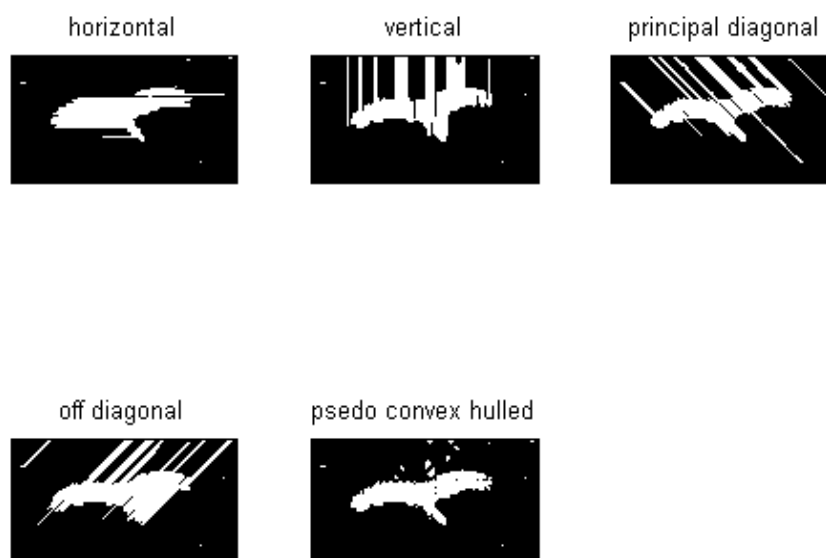


Fig 4.12
Image (4.11) after being sampled at specified angles and then hulled

- ii. Removal of small objects by component labeling.
After computing the wedge-convex hull we get an initial estimate of the dominant object. We also get some small objects arising out of scattered extraneous edge pixels in the background. These can be

isolated by component labeling and subsequently removed keeping only the biggest one.

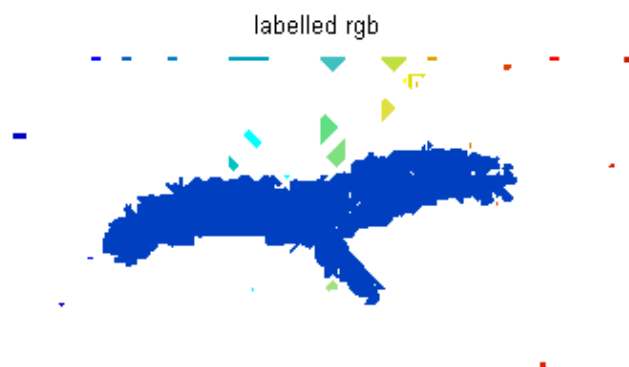


Fig 4.13
Image (4.12) after connected component analysis and component labelling

- iii. Final extraction of object region.
After removal of small objects we finally determine the dominant object region by applying pseudo-convex hull algorithm with $th = 1$ on the point set of largest connected component obtained from previous step.

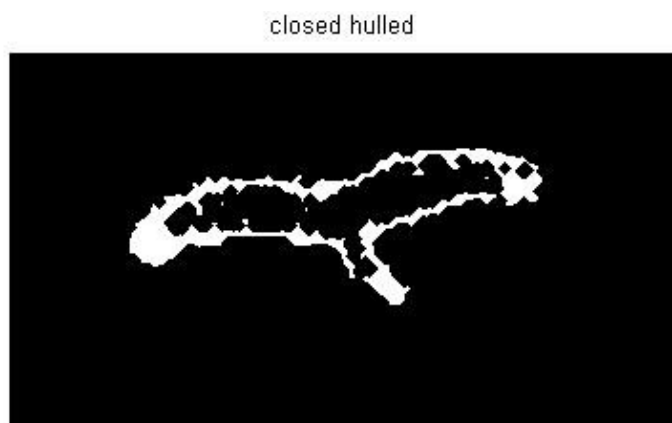


Fig 4.14
Image (4.13) after largest component extraction and application of morphological close operation

4.4.2 Results of pseudo convex hull algorithm

Our approach to pseudo convex hulling involved obtaining edge information using DWT followed by application of median filtering to remove noise. The resulting image is binarised and pseudo convex hull algorithm is applied to obtain the object. Connected component analysis is done to select the largest object and eliminate noise. Pseudo convex algorithm is applied again to avoid intrusions into the object. Morphological operator is applied on the result obtained, to close the object edges.

This is not the most efficient method for segmentation as it is very sensitive to noise and is unable to extract the required central object of interest if there are other elements present in the background.

In the presence of certain unavoidable objects like branches of trees (on which the bird is sitting) in the background, the result is very poor which cannot be used for further processing. Hence this method was dropped.

4.5 Connected component analysis

Connected component labelling analysis is an algorithmic application of graph theory, where subsets of connected components are uniquely labelled based on a given heuristic. Connected component labelling is not to be confused with segmentation. Connected component labelling is used in computer vision to detect unconnected regions in binary digital images, although colour images and data with higher-dimensionality can also be processed. When integrated into an image recognition system or human-computer interaction interface, connected component labelling can operate on a variety of information.

Connected components labelling 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 labelled with a graylevel or a colour (colour labelling) according to the component it was assigned to. Extracting and labelling of various disjoint and connected components in an image is central to many automated image analysis applications.

Connected component labelling works by scanning an image, pixel-by-pixel (from top to bottom and left to right) in order to identify connected pixel regions, i.e. regions of adjacent pixels which share the same set of intensity values V . (For a binary image $V=\{1\}$; in a graylevel image V will take on a range of values, for example: $V=\{51, 52, 53, \dots, 77, 78, 79, 80\}$.)

Connected component labelling works on binary or graylevel images and different measures of connectivity are possible. However, for the following we assume binary input images and 8-connectivity. The connected components labelling operator scans

the image by moving along a row until it comes to a point p (where p denotes the pixel to be labelled at any stage in the scanning process) for which $V=\{1\}$. When this is true, it examines the four neighbours of p which have already been encountered in the scan (i.e. the neighbours (i) to the left of p , (ii) above it, and (iii and iv) the two upper diagonal terms). Based on this information, the labelling of p occurs as follows:

1. If all four neighbours are 0, assign a new label to p , else
2. If only one neighbour has $V=\{1\}$, assign its label to p , else
3. If more than one of the neighbours have $V=\{1\}$, assign one of the labels to p and make a note of the equivalences.

After completing the scan, the equivalent label pairs are sorted into equivalence classes and a unique label is assigned to each class. As a final step, a second scan is made through the image, during which each label is replaced by the label assigned to its equivalence classes. For display, the labels might be different graylevels or colours.

The component with the maximum number of pixels with the same label is extracted as this represents the largest connected component in the image.

Connected component analysis has been made use of in a certain cases to obtain the bird separately:

1. To extract the bird from the edge information as obtained from the detailed coefficients of Daubechies wavelet transform, so that further processing becomes easier.
2. After convex hulling to extract the single largest object and remove noise.

4.6 Morphological closing operation

Morphological image processing consists of a set of operators that transform the image according to topological and geometrical continuous-space concepts such as size, shape, convexity, connectivity, and geodesic distance on both continuous and discrete spaces. Mathematical morphology is also the foundation of morphological image processing is a theory and technique for the analysis and processing of geometrical structures, based on set theory, lattice theory, topology, and random functions. Mathematical morphology is most commonly applied to digital images.

There are basically two morphological operators that are shift-invariant and related to Minkowski addition viz. Dilation and erosion.

4.6.1 Dilation

The basic effect of the operator on a binary image is to gradually enlarge the boundaries of regions of foreground pixels (i.e. white pixels, typically). Thus areas of foreground pixels grow in size while holes within those regions become smaller. The dilation operator takes two pieces of data as inputs. The first is the image which is to be dilated. The second is a (usually small) set of coordinate points known as a structuring element (also known as a kernel). It is this structuring element that determines the precise effect of the dilation on the input image.

Mathematically, dilation for a binary image is :

Suppose that X is the set of Euclidean coordinates corresponding to the input binary image, and that K is the set of coordinates for the structuring element.

Let K_x denote the translation of K so that its origin is at x .

Then the dilation of X by K is simply the set of all points x such that the intersection of K_x with X is non-empty.

4.6.2 Erosion

The basic effect of the operator on a binary image is to erode away the boundaries of regions of foreground pixels (i.e. white pixels, typically). Thus areas of foreground pixels shrink in size, and holes within those areas become larger. The erosion operator takes two pieces of data as inputs. The first is the image which is to be eroded. The second is a (usually small) set of coordinate points known as a structuring element (also known as a kernel). It is this structuring element that determines the precise effect of the erosion on the input image.

