GENERALIZED BEAM ANGLE STATISTICS FOR SHAPE DESCRIPTION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÖMER ÖNDER TOLA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

AUGUST 2004

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Ayşe Kiper
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Dr. Nafiz Arıca
Co-Supervisor

Prof. Dr. Fatoş Yarman - Vural
Supervisor

Examining Committee Members

Prof. Dr. Faruk Polat (METU,CENG) _____

Prof. Dr. Fatoş Yarman - Vural (METU,CENG) _____

Asst. Prof. Dr. Erol Şahin (METU,CENG) _____

Asst. Prof. Dr. Pınar Duygulu - Şahin (BİLKENT,CENG) _____

Mutlu Uysal (STM SAV. TEK. MÜH. ve TİC. A.Ş.) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Lastname: Ömer Önder Tola

Signature :

ABSTRACT

GENERALIZED BEAM ANGLE STATISTICS FOR SHAPE DESCRIPTION

Tola, Ömer Önder

M.S., Department of Computer Engineering

Supervisor: Prof. Dr. Fatoş Yarman - Vural

Co-Supervisor: Dr. Nafiz Arıca

August 2004, 84 pages

In this thesis, we introduce a new shape descriptor and a graph based matching algorithm to detect a template shape in an image that contains a single object. The shape descriptor, Generalized Beam Angle Statistics, GBAS is obtained with the generalization of the boundary based shape descriptor, Beam Angle Statistics, BAS [5]. GBAS improves BAS so that it can compute the feature vector of a boundary point without the requirement of the parametric boundary representation. This way, it can be used in matching an individual edge pixel with a boundary point of template shape, even if it is not possible to extract the shape boundary in the image with the available techniques.

Given a template shape, the matching algorithm solves the correspondence problem between the sampled boundary points of the template and the edges of the query image, using the GBAS feature vectors and the spatial information of edges. The match graph represents the correspondence problem and the optimum path on this graph gives the solution of it. Optimum path is found using a polynomial time algorithm that is based on the dynamic programming approach.

In the experiments, we show that the proposed shape descriptor is very powerful and the matching algorithm is capable of detecting a template shape in edge detected

images under a variety of transformations and noise.

Keywords: Shape Recognition, Beam Angle Statistics, Generalized Beam Angle Statistics, Matching Algorithm

ÖZ

ŞEKİL BETİMLEMEK İÇİN GENELLEŞTİRİLMİŞ KERTERİZ AÇISI İSTATİSTİKLERİ

Tola, Ömer Önder

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Fatoş Yarman - Vural

Ortak Tez Yöneticisi: Dr. Nafiz Arıca

Ağustos 2004, 84 sayfa

Bu tezde, bir şablon şekli tek bir nesne içeren bir imgede bulmak için yeni bir şekil betimleyicisi ve çizgeye dayanan bir eşleştirme algoritması önerilmektedir. Genelleştirilmiş Kerteriz Açısı İstatistikleri, GKAİ adını verdiğimiz bu şekil betimleyicisi daha önce önerilen ve bir sınır tabanlı şekil betimleyicisi olan, Kerteriz Açısı İstatistikleri, KAİ'nin genelleştirilmesi ile elde edilmiştir [5]. GKAİ, KAİ'yi parametrik sınırlara gerek duymadan bir sınır noktasının öznitelik vektörünü hesaplayacan şekilde geliştirir. Böylece, resimdeki şekil sınırlarını var olan yöntemlerle elde etmek mümkün olmasa dahi, bireysel bir kenar pikselini şablon şeklin bir sınır noktasıyla eşleştirmek mümkün olur.

Bir şablon şekil verildiğinde, eşleştirme algoritması, şablon şeklin örneklenmiş sınır noktaları ile sorgu imgesinin kenarları arasındaki eşleşme problemini kenarların GKAİ öznitelik vektörleri ve uzaysal bilgisini kullanarak çözer. Eşleştirme çizgesi, eşleşme problemini temsil eder ve bu çizge üzerindeki en uygun yol problemin çözümü verir. En uygun yol, dinamik programlama yaklaşımına dayanan polinom zamanlı bir algoritma ile bulunur.

Yapılan deneylerde, önerilen şekil betimleyicisinin çok güçlü olduğunu ve eşleştirme algoritmasının çeşitli dönüşüm ve gürültü tipleri altında bir şablon şekli kenarları

taranmış bir imge içerisinde tespit edebildiğini gösteriyoruz.

Anahtar Kelimeler: Şekil Tanıma, Kerteriz Açısı İstatistikleri, Genelleştirilmiş Kerteriz Açısı İstatistikleri, Eşleştirme Algoritması

ACKNOWLEDGMENTS

I would like to thank to my family, friends, supervisors, examining committee members and the society of the METU, Computer Engineering Department for their support, encouragement and invaluable ideas during my thesis study. Thank you all.

In memory of my grandfathers Ramazan Can and Ömer Tola.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Shape is an important visual property of objects. It has much superiority compared to other visual features such as color and texture. Firstly, humans make abstractions of objects with their shapes. When two people talk about an object, the common property that both of them imagine is mostly its shape. Other properties such as the color and texture might be a secondary issue. Secondly, shape is a discriminative property of objects that lead to object identity but, other properties are not. This is especially true for man-made objects. Moreover, humans give different names to objects that have different shapes. For instance, car manufactures give a distinctive name for each model of car they produce. Therefore, formation of object classes is strongly influenced with the shapes of objects rather than other low level visual features. In addition, the shapes of objects are closely related to their function. For example; all birds have wings, all fish have fins, etc. Importance of this fact from the perspective of object recognition is that object classes are formed according to their functions. In other words, objects that have similar functions are grouped into the same classes. As a result, shape features provide a powerful clue to object identity and functionality and determine their class.

Shape recognition is a difficult problem because the detection of shapes in images cannot be performed successfully. This has many reasons ranging from the complexity of the shapes, to the lighting conditions such as shadows and specular reflection. Additionally, depth information is lost during the formation of shapes with the perspective projection of the objects on the 2D imaging plane. Therefore, even if the shape is

detected correctly, the loss of information during the projection must be handled. Yet another problem is the noise, which is present in images due to many sources, changes the forms of shapes. Occlusion is another factor that prevents the detection of shapes. Accordingly, shapes must be detected even if some parts of them are not available.

Shape recognition involves the detection and classification of shapes. Related methods can be divided into two classes. In the first class, detection and classification phases are separated from each other. First, the object is detected and then it is classified. Detection phase uses only the information gathered from the image. Segmentation is the usual process for detection. Discontinuities between unrelated parts and similarity of the related parts are used in the process. After the segmentation, each connected part is assumed to be an object. Then, various features of these object regions are used to classify them.

In the second approach, templates or models are used to recognize shapes [34, 20, 23, 1, 18]. These methods locate the template shape within the image. Since the class of the template shape is known a priori, the classification step is eliminated. This way, only the objects of interest are recognized within images. The main advantage of template based recognition is the improvement of the shape recognition process combining the detection and recognition phases.

During the MPEG-7 studies, it is shown that boundary based shape descriptors are more successful in the retrieval of shapes. The boundary based shape descriptor, BAS outperformed many other shape description methods found in the literature [5]. However, as many other similar shape descriptors [25], BAS requires the extraction of the shape boundary for description. Therefore, it has limited applicability.

In this study, we follow the template based shape recognition paradigm in order to extend the applicability of the boundary based shape descriptor BAS to general images. Accordingly, we suggest a method that performs the detection of a given template shape in images that contain a single object using the shape descriptor, Generalized Beam Angle Statistics, GBAS and a graph based matching algorithm. GBAS, which is obtained with the generalization of BAS, improves BAS so that it can compute the feature vectors of boundary points in the lack of parametric boundary representation. This way, it can be used to compute the features vectors of edge pixels even if it is not possible to extract the shape boundary with the available

techniques. Then, the detection problem is reduced to matching the boundary points of the template shape and the edges. This is achieved by a graph based matching algorithm, which transforms the problem to the determination of the optimum path. A polynomial time algorithm based on the dynamic programming approach is employed for the determination of the optimum path.

The thesis is organized as follows. The related background is given in chapter 2. Next, the proposed shape descriptor Generalized Beam Angle Statistics, (GBAS) is introduced in chapter 3. Chapter 4 describes the matching algorithm used for object detection. The experiments that are carried out for the evaluation of the performance of GBAS and the matching algorithm are given in chapter 5. Finally, chapter 6 concludes the thesis and gives the future research directions.

CHAPTER 2

BACKGROUND

In this chapter, first we give an overview of shape representation techniques. Then, we describe Beam Angle Statistics (BAS) [5], the shape descriptor that we have generalized to create a new shape descriptor, Generalized Beam Angle Statistics, GBAS for object detection. Finally, we review the edge detection methods and discuss the Canny edge detector which we use as a preliminary step for description.

2.1 Shape Representation Techniques

Shape representation techniques can be analyzed in two main classes: region and boundary based techniques [25]. Region based techniques use the whole information content of shapes for description. Basic representation methods are region decompositions into simple surface primitives such as polygons and quad-trees, bounding regions such as the minimum enclosing rectangle and the convex hull and internal features such as the skeletons and shape matrices [12]. Examples of region based shape descriptors are [36, 28, 22, 3, 21]. Since we utilize a boundary based shape descriptor we do not elaborate on the region based shape representation techniques any further.

Boundary based shape representation techniques can be classified into the following classes: simple boundary descriptors, parametric contours, set of boundary points, curve approximations and transform domain techniques.

Simple boundary descriptors mainly focus on a single property of shapes and represent that property with a number. These are global descriptors so that they

do not represent local shape characteristics. However, such characteristics affect some of these descriptors drastically. Boundary length, diameter, centroid, bending energy, eccentricity, convexity, major axis orientation, circular variance and elliptic variance [12, 17, 26, 37] are among the simple boundary descriptors. Disadvantage of these descriptors is that they cannot be used to describe shapes alone because completely different shapes can be mapped to similar simple boundary descriptors. Therefore, they are usually used in conjunction with other methods to eliminate false alarms.

Parametric boundary descriptors represent the shape boundary as an ordered set of boundary points, which are accessible through a parameter. The simplest parameterization is the indices of boundary points. This corresponds to the digital equivalent of the arc length parameterization, which is useful for continuous curves. However, the index parametrization is not equivalent to the arc length parameterization of digital curves, for instance in 8-connected boundaries. Most of the parametric boundary descriptors are also boundary representations because they define the boundary uniquely. Examples of parametric boundary descriptors are vectors, complex signals, chain codes and shape numbers [12].

Set of boundary points is an alternative to the parametric representations. It aims the removal of the dependency of shape descriptors to the ordering of boundary points. This is especially useful in scenarios where it is required to describe shapes even if the boundary cannot be extracted with the available techniques. Contour following algorithms extract digital shape boundaries from pre-segmented binary images. However, extraction of shape boundaries is not always possible in grayscale or colored images. Therefore, shape descriptors which does not depend on the parametric boundary representation, are useful for the detection of objects in general images.

Chords [26] and shape contexts [21], are examples of the set of boundary points descriptors. The shape descriptor proposed in this thesis, GBAS is also based on the set of boundary points representation. GBAS assigns a feature vector to each boundary point without using the parametric boundary information. This makes it a suitable shape descriptor to be used in object detection.

Parametric boundary descriptors such as vectors and complex signals are very redundant. Curve approximation methods try to reduce the redundancy of these descriptors by using only the "important" boundary points for shape description. They

partition the curves into parts and then represent these parts using predefined geometric primitives such as lines, arcs and B-splines. Polygon approximation methods are the most popular curve approximation methods, which use line segments to represent these parts. These methods are not practical for natural objects [37].

Transformations are useful mathematical tools that allow one to replace the problem domain with a more convenient one in which the solution is more easily constructed. Shape description techniques make use of the transforms as well. Transform domain techniques, which are divided into the mono-scale (Fourier, Sine, Cosine, Laplace, Z, etc.) and multi-scale (short-time Fourier, Wavelets, Gabor, Scale-space, etc.) transforms, use transform coefficients, transform measures such as energies and transform statistics as transform domain features [12]. Main difference between the mono-scale and multi-scale transforms is that while the former are one-to-one, latter are usually one-to-many. Multi-scale transforms map the input signal based on the supplied scale parameter to the transform domain. On the other hand, mono-scale transforms do not require any parameter for mapping. Fourier descriptors [17], curvature scale space [15, 16, 31] and wavelet descriptors [35] are examples of transform domain descriptors.

2.2 Beam Angle Statistics: BAS

Beam Angle Statistics, BAS, is a boundary based shape descriptor that describes shapes by a set of moment functions derived from the beam angles of boundary points. Like many other shape descriptors [37, 25], it requires parametric boundary representation for description.

In this section, first we give some introductory definitions and then define BAS and discuss its properties. Finally, we conclude with a discussion that argues the additional requirements of a BAS based shape descriptor which will be used in shape detection.

2.2.1 Shape Boundary

Shape boundary is an ordered set of points lying on the discrete cartesian coordinate system, Z^2. A shape boundary composed of L points is represented as;

$$S = \{s_1(x_1, y_1), s_2(x_2, y_2), \ldots, s_L(x_L, y_L)\} \tag{2.1}$$

where $s_1(x, y)$ and $s_L(x, y)$ are the starting and ending points of the boundary respectively. For closed boundaries, $s_1 = s_L$.

2.2.2 Curvature

Curvature is the rate of change in the slope of the curve tangent. Intuitively, it is an indication of the amount of bended ness in the curve. For instance, the curvature of a straight line is zero. This result can be verified intuitively since a straight line is not bended. Consider a circle. The curvature of each point in a circle is constant. Again, intuitively, a circle is bended by a constant amount from all points. Besides these intuitive understandings, curvature can be used to extract other geometrical means such as convex or concave corners, inflection points, etc. [12].

In mathematical terms, the curvature, $k(t)$ of a two-dimensional parametric curve, $c(t) = (x(t), y(t))$ is defined as follows;

$$k(t) = \frac{x'(t)y''(t) - x''(t)y'(t)}{(x'(t)^2 + y'(t)^2)^{\frac{3}{2}}} \tag{2.2}$$

Clearly, in order to compute the curvature of $c(t)$, first and second order derivatives of $x(t)$ and $y(t)$ are required. Note that, this definition serves for the curvature analysis of continuous curves.

Curvature is an important shape feature and in its stand-alone form it can be used as a shape descriptor. However, this can be expensive because as the length of the curve increases, the length of the feature vector increases as well. In order to overcome this difficulty one can use different methods for defining a curvature based feature vector. Uniform sampling of the curvature function, non-uniform sampling of the curvature function using the important curve points such as high-curvature points, convex or concave corner points, etc., statistics of global measures derived from the curvature function such as mean-curvature, variance, standard deviation, etc., and important global measures of shape complexity such as bending-energy can all be used as features derived from curvature [12].

2.2.3 Beam Angle Function

Beam-Angle function is a measure of curvature of a discrete curve. Given a shape boundary, S, the Beam-Angle function, $c_n(s_i)$ of a boundary point $s_i(x, y)$, is defined

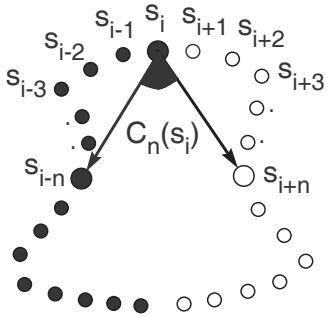


Figure 2.1: Beam Angle Function, $c_n(s_i)$.

as the angle between the vectors $F_n(s_i) = \overrightarrow{(s_i, s_{i+n})}$ and $B_n(s_i) = \overrightarrow{(s_i, s_{i-n})}$ where n indicates the number of points to the right and left of s_i that will be used in the computation (see figure 2.1). One can think n as a parameter that arranges the scale at which the beam angle of s_i would be evaluated. Beam angle function is introduced in [5].

For a straight line, the Beam-Angle function of any point on the line must be π. For a convex point, the value of the Beam-Angle function must be less than π, in a sufficiently large neighborhood, n. Similarly, for a concave point, the value of the Beam-Angle function must be greater than π, for a sufficiently large neighborhood n. Other geometrical means related to curvature can be represented similarly by the Beam-Angle function as well.

Beam Angle Statistics is an innovative way of defining a curvature based shape descriptor. In all of the techniques discussed in 2.2.2, an estimation of the curvature at each boundary point must be performed before defining curvature based features. This is problematic because the resultant descriptor depends on the scale, n which is used to compute the curvature. However, BAS does not require the estimation of curvature at each boundary point. Rather, it uses the curvature estimates performed at all scales. For each boundary point, it creates the feature vector from the moments of curvature estimates performed at all scales. This results in a robust shape descriptor which is independent of the scale.

2.2.4 Beam Angle Statistics (BAS)

Given a shape-boundary $S = \{s_1(x_1, y_1), s_2(x_2, y_2), \ldots, s_L(x_L, y_L)\}$, there exists a set, $B_i = \{c_1(s_i), c_2(s_i), \ldots, c_M(s_i)\}$ of beam angles for each boundary point s_i where $M = L/2$. Then, each boundary point s_i is assigned a feature vector derived from its set of beam angles, B_i. The feature vector is composed of the beam angles' central moments;

$$\Xi\{c^m(s_i)\} = \frac{1}{M}\sum_{j=0}^{M}(c_j(s_i) - \mu_i)^m, \text{ for } m = 1, 2, \ldots, D \tag{2.3}$$

where

$$\mu_i = \frac{1}{M}\sum_{j=0}^{M} c_j(s_i) \tag{2.4}$$

Here, Ξ is the expected value operator. Then, the D-dimensional feature vector of s_i is given by;

$$F(s_i) = \left[\Xi\{c^1(s_i)\}, \Xi\{c^2(s_i)\}, \ldots, \Xi\{c^D(s_i)\}\right] \tag{2.5}$$

Combining the corresponding entries of all boundary points' feature vectors;

$$\Gamma(j) = \{\Xi\{c^j(s_i)\} | i = 1, 2, \ldots, L\} \tag{2.6}$$

one can create a set of D-function that represents the shape being described;

$$\Gamma = \{\Gamma(1), \Gamma 2, \ldots, \Gamma(D)\} \tag{2.7}$$

These functions are the 1^{st}, 2^{nd} and D^{th} moments of the boundary points' beam angles. The statistics created in this fashion are called Beam Angle Statistics [5].

In figures 2.2 (a) and (b), a template shape and its BAS moment functions are presented, respectively. In (b), horizontal axis shows boundary points and the vertical axis shows the moments of the boundary points. Moreover, thick and thin lines are the 1^{st} and 2^{nd} moments of the BAS shape descriptor, respectively. Additionally, in the figures, some boundary points and their corresponding BAS moment function points are marked with numbers from 1 to 4. Observe that, in BAS, there exists a local minima for each convex point and a local maxima for each concave point. For instance, the marked points 1, 2 and 4 are convex points which have a local minima in the BAS functions. On the other hand, point 3 is a concave point which has a local maxima. We say that BAS preserves the concavities and convexities of shapes, therefore BAS moment functions are consistent with the human visual system. Note

that, in the BAS moment functions 1^{st} point is marked twice which is due to the periodicity of BAS.

2.2.5 Properties of BAS

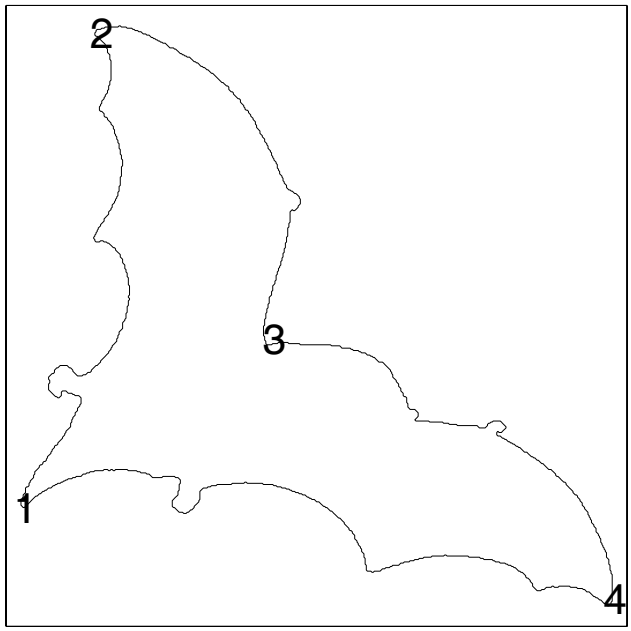
In this section, first we discuss the properties of the BAS shape descriptor. Then, we analyze the weaknesses of BAS which leads us to develop a new shape descriptor for object detection.

2.2.5.1 Superiorities of BAS

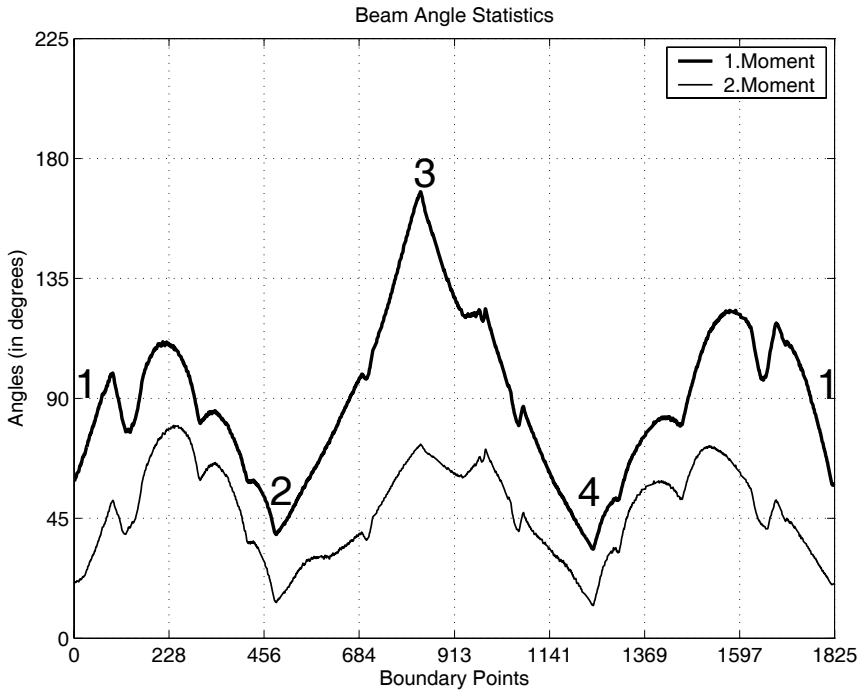
1. **BAS is consistent with the Human Visual System:** BAS shape descriptor is successful in capturing the concavities and convexities of shapes which are known to be important visual clues in human recognition of shapes. Each convexity and concavity has a corresponding minima and maxima in BAS, respectively. This leads to the representation of parts of a shape as well. Consequently, BAS is one of the best curvature based shape descriptors and it is shown that it performs better than most of the shape descriptors found in the literature [5].

2. **BAS is robust to noise:** One of the desirable properties of a shape descriptor is its robustness under noise. Consider a smooth shape, such as a circle. Now consider the same shape with a little perturbation noise added. Clearly, a human observer would recognize the shape correctly and it would describe the shape as a "noisy circle". We expect a similar behavior from a shape descriptor. The noise added to the shape must not change the overall form of the descriptor much but there must be a difference with the original descriptor so that noise can be captured. In BAS, noise does not effect the overall form of the shape descriptor but the effect of noise can be captured. Note that as the amount of noise increases, then the effects of it increases as well.

3. **BAS is robust to occlusion:** Occlusion is a common deformation that is produced in natural images due to the ordering of objects. Therefore, insensitivity of a shape descriptor to occlusion is a desirable property. Although BAS uses whole shape boundary to compute the feature vector of each boundary point, for modest levels of occlusion it performs well.



(a)



(b)

Figure 2.2: A sample shape from the MPEG-7 CE Shape-1 Data Set is shown in (a). The BAS shape descriptor of shape in (a) is given in (b).

4. **BAS is invariant to Rotation, Scaling, Translation and Reflection:** BAS shape descriptor is shown to be invariant under rotation, scaling, translation and reflection transformations. These are very important affine transformations that play role in the evaluation of shape similarity. Therefore, it is a big plus for BAS.

2.2.5.2 Weaknesses of BAS

1. **BAS requires parametric boundary representation:** BAS implicitly assumes that the boundary of the shape to be described is extracted and made available for description. This assumption of processing a digital image in a sequence of stages producing different representations at each stage dates back to David Marr. In his book about computer vision [27], he suggests a step-by-step algorithm for the extraction of information from images. At first glance, this idea seemed to be valuable and then scientists begin to attack different aspects of the ultimate problem of information extraction from digital images. However, it become apparent that, each representation produced at the output of every step making explicit some parts of the information contained, also results in the loss of some useful information. Thus, attacking the ultimate problem of information extraction from images, we might be more successful if we can exploit the whole information present in the image in a single step or in as small number of steps as possible. BAS is not applicable to a raw image.

2. **BAS is not suitable to the output of poor segmentation algorithms:** To our best knowledge, segmentation algorithms proposed so far are far from being optimally segmenting images, in the sense of the segmentation that would be expected from a human operator. Consequently, if one wants to detect shapes within images first applying the existing segmentation algorithms and then trying to extract shape information using different means, then it must be prepared to handle poor segmentation results. Using a very optimistic shape descriptor such as BAS, this is not possible. It is desirable to make BAS more robust to poor segmentation results.

3. **BAS' statistical stability decreases as the resolution of shapes decreases:** BAS shape descriptor is proven to be scale invariant. However, as

the scale of shapes decreases, then the number of beam-angles decreases proportional to the length of the boundary. Therefore, for shapes having small number of points, sufficient statistics cannot be computed. This affects noise tolerance of BAS for small scaled shapes negatively.

2.2.5.3 Towards a new Shape Descriptor

We have investigated most of the important properties of BAS. In conclusion, we can say that, BAS is a good shape descriptor as long as the input, shape-boundary, is presented to BAS carefully. On the other hand, in real images, many forms of noise exist and we do not have a procedure that will present input to BAS in the required form. Thus, in order to use BAS as a shape descriptor when several forms of noise exist in the input it must be improved. Desired properties of a BAS based shape descriptor that will be used for object detection in real images are as follows;

1. The shape descriptor must be able to compute the feature vectors of boundary points even when full ordering of the shape-boundary points is not available.

2. The shape descriptor must be able to compute the feature vectors of boundary points in case there are excessive or missing parts in the input as a result of poor segmentation results.

3. The shape descriptor must behave well under different kinds of noise that can be present in images.

4. The shape descriptor must be able to extract sufficient number of beam-angles in cases where the scale of the shape is small.

In the following chapters, we will show how such a shape descriptor can be defined based on BAS.

2.3 Edge Detection

Segmentation is one of the first steps that are performed in most of the image analysis problems. Detection of discontinuities is a basic approach for segmentation. Edges are the most general means of discontinuities and can be defined as the regions of an image where gray-level variations are sharp (abrupt). Although detection of edges is

not enough to discover meaningful structures in an image because the sources of edges can be quite diverse including noise, shadows, etc., it has been widely used prior to object detection in many studies such as the Hough Transforms [19, 24].

Since boundaries of objects produce strong edges, we use edge detection to extract possible regions of images that include object boundaries. In this section, first we review general edge detection approaches and then we describe the Canny edge detector, which we have used to detect edges in our object detection method.

2.3.1 Edge Detection Methods

Edge detection methods can be classified into the image domain and the transform domain techniques. In the image domain, primary mathematical tool used for edge detection is differentiation. On the other hand, Fourier transform is the most popular transform domain based technique used for edge detection.

Differentiation based edge detection methods either use first-order differentiation with the Gradient operator or second-order differentiation with the Laplacian operator. First order differentiation operator Gradient can be used to detect maximum rate of change. Usually, a large response is expected from edge pixels and by use of an appropriate threshold, these pixels can be detected. On the other hand, second order differentiation operator Laplacian produces zero-crossings at the edges. These locations effectively describe the edges.

Fourier transform based edge detection relies on the fact that edges produce high frequency components in the frequency domain. Using this property and applying high-pass filtering enhances edges. Again, a threshold value can be used to detect edges after the filtering operation.

2.3.2 Problems of Edge Detection

Detection of edges using both of these mathematical tools has three major problems: noise, localization and thresholding [8].

Firstly, noisy pixels cannot be separated from edges by these operators. Consequently, before using any of these tools, low-pass filtering or smoothing is applied for noise reduction. However, there is a side effect of low-pass filtering. As the scale of low-pass filter increases, so as the spreading and the displacement of edges. This

means that there is a trade-off between the amount of noise reduced and the error made in the localization of edges.

Third problem is the use of a single threshold value. Using a simple threshold does not provide the best possible edge localization results that can be obtained from edge enhanced images because a large value of the threshold discards weak edges producing too many fragmentations and a small value of the threshold introduces too many edges. All of these problems are addressed in the Canny edge detector.

2.3.3 The Canny Edge Detection Method

Canny edge detection was proposed by Canny in [10]. It provides solutions for the major problems of edge detection; noise reduction, edge localization and thresholding. Firstly, for the trade-off between noise reduction and edge localization, it is shown that, Canny edge detector makes the optimal choice between these issues in the case of step edges and Gaussian noise [14]. Secondly, for the edge localization problem, Canny edge detector performs non-maximal suppression and hysteresis thresholding. Non-maximal suppression simply thins the magnitude of gradient to prevent thick edges at the output. Hysteresis thresholding employs two threshold values, low and high, to decide which of the edge-enhanced pixels correspond to edges. This ensures the appearance of weak edges which are detected using the low threshold value, in the final output if they are connected to strong edges which are determined by the high threshold value. This way, unnecessary weak edges are avoided but weak edges that might be useful are kept. Details of the standard implementations of Canny edge detection are as follows;

1. **Reduction of Noise:**
   In the first step, before trying to detect edges, noise reduction is performed. This is usually achieved by Gaussian smoothing.

2. **Computation of the Gradient Vector:**
   Both the direction and the magnitude of gradient vector is used in Canny edge detection. Gradient can be computed by different operators such as Sobel, Prewitt and Roberts.

3. **Non-Maximum Suppression:**

Direction of the gradient is used in non-maximum suppression. Direction along the edge is traced and any values smaller than the local maximum is suppressed. As a result, gradient image is thinned to produce a single response for a single edge.

4. **Hysteresis Thresholding:**

   Hysteresis thresholding eliminates the fragmentation of edge segments by employing two thresholds: low and high. Any pixel having gradient magnitude greater than high threshold is immediately considered as an edge. Any pixel having a gradient magnitude less than the low threshold is eliminated to be an edge. Finally, the pixels that have gradient magnitudes between low and high thresholds are considered as edges only if they are connected to an edge having an edge magnitude greater than high threshold.

In conclusion, Canny edge detector performs a sophisticated edge detection operation and it employs very important features that are necessary to detect edges effectively.

CHAPTER 3

GENERALIZED BEAM ANGLE STATISTICS (GBAS)

Generalized Beam Angle Statistics, (GBAS) is a boundary based shape descriptor which describes shapes with the statistics of generalized beam angles produced between the beams originating at the reference point and directed to the rest of the points of the boundary. GBAS is obtained with the generalization of Beam Angle Statistics, BAS [5]. Although BAS requires the parametric boundary representation in order to compute the feature vectors of boundary points, GBAS drops this requirement and allows the computation of the feature vectors in the lack of parametric boundary representations. This property is essential for the description of edges in edge detected images. As a result, the feature vector of a boundary point in a template shape and the feature vector of an edge pixel in an edge image can be computed in the same manner. Then, these feature vectors are used to match the points of the template shape with the edges.

The chapter is organized as follows. In section 3.1, we define the GBAS shape descriptor formally. Invariance properties of GBAS under translation, rotation, scaling, and reflection transformations are given in section 3.2. Then, we show the robustness of GBAS shape descriptor to shearing, noise and occlusion in sections 3.3, 3.4 and 3.5, respectively. Finally, we discuss the computational complexity of the descriptor in section 3.6.

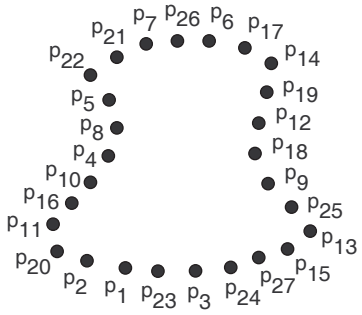


Figure 3.1: Boundary Pixels Set.

3.1 Definition

In this section we first give some definitions and then introduce Generalized Beam Angle Statistics.

3.1.1 Boundary Pixels Set

Boundary pixels set, $P = \{p_1, p_2, \ldots, p_N\}$ is a set of edge pixels that are extracted from an image using an edge detection algorithm (see Figure 3.1). The indices of the boundary pixels are assigned arbitrarily. Note that this representation is not parametric since the ordering of the pixels constituting the boundary is unknown.

Given a binary image containing a single shape, one can obtain the parameterized boundary of the shape using a contour following algorithm. However, in most of the complex grayscale or colored images, the problem of extraction of a shape boundary is not guaranteed to have a solution. Therefore, in this study, we focus on the set of points or non-parametric representation of boundaries.

3.1.2 Beam Vector

*B*eam Vector, $V(p_i, p_j)$, is defined as a vector from an edge pixel, p_i, to another edge pixel, p_j (see Figure 3.2). It is essential in the definition of beam angles.

3.1.3 Mean Beam Vector

In order to compute the GBAS feature vector of an edge pixel p_i, first we partition the set of edge pixels P into two disjoint sets relative to p_i. For this purpose, the Mean

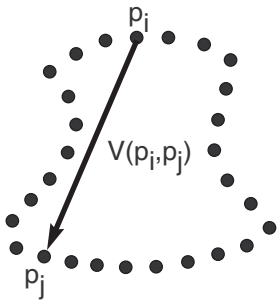


Figure 3.2: Beam Vector

Beam Vector, $OK(p_i)$, of the edge pixel p_i is computed (See Figure 3.3). $OK(p_i)$ is the mean vector of all beam vectors originating at p_i and directed to all other edge pixels in P. Thus; in mathematical terms;

$$OK(p_i) = \sum_{j=1}^{N} V(p_i, p_j) \tag{3.1}$$

where $\sum$, denotes vector addition operation. Mean beam vector, $OK(p_i)$ partitions

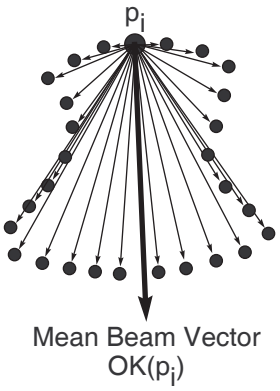


Figure 3.3: Mean Beam Vector, $OK(p_i)$.

the edge pixels set P into two disjoint sets, namely, the *forward* I and *backward* G, edge pixel sets (Figure 3.4) relative to p_i.

3.1.4 Forward and Backward Edge Pixels Sets

The forward edge pixels set of an edge pixel p_i, $I = \{i_1, i_2, \ldots, i_S\}$, is the union of the edge pixels, p_j, whose beam vectors, $V(p_i, p_j)$, have less than π degrees with the mean beam vector, $OK(p_i)$, in the counter clockwise direction. In Figure 3.4, edge pixels in

the forward edge pixels set are shown as white circles. Note that each boundary pixel, p_i has a distinct forward edge pixels set determined by the mean beam vector. Thus, the number, S, and elements of the forward edge pixels set changes from boundary pixel to boundary pixel.

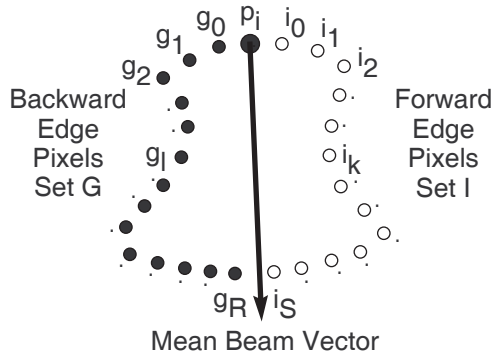


Figure 3.4: Forward I and Backward G Edge Pixels Sets.

The backward edge pixels set of an edge pixel p_i, $G = \{g_1, g_2, \ldots, g_R\}$, is the union of the edge pixels, p_j, whose beam vectors, $V(p_i, p_j)$, have less than π degrees with the mean beam vector, $OK(p_i)$, in the clockwise direction. In Figure 3.4, edge pixels in the backward edge pixels set are shown as black circles. Note that each boundary pixel, p_i has a distinct backward edge pixels set determined by the mean beam vector. Thus, the number, R, and elements of the backward edge pixels set changes from boundary pixel to boundary pixel.

3.1.5 Generalized Beam Angle

Generalized beam angle $C_{k,l}(p_i)$ of an edge pixel p_i, is an angle formed by forward $V(p_i, i_k)$ and backward $V(p_i, g_l)$ beam vectors that originate at p_i and are directed to i_k forward and g_l backward edge pixels, respectively (Figure 3.5).

This definition somewhat generalizes the beam angle definition in [5]. The beam angle of [5] is defined as the angle between the beam vectors left and right to the reference pixel p_i. Obviously, the determination of "left" and "right" boundary pixels requires parametric boundary representation in [5]. The generalized beam angle definition introduced here, eliminates this requirement. This simple modification of beam angle definition with the introduction of Mean Beam Vector that provides the partial

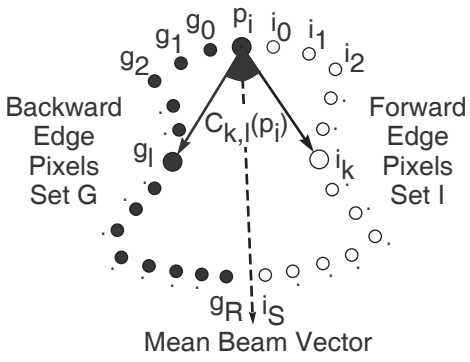


Figure 3.5: Generalized beam angle, $C_{k,l}(p_i)$.

ordering of boundary pixels frees the computation of beam angle statistics from the parametric boundary representation.