Mathematically, erosion for a binary image is :

Suppose that X is the set of Euclidean coordinates corresponding to the input binary image, and that K is the set of coordinates for the structuring element.

Let K_x denote the translation of K so that its origin is at x .

Then the erosion of X by K is simply the set of all points x such that K_x is a subset of X .

4.6.3 Closing

Closing is similar in some ways to dilation in that it tends to enlarge the boundaries of foreground (bright) regions in an image (and shrink background color holes in such regions), but it is less destructive of the original boundary shape. As with other morphological operators, the exact operation is determined by a structuring element. The effect of the operator is to preserve background regions that have a similar shape to this structuring element, or that can completely contain the structuring element, while eliminating all other regions of background pixels. Closing is opening performed in reverse. It is defined simply as dilation followed by erosion using the same structuring element for both operations. The closing operator therefore requires two inputs: an image to be closed and a structuring element.

Closing is the dual of opening, i.e. closing the foreground pixels with a particular

structuring element, is equivalent to closing the background with the same element.

The closing operation was performed on the image obtained after pseudo convex hulling.

4.7 Picking up frames from videos

To ensure correct bird recognition we have to study the bird in different postures. This is possible by extracting frames from videos which can provide maximum information about the bird. We automated this process by picking up best ten frames based on the criteria that the largest connected component in these frames are the ten best out of all the frames. Unfortunately this method didn't turn out to be of any use as we couldn't retrieve the frames with side views. Most frames obtained had a large exposed surface area of the bird, but it was not a side view. Hence, we had to pick up suitable frames from the videos manually.

4.8 Template matching

Template matching was used as a pre-processing step to check if a bird was indeed present in the image or not.

Template matching is the process of matching a template to an image, where the template is a sub-image that contains the shape we are trying to find. We centre the template on an image point and count up how many points in the template match those in the image. The procedure is repeated for the entire image and the point which led to the best match, the maximum count, is deemed to be the point where the shape lies within the image. Template matching is used to find the existence of the bird in the image by matching a template of the beak with the image. We used three methods for the same which are explained as below:

4.8.1 Generalised Hough transform

The Hough Transform is a technique that locates shapes in images. It has been used to extract lines, circles and ellipses. 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 hough transform. These techniques again give results equivalent to those delivered by matched template filtering, but with the computational advantage of the evidence gathering approach. An early approach offered only limited capability for arbitrary shapes. The full mapping is called the Generalised hough transform and can be used to locate arbitrary shapes with unknown position, size and orientation. The GHT can

be formally defined by considering the duality of a curve.

The formal analysis of the HT provides the route for generalising it to arbitrary shapes [19]. Let the model curve be:

$$\mathbf{v}(\theta) = x(\theta) \begin{bmatrix} 1 \\ 0 \end{bmatrix} + y(\theta) \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (4.8)$$

In general, we are interested in matching the model shape against a shape in an image. The shape in the image has a different location, orientation and scale. The image shape can be defined by considering translation, rotation and change of scale. Thus, the shape in the image can be defined as

$$\boldsymbol{\omega}(\theta, \mathbf{b}, \lambda, \rho) = \mathbf{b} + \lambda \mathbf{R}(\rho) \mathbf{v}(\theta) \quad (4.9)$$

where $\mathbf{b} = (x_0, y_0)$ is the translation vector λ is a scale factor and $\mathbf{R}(\rho)$ is a rotation matrix. The location of the shape is given by:

$$\mathbf{b} = \boldsymbol{\omega}(\theta) - \lambda \mathbf{R}(\rho) \mathbf{v}(\theta) \quad (4.10)$$

Given a shape $\boldsymbol{\omega}(\theta)$ and a set of parameters \mathbf{b} , λ and ρ , this equation defines the location of the shape. We do not know the shape $\boldsymbol{\omega}(\theta)$ (since it depends on the parameters that we are looking for), but we only have a point in the curve. If we call $\omega_i = (\omega_{xi}, \omega_{yi})$ the point in the image, then

$$\mathbf{b} = \omega_i - \lambda \mathbf{R}(\rho) \mathbf{v}(\theta) \quad (4.11)$$

defines a system with four unknowns and with as many equations as points in the image. In order to find the solution we can gather evidence by using a four-dimensional accumulator space. For each potential value of \mathbf{b} , λ and ρ , we trace a point spread function by considering all the values of θ . That is, all the points in the curve $\mathbf{v}(\theta)$. 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 information. The gradient direction at a point in the arbitrary model is given by:

$$\phi'(\theta) = \frac{y'(\theta)}{x'(\theta)} \quad \text{and} \quad \hat{\phi}'(\theta) = \tan^{-1}(\phi'(\theta)) \quad (4.12)$$

This equation 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.

That is, a point spread function for a given edge point ω_i is obtained by selecting a subset of points in $\mathbf{v}(\theta)$ such that the edge direction at the image point rotated by ρ

equals the gradient direction at the model point.

4.8.1.1 Method

Given an image point ω_i we have to find a displacement vector $\gamma(\lambda, \rho)$. When the vector is placed at ω_i , then its end is at the point b . In the GHT jargon, this point is called the reference point. The vector $\gamma(\lambda, \rho)$ can be obtained as $\lambda R(\rho) \nu(\theta)$. In order to evaluate these equations, we need to know the point $\nu(\theta)$. This is the crucial step in the evidence gathering process.

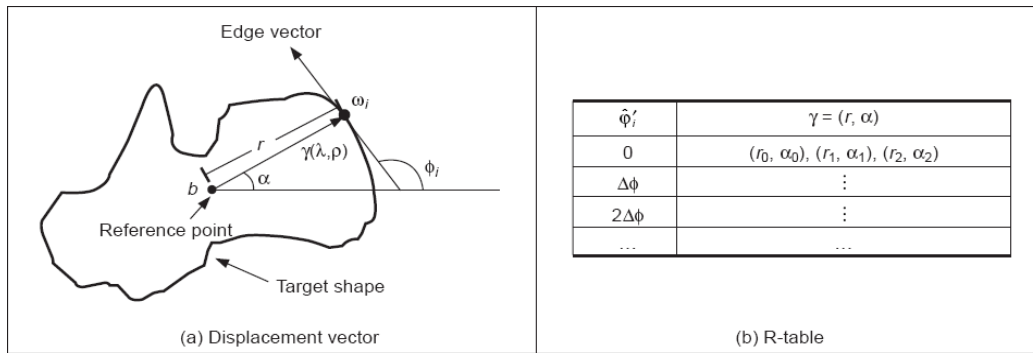


Fig 4.15

(a) Geometry of GHT (b) Structure of R-table

The process of determining $\nu(\theta)$ centres on solving Equation 5.76. According to this equation, since we know $\hat{\phi}_i'$, then we need to find the point $\nu(\theta)$ whose gradient direction is

$$\hat{\phi}_i' + \rho = 0.$$

Then we must use $\nu(\theta)$ to obtain the displacement vector $\gamma(\lambda, \rho)$. The GHT pre-computes the solution of this problem and stores it in an array called the *R-table*. The R-table stores for each value of $\hat{\phi}_i'$ the vector $\gamma(\lambda, \rho)$ for $\rho = 0$ and $\lambda = 1$. In polar form, the vectors are stored as a magnitude direction pair and in Cartesian form as a co-ordinate pair.

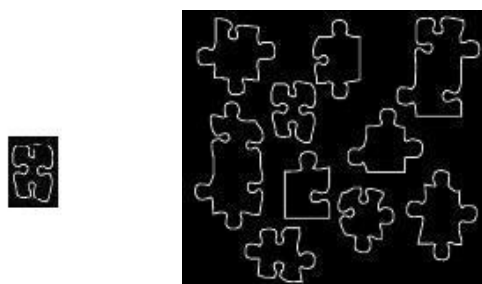
The possible range for $\hat{\phi}_i'$ is between $-\pi/2$ and $\pi/2$ radians. This range is split into N equispaced slots, or bins. These slots become rows of data in the R-table. The edge direction at each border point determines the appropriate row in the R-table. The length, r , and direction, α , from the reference point is entered into a new column element, at that row, for each border point in the shape. In this manner, the N rows of the R-table have elements related to the border information, elements for which there is no information contain null vectors. The length of each row is given by the number of edge points that have the edge direction corresponding to that row; the total number of elements in the R-table equals the number of edge points above a chosen threshold. The structure of the R-table for N edge direction bins and m template border points is illustrated in Fig 4.15 (b).