3.1.6 Beam Angles Matrix

Next, we define the *Beam Angles Matrix*, $K(p_i)$, of the edge pixel p_i, in order to compute its GBAS feature vector. Beam angles matrix, $K(p_i)$, denotes all generalized beam angles of the boundary pixel p_i. In mathematical terms it is defined as;

$$K(p_i) = \lfloor C_{k,l}(p_i) \rfloor \quad k = 1, 2, \ldots, S \quad l = 1, 2, \ldots, R \tag{3.2}$$

Remember that, $C_{k,l}(p_i)$ denotes the generalized beam angle that is formed by the forward $V(p_i, i_k)$ and backward $V(p_i, g_l)$ beam vectors that originate at p_i and are directed to i_k forward and g_l backward edge pixels, respectively. Since $K(p_i)$ incorporates all beam angles formed by the beam vectors that are created by the forward and backward edge pixel sets I and G, respectively, it has $S * R$ elements. Note that since the number and elements of forward and backward edge pixels sets is different for each boundary pixel, the number of elements of the beam angles matrix may differ from boundary pixel to boundary pixel as well.

3.1.7 Generalized Beam Angle Statistics

In this study, for each edge pixel p_i, the Generalized Beam Angle $C_{k,l}(p_i)$ is taken as a random variable with the probability density function $P_i(C_{k,l}(p_i))$ and considered as an outcome of the stochastic process which generates the shape being described

at different scales. The probability density function P_i is approximated with the histogram of the generalized beam angles in the beam angles matrix, $K(p_i)$. As a result, moments of the beam angle random variable, $C_{k,l}(p_i)$, are defined as follows;

$$\Xi\left[C(p_i)^m\right] = \sum_{k=1}^{S}\sum_{l=1}^{R} C_{k,l}(p_i)^m P_i(C_{k,l}(p_i)),\ m = 1, 2, \ldots, d. \tag{3.3}$$

In the above equation, Ξ indicates the expected value operator. The moments describe the statistical behavior of the generalized beam angles that form edge pixel p_i's beam angles matrix $K(p_i)$.

3.1.8 GBAS Feature Vector

Each edge pixel p_i, is described by the following *GBAS feature vector*;

$$\Gamma(p_i) = \left[\Gamma^1(p_i), \Gamma^2(p_i), \ldots, \Gamma^d(p_i)\right],\ i = 1, 2, \ldots, N. \tag{3.4}$$

The components of the feature vector are the moment statistics of Generalized Beam Angles;

$$\Gamma^m(p_i) = \Xi\left[C(p_i)^m\right],\ m = 1, 2, \ldots, d. \tag{3.5}$$

Note that, the number of moments required to describe the shape depends on the form of the probability density function, $P_i(C_{k,l}(p_i))$, of the beam angles matrix, $K(p_i)$.

3.1.9 GBAS Moment Functions

GBAS Moment functions of a shape are formed by combining the corresponding components of boundary pixels' GBAS feature vectors end to end. In mathematical terms, the moment functions of a shape, A, which is represented by a set of edge pixels, P, are defined as follows;

$$GBAS_m(A) = < \Gamma^m(p_1), \Gamma^m(p_2), \ldots, \Gamma^m(p_N) >,\ m = 1, 2, \ldots, d. \tag{3.6}$$

In figures 3.6, 3.8, 3.10 and 3.12 some sample shapes and in figures 3.7, 3.9, 3.11 and 3.13 their GBAS Moment functions are shown. Observe that local maxima and minima points of GBAS Moment functions correspond to concave and convex corner points of shapes. For example, in figure 3.7 two minimas correspond to apples' handles. The maxima between the minimas is due to the concavity between the two handles of the apple. In figure 3.9, the local minimas at the center of the graph correspond to

the two extremes of the bat's wings at the top left and bottom right of the figure 3.8. On the other hand, the local minima divided at the edges of the figure 3.9 correspond to the tail of the bat at the lower left corner of the shape. In figure 3.10, there is a triangle whose edges are slightly convex. Three local minimas of the GBAS Moment functions correspond to the corners of the triangle. On the other hand, three maximas correspond to the centers of the triangles' edges. In figure 3.12, there is a flower having 10 leaves. The corner of each leave correspond to a local minima. Therefore, it is possible to count the number of leaves by counting the number of local minimas in the GBAS Moment functions. These examples show that concavities and convexities produce local maximas and minimas in the GBAS Moment functions.

3.2 Properties of GBAS

In this section, we investigate the properties of GBAS shape descriptor in a sub-class of affine transformations. We show that GBAS is invariant to translation, rotation, scaling (uniform), and reflection. An immediate consequence of invariance of the GBAS shape descriptor to these transformations is that the shape descriptor is also invariant to the following types of affine transformations:

- euclidean motion (rotation + translation),
- rigid body transformation (rotation + translation + reflection) and
- similarity transformation (rotation + translation + scaling).

Euclidean motion, rigid body and similarity transformations are formed by a combination of the rotation, translation, reflection and scaling transformations. But, GBAS is invariant under rotation, translation, reflection and scaling transformations. Therefore, it is also invariant to the Euclidean motion, rigid body and similarity transformations.

3.2.1 Translation Invariance

GBAS shape descriptor is translation invariant. GBAS shape descriptor of a boundary point p_i depends on its mean beam vector $OK(p_i)$ and the moments of its generalized beam angles $C_{k,l}(p_i)$. Neither $OK(p_i)$ nor $C_{k,l}(p_i)$ depends on the absolute coordinates

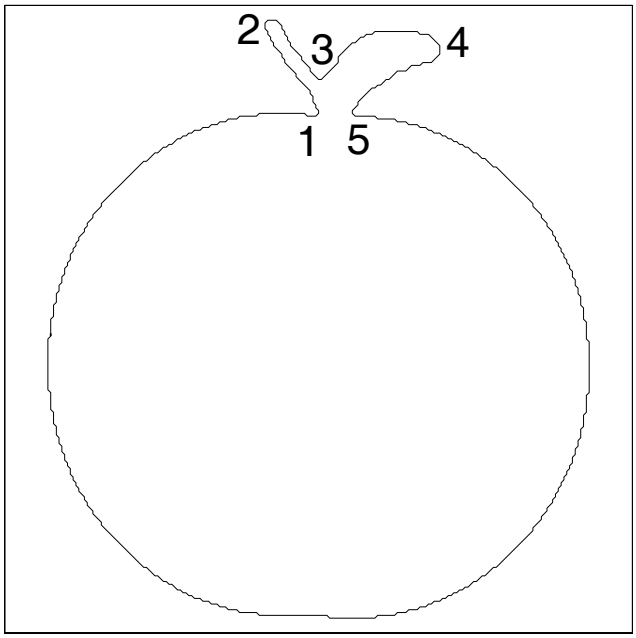


Figure 3.6: A sample shape from MPEG-7 CE Shape-1 Data Set.

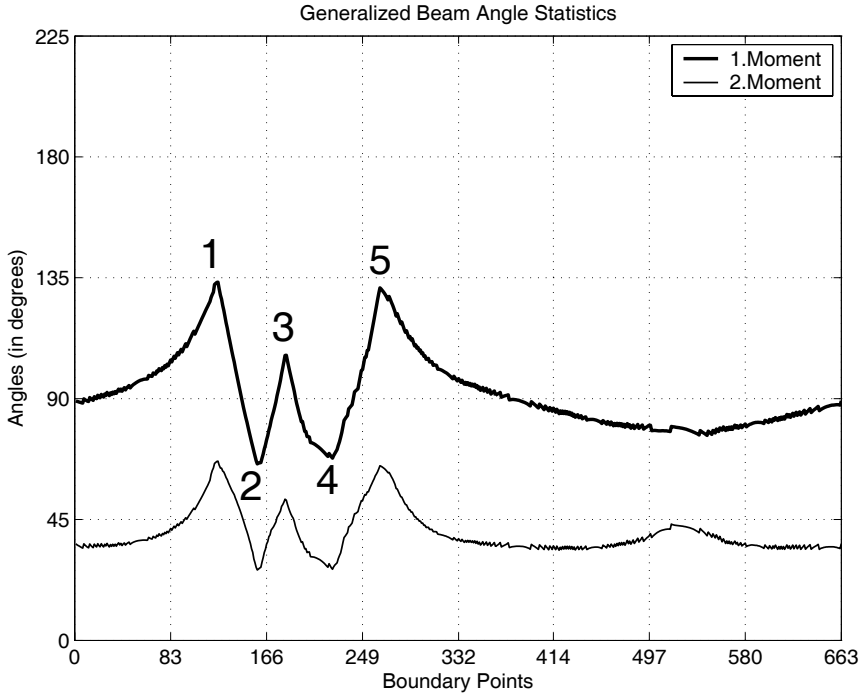


Figure 3.7: GBAS Moment functions of shape in figure 3.6.

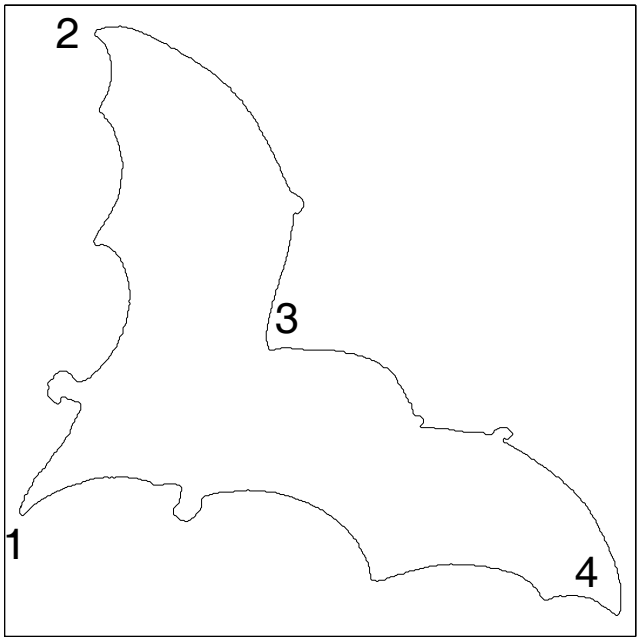


Figure 3.8: A sample shape from MPEG-7 CE Shape-1 Data Set.

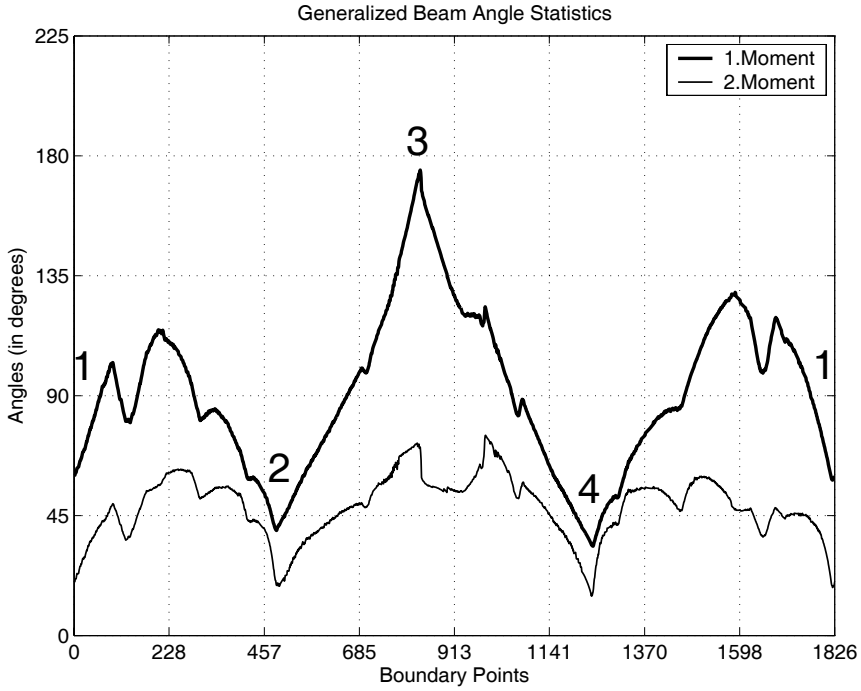


Figure 3.9: GBAS Moment functions of shape in figure 3.8.

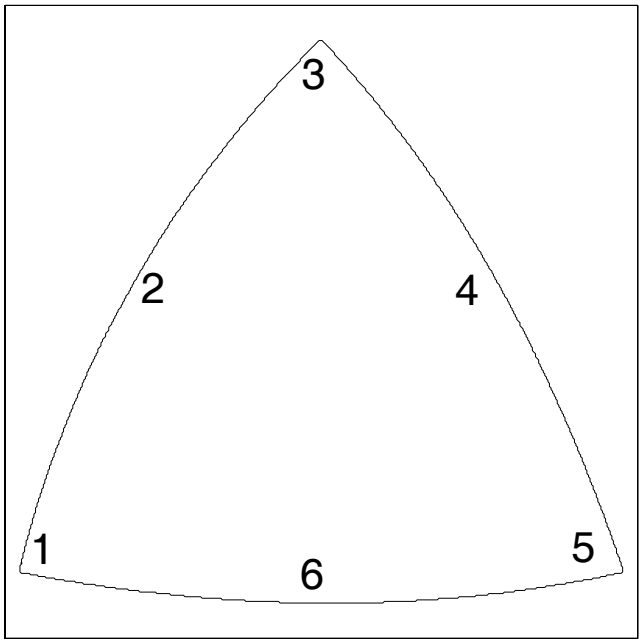


Figure 3.10: A sample shape from MPEG-7 CE Shape-1 Data Set.

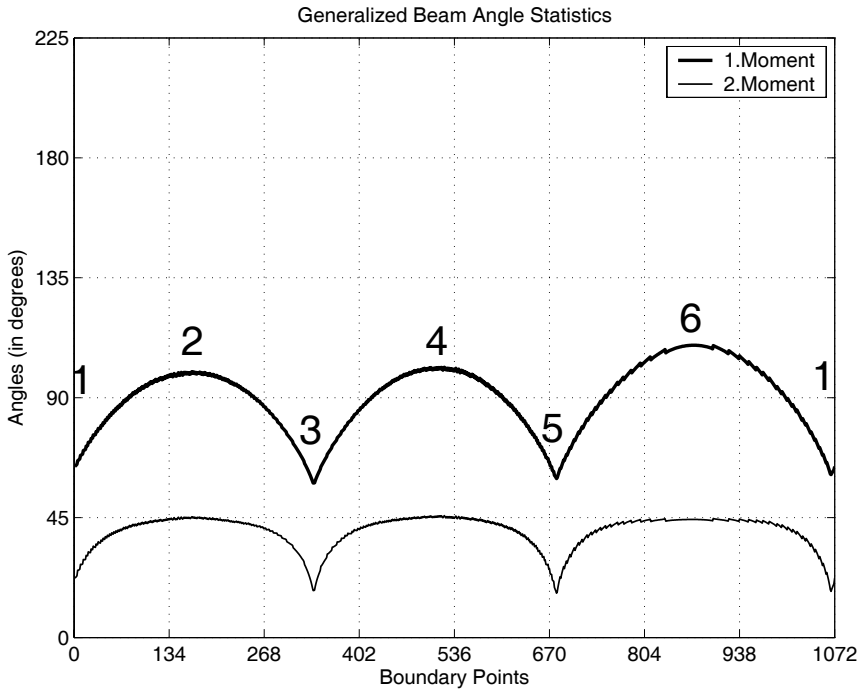


Figure 3.11: GBAS Moment functions of shape in figure 3.10.

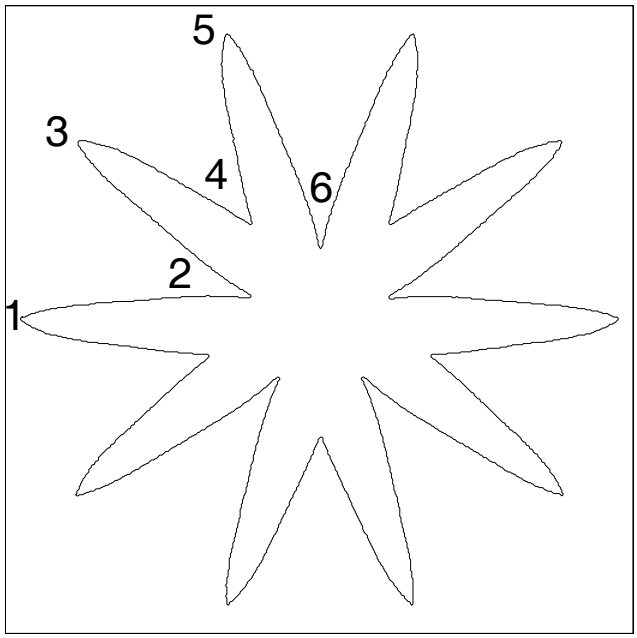


Figure 3.12: A sample shape from MPEG-7 CE Shape-1 Data Set.

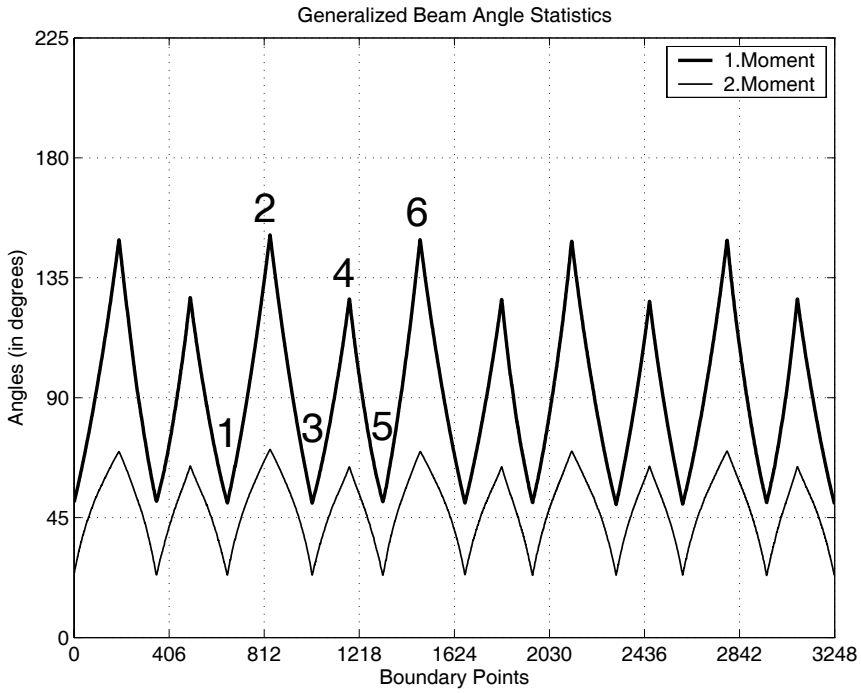


Figure 3.13: GBAS Moment functions of shape in figure 3.12.

of boundary pixels. They depend on beam vectors $V(p_i, p_j)$ directed from the reference boundary pixel p_i to another boundary pixel p_j. Consequently, GBAS shape descriptor is translation invariant.

3.2.2 Rotation Invariance

GBAS shape descriptor is rotation invariant. If the shape is rotated by α degrees then $V(p_i, p_j)$ and $OK(p_i)$ are rotated α degrees, as well. Additionally, the partitioning of the pixel set, P, into the forward, I and backward G edge pixels sets is preserved. Therefore, p_i's generalized beam angles, $C_{k,l}(p_i)$ do not change. However, if the starting pixels of the shape boundaries are not same, then GBAS Moment functions are circularly shifted. In other words, there exists a phase difference between the GBAS Moment functions. This phase difference is typically handled in the descriptor matching algorithm. Since the phase difference is due to the change of the starting pixel, it can be observed after any transformation.

3.2.3 Scale Invariance

GBAS shape descriptor is scale invariant. As the shape is scaled by a factor of α uniformly;

- the length of shape boundary increases ($\alpha > 1$) or decreases ($\alpha < 1$),
- the beam vectors, $V(p_i, p_j)$ starting from the reference pixel, p_i, ending in rest of the boundary pixels are preserved, however their number either increases ($\alpha > 1$) or decreases ($\alpha < 1$),
- the mean beam vector, $OK(p_i)$ of the boundary pixel p_i is preserved,
- the partitioning of the pixel set, P, into its forward, I and bacward G, edge pixels sets is preserved but the number of elements in each of these sets either increases ($\alpha > 1$) or decreases ($\alpha < 1$)

It follows that p_i's generalized beam angles, $C_{k,l}(p_i)$ do not change. However, as the length of the boundary pixels set P increases, then, the number of generalized beam angles, $C_{k,l}(p_i)$ used to compute p_i's GBAS feature vector increases as well. Then, statistical stability of the descriptor also increases. Conversely, as the length of the

boundary pixels set P decreases, then the number of generalized beam angles, $C_{k,l}(p_i)$ used to compute p_i's GBAS feature vector decreases as well. Then, statistical stability of the descriptor decreases. Consequently, as the scale of the shape increases, then the generalized beam angle statistics is enhanced and as the scale of the shape decreases, generalized beam angle statistics deteriorates.

In figure 3.14 a sample shape and its 2.0, 0.30, 0.25, 0.20 and 0.10 scaled instances are shown. 1^{st} and 2^{nd} GBAS Moment functions of these shapes are shown in figures 3.15 and 3.16, respectively. Observe that GBAS Moment functions are very similar however, as the scale of the shape decreases, then the fluctuations of the shape descriptor increases. This is due to the decrease in the statistical stability. Note that there is a phase difference between the 0.10 scaled shape and others since the starting points of the shapes are different. Note also that, GBAS Moment functions are down-sampled to the size of the shortest boundary for illustration purposes. In conclusion, GBAS is scale invariant.

3.2.4 Reflectance Invariance

GBAS shape descriptor is reflectance invariant. As the shape is reflected:

- the beam vectors, $V(p_i, p_j)$ starting from the reference pixel, p_i, ending in rest of the boundary pixels are reflected,
- the mean beam vector, $OK(p_i)$ of the boundary pixel p_i is reflected,
- the partitioning of the pixel set, P, into its forward, I and backward G, edge pixels sets is preserved.

From these it follows that p_i's generalized beam angles, $C_{k,l}(p_i)$ do not change.

In figure 3.17 (a), (b), (c) and (d) a sample shape and its reflected instances about the x, y and both of x and y axis are shown. In figures 3.18 and 3.19 the 1^{st} and 2^{nd} GBAS Moment functions of the sample and reflected shapes are shown, respectively. Observe that GBAS shape descriptors of these shapes are exactly the same.

3.3 Robustness to Shear

Shearing changes the curvature of the boundary points. GBAS is a curvature based shape descriptor, thus, as the shape is sheared, it changes. However, the change of the

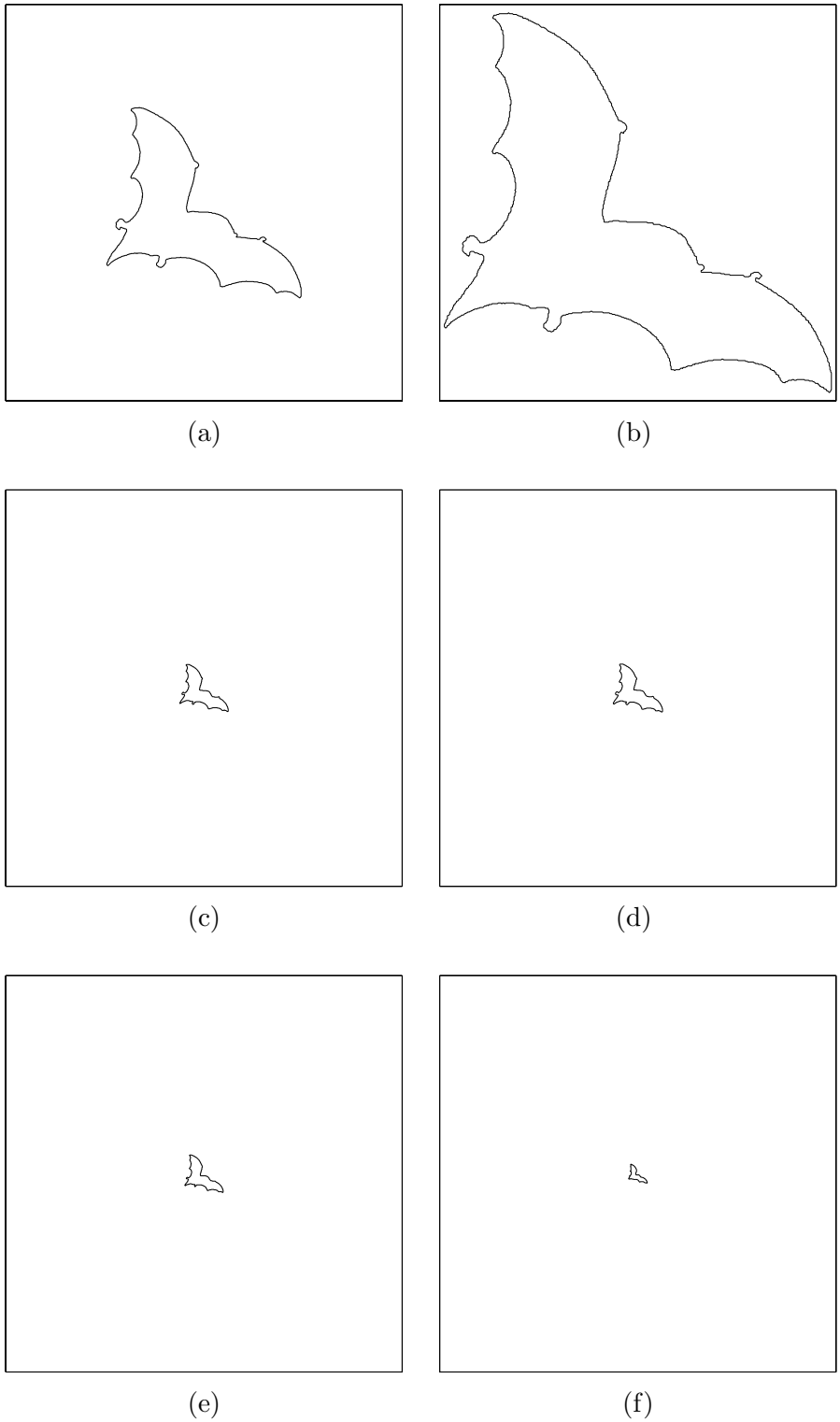


Figure 3.14: Figures (a) to (f) show a sample shape and its 2.0, 0.3, 0.25, 0.2 and 0.1 scaled instances respectively.

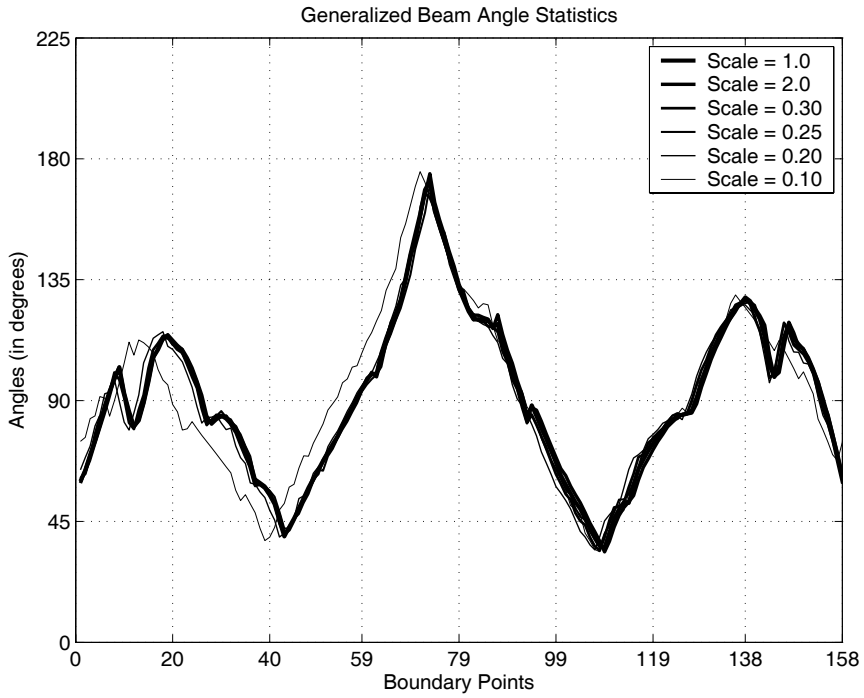


Figure 3.15: 1^{st} GBAS Moment functions of the sample and scaled shapes in figure 3.14.

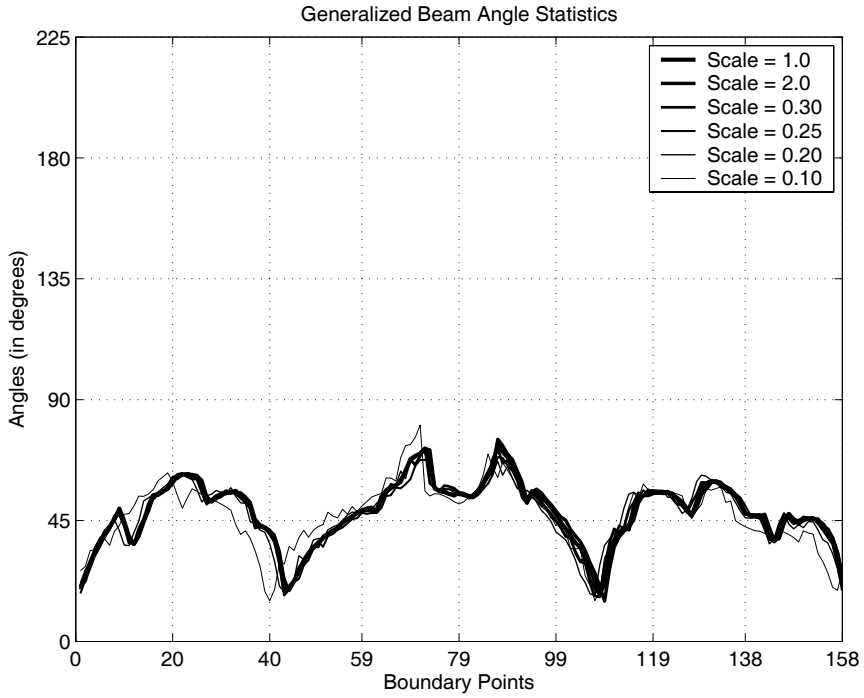


Figure 3.16: 2^{nd} GBAS Moment functions of the sample and scaled shapes in figure 3.14.

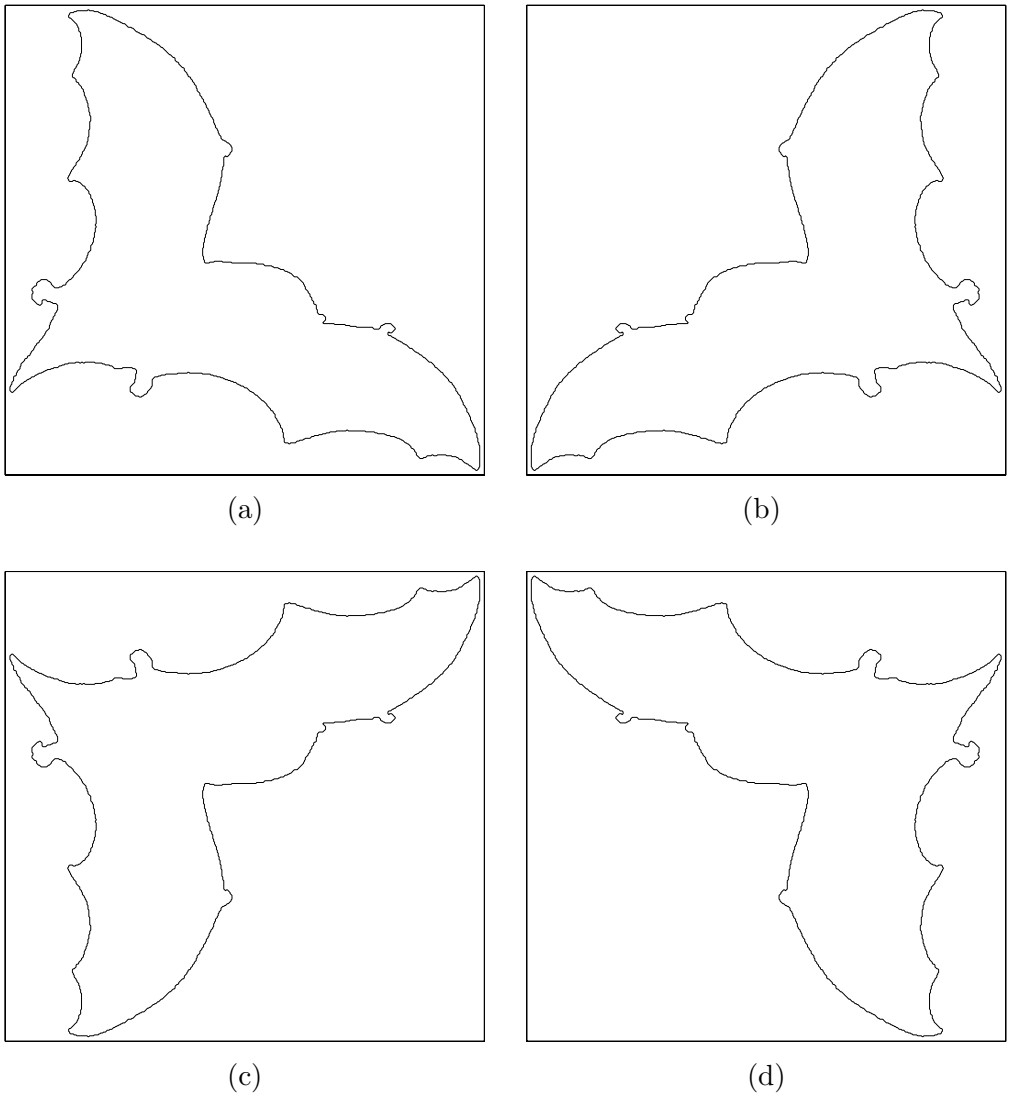


Figure 3.17: Sample shapes for demonstrating the effects of reflection. In (a) original shape is shown. In (b), (c) and (d) reflected versions of the original shape about the x, y and both of x and y axis are shown.

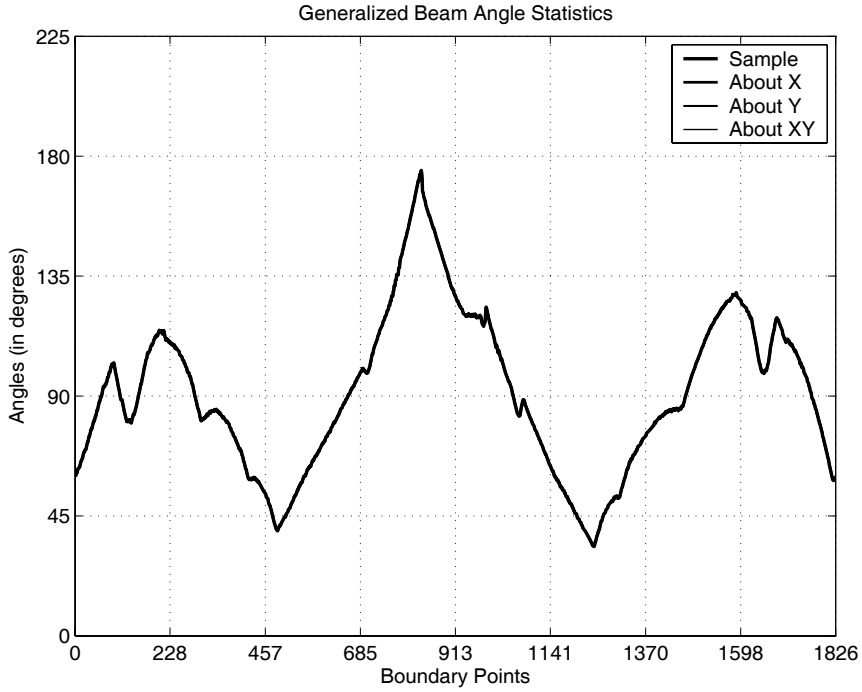


Figure 3.18: 1^{st} GBAS Moment functions of the sample and reflected shapes in figure 3.17.

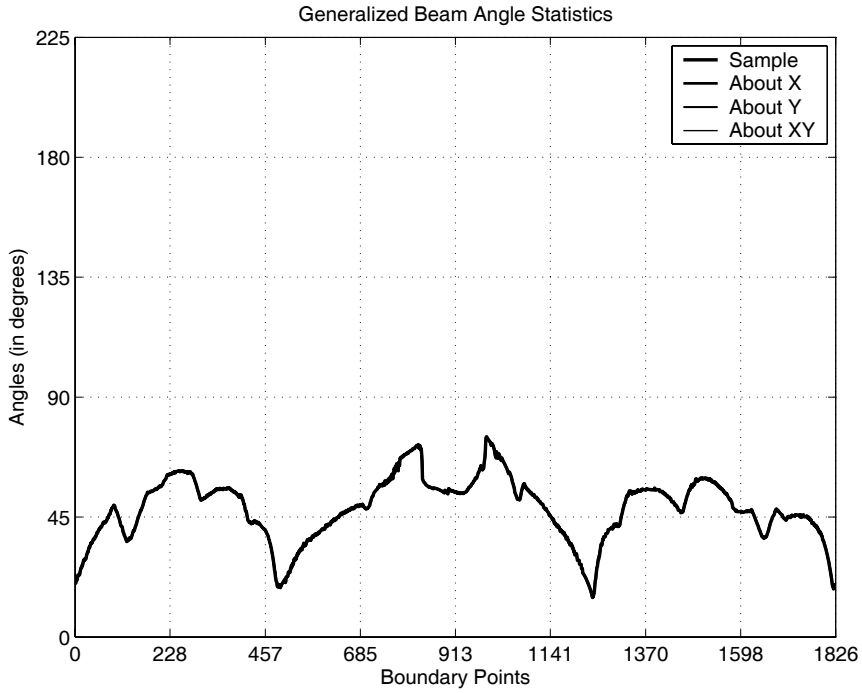


Figure 3.19: 2^{nd} GBAS Moment functions of the sample and reflected shapes in figure 3.17.

descriptor is closely related with the magnitude of the shearing. For small amounts of shearing the form of the descriptor does not change significantly.