Now using the data stored in the R-table we build the accumulator array. The co-ordinates of points given by evaluation of all R-table points for the particular row

indexed by the gradient magnitude are used to increment cells in the accumulator array. The maximum number of votes occurs at the location of the original reference point. After all edge points have been inspected, the location of the shape is given by the maximum of an accumulator array.

4.8.1.2 Results and conclusion

Although GHT is an effective method for shape extraction, there are several inherent difficulties in its formulation. The most evident problem is that the table does not provide an accurate representation when objects are scaled and translated. This is because the table implicitly assumes that the curve is represented in discrete form. Thus, the GHT maps a discrete form into a discrete parameter space. Additionally, the transformation of scale and rotation can induce other discretisation errors. This is because when discrete images are mapped to be larger, or when they are rotated, loci which are unbroken sets of points rarely map to unbroken sets in the new image. Another important problem is the excessive computation required by the four-dimensional parameter space. This makes the technique impractical. Also, the GHT is clearly dependent on the accuracy of directional information. By these factors, the results provided by the GHT can become less reliable. A solution is to use an analytic form instead of a table. This avoids discretisation errors and makes the technique more reliable. This also allows the extension to affine or other transformations. However, this technique requires solving for the point $u(\theta)$ in an analytic way, increasing the computational load. A solution is to reduce the number of points by considering characteristic points defined as points of high curvature. This still requires the use of a four-dimensional accumulator. An alternative to reduce this computational load is to include the concept of invariance in the GHT mapping.



(a) *Template*

Fig 4.16

(b) *Image to searched in*

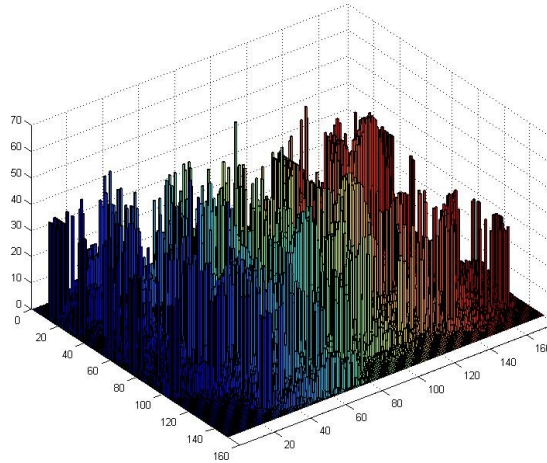


Fig 4.17
Accumulator array for the above template matching in GHT

4.8.2 Invariant Generalised Hough Transform

The problem with the GHT (and other extensions of the HT) is that they are very general. That is, the HT gathers evidence for a single point in the image, a point on its own provides little information. Thus, it is necessary to consider a large parameter space to cover all the potential shapes defined by a given image point. The GHT improves evidence gathering by considering a point and its gradient direction. However, since gradient direction changes with rotation, then the evidence gathering is improved in terms of noise handling, but little is done about computational complexity. 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.

Let us explain this idea in more detail. The main aim of the constraint in Equation 4.12, is to include gradient direction to reduce the number of votes in the accumulator by identifying a point $v(\theta)$. Once this point is known, then we obtain the displacement vector $\gamma(\lambda, \rho)$. For each value of rotation, we have a different point in $v(\theta)$. The constraint is replaced by a constraint of the form

$$Q(\omega i) = Q(v(\theta)) \quad (4.13)$$

The function Q is said to be invariant and it computes a feature at the point. This feature can be, for example, the colour of the point, or any other property that does not change in the model and in the image. By considering Equation 4.13, we have

$$Q(\omega i) - Q(v(\theta)) = 0 \quad (4.14)$$

That is, 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. The definition of Q depends on the application and the type of transformation. The most general invariant properties can be obtained by considering geometric definitions. In the case of rotation and scale changes (i.e. similarity transformations) the fundamental invariant property is given by the concept of angle. An angle 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. That is,

$$Q(\omega_i) = \frac{X_j Y_i - X_i Y_j}{X_i X_j + Y_i Y_j} \quad (4.15)$$

where

$$\begin{aligned} \mathbf{X}_k &= \boldsymbol{\omega}_k - \boldsymbol{\omega}_T, \\ \mathbf{Y}_k &= \boldsymbol{\omega}_k - \boldsymbol{\omega}_T. \end{aligned}$$

In general, we can define the points $[\omega_j, \omega_T]$ in different ways. An alternative geometric arrangement is shown in Figure 4.18(a). Given the points ω_i and a fixed angle ν , 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 ν . The third point is defined by the intersection of the tangent lines at ω_i and ω_j .

$$Q(\omega_i) = \frac{\phi'_i - \phi'_j}{1 + \phi'_i \phi'_j} \quad (4.16)$$

We can replace the gradient angle in the R-table by the angle β . The form of the new invariant table is shown in Figure 4.18(c). Since the angle β does not change with rotation or change of scale, 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.

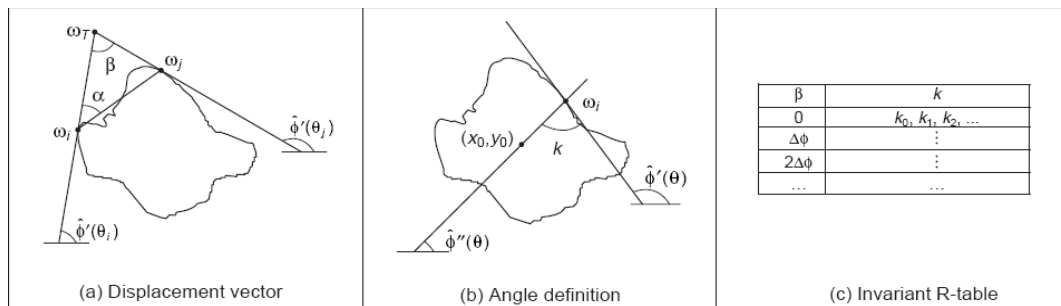


Fig 4.18

(a), (b) Geometry of invariant GHT

(c) Structure of R-table

In order to locate the point b we can generalise the ideas presented in Figure 4.18(b). As in the case of the circle and ellipse, we can locate the shape by considering a line of votes that passes through the point b . This line is determined by the value of ϕ_i'' . First, we will find an invariant definition of this value. Second, we will include it on the GHT table. We can develop

$$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} \omega_{xi} \\ \omega_{yi} \end{bmatrix} + \lambda \begin{bmatrix} \cos(\rho) & \sin(\rho) \\ -\sin(\rho) & \cos(\rho) \end{bmatrix} \begin{bmatrix} x(\theta) \\ y(\theta) \end{bmatrix} \quad (4.17)$$

$$\phi_i'' = \frac{\omega_{yi} - y_0}{\omega_{xi} - x_0} = \frac{[-\sin(\rho) \quad \cos(\rho)]y(\theta)}{[\cos(\rho) \quad \sin(\rho)]x(\theta)} \quad (4.18)$$

In order to define ϕ_i'' we can consider the tangent angle at the point ω_i . By considering the derivative of Equation 4.12 we have that

$$\phi_i' = \frac{[-\sin(\rho) \quad \cos(\rho)]y'(\theta)}{[\cos(\rho) \quad \sin(\rho)]x'(\theta)} \quad (4.19)$$

Thus,

$$\phi_i' = \tan(\phi - \rho) \quad (4.20)$$

where

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

We define

$$\hat{\phi}_i' = k + \phi_i' \quad (4.21)$$

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 characterisation. In order to see that k is invariant, we solve it for Equation 5.97. That is,

$$k = \hat{\phi}_i' - \phi_i' \quad (4.22)$$

Thus,

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

That is,

$$k = \xi - \phi \quad (4.23)$$

That is, independent of rotation. The definition of k has a simple geometric interpretation illustrated in Figure 4.18(b).

In order to obtain an invariant GHT, it is necessary to know for each point ω_i , the

corresponding point $u(\theta)$ and then compute the value of φ_i'' . Then evidence can be gathered by the line in Equation 4.18.

That is,

$$y_0 = \varphi_i''(x_0 - \omega_{xi}) + \omega_{yi} \quad (4.24)$$

In order to compute φ_i'' we can obtain k and then use Equation 4.22. In the standard tabular form the value of k can be precomputed and stored as function of the angle β . During implementation the value of α is set to $\pi/4$ and each element of the table stores a single value computed according to Equation 4.22. The more cumbersome part of the code 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 use the value of β defined in Equation 4.16 to index the. The data k recovered from the table is used to compute the slope of the angle defined in Equation 4.22. This is the slope of the line of votes traced in the accumulators

4.8.3 Template matching invariant to rotation, scaling and translation – The Ciratefi Algorithm

This method [11] proposes a much more efficient approach of finding a query template grayscale image Q in another grayscale image A to analyze, invariant to rotation, scale, translation, brightness and contrast (RSTBC), without previous simplification of A and Q that discards grayscale information, like detection of edges, detection of interest points and segmentation/binarization. These image-simplifying operations throw away the rich grayscale information, are noise-sensitive and prone to errors, decreasing the robustness of the matching.

The brute force solution to this problem performs a series of conventional template matching between the image to analyze A and the query template Q . Image Q must be rotated by every angle, translated to every position and scaled by every factor (within some specified range of scale factors) and a conventional BC-invariant template matching is executed for each instance of the transformed Q . Possibly, the brute force algorithm yields the most precise solution to this problem. To obtain RSTBC-invariant template matching the query shape Q must be rotated by every angle and scaled by every factor. In practice, it is not possible to rotate and scale Q by every angle and scale, but only by some discrete set of angles and scales. To avoid that, a small misalignment may cause a large mismatching, a low-pass filter (for example, the Gaussian filter) smoothes both images A and Q . This low-pass filtering lessens the errors introduced by using discrete scales and angles.

Then, each pixel p of A is tested for matching against all the transformed templates. If the largest absolute value of the contrast/brightness-aware correlation at pixel p is above some threshold t , the template is considered to be found at p .

This process is divided into three parts described as follows.

4.8.3.1 Circular sampling filter

Circular sampling filter (Cifi) uses the projections of the images A and Q on a set of rings to detect the first grade candidate pixels and their probable scales. The correct choice of number of circles l is not essential to our algorithm, because the next two steps will further filter the first grade candidate pixels.

4.8.3.2 Radial sampling filter

The second filter is called radial sampling filter (Rafi) and uses the projections of images A and Q on a set of radial lines to upgrade some of the first grade candidate pixels to second grade. The pixels that are not upgraded are discarded. It also assigns a probable rotation angle to each second grade candidate pixel.

4.8.3.3 Template sampling filter

The third filter is called Tefi and it is simply the BC-invariant template matching, applied only at the second grade candidate pixels, using the probable scale and angle determined respectively by Cifi and Rafi.

4.9 Deskewing

It was necessary to obtain a deskewed image of the bird as a preprocessing stage, hough and radon transform was applied on the image to extract features of the bird. These two transforms are able to transform two dimensional images with lines into a domain of possible line parameters, where each line in the image will give a peak positioned at the corresponding line parameters. They have lead to many line detection applications within image processing, computer vision, and seismic.

4.9.1 Hough Transform

The Hough transform is a feature extraction technique used in image analysis, computer vision, and digital image processing. The purpose of the technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm for computing the Hough transform. The Hough transform algorithm uses an array, called accumulator, to detect the existence of a line $y = mx + b$. The dimension of the accumulator is equal to the number of unknown parameters of the Hough transform problem. For example, the linear Hough transform problem has two unknown parameters: m and b . The two dimensions of the accumulator array would correspond to quantized values for m and b . For each pixel

and its neighbourhood, the Hough transform algorithm determines if there is enough evidence of an edge at that pixel. If so, it will calculate the parameters of that line, and then look for the accumulator's bin that the parameters fall into, and increase the value of that bin. By finding the bins with the highest values, typically by looking for local maxima in the accumulator space, the most likely lines can be extracted, and their geometric definitions read off. The simplest way of finding these peaks is by applying some form of threshold, but different techniques may yield better results in different circumstances - determining which lines are found as well as how many. Since the lines returned do not contain any length information, it is often next necessary to find which parts of the image match up with which lines. Moreover, due to imperfection errors in the edge detection step, there will usually be errors in the accumulator space, which may make it non-trivial to find the appropriate peaks, and thus the appropriate lines.

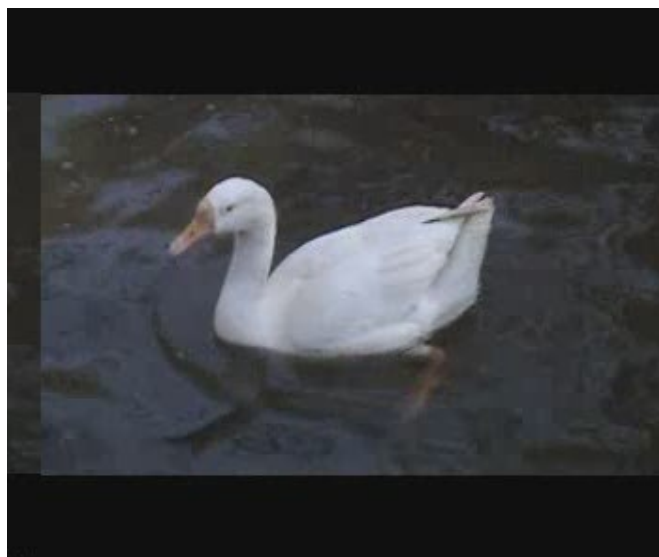


Fig 4.19
Original image – before Hough transform



Fig 4.20
Result of Hough transform on binarized Canny edge detected image



Fig 4.21
Deskewed image after application of Hough transform



Fig 4.22
Original input image for Hough transform



Fig 4.23
Image after applying Hough transform

4.9.2 Radon transform

In mathematics, the Radon transform in two dimensions is the integral transform consisting of the integral of a function over straight lines. The radon transform computes projections of an image matrix along specified directions. A projection of a two-dimensional function $f(x,y)$ is a set of line integrals. The radon transform computes the line integrals from multiple sources along parallel paths, or beams, in a certain direction. The beams are spaced 1 pixel unit apart. To represent an image, the radon transform takes multiple, parallel-beam projections of the image from different angles by rotating the source around the centre of the image. Consider the straight line defined parametrically by

$$(x(t),y(t)) = t(\sin\alpha, -\cos\alpha) + s(\cos\alpha,\sin\alpha) \quad (4.25)$$

where s is the distance from the origin and α is the angle from the x axis. We define the Radon transform of a function f on the plane (where it is assumed that the function is continuous and vanished outside a disc of some finite radius) by

$$\mathcal{R}[f](\alpha, s) = \int_{-\infty}^{\infty} f(x(t), y(t)) dt = \int_{-\infty}^{\infty} f(t(\sin \alpha, -\cos \alpha) + s(\cos \alpha, \sin \alpha)) dt$$

(4.26)

A simple thresholding algorithm could then be used to pick out the line parameters, and given that the transform is linear many lines will just give rise to a set of distinct

point in the Radon domain.



Fig 4.24
Deskewed image(4.19) after Radon transform



Fig 4.25
Image(4.22) after applying Radon transform

Hough and radon transforms give us an idea of the angles at which straight lines are oriented in the image. Using this information we find the longest line segment and the angle at which it is oriented. The image is rotated in the reverse sense as the angle obtained to deskew the bird image.

This method of deskewing did not turn out to be efficient as in cases where there were objects in the background like branches of trees or wires on which the bird is sitting, deskewing was done about these objects which gave completely wrong results.

4.10 Clustering and calculation of inter cluster distances

Clustering is the process of grouping a set of physical or abstract objects into classes of similar objects. It is also called unsupervised learning and is used in a number of applications. An important facet of clustering is the similarity function used. In case of numeric data, a similarity function based on distance such as Euclidean distance is usually used. In our approach, we have implemented the classic k-means algorithm to obtain a predefined number of clusters; six in this case.

4.10.1 K-means Clustering

In this clustering method, the pixels in the image are clustered into a predefined number of clusters. We chose 6 clusters corresponding to the major parts of a bird (bill, crown and nape; back; rump; tail; breast; belly).

A set of 6 points are chosen from the edge extracted image. We went about choosing the initial six points in the following fashion; the edge extracted image is divided into six parts; halved along the row and divided into three along the columns. For each block so obtained, the point where the intensity of the edge detail was maximum, was chosen as the centre of that cluster.

With these six points, then, the clustering process begins. For each point in the edge extracted image (i.e. each point on the edges), the similarity function is calculated. We used Euclidean distance as the similarity function. The point was then allocated to that cluster whose cluster center it is closest to.

The cluster center is recalculated as the mean of the points in the cluster. The process is repeated until the cluster centers stop changing significantly between successive iterations.

We chose this algorithm for its simplicity. The results were sufficient for the purpose of calculating inter cluster distances. These inter cluster distances serve as features that are sent as input to a DST based decision maker that makes a decision on the species of the bird in the query image giving a degree of belief based on the evidence i.e. the features.

4.10.2 Bhattacharyya distance

In statistics, the Bhattacharyya distance measures the similarity of two discrete probability distributions. It is normally used to measure the separability of classes in classification or clusters in clustering.