In figure 3.21 sample shapes for visualizing the effects of shearing are shown. Original shape of figure 3.20 is sheared along the direction of horizontal axis. Figures 3.21 (a), (b), (c), and (d) show sheared instances of the original shape by factors of 0.8, 0.6, 0.4 and 0.2 respectively. Figures 3.22 and 3.23 show the 1^{st} and 2^{nd} GBAS Moment functions of the sample and sheared shapes. It is clear that, up to shear factor of 0.6, GBAS shape descriptors have strong resemblances, i.e. local maximas and minimas of the descriptor do not change much. Below this factor, the differences between shape descriptors become more apparent.

3.4 Robustness to Noise

Noise is a common problem for image processing systems. There are various sources of noise. Imperfections of imaging sensors results in noisy images. Additionally, noise can be introduced in early stages of image processing such as segmentation and edge detection. Therefore, noise robustness is a desirable property of shape descriptors. In this section, we demonstrate noise robustness of GBAS. For this purpose, Gaussian noise having various variances is added to a shape. Noise is taken as the magnitude of the displacement vector of each boundary point. Direction of this vector is given by the direction of the gradient.

In figure 3.24 a sample shape and various levels of noisy instances of it are shown. In figures 3.25, and 3.26 it can be observed that the 1^{st} and 2^{nd} GBAS Moment functions of the noisy shapes have small variations. As the amount of noise increases then the amount of variations increases as well. This shows that, GBAS shape descriptor is insensitive to Gaussian noise.

3.5 Robustness to Occlusion

Occlusion is a common deformation that can be encountered in images because of the ordering of objects in the scene. In this case, only a portion of the shape is visible and it is requested to detect the whole shape from the visible part. Shape descriptors that exploit local information can be successful to cope up with this problem. However, GBAS computes the statistics of beam angles which are directed to all

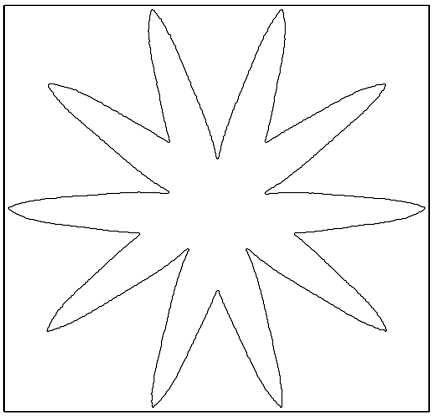


Figure 3.20: Sample shape used in the shearing test.

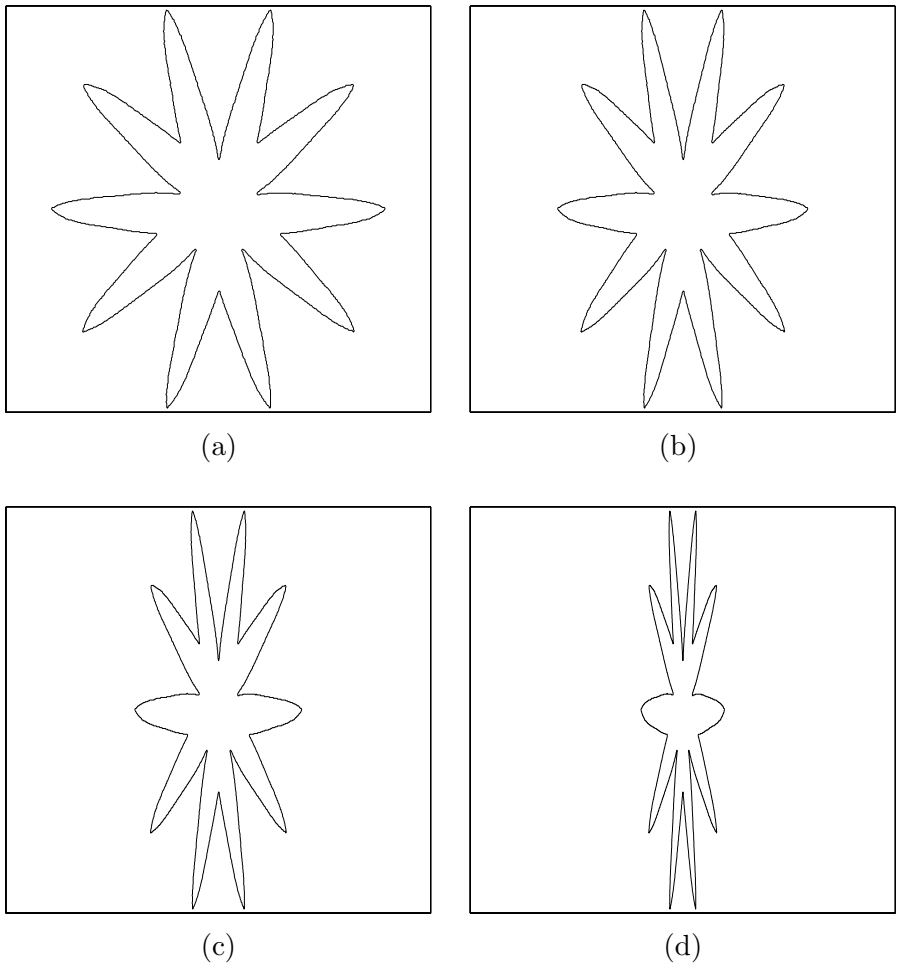


Figure 3.21: Sample shapes for demonstrating the effects of shearing. In figures (a) to (d) sheared instances of the original shape in figure 3.21 by factors of 0.8, 0.6, 0.4 and 0.2 are shown.

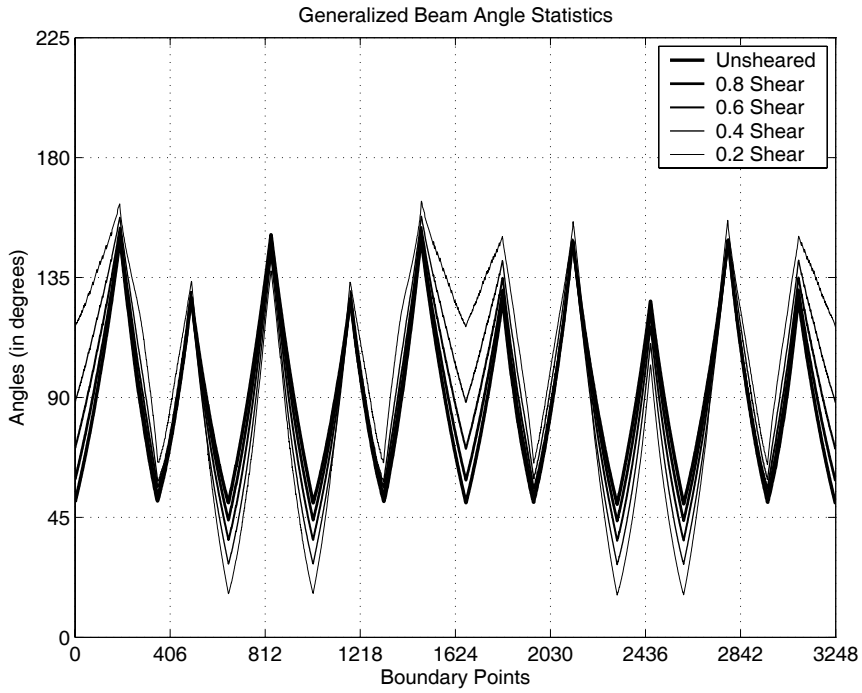


Figure 3.22: 1^{st} GBAS Moment functions of the sample and sheared shapes in figures 3.20 and 3.21, respectively.

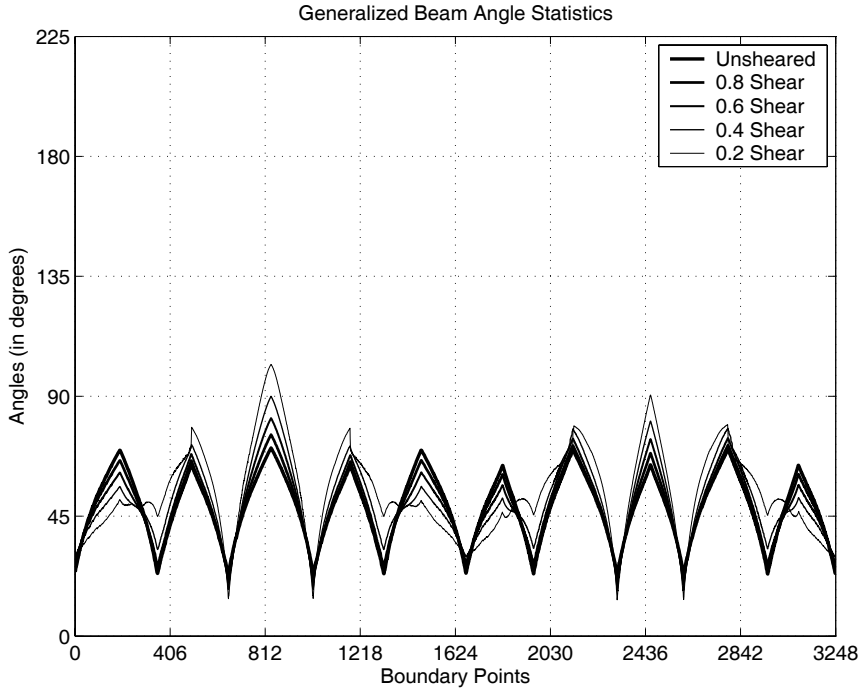
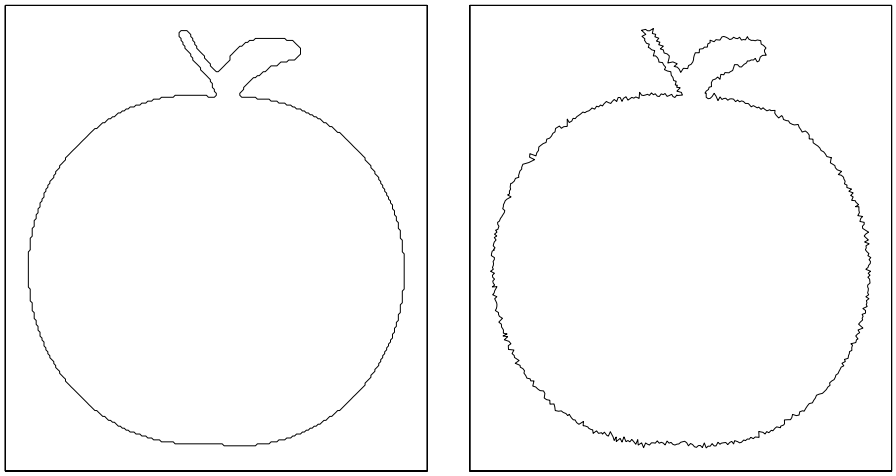
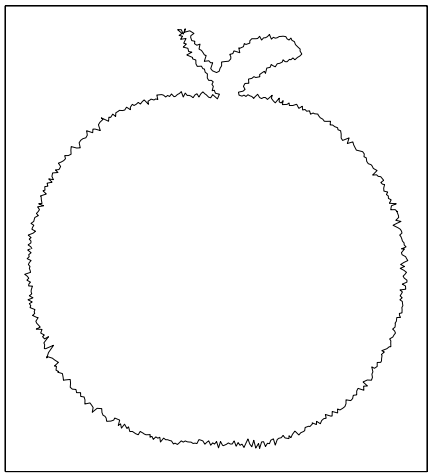


Figure 3.23: 2^{nd} GBAS Moment functions of the sample and sheared shapes in figures 3.20 and 3.21, respectively.

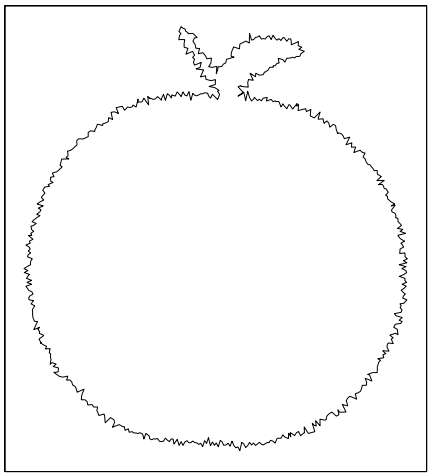


(a)

(b)



(c)



(d)

Figure 3.24: Sample shapes for demonstrating the effects of noise. In (a) original shape is seen. In figures (b) to (d) Gaussian noise having zero mean and 0.5, 1.0, and 1.5 variance added versions of the shape are seen.

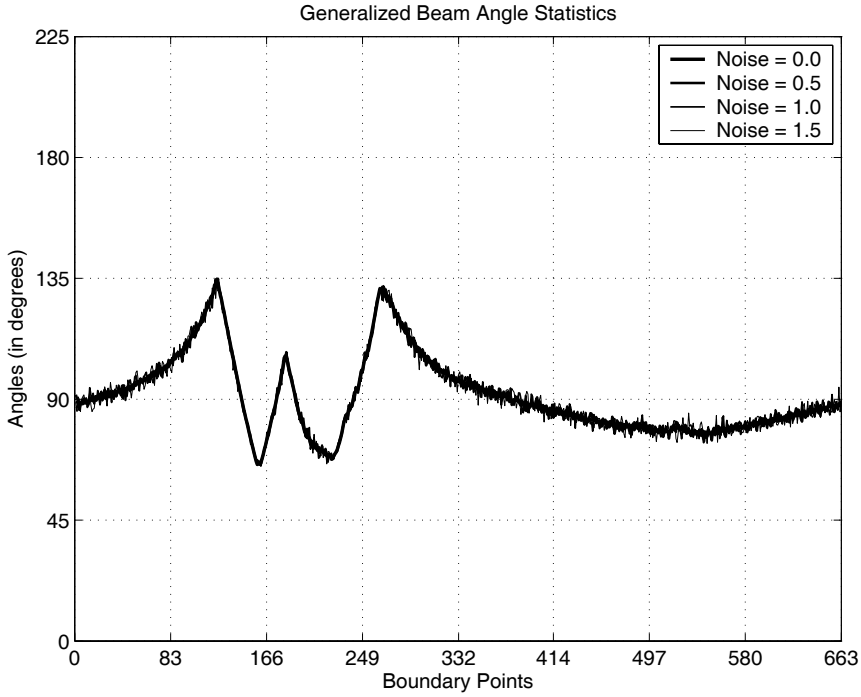


Figure 3.25: 1^{st} GBAS Moment functions of the sample and noisy shapes in figure 3.24.

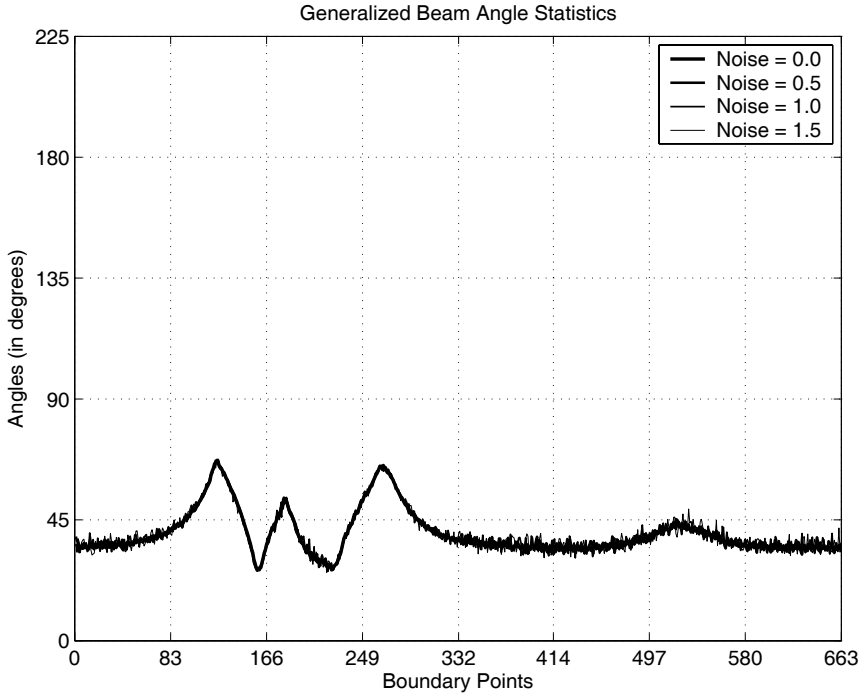


Figure 3.26: 2^{nd} GBAS Moment functions of the sample and noisy shapes in figure 3.24.

Algorithm 3.1: Find GBAS shape descriptor of a given shape boundary P.

(1) **for** each pixel p_i in P **do**
(2) **compute** $OK(p_i)$
(3) **partition** P using $OK(p_i)$ into I and G
(4) **for** each forward pixel i_l in I **do**
(5) **for** each backward pixel g_k in G **do**
(6) **compute** $C_{k,l}(p_i)$ using $V(p_i, i_l)$, $V(p_i, g_k)$ and $OK(p_i)$
(7) **compute** $\Gamma(p_i)$ of p_i using angles $C_{k,l}(p_i)$

boundary points. Therefore, since occluded boundary points are not available, GBAS shape descriptor is inevitably effected by occlusion. Nevertheless, for small amounts of occlusion it performs well.

Figures 3.27 and 3.28 show a sample shape and various levels of occluded instances of it. In figures 3.29 and 3.30, 1^{st} and 2^{nd} GBAS Moment functions of the sample and occluded shapes are shown, respectively. Observe that the shapes of the moment functions are very similar. The magnitude of error increases at the points near the occlusion. We can say that, up to 30% occlusion, results are satisfactory.

3.6 Computational Complexity

In this section, we investigate the computational complexity of the proposed shape descriptor, GBAS. We show that although naive implementation of the shape descriptor has complexity $O(N^3)$, it can be improved to $O(N^2)$. This time complexity is the same as the time complexity of the BAS shape descriptor, which is $O(N^2)$. This means that, for the added capability of description in the absence of parametric boundary representation, we do not pay any additional cost. Note that, although the time complexity of the two shape descriptors GBAS and BAS is $O(N^2)$, practically BAS runs slightly faster than GBAS. This is due to the constant which is not shown in the complexity analysis.

3.6.1 Naive Implementation

Naive method to compute the GBAS shape descriptor of a shape boundary is summarized in Algorithm 3.1, where for each pixel, mean beam vector $OK(p_i)$ is computed.

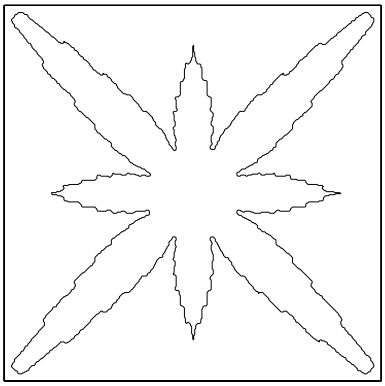


Figure 3.27: Sample shape for occlusion.