In its simplest formulation, Bhattacharyya distance can be calculated from the mean and variance of each class in the following way [16]:

$$D_B(k_1, k_2) = \frac{1}{4} \ln \left\{ \frac{1}{4} \left(\frac{\sigma_{k_1}^2}{\sigma_{k_2}^2} + \frac{\sigma_{k_2}^2}{\sigma_{k_1}^2} + 2 \right) \right\} + \frac{1}{4} \left\{ \frac{(\mu_{k_1} - \mu_{k_2})^2}{\sigma_{k_1}^2 + \sigma_{k_2}^2} \right\}, \quad (4.27)$$

where $D_B(k_1, k_2)$ is the Bhattacharyya distance between the k_1^{th} and k_2^{th} cluster, $\sigma_{k_1}^2$ and $\sigma_{k_2}^2$ are the variance of the k_1^{th} and k_2^{th} cluster respectively and μ_{k_1} and μ_{k_2} are the means of the k_1^{th} and k_2^{th} cluster respectively.

In a recent study [15], a number of measures such as Bhattacharyya, Kullback-Leibler, Euclidean, Fischer were studied for image discrimination and it was concluded that Bhattacharyya distance is the most effective texture discrimination for sub-band filtering schemes.

4.10.3 Hausdorff distance

Named after Felix Hausdorff (1868-1942), Hausdorff distance is the maximum distance of a set to the nearest point in the other set. More formally, Hausdorff distance from set A to set B is a maximin function, defined as:

$$h(A, B) = \max_{a \in A} \{ \min_{b \in B} \{ d(a, b) \} \} \quad (4.28)$$

where a and b are points of sets A and B respectively, and $d(a, b)$ is any metric between these points ; for simplicity, we'll take $d(a, b)$ as the Euclidean distance between a and b.

The distance formula above is a directed Hausdorff distance value. In general however, Hausdorff distance is defined as:

$$H(A, B) = \max(h(A, B), h(B, A)) \quad (4.29)$$

Thus, it measures the degree of mismatch between two sets by measuring the distance of the point of A that is farthest from any point of B and vice versa. Intuitively, if the Hausdorff distance is d, then every point of A must be within a distance d of some point of B and vice versa [17]. Thus, the notion of resemblance encoded by this distance is that each member of A be near some member of B and vice versa. Unlike most methods of comparing shapes, there is no explicit pairing of points of A with points of B.

4.10.4 Euclidean distance

This is one of the simplest features that can be extracted from the clusters. Euclidean distance between points $P=\{p_1,p_2,\dots,p_n\}$ and $Q=\{q_1,q_2,\dots,q_n\}$ in Euclidean n -space is defined as :

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}. \quad (4.30)$$

4.11 Iterative Closest Point (ICP)

The iterative closest point algorithm (ICP) is designed to fit points in a target image to points in a control image. The ultimate goal of the algorithm is to minimize the sum of square errors with respect to the closest target points and their corresponding control points. It is important that an initial estimate is made regarding where the overlay of the images should be. An appropriate transformation should be applied based on this initial estimate to align the images coarsely before ICP is applied. The base component of the algorithm calculates the smallest distance between each point in the target image to a point in the control image. These calculated points are then used to form a translation and rotation matrix that is applied over all points in the target image to adjust them towards the control image. This process is repeated numerous times, thus an iterative algorithm, with the end result being a target image with points that are within a specified squared error distance of their corresponding points in the control image.

4.11.1 Introduction to ICP

The data types that ICP can work on are point sets, implicit curves, parametric curves, line segment sets, triangle sets (meshes), implicit surfaces and parametric surfaces. The motivations behind the use of ICP could be as many as shape inspection, motion estimation, appearance analysis, texture mapping and tracking. In our approach, we felt that the bird's colours alone should not be a deciding factor to classify it as it occludes many cases where the birds are not brightly coloured. Also variations like spots, streaks etc. are too subtle to be caught by simple colour dependent schemes. Thus our approach makes use of texture mapping. We aim to map the texture of the query image with the standard images in the database.

4.11.2 Implementation of ICP

ICP always converges monotonically to the local minima with respect to the mean squared error (MSE) objective function.

The implementation consists of four steps discussed below:

1. Closest points calculation

It is important that during this calculation all regions that are not part of the overlying area be removed from further calculations. Examining all the closest distances and removing any correspondences that have distances above a specified threshold will accomplish this. If a closest point calculation is above the specified threshold it signifies that this target model point does not belong in the overlapping section of the control model. This assumes that the initial estimation for transformation given was within reason.

Given two points r_1 and r_2 , the Euclidean distance between them is defined as

$$d(r_1, r_2) = \|r_1 - r_2\| \quad (4.31)$$

Given a point r_1 and a set of points A , the Euclidean distance is defined as

$$d(s, M) = \min_{m \in M} \|m - s\| \quad (4.32)$$

The scene shape is angled to be in the best alignment with model shape M .

$$\begin{aligned} d(s, M) &= \min_{m \in M} \|m - s\| = d(s, y) \\ y &\in M \\ Y &= CP(S, M) \\ Y &\subseteq M \end{aligned} \quad (4.33)$$

(11.3)

where CP is the closest point operator and Y is the set of points closest to M .

Given Y , we can calculate $(rot, trans) = T(S, Y)$ where rot and $trans$ are the rotation and translation matrix obtained after applying the transformation T to points in S and Y . The steps involved in determining rot and $trans$ are outlined below.

2. Covariance matrix calculation

Once the closest distances for each point in the target model to a point in the control model have been discovered it is necessary to calculate a covariance matrix. First it is required to compute the mean value over all the valid points in the target model and then a second mean over all the points they corresponded to in the control model.

The end result is two mean points (xmean , ymean , zmean). Utilizing all valid points from both models and the two mean points as row vectors the covariance matrix calculation is done.

n = Number of valid points used in closest point calculation

A = model point vector $[x_k, y_k, z_k]$

B = corresponding target model point vector $[x_k, y_k, z_k]$

E = control model mean point vector $[x_1, y_1, z_1]$

F = target model mean point vector $[x_2, y_2, z_2]$

$$\text{CovarianceMat} = \sum_{k=1}^n A * B^T - E * F^T \quad (4.34)$$

3. Quarternion matrix calculation

The Quarternion matrix is calculated from the covariance matrix. Let it be denoted by Q.

4. Transformation matrix calculation

The final step in the ICP algorithm is to calculate a transformation matrix. The first step in calculating the transformation matrix is to find the maximum eigen value and its corresponding eigen vector for Q.

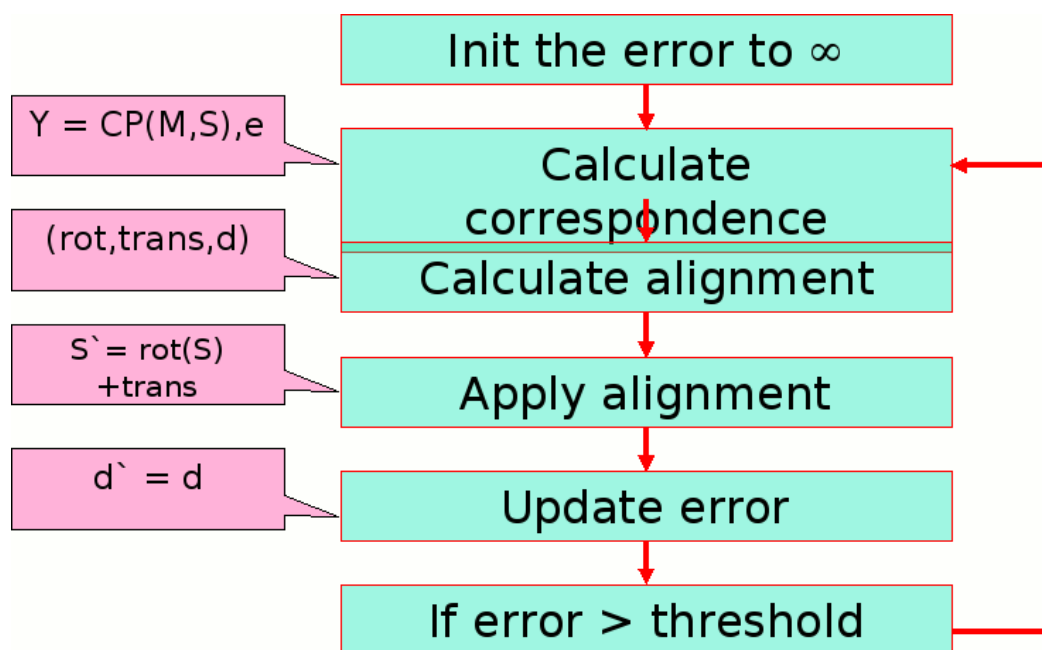


Fig 4.26
ICP Algorithm

The transformation matrix so obtained is applied to the scene to update it and the error of alignment is recalculated. If the error is above a certain threshold, the process is repeated again (iterative closest point).