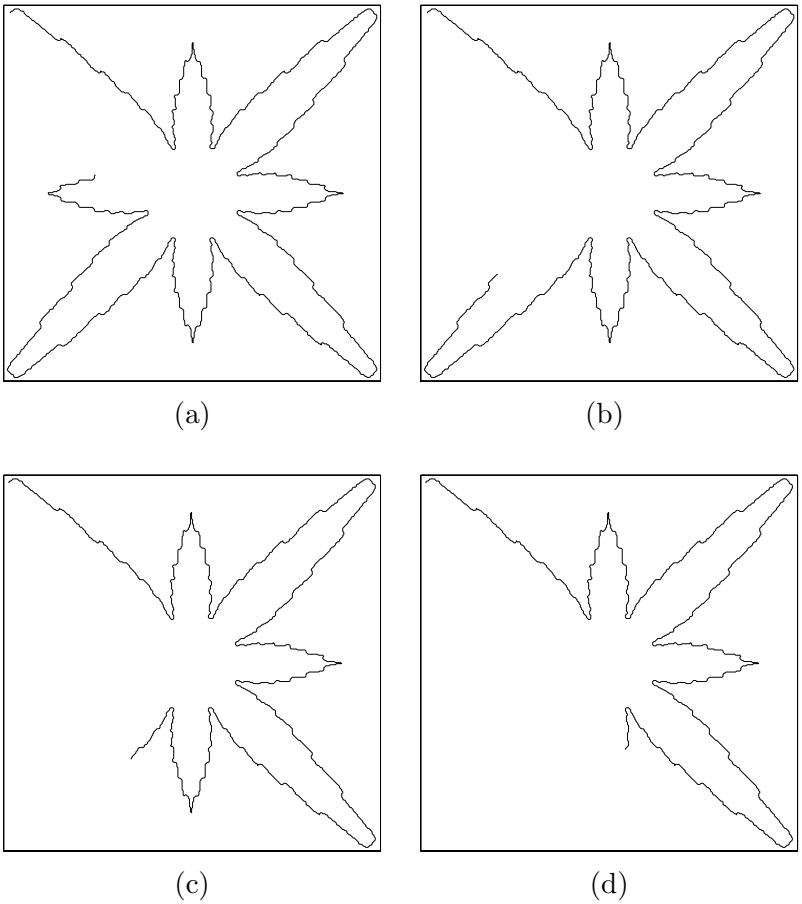


Figure 3.28: Sample shapes for demonstrating the effects of occlusion. In figures (a) to (d), 10%, 20%, 30% and 40% occluded instances of the sample shape in figure 3.27 are shown, respectively.

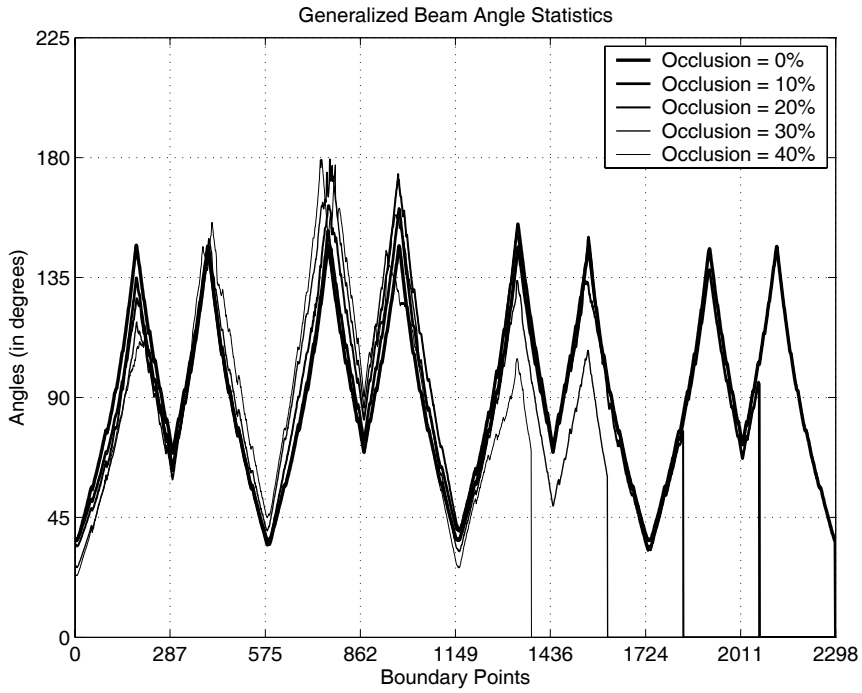


Figure 3.29: 1^{st} GBAS Moment functions of the sample and occluded shapes in figures 3.27 and 3.28, respectively.

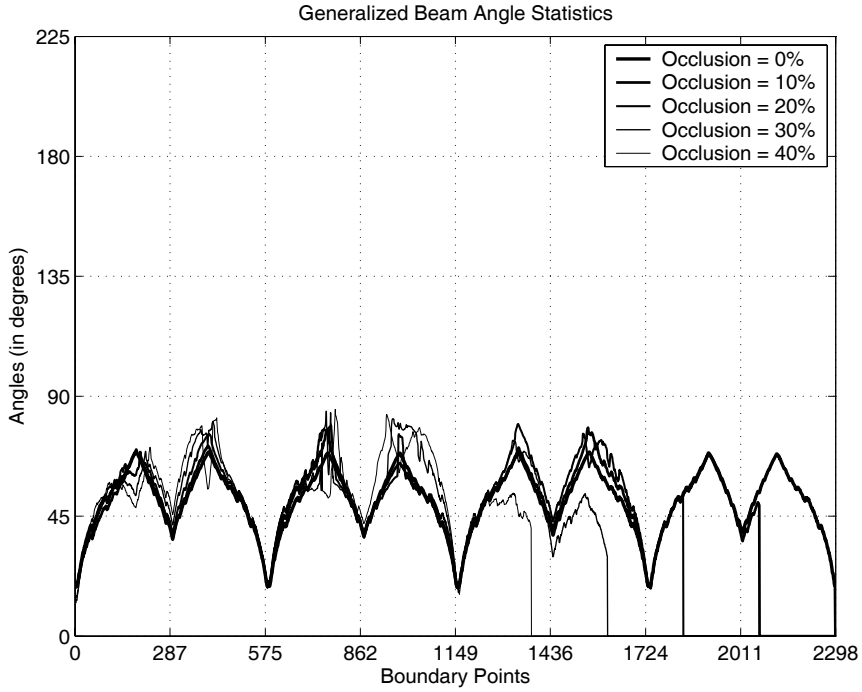


Figure 3.30: 2^{nd} GBAS Moment functions of the sample and occluded shapes in figures 3.27 and 3.28, respectively.

This can be achieved in $O(N)$ time, since computation involves the summation of N beam vectors, $V(p_i, p_j)$, directed from p_i to all other pixels in P. In step 3 of the algorithm, pixel set P is partitioned into forward, I and backward, G pixel sets using $OK(p_i)$. This can be achieved in $O(N)$ time as well, since computation is performed once for each pixel in P. In steps 4, 5 and 6 of the algorithm, Generalized Beam Angles of pixel p_i are computed. Clearly p_i has $S * R$ Generalized Beam Angles, $C_{k,l}(p_i)$. Here S and R are the number of pixels in the forward, I and backward, G boundary pixels sets, respectively. $S * R$ takes its largest value when $S = R = N/2$. Consequently, time required to compute generalized beam angles is $O(S * R)$ which is $O(N/2 * N/2)$ or $O(N^2)$ in the worst case. Finally, in step 7, moment statistics of generalized beam angles are computed. This can be performed in $O(N^2)$ time because during the computation each Generalized Beam Angle is processed only once. As a result, total time required to compute GBAS feature vector of a single boundary pixel is $O(N + N + N^2 + N^2)$ which is simply $O(N^2)$. Since this operation will be performed for each boundary pixel, the computational cost of naive implementation for the computation of the GBAS shape descriptor of a shape having boundary length N is $O(N^3)$.

3.6.2 Efficient Implementation

In this section, we propose an efficient algorithm for the computation of GBAS shape descriptor. The algorithm is based on the idea that, if one can represent generalized beam angles, $C_{k,l}(p_i)$ in terms of each other, then without computing all of them, GBAS shape descriptor can be computed. Based on this idea, computational cost of GBAS shape descriptor is decreased to $O(N^2)$.

A forward beam vector, $F(p_i, i_k)$ is a beam vector that starts from the reference pixel p_i and ends in a pixel, i_k which is an element of the forward pixels set, I of p_i. Similarly, a backward beam vector, $B(p_i, g_l)$ is a beam vector that starts from the reference pixel p_i and ends in a pixel, g_l which is an element of the backward pixels set, G. See figure 3.31.

A forward beam angle, $F_k(p_i)$ is an angle formed by the mean beam vector, $OK(p_i)$ and a forward beam vector, $F(p_i, i_k)$ of the reference pixel, p_i. Similarly, a backward beam angle, $B_l(p_i)$ is an angle formed by the mean beam vector, $OK(p_i)$

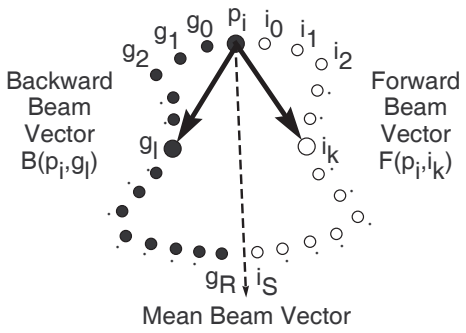


Figure 3.31: Forward and Backward Beam Vectors.

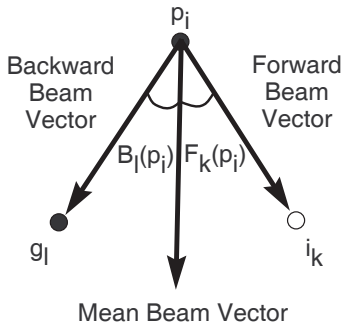


Figure 3.32: Forward and Backward Beam Angles.

and a backward beam vector, $B(p_i, g_l)$ of the reference pixel, p_i. Note that forward and backward beam angles of a boundary pixel p_i are always positive by definition. See figure 3.32.

A generalized beam angle, $C_{k,l}(p_i)$ of a reference pixel, p_i can be represented by the summation of a forward, $F_k(p_i)$ and a backward, $B_l(p_i)$ beam angle.

A forward difference, $\delta F_k(p_i)$ of a reference pixel, p_i is the difference of its two consecutive forward beam angles, $F_k(p_i)$ and $F_{k-1}(p_i)$:

$$\delta F_k(p_i) = F_k(p_i) - F_{k-1}(p_i),\ k = 2, 3, \ldots, S. \tag{3.7}$$

Similarly, a backward difference, $\delta B_l(p_i)$ of a reference pixel, p_i is the difference of its two consecutive backward beam angles, $B_l(p_i)$ and $B_{l-1}(p_i)$:

$$\delta B_l(p_i) = B_l(p_i) - B_{l-1}(p_i),\ l = 2, 3, \ldots, R. \tag{3.8}$$

Main beam angle, $A(p_i)$ of a reference pixel, p_i is the Generalized Beam Angle $C_{1,1}(p_i)$ that is formed by the first forward, $F_1(p_i)$ and backward, $B_1(p_i)$ beam

angles:

$$A(p_i) = F_1(p_i) + B_1(p_i). \tag{3.9}$$

It is possible to represent all Generalized Beam Angles of a boundary pixel, p_i by its main beam angle, $A(p_i)$ and forward, $\delta F_k(p_i)$ and backward, $\delta B_l(p_i)$ differences:

$$C_{k,l}(p_i) = A(p_i) + \sum_{u=2}^{k} \delta F_u(p_i) + \sum_{v=2}^{l} \delta B_v(p_i). \tag{3.10}$$

This formulation reduces the computation of Generalized Beam Angle statistics to $O(N^2)$ time. In the next section, we show that, using equation 3.10, it is possible to compute GBAS without computing all generalized beam angles of a boundary point. The computation of main beam angle, $A(p_i)$, and terms of forward $\delta F_k(p_i)$ and backward $\delta B_l(p_i)$ differences are sufficient to obtain the statistics.

3.6.2.1 Computation of GBAS

In this section, we derive formulations to compute GBAS of a boundary pixel, p_i. For the simplicity of notation, we drop p_i terms in all of the equations. Therefore, equation 3.10 can be written in the new notation as follows;

$$C_{k,l} = A + \sum_{u=2}^{k} \delta F_u + \sum_{v=2}^{l} \delta B_v. \tag{3.11}$$

3.6.2.2 Mean

Mean value is given by;

$$\mu = \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} C_{k,l}. \tag{3.12}$$

Expanding the generalized beam angles term, $C_{k,l}$ using equation 3.10 we obtain;

$$\mu = \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} \left(A + \sum_{u=2}^{k} \delta F_u + \sum_{v=2}^{l} \delta B_v \right). \tag{3.13}$$

This equation can be simplified to;

$$\mu = A + \frac{1}{S} \sum_{k=1}^{S} \left(\sum_{u=2}^{k} \delta F_u \right) + \frac{1}{R} \sum_{l=1}^{R} \left(\sum_{v=2}^{l} \delta B_v \right). \tag{3.14}$$

Introducing the definitions;

$$E(F)^{(n)} = \sum_{k=1}^{S} \left(\sum_{u=2}^{k} \delta F_u \right)^n \tag{3.15}$$

and

$$E(B)^{(n)} = \sum_{l=1}^{R} \left(\sum_{v=2}^{l} \delta B_v \right)^n , \tag{3.16}$$

equation 3.14 can be written as;

$$\mu = A + \frac{E(F)}{S} + \frac{E(B)}{R}. \tag{3.17}$$

3.6.2.3 Computation of powers of $E(X)$

Consider the computation of $E(F)$:

$$E(F) = \sum_{k=1}^{S} \left(\sum_{u=2}^{k} \delta F_u \right). \tag{3.18}$$

Clearly, the term $\sum_{u=2}^{k} \delta F_u$ inside the summation will be evaluated for each k from 1 to S. However, for the evaluation of the $k+1^{th}$ term, k^{th} term will be used:

$$\sum_{u=2}^{k+1} \delta F_u = \sum_{u=2}^{k} \delta F_u + \delta F_{k+1}. \tag{3.19}$$

Therefore, the computation of the terms for $k = 1$ to S can be performed in $O(1)$ time. Since the outer summation has complexity $O(S)$, then the total computational cost of computing $E(F)$ is $O(S*1)$ which is simply $O(N)$. Therefore, $E(F)$ can be computed in $O(N)$ time. This result can be generalized to $E(F)^{(n)}$ and $E(B)^{(n)}$ similarly.

3.6.2.4 Variance

Variance is given by;

$$\sigma^2 = \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} (C_{k,l} - \mu)^2. \tag{3.20}$$

Expanding the square we obtain;

$$\sigma^2 = \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} \left(C_{k,l}^2 - 2\mu C_{k,l} + \mu^2 \right). \tag{3.21}$$

This equation can be simplified to;

$$\sigma^2 = \left(\frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} C_{k,l}^2 \right) - \mu^2. \tag{3.22}$$

First term of the equation can be expanded using equation 3.10 into;

$$\mu^{(2)} = \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} \left(A + \sum_{u=2}^{k} \delta F_u + \sum_{v=2}^{l} \delta B_v \right)^2 . \tag{3.23}$$

Introducing the definition below;

$$X = \sum_{u=2}^{k} \delta F_u + \sum_{v=2}^{l} \delta B_v, \tag{3.24}$$

equation 3.23 can be written as;

$$\mu^{(2)} = \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} (A + X)^2 . \tag{3.25}$$

Applying the square;

$$\mu^{(2)} = \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} \left(A^2 + 2AX + X^2\right), \tag{3.26}$$

This equation can be simplified to;

$$\mu^{(2)} = A^2 + \frac{2A}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} X + \frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} X^2. \tag{3.27}$$

Expanding the X in second term we obtain

$$\frac{2A}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} \left(\sum_{u=2}^{k} \delta F_u + \sum_{v=2}^{l} \delta B_v \right), \tag{3.28}$$

which is equal to;

$$\frac{2A}{S} E(F) + \frac{2A}{R} E(B). \tag{3.29}$$

Next, expanding the X in the third term of equation 3.27 we obtain;

$$\frac{1}{SR} \sum_{k=1}^{S} \sum_{l=1}^{R} \left(\sum_{u=2}^{k} \delta F_u + \sum_{v=2}^{l} \delta B_v \right)^2, \tag{3.30}$$

which is equal to;

$$\frac{1}{S} \sum_{k=1}^{S} \left(\sum_{u=2}^{k} \delta F_u \right)^2 + \frac{2}{SR} \left(\sum_{k=1}^{S} \sum_{u=2}^{k} \delta F_u \right) \left(\sum_{l=1}^{R} \sum_{v=2}^{l} \delta B_v \right) + \frac{1}{R} \sum_{l=1}^{R} \left(\sum_{v=2}^{l} \delta B_v \right)^2 . \tag{3.31}$$

Using equations 3.15 and 3.16 equation 3.31 can be written as;

$$\frac{E(F)^{(2)}}{S} + \frac{2E(F)E(B)}{SR} + \frac{E(B)^{(2)}}{R}. \tag{3.32}$$

Then, we can write equation 3.27 using equations 3.29 and 3.32 as follows;

$$\mu^{(2)} = A^2 + \frac{2AE(F) + E(F)^{(2)}}{S} + \frac{2E(F)E(B)}{SR} + \frac{2AE(B) + E(B)^{(2)}}{R}. \tag{3.33}$$

Finally, using equation 3.22 and 3.33 we obtain variance as follows;

$$\sigma^2 = A^2 + \frac{2AE(F) + E(F)^{(2)}}{S} + \frac{2E(F)E(B) - \mu^2}{SR} + \frac{2AE(B) + E(B)^{(2)}}{R}. \quad (3.34)$$

Higher order moments of GBAS can be written similar to mean and variance using the terms introduced in equations 3.15 and 3.16. These terms are called *moment representing functions* of GBAS. They can be computed in linear time, $O(N)$ with the length of the shape boundary, N.

3.6.2.5 Fast GBAS and its Complexity

Based on the derivations presented in the previous section, algorithm 3.2 computes the GBAS shape descriptor of a shape having N boundary points in $O(N^2)$ time. In the algorithm, firstly, mean beam vector is computed and pixel set is partitioned into forward and backward pixels sets using the mean beam vector. Secondly, forward, backward beam angles and moment representing functions are computed. Thirdly, main beam angle is found. Finally, moment statistics are computed using the moment representing functions and main beam angle.

The 2^{nd} step of the Algorithm 3.2, computes the mean beam vector of the reference pixel, p_i in $O(N)$ time. The 3^{rd} step of the algorithm, partitions pixel set P into the forward, I and backward, G pixel sets in $O(N)$ time. In steps 4 and 5 of the algorithm, forward beam angles, $F_k(p_i)$ of the reference pixel are evaluated. Therefore, the operation is of order $O(S)$, which is on the order of $O(N)$. In steps 6 to 10, moment representing functions of forward pixels are computed. The number of operations that must be performed is $O(S * m)$. S is on the order of $O(N)$ and m is the number of moments to be computed, which is independent of the boundary length. Thus, this part of the algorithm is $O(N)$. In steps 11 to 17, every computation performed for forward pixels is performed for backward pixels, in $O(N)$ time as well. In steps 18 to 23, Generalized Beam Angle Statistics of the reference pixel is computed using the moment representing functions previously computed. These steps are independent of the size of the shape boundary, therefore they are completed in $O(1)$ time. As a result, computational complexity of generating feature vector of a single boundary pixel is $O(N)$. Consequently, the computational complexity of producing the GBAS Moment functions of a shape boundary having length N is $O(N^2)$.

Algorithm 3.2: Find GBAS shape descriptor of a given shape boundary P.

(1) **for** each pixel p_i in P **do**
(2) **compute** $OK(p_i)$
(3) **partition** P using $OK(p_i)$ into I & G
(4) **for** each forward pixel i_k in I **do**
(5) $F_k = atan(V(p_i, i_k)) - atan(OK(p_i))$
(6) **for** $k = 2$ to S **do**
(7) $\delta F_k = F_k - F_{k-1}$
(8) $Sum(\delta F_k) = Sum(\delta F_k) + \delta F_k$
(9) **for** $m = 1$ to d **do**
(10) $P_F(m) = P_F(m) + Sum(\delta F_k)^m$
(11) **for** each backward pixel g_l in G **do**
(12) $B_l = atan(V(p_i, g_l)) - atan(OK(p_i))$
(13) **for** $l = 2$ to R **do**
(14) $\delta B_l = B_l - B_{l-1}$
(15) $Sum(\delta B_l) = Sum(\delta B_l) + \delta B_l$
(16) **for** $m = 1$ to d **do**
(17) $P_B(m) = P_B(m) + Sum(\delta B_l)^m$
(18) $A = F_1 + B_1$
(19) $\alpha = S * R$
(20) $T_1 = P_F(1)/S + P_B(1)/R$
(21) $T_2 = P_F(2)/S + 2 * P_F(1) * P_B(1)/(R * \alpha) + P_B(2)/R$
(22) $\Gamma^1(p_i) = A + T_1$
(23) $\Gamma^2(p_i) = A^2 + 2AT_1 + T_2$

CHAPTER 4

MATCHING ALGORITHM

In the previous chapter, we introduced GBAS shape descriptor which represents shapes without a parametric boundary. In this chapter, we describe an algorithm to detect objects in images using the GBAS shape descriptor. We assume that the image contains a single object for which boundary is not available. The only preprocessing method we use is Canny edge detection. Matching algorithm selects the most similar boundary pixels of the template shape to the edge pixels in the edge detected image. In the experiments, we show that matching algorithm successfully extracts the template shape within the edges in the input image.

The chapter is organized as follows. In section 4.1, we define the matching problem formally. Section 4.2 introduces the match graph which is used to represent the matching problem in mathematical terms. Match path and the proposed algorithm to solve the matching problem are given in sections 4.3 and 4.4 respectively. Finally, in section 4.5, we investigate the computational complexity of the matching algorithm.

4.1 Matching Problem

In this study, *Matching* is defined as the correspondence;

$$M = \{(s_1, p(s_1)), (s_2, p(s_2)), \ldots, (s_L, p(s_L))\} \tag{4.1}$$

between the ordered set of boundary pixels of a template shape;

$$S = \{s_1(x_1, y_1), s_2(x_2, y_2), \ldots, s_L(x_L, y_L)\} \tag{4.2}$$

and the unordered set of edge pixels;

$$P = \{p_1(x_1, y_1), p_2(x_2, y_2), \ldots, p_N(x_N, y_N)\} \tag{4.3}$$

where $p(s_i)$ denotes the edge pixel that is matched with the boundary pixel s_i. Clearly, the solution of the matching problem is the parametric representation of the detected shape boundary;

$$B = \{p(s_1), p(s_2), \ldots, p(s_L)\}. \tag{4.4}$$

Therefore, the matching algorithm proposed in this study can be considered as a boundary extraction method as well.

Matching problem is illustrated in figure 4.1 where the two inputs are a template shape and an edge detected image. The output is the set of edges corresponding to the pixels of the template shape. The figure also shows that the matching algorithm uses the GBAS feature vectors and the Euclidean distances(spatial information) of the edge pixels to solve the correspondence problem.

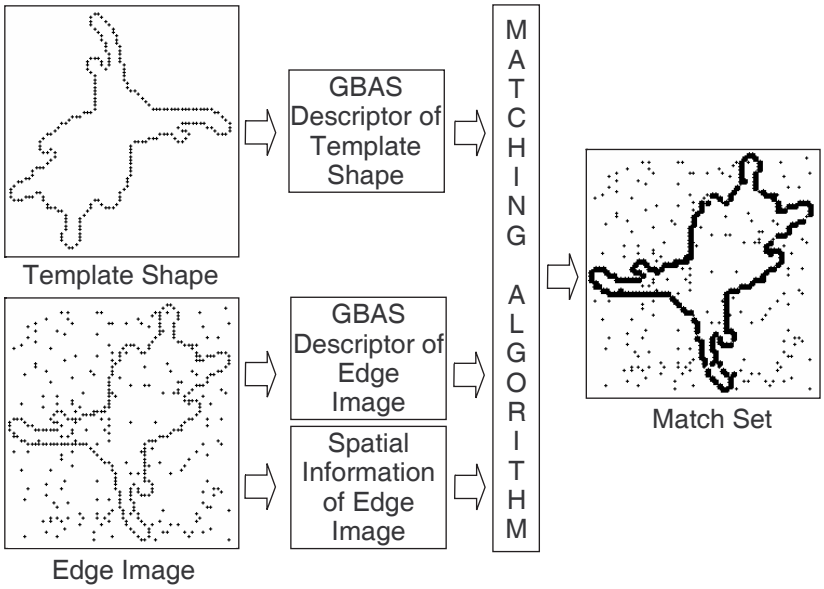


Figure 4.1: Matching Problem

4.2 Match Graph

Match graph is a mathematical representation of the matching problem. In the graph, there exists a node for a matching alternative between the template shapes boundary

pixels and the edge pixels. On the other hand, arcs of the match graph represent the ordering of the template shapes' boundary pixels. This allows the exploitation of the neighborhood relationships of both boundary pixels and edge pixels during the matching process. Moreover, it ensures a parametric boundary representation of the detected shape. This representation can be used in many applications, but a useful one for the matching process itself is the verification of the fitness of the detected shape to the template. This can be used in the performance evaluation of the matching algorithm. Now, let's define match graph formally.

4.2.1 Definition: Match Graph

Given a template shape S and an edge image P, *match graph*, $Z(E, F)$ is a multistage di-graph where the nodes $E(i, j) \in E$ represent the matching between s_i in the boundary pixels set S and p_j in the edge pixels set P.

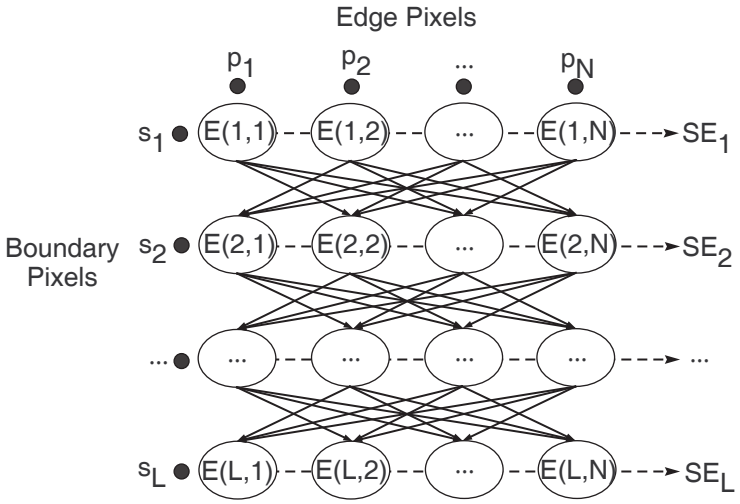


Figure 4.2: Matching Graph.

In the match graph, each row corresponds to the set of matching nodes of the boundary pixel s_i and given by;

$$SE_i = \{E(i, j) | j = 1, 2, \ldots, N\} \tag{4.5}$$

Each row of the graph is called the field of the corresponding boundary pixel.

The arcs in F are of the form $< E(i, j), E(i + 1, k) >$ for all $1 \leq k \leq N$ and $k \neq j$. In other words, in the match graph, there exists an arc from the node $E(i, j)$ to the

rest of the nodes in SE_{i+1}. The exception of this rule is the case where the matching nodes $E(i,j)$ and $E(i+1,k)$ are matched with the same edge pixel; $j = k$. Then, a match graph can be represented by a multi-stage di-graph, as shown in Figure 4.2.

In the match graph, the cost of an arc $< E(i,j), E(i+1,k) >$ is defined as follows;

$$F((i,j),(i+1,k)) = D(i,j) + \alpha U(p_j, p_k), \tag{4.6}$$

where $D(i,j)$ is the distance of GBAS feature vectors of s_i and p_j;

$$D(i,j) = \sum_{b=1}^{d} |\Gamma^b(s_i) - \Gamma^b(p_j)| \tag{4.7}$$

and $U(p_j, p_k)$ is Euclidean distance of the p_j and p_k edge pixels;

$$U(p_j, p_k) = \sqrt{(x_k - x_j)^2 + (y_k - y_j)^2}. \tag{4.8}$$

The parameter α is the normalization factor for making the algorithm independent of the image size. The value of this parameter restricts the allowed distances between the matched edge pixels of two consecutive boundary pixels. As the value of it increases, then the algorithm selects the next edge pixel to be matched closer to the current pixel.

4.3 Match Path

A match path is a path that starts from SE_1 of the first boundary pixel s_1, ending in SE_L of the last boundary pixel s_L, forming a matching set such that each boundary pixel is matched with a unique edge pixel in the set of edge pixels, P. Clearly, the match path represents the correspondences of the boundary pixels and edge pixels. Moreover, it determines the parametric boundary representation of the detected object in the edge image. Among many alternatives of match paths, optimum match path which has the minimum cost is of up most interest.

4.4 Matching Algorithm

The problem of matching the boundary of the template shapes to the edge pixels of an image is reduced into the problem of finding the minimum cost path (optimum path) in the match graph.

Algorithm 4.1: Find optimum match path Y of a match graph $Z(E, F)$.

Initialization:

for $1 \leq j \leq N$ **do**

$f_1(j) = D(1, j)$

Iteration:

for $2 \leq i \leq L$ **do**

for $1 \leq j \leq N$ **do**

$t_{min} = \mathbf{min}_t\ F((t-1, t), (i, j)) + f_{(i-1)}(t)$ **such that**

$t_{min} \in \{1 \leq t \leq N\} \wedge p_j \notin Path(E(t-1, t_{min}))$

$f_i(j) = F((t-1, t_{min}), (i, j)) + f_{i-1}(t_{min})$

$Q_i(j) = E(i-1, t_{min})$

Termination:

$\mathbf{min}_t\ f_L(t)$ **such that**

$t_{min} \in \{1 \leq t \leq N\}$

$Y = Path(E(L, t_{min}))$

As described above, the algorithm constructs a match graph, which represents all the matching alternatives between the boundary pixels of the template shapes and the edge pixels of the edge image. It, then finds an optimum matching using dynamic programming method.

Let, $f_i(j)$ be the minimum accumulated cost at node $E(i, j)$ in row i and $Q_i(j)$ be the column of the node in SE_{i-1} on the shortest path to the node $E(i, j)$ in SE_i, we define the path with the minimum accumulated cost as the optimum matching searched by the algorithm 4.1.

In the algorithm, $Path(E(i, j))$ denotes the sub-path that starts from the $E(i, j)$ match node of the match graph that is constructed by following $Q_i(j)$. Optimum match path Y, is constructed by following $Q_L(t_{min})$ that is in the field SE_L of the last boundary pixel s_L and has the minimum cost.

4.5 Computational Complexity

In this section, we investigate the computational complexity of the matching algorithm.

In the initial step, all matching nodes in the first row are processed and assigned the difference of GBAS feature vectors. The number of operations performed in this step is proportional to the number of edge pixels, N. Therefore, the computational complexity of this step is $O(N)$.

In the second step of the algorithm, all remaining matching nodes are processed. The number of matching nodes is equal to $(L-1)*N$. For each matching node, all previous matching nodes are investigated and the path having minimum cost is identified. This requires operations on the order of the number of edge pixels, $O(N)$. However, for each processed path, it is ensured that the edge pixel of the current matching node is not found on the path. This requires operations on the order of the length of the path formed. This means that on the average $O(L/2)$ operations will be performed. Total cost of this step can be computed by multiplying all these costs; $O((L-1)*N*N*L/2)$ which is simply $O(N^2L^2)$.

Finally, in the last phase of the algorithm, the matching node on the last row having minimum accumulated cost is identified. This requires the investigation of $O(N)$ matching nodes.

Clearly, the most demanding phase of the algorithm is the second phase. Thus, the computational complexity of the algorithm is $O(N^2L^2)$.

4.6 Increasing the Efficiency of the Matching Algorithm

Matching algorithm considers all match nodes of the match graph. For the match node, $E(i,j)$ of a boundary point s_i and an edge p_j, it iterates through the match nodes, SE_{i-1} of the previous boundary point, s_{i-1} and then finds the optimum matching node, $E(i-1,t)$ such that the cost,

$$E(i,j).cost = D(i,j) + E(i-1,t).cost + \alpha U(p_j, p_t) \tag{4.9}$$

is minimized. Recall that $U(p_j, p_t)$ is the Euclidean distance between the edges p_j and p_t. Consequently, processing the match nodes of the edges that are close to p_j is enough for the determination of the optimum match node. Therefore, processing all match nodes is an overhead and implementation of a scheme that will eliminate unnecessary match nodes can improve the performance of the matching algorithm. One of the possible methods is the use of range queries provided by multi-dimensional

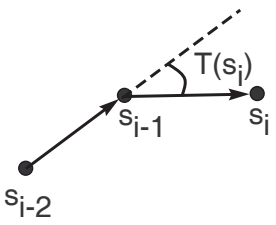


Figure 4.3: Turning angle, $T(s_i)$ of boundary point s_i.

indexing mechanisms [9].

Ensuring that each boundary point is matched with a unique edge is another overhead of the matching algorithm. One way of implementing this requirement is through the traversal of the sub-path, constructed by the algorithm and checking if p_j is already included on the path or not. The number of operations is proportional to the length of the sub-path and on the average it is of complexity $O(L)$, where L is the number of boundary points. This procedure can be viewed as a global optimization which ensures that each boundary point is matched with a unique edge. A more efficient solution can be obtained by employing a local optimization method, in which the probability of matching multiple boundary points with the same edge is low. We propose a local optimization method which is based on the idea of preventing the match of successive boundary points with the same edge. It is implemented with the addition of the turning angle constraint, $\beta(i,j)$. Accordingly, the turning angle at the boundary point, s_i must be close to the turning angle at the match edge, p_j. Turning angle, $T(s_i)$ of a boundary point s_i is defined as the positive angle between the vectors $V(s_{i-2}, s_{i-1})$ and $V(s_{i-1}, s_i)$ (see figure 4.3). Defining the turning angle constraint, $\beta(i,j)$ as follows;

$$\beta(i,j) = \parallel T(s_i) - T(p_j) \parallel \tag{4.10}$$

the cost function to be minimized becomes;

$$E(i,j).cost = D(i,j) + E(i-1,t).cost + \alpha U(p_j, p_t) + \beta(i,j). \tag{4.11}$$

Note that, the terms of the cost function except for $D(i,j)$ which represents the distance of the GBAS feature vectors of the boundary point s_i and edge p_j, are subject to change with respect to the previous match node, $E(i-1,t)$. The turn-angle constraint can be implemented without the introduction of additional cost to

the matching algorithm, therefore a performance gain of factor $O(L)$ can be achieved.

CHAPTER 5

EXPERIMENTS

In this chapter, we evaluate the performance of the proposed shape descriptor, Generalized Beam Angle Statistics, GBAS and the matching algorithm through experiments. Experiments are collected in two main classes:

- Descriptor performance of GBAS and
- Detection performance of the Matching algorithm.

In the first class of experiments, the descriptor performance of GBAS is evaluated using the MPEG-7 CE Shape-1 database. The results presented in section 5.1 indicate that GBAS is a shape descriptor as powerful as its predecessor: BAS. Therefore, generalizing the BAS for the purpose of object detection, we do not loose the power of it.

In the second class of experiments we evaluate the performance of the proposed matching algorithm using artificial and real edge detected images. Various experiments, which are presented in section 5.2 show that the matching algorithm can be used to detect a given template shape in edge detected images under different conditions.

5.1 The Performance of the GBAS Shape Descriptor

In this section, we demonstrate the performance of the proposed shape descriptor, GBAS. For this purpose, MPEG-7 Core Experiments Shape-1 data set is used. This

data set was formed during the MPEG-7 activities, which aimed to create standards on multimedia database indexing. The performance evaluation of shape descriptors through this data set is important because it allows the comparison of the performance with many other similar studies.

5.1.1 MPEG-7 Core Experiments Shape-1 Data Set

MPEG-7 Core Experiments Shape-1 data set contains three test cases: Part-A, Part-B and Part-C. In these test cases, all images are binary that contain a single object. Therefore, before the application of the tests contours are extracted using a contour following algorithm. Then, tests are performed as specified in the related sections.

5.1.1.1 Part-A

Part-A evaluates the performance of shape descriptors under scale and rotation transformations. It is divided into the Part-A1 and Part-A2 test classes that are obtained from 70 original digital images with the application of scaling by 0.1, 0.2, 0.25, 0.30 and 2.0 scale factors and rotation by 9, 36, 45, 90 and 150 degrees, respectively. Therefore, each test case contains 70 classes of 420 shapes. A class of shapes from Part-A1 and Part-A2 data sets are shown in figures 5.1 (a) to (f) and (g) to (l), respectively.

In the experiments, all of the 420 shapes are used for query. Then, top 6 retrievals are considered as matches. If an algorithm retrieves all 6 members of a class in the matches for all shapes, then it has 100% success.

5.1.1.2 Part-B

Part-B evaluates the performance of shape descriptors under similarity. It contains 70 classes of 1400 shapes. There are 20 shapes in each class. Classes are formed based on the similarity of the shapes. Example images from different classes of the data set are shown in figures 5.2 (a) to (f). On the other hand, figures 5.2 (g) to (l) demonstrate the diversity of shapes within a class of the data set.

In the experiments, all of the 1400 shapes are used as a query. Then, top 40 retrievals are considered as matches. If an algorithm retrieves all 20 members of a class in the matches for all shapes, then it has 100% success.