4.11.3 Analysis of ICP

Each iteration includes 3 main steps. Assume that S is the scene and M the model to which the scene needs to be matched. Then N_S is the number of points in the scene and N_M is the number of points in the model.

1. Finding the closest points :

$O(N_M)$ per each point

$O(N_M * N_S)$ total.

2. Calculating the alignment: $O(N_S)$

3. Updating the scene: $O(N_S)$

It is important that the initial estimate placing the scene next to the model is a strong one so ICP does not have too far to converge. In order for ICP to converge at the correct position it is important the invalid data points do not contribute to any of the calculations.

Applying ICP to the target terrain(yellow) over the control terrain (white) :

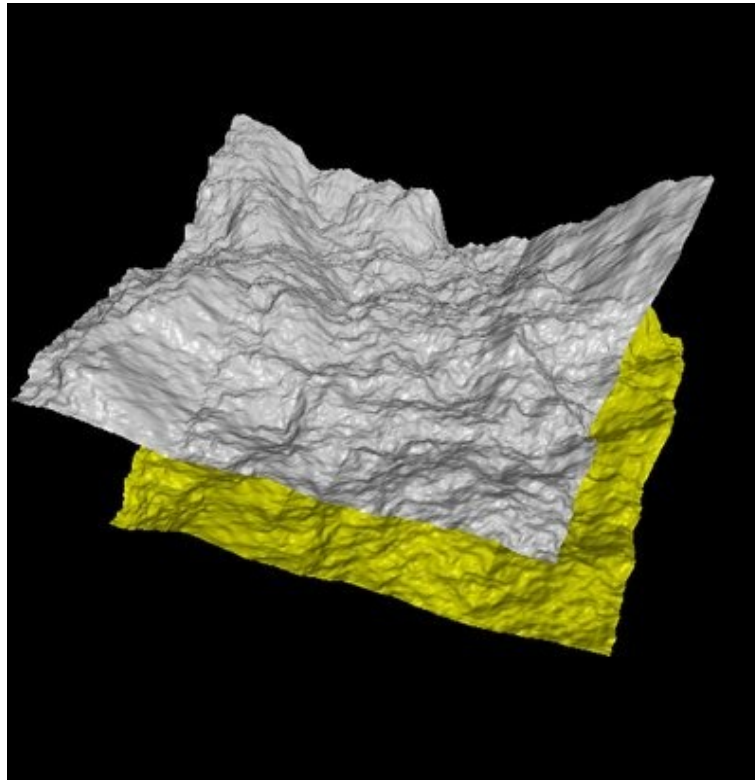


Fig 4.27

Model and Target terrain before convergence with ICP

After applying ICP, the target terrain(yellow) converges as close as possible to target terrain (white).

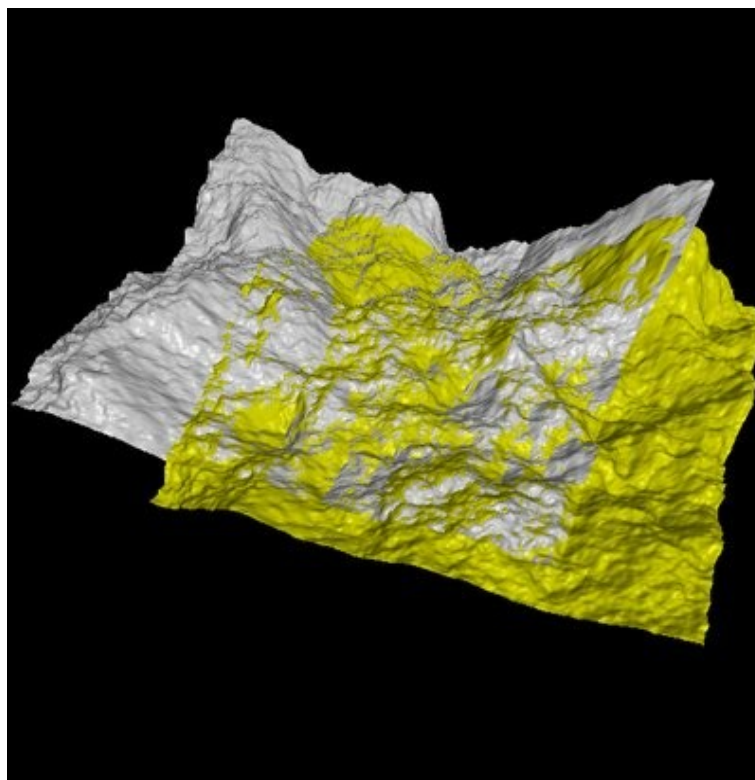


Fig 4.28
Model and Target terrain after convergence with ICP

When handling large data sets it is important that not all valid points are taken into consideration. Some methods for handling these data sets are to randomly sample points, to choose every x^{th} position point, or to only look for points within a specified region of the current point. By randomly sampling points or specifying which points to sample you can dynamically choose how many points you sample on each iteration of ICP. ICP only needs four valid points to compute a transformation matrix, however, it is important that more points are chosen in order for the algorithm to be accurate. If more valid points are used it is more likely the algorithm will converge at an appropriate position. As the algorithm gets closer to convergence it becomes less important that more points are used so you can gradually decrease the points you are sampling.

Variants of the basic ICP algorithm have been proposed:

1. Selecting sample points (from one or both meshes)
Use all points, uniform sampling, random sampling at each iteration, ensuring that samples have normals distributed as uniformly as possible.
2. Matching to points in the other mesh
Closest point, normal shooting, reverse calibration, restricting matches to compatible matches based on colour, intensity, normals etc.
3. Weighting the correspondences

Constant weights, assigning lower weight to point pairs with greater point to point distance, assigning weights based on normal compatibility etc.

4. Rejecting certain (outlier) point pairs
Corresponding points with point to point distance higher than a given hreshold, rejection of worst n% pairs based on some metric, pairs containing points on end vertices etc.
5. Assigning an error metric to the current transform.
6. Minimizing the error metric with respect to transformation.

4.12 Dempster Shafer Theory

The Dempster-Shafer theory [20] , also known as the theory of belief functions, is a generalization of the Bayesian theory of subjective probability. Whereas the Bayesian theory requires probabilities for each question of interest, belief functions allow us to base degrees of belief for one question on probabilities for a related question. These degrees of belief may or may not have the mathematical properties of probabilities; how much they differ from probabilities will depend on how closely the two questions are related.

4.12.1 Introduction to Dempster Shafer theory

Implementing the Dempster-Shafer theory in a specific problem generally involves solving two related problems. First, we must sort the uncertainties in the problem into a priori independent items of evidence. Second, we must carry out Dempster's rule computationally. These two problems and and their solutions are closely related. Sorting the uncertainties into independent items leads to a structure involving items of evidence that bear on different but related questions, and this structure can be used to make computations

There are three important functions in Dempster-Shafer theory:

The basic probability assignment function (bpa or m), the Belief function (Bel), and the Plausibility function (Pl).

The basic probability assignment (bpa) is a primitive of evidence theory. Generally speaking, the term “basic probability assignment” does not refer to probability in the classical sense. The bpa, represented by m , defines a mapping of the power set to the interval between 0 and 1, where the bpa of the null set is 0 and the summation of the bpa's of all the subsets of the power set is 1. The value of the bpa for a given set A (represented as $m(A)$), expresses the proportion of all relevant and available evidence that supports the claim that a particular element of X (the universal set) belongs to the set A but to no particular subset of A . Formally, a function $m:P(x)\rightarrow[0,1]$, is called a basic probability assignment (BPA), when it verifies two axioms. First, the mass of the empty set is zero:

$$m(\emptyset)=0; \tag{4.35}$$

Second, the masses of the remaining members of the power set add up to a total of 1:

$$1 = \sum_{A \in \mathbb{P}(X)} m(A). \quad (4.36)$$

The mass $m(A)$ of a given member of the power set, A , expresses the proportion of all relevant and available evidence that supports the claim that the actual state belongs to A but to no particular subset of A . The value of $m(A)$ pertains only to the set A and makes no additional claims about any subsets of A , each of which have, by definition, their own mass.

From the mass assignments, the upper and lower bounds of a probability interval can be defined. This interval contains the precise probability of a set of interest (in the classical sense), and is bounded by two non-additive continuous measures called belief (or support) and plausibility:

$$\text{bel}(A) \leq P(A) \leq \text{pl}(A) \quad (4.37)$$

The belief $\text{bel}(A)$ for a set A is defined as the sum of all the masses of (not necessarily proper) subsets of the set of interest:

$$\text{bel}(A) = \sum_{B|B \subseteq A} m(B). \quad (4.38)$$

The plausibility $\text{pl}(A)$ is the sum of all the masses of the sets B that intersect the set of interest A :

$$\text{pl}(A) = \sum_{B|B \cap A \neq \emptyset} m(B). \quad (4.39)$$

Also, plausibility and belief are associated by :

$$\text{pl}(A) = 1 - \text{bel}(\overline{A}). \quad (4.40)$$

4.12.2 Motivation behind using DST

The motivation for selecting Dempster-Shafer theory can be characterized by the following reasons:

1. The relatively high degree of theoretical development among the non-

traditional theories for characterizing uncertainty.

2. The relation of Dempster-Shafer theory to traditional probability theory and set theory.
3. The large number of examples of applications of Dempster-Shafer theory in engineering in the past ten years.
4. The versatility of the Dempster-Shafer theory to represent and combine different types of evidence obtained from multiple sources.

4.12.3 Combining the evidence

The purpose of aggregation of information is to meaningfully summarize and simplify a corpus of data whether the data is coming from a single source or multiple sources[23][24]. Familiar examples of aggregation techniques include arithmetic averages, geometric averages, harmonic averages, maximum values, and minimum values. Combination rules are the special types of aggregation methods for data obtained from multiple sources. These multiple sources provide different assessments for the same frame of discernment and Dempster-Shafer theory is based on the assumption that these sources are independent.

Dempster-Shafer Theory as a theory of evidence has to account for the combination of different sources of evidence. Dempster & Shafer's Rule of Combination is an essential step in providing such a theory. This rule is an intuitive axiom that can best be seen as a heuristic rule rather than a well-grounded axiom. In addition to the Dempster rule of combination which is a generalization of Bayes rule, Yager's rule, Inagaki's unified combination rule, Zhang's center combination rule and Dubois and Prade's disjunctive pooling rule are used to aggregate evidence to form a single conclusion.

There are a number of considerations that need to be addressed when combining evidence in Dempster-Shafer theory. Generally speaking, these include the evidence itself, the sources of information, the context of the application, and the operation used to combine the evidence.

4.12.4 DST applied to bird recognition

In a manner similar to application of DST in problems such as facial feature recognition[13] and colour image segmentation[14], we formulate the bird recognition problem as a problem of evidence collection. Each inter cluster distance calculated serves as a feature which can support a definite subset of birds. To solve this problem, Dempster-Shafer theory is employed.

4.12.5 Problem formulation under DST

1. Frame of discernment – The set Θ of possible candidates under our problem domain i.e. the set of birds that we wish to recognize. In our current work, we used only one bird viz a duck.
2. Evidence X is a feature value i.e. inter cluster distances viz. Avg Bhattacharyya distance, avg. Hausdorff distance and avg. Euclidean distance
3. Hypothesis H is the set of candidates selected from the problem domain i.e. a subset of Θ .
4. Mass m in the range of [0,1] assigned to each hypothesis. It is the degree to which the evidence supports the hypothesis.

4.12.6 Implementation of a very simple DST

1. Calculate average values for the three inter cluster distances (our features) viz. Bhattacharyya, Hausdorff and Euclidean distance for the frames picked from the user query video.
2. Define threshold for each for each feature as the average value of the feature values calculated for the duck in the database.
3. Determine BPA for a hypothesis; in our case; duck; given evidence of the existence of a particular feature [18]. i.e.

$$m_{BD}(\{\text{duck}\}) = 1 + \exp(\text{val-thresh} - 1) \quad (4.41)$$

$$m_{BD}(\{\text{not duck}\}) = 1 - M_{BD}(\{\text{duck}\}) \quad (4.42)$$

4. Make similar calculations for each of the other two features.
5. Now combine the result using Dempster's rule.
For each hypothesis processing, the combined effect of two pieces of evidence for a given new hypothesis H on the resulting mass is:

$$m_{ab} = K^{-1} \cdot \sum_{A \cap B = H} m_a(A) \cdot m_b(B)$$

where $K = 1 - \sum_{A \cap B = \emptyset} m_a(A) \cdot m_b(B)$

(4.43)

In our case,

$$m(\{\text{duck}\}) = \frac{(m_{BD}(\{\text{duck}\}) * m_{HD}(\{\text{duck}\}) * m_{ED}(\{\text{duck}\}))}{1 - (m_{BD}(\{\text{duck}\}) * m_{BD}(\{\text{notduck}\}) + m_{HD}(\{\text{duck}\}) * m_{HD}(\{\text{notduck}\}) + m_{ED}(\{\text{duck}\}) * m_{ED}(\{\text{notduck}\}))}$$

5. Calculate $m(\{\text{not duck}\}) = 1 - m(\{\text{duck}\})$
6. The maximum of $m(\{\text{duck}\})$ and $m(\{\text{not duck}\})$ gives the result of the query.

Chapter 5 Results

We built our database for a duck with ten frames picked from a sample video that we shot at the local zoo.

A few of the sample frames after background removal are shown below:

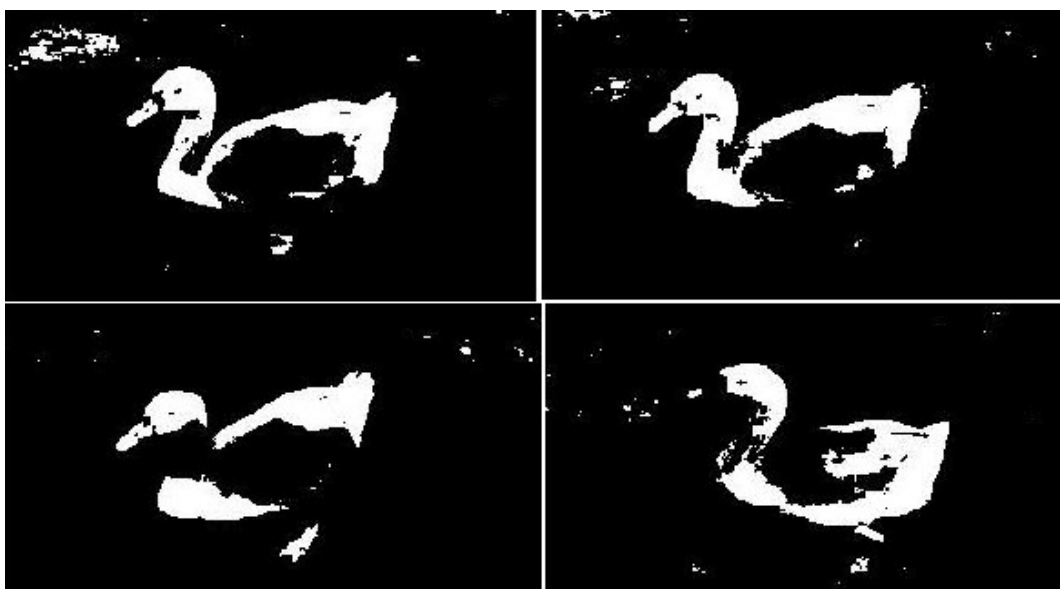


Fig 4.29

*Few sample frames used to build database
LtoR and TtoB: Figure 2, Figure 3, Figure 6, Figure 9*




The feature values were calculated for the ten frames and were tabulated as follows:

FIGURE NO.	BHATTACHARYYA DISTANCE		HAUSDORFF DISTANCE		EUCLIDEAN DISTANCE	
	Minimum	Maximum	Minimum	Maximum	Minimum	Maximum
1	0.0004	0.2821	72.1388	269.8981	8.7034	102.5965
2	0.0001	0.2925	68.2495	230.2542	8.7034	98.9726
3	0.0002	0.1104	59.2368	273.0568	20.5284	88.4738
4	0.0016	0.1469	59.4811	247.3095	20.9610	96.7673
5	0.0001	0.1745	61.0328	246.8441	10.7688	89.7082
6	0.0002	0.0652	68.7314	303.2903	12.7820	118.9067
7	0.0009	0.1702	64.2028	256.7041	23.9408	129.0289
8	0.0001	0.0832	59.4138	214.8046	25.3371	120.1114
9	0.0002	0.0233	60.9262	275.7481	22.3017	115.8198
10	0.0002	0.0624	74.6257	280.4140	3.7558	123.2256

Table 1

Standard feature values for the sample frames of a duck

The user query video is analysed, its background is removed and the procedure for identification is run on selected frames. The results are tabulated as below:

FIGURE	SUPPORT ({Duck})	SUPPORT ({Not Duck})	RESULT	TRUE/ FALSE
	0.0748	0.9252	Not Duck	True
	0.0817	0.9183	Not Duck	True
	0.0523	0.9477	Not Duck	True


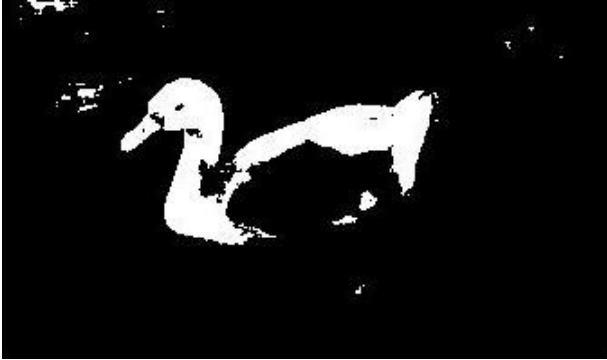
	0.2429	0.7571	Not Duck	False
	0.9041	0.0959	Duck	True

Table 2
Calculated result of comparing user query to database images

Chapter 6 Conclusion and Future Scope

A new approach for bird recognition is proposed using texture pattern in the image. Bird segmentation is the key step involved in bird recognition. Bird segmentation is a difficult task as the birds are normally camouflaged with the background and they are either constantly moving (with respect to the background) or stay still at a place due to which the choice of background subtraction algorithm greatly affects the segmentation process. After background subtraction, wavelets were used to extract edges information, which is further used to cluster the bird. The DST decision maker then makes a decision regarding the bird based on the inter cluster distances such as Bhattacharya distance, Hausdorff distance and Euclidean distances.

An efficient background subtraction technique would have to be developed that would address the problems in this particular domain. Bird movement is quite unpredictable and they tend to blend into their background. A better clustering technique such as super vector machine (SVM) clustering can also be employed.

In the technique outlined in this work, we have worked with average feature values (inter cluster distances). By averaging the values, a lot of information is lost; even though the decision process is rendered simple by it. To truly utilize information regarding texture of the bird, we need to retain all these values. Further, intra cluster distances also need to be made part of decision making process. Texture information extracted from such large amounts of data can narrow the choice of possible birds for a user query.

Further, the database needs to be expanded to many more birds. The inter and intra cluster distances calculated for these database images need to be analyzed. Patterns may be discerned from these values i.e. we can determine the variation in feature values as the poses of the bird vary. Texture information for the same bird in different positions yields more insights into classification and helps the decision maker make a more informed decision.

Chapter 7 References

7.1.1 Papers

- [1] Madirakshi Das and R. Manmatha. Automatic Segmentation and Indexing in a Database of Bird Images.
- [2] G. Pass and R. Zabih. Histogram refinement for contentbased image retrieval. *Workshop on Applications of Computer Vision*, pages 96-102, 1996.
- [3] J. Ashley and et al. Automatic and semi-automatic methods for image annotation and retrieval in qbic. SPIE Conference on Storage and Retrieval for Image and Video Databases, pages 24-35, 1995.
- [4] M. Das, R. Manmatha and E. M. Riseman. Indexing flowers by color names using domain knowledge-driven segmentation. IEEE Workshop on Applications of Computer Vision, pages 94–99, Oct 1998.
- [5] M. Das, R. Manmatha and E. M. Riseman. Indexing flower patent images using domain knowledge. IEEE Intelligent Systems, 14(5):24–33, Oct 1999.
- [6] N.J.B. McFarlane and C.P. Schofield. Segmentation and tracking of piglets in images. *Machine Vision and Applications* (1995) 8:187-193
- [7] C.C. Reyes-Aldasoro and A. Bhalerao. The Bhattacharya space for feature selection and its application to texture segmentation.
- [8] Zhengyou Zhang. Iterative Point Matching for Registration of Free-form Curves (1992).
- [9] Horace H. S. Ip and Richard C. K. Chiu. Evidential reasoning for facial gesture recognition from cartoon images.
- [10] Sanjoy Kumar Saha, Amit Kumar Das, and Bhabatosh Chanda. An Automatic Image Segmentation Technique Based on Pseudo-convex Hull
- [11] Hae Yong Kim and Sidnei Alves de Araujo. Grayscale template matching invariant to rotation, scale, translation.
- [12] Amara Graps. An introduction to wavelets. IEEE Computational Science and Engineering, Summer 1995, vol. 2, num. 2
- [13] Horace H. S. Ip and Richard C. K. Chiu. Evidential Reasoning for Facial

Gestures Recognition from Cartoon Images. Australian and New Zealand Conference on Intelligent Information Systems Proceedings: 397-401 (1994).

[14] J. B. Mena, J.A. Malpica. Color image segmentation using Dempster-Shafer theory of evidence for the fusion of texture..

[15] A. Bhalerao, N. Rajpoot. Selecting discriminant sub-bands for texture classification. BMVC, Norwich, UK, September 2003.

[16] C.C. Reyes-Aldasoroa, A. Bhaleraob. The Bhattacharyya space for feature selection and its application to texture segmentation

[17] Daniel P. Huttenlocher, Gregory A. Klanderman, and William J. Rucklidge. Comparing Images Using the Hausdorff Distance. IEEE Transactions on pattern analysis and machine intelligence, Vol 15, No 9, September 1993.

[18] Franz Rottensteiner, John Trinder, Simon Clode, Kurt Kubik, Brian Lovell. Building Detection by Dempster-Shafer Fusion of LIDAR Data and Multispectral Aerial Imagery. 17th International Conference on Pattern Recognition (ICPR'04) IEEE

7.1.2 Books

[19] Feature Extraction and Image Processing by Mark S. Nixon and Alberto S. Aguado.

[20] Dependability Modelling Under Uncertainty: An Imprecise Probabilistic Approach by Philipp Limbourg.

7.1.3 Web pages

[21] Amara's wavelet page : <http://www.amara.com/current/wavelet.html>

[22] Index to series of tutorials to wavelet transform by Robi Polikar : <http://users.rowan.edu/~polikar/WAVELETS/WTtutorial.html>

[23] Shafer, Glenn; Dempster–Shafer theory, 2002
<http://www.glennshafer.com/assets/downloads/articles/article48.pdf>

[24] Combination of Evidence in Dempster-Shafer Theory
www.sandia.gov/epistemic/Reports/SAND2002-0835.pdf

Chapter 8 List of Figures

Figure no.	title	page no.
4.1	A sequence of frames obtained after applying approximate median filter background subtraction to a video	12
4.2	Coverage of time-frequency plane by Fourier basis functions in STFT where the window is a square wave (from [21])	14
4.3	Coverage in time-frequency plane by Daubechies wavelet basis (from [21])	15
4.4	Stages of multiresolution in calculation of DWT using FWT Daubechies and Mallat's Algorithm	17
4.5	Image and its 1,2,3 level dyadic wavelet transform	18
4.6	First level decomposition using db4 DWT	19
4.7	Second level decomposition using db4 DWT	19
4.8	Different types of objects (a) Pseudo-convex (b) Concave(c) Convex	22
4.9	Input image to pseudo convex hull algorithm	24
4.10	Image after edge detection by DWT	24
4.11	Binary image (4.10) after thresholding	25
4.12	Image (4.11) after being sampled at specified angles and then hulled	25
4.13	Image (4.12) after connected component analysis and component labelling	26
4.14	Image (4.13) after largest component extraction and application of morphological close operation	26
4.15	(a) Geometry of GHT (b) Structure of R-table	32
4.16	(a) Template (b) Image to searched in	33
4.17	Accumulator array for the above template matching in GHT	34
4.18	(a), (b) Geometry of invariant GHT (b) Structure of R-table	35
4.19	Original image – before Hough transform	39
4.20	Result of Hough transform on binarized Canny edge detected image	39
4.21	Deskewed image after application of Hough transform	40
4.22	Original input image for Hough transform	40
4.23	Image after applying Hough transform	41
4.24	Deskewed image(4.19) after Radon transform	42
4.25	Image(4.22) after applying Radon transform	42
4.26	ICP Algorithm	48
4.27	Model and Target terrain before convergence with ICP	49
4.28	Model and Target terrain after convergence with ICP	50
4.29	Few sample frames used to build database	55

Chapter 9 List of Tables

TABLE NO.	TITLE	PAGE NO.
1	Standard feature values for the sample frames of a duck	55
2	Calculated result of comparing user query to database images	56