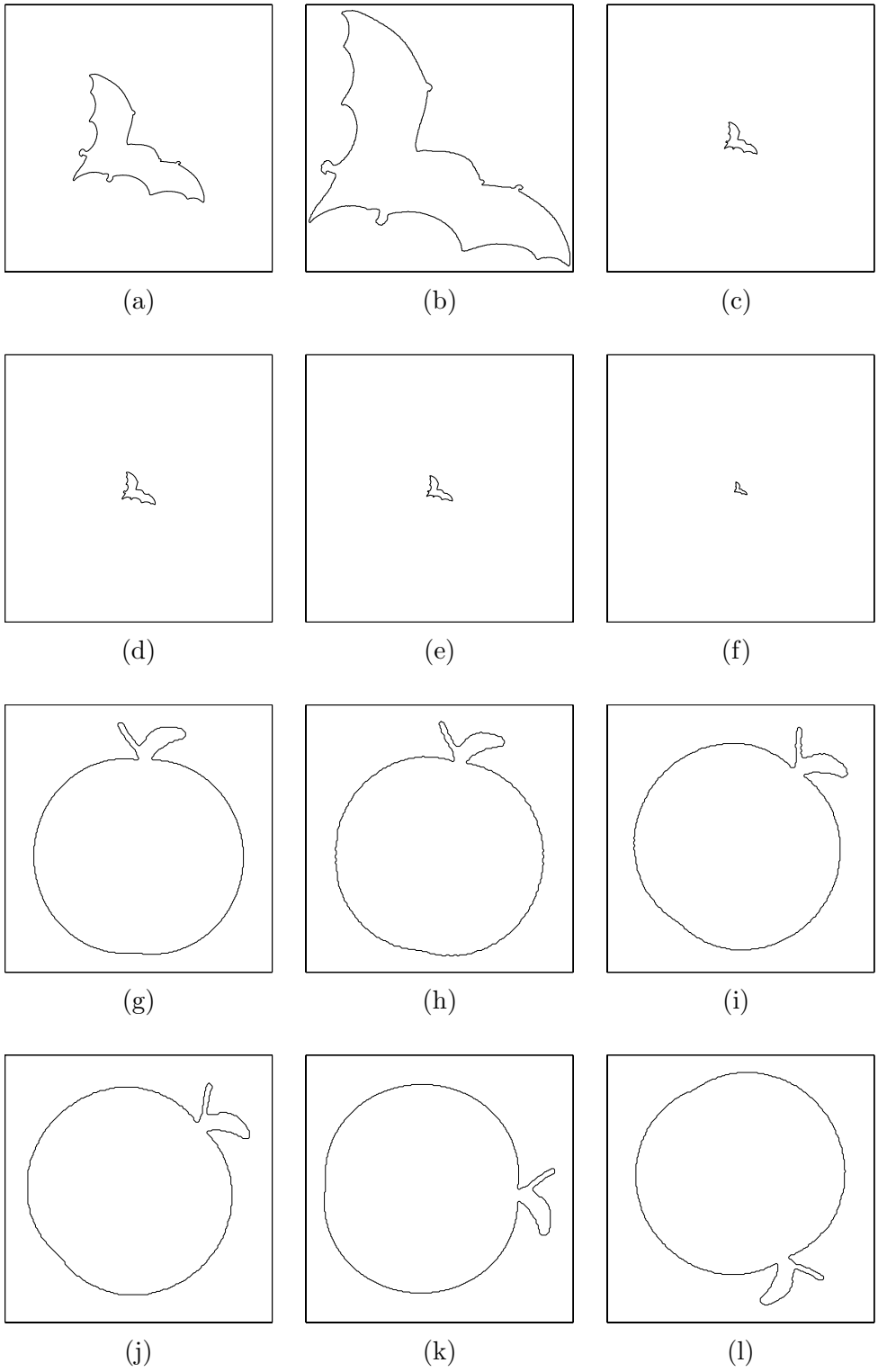


Figure 5.1: A class of shapes from the MPEG-7 CE Shape-1 Part-A1 and Part-A2 data sets are shown in figures (a) to (f) and (g) to (l), respectively. Original images are given in figures (a) and (g). Scales of figures (b) to (f) and rotations of figures (h) to (l) are 2.0, 0.3, 0.25, 0.2 and 0.1 and 9, 36, 45, 90 and 150 degrees, respectively.

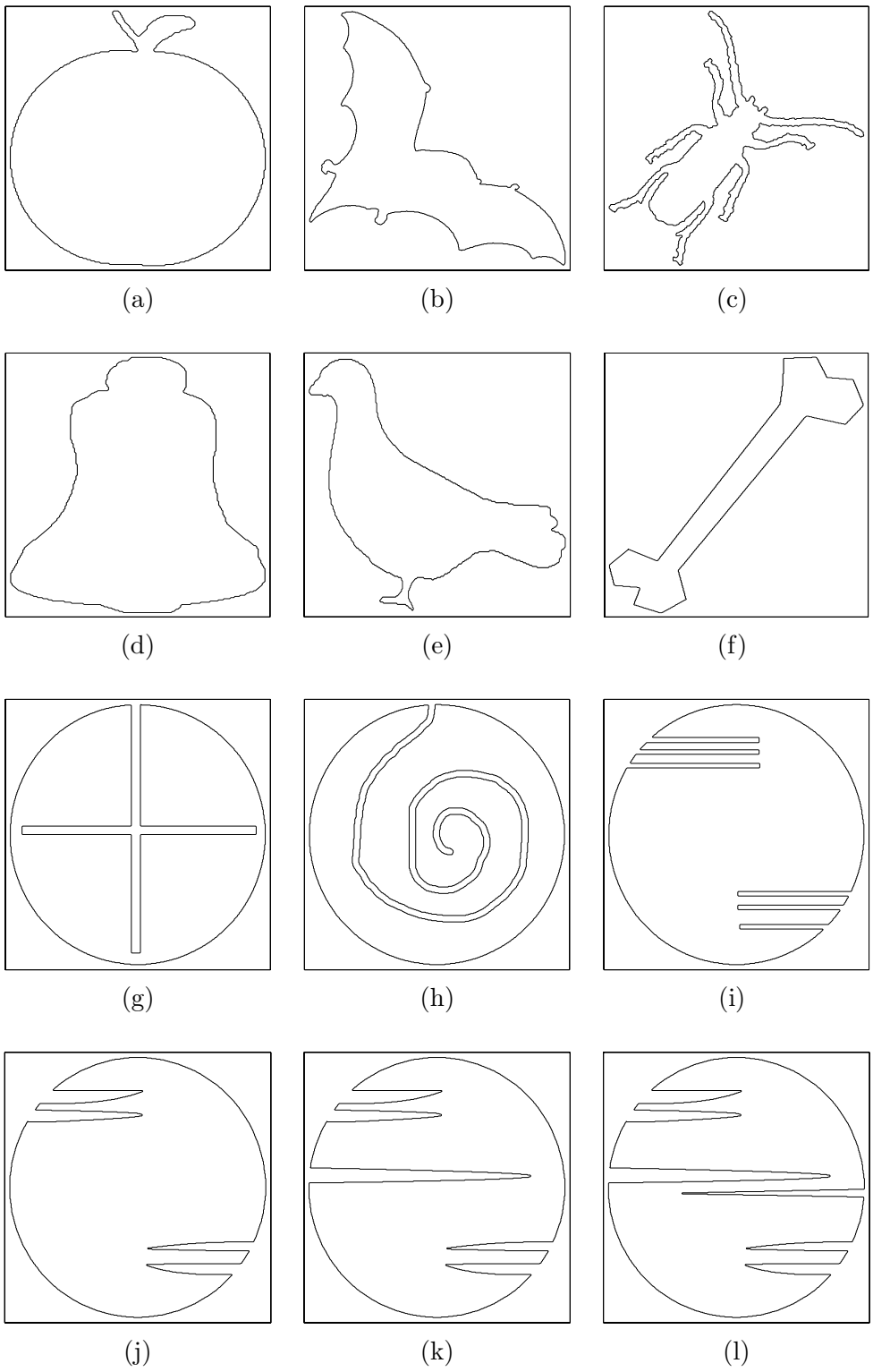


Figure 5.2: Example images from the MPEG-7 CE Shape-1 Part-B data set. Shapes in figures (g) to (l) are taken from a class of the set and they demonstrate the within class diversity of Part-B.

5.1.1.3 Part-C

Part-C, which contains 1300 images, evaluates the performance of shape descriptors under motion and non-rigid deformations. 200 images of the data set are the motion pictures of a bream fish. Examples of these images are given in figures 5.4 (a) to (n). Remaining 1100 images are of various marine animals. See figure 5.4 (o) to (ab) for examples of them.

In the experiments, only the query bream fish, which is shown in figure 5.3, is used for retrieval. Then, top 200 retrievals are considered as matches. If an algorithm retrieves all 200 images of bream fishes' motion pictures in the matches, then it has 100% success.

5.1.2 Feature Extraction

Feature extraction aims the reduction of the shape descriptor to smaller sizes. We extract features from the GBAS shape descriptor using a simple method. Accordingly, uniformly spaced samples taken from the GBAS shape descriptor are used as features. In all tests, 64 samples are taken.

5.1.3 Similarity Measurement

For similarity measurement, a Dynamic Time Warping (DTW) algorithm is employed. Dynamic Time Warping algorithms are originally developed for speech recognition [33, 30, 11, 32]. They are applied to the recognition of isolated musical patterns [2] and hand-writings [7] as well. DTW is suitable for the matching of signals that have local non-linear variations. It achieves this goal by stretching and compressing parts of the signals [4]. The details of the algorithm are given in [4].

5.1.4 Results

In the experiments, firstly feature vectors of shapes are computed and saved to descriptor files. Next, in the feature extraction phase 64 samples are taken by uniformly sampling the shape descriptors. Then, queries are performed as explained in the previous section. The results of the experiments along with the results of other shape descriptors are given in table 5.1.

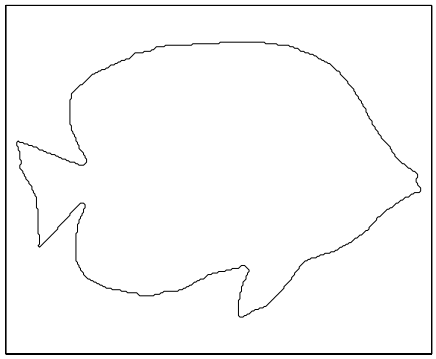


Figure 5.3: Query image of the MPEG-7 CE Shape-1 Part-C data set.

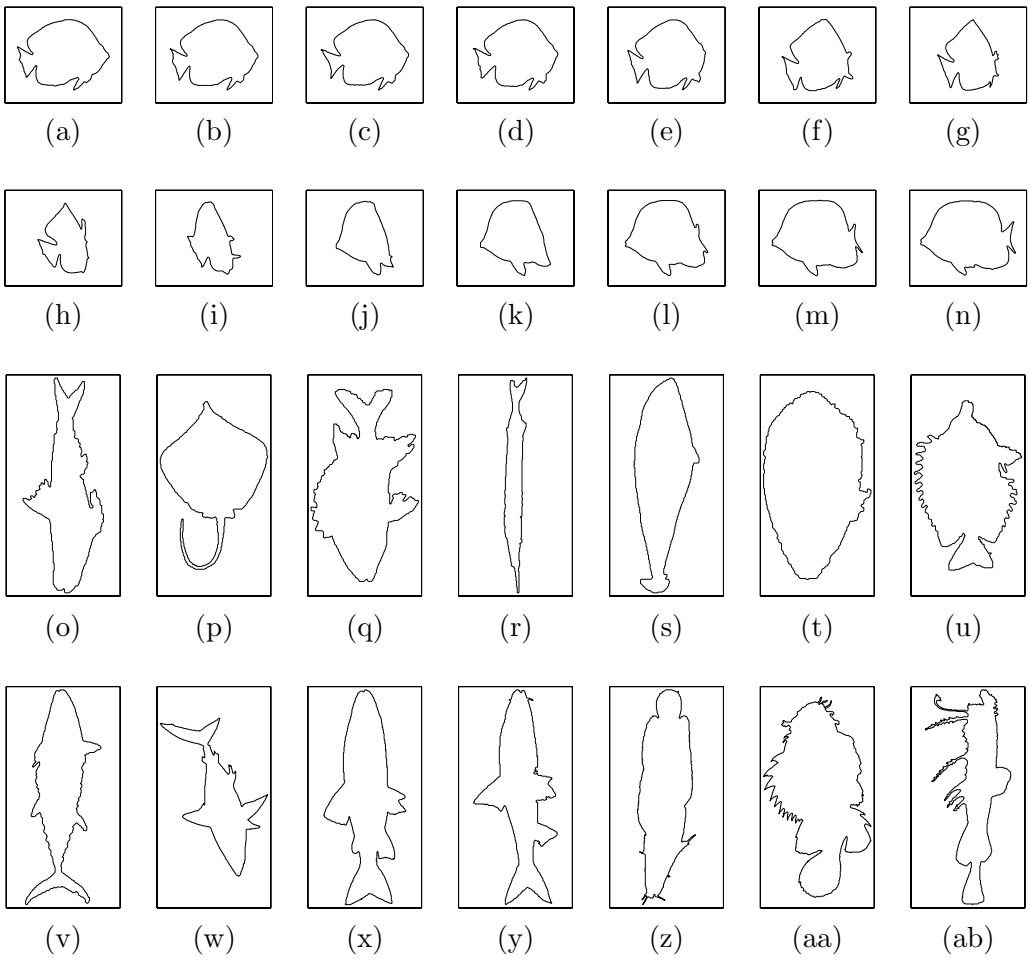


Figure 5.4: Example images from the motion pictures of the bream fish and the pictures of marine animals of the MPEG-7 CE Shape-1 Part-C data set are given in (a) to (n) and (o) to (ab), respectively.

Table 5.1: Performance comparison of GBAS in the MPEG-7 Core Experiments Shape-1 Data Set.

Descriptor	Part A1	Part A2	Part B	Part C
Shape Context	-	-	76.51	-
Tangent Space	88.65	**100**	75.45	92
Curvature Scale Space	89.76	99.37	75.44	**96**
Zernike Moments	**92.54**	99.60	70.22	94.5
Wavelet	88.04	97.46	67.76	93
DAG	85	85	60	83
BAS(60)	90.87	**100**	**82.37**	93.5
GBAS(64)	92.26	**100**	82.07	93.0
BEST Performance	92.54	100	82.37	96

Part-A1 or the scaling test performance of GBAS is 92.26%, which is a better performance than the one obtained with BAS, 90.87%. Moreover, this result is very close to the best performance exhibited by Zernike Moments, 92.54%. The superiority of GBAS' performance to BAS in the Part-A1 test indicates that GBAS is more successful in collecting sufficient statistics when the scale of shapes and consequently the length of the shape boundaries are small. In other words, this result shows that the statistical stability of GBAS is better than that of BAS.

Part-A2 or the rotation test performance of GBAS is 100%. This is very natural because rotation does not change the GBAS shape descriptor.

In Part-B or the similarity test, which is the most important test of MPEG-7, the performance of GBAS, 82.07% is nearly equal to the performance of BAS, 82.37% which is the best performance of this test. In this test, BAS and GBAS perform much more better than the other most competitive descriptor, Shape Context which has a performance of 76.51%. Therefore, in the similarity test GBAS' performance decreases compared to BAS, however it is still much more better than other studies in the literature.

In Part-C or the motion and non-rigid deformations test, the performance of GBAS is 93.0% which is slightly worse than BAS' performance, 93.5%. Part-C performance of GBAS is worse than Curvature Scale Space and Zernike Moment's performances that are 96% and 94.5% respectively. GBAS detects 186 of 200 motion pictures of bream fish correctly, however for 14 of them it fails. Explanation of this fact is that, although the object in all of the 200 images is the same, its shape differs from frame to

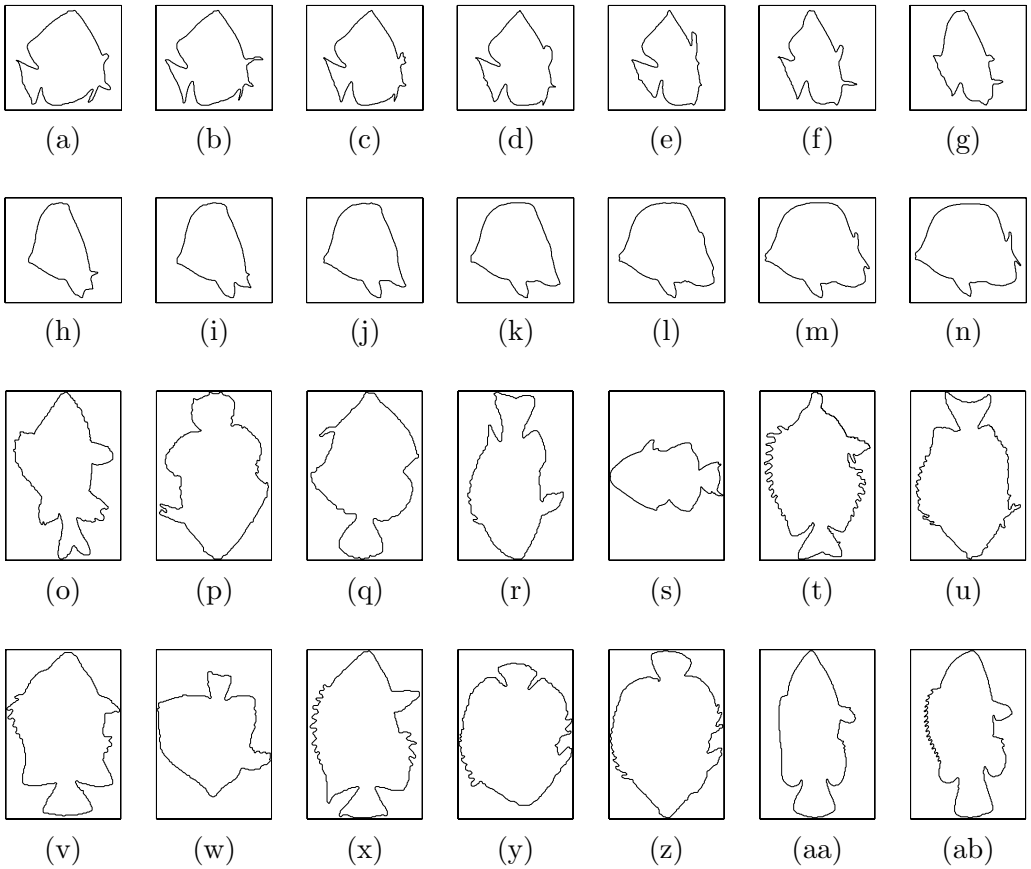


Figure 5.5: Missed and misclassified shapes in the MPEG-7 CE Shape-1 Part-C test are given in figures (a) to (n) and (o) to (ab), respectively.

frame due to the perspective projection involved. This produces most severe effects, when the fish turns around in the 3D space, creating a completely different set of shapes in the 2D imaging plane. In figures 5.5 (a) to (n) and (o) to (ab), missed and misclassified shapes are shown, respectively. Observe that, the query shape is more similar to the misclassified shapes than the missed shapes.

As a result, tests performed on MPEG-7 CE Shape-1 data set show that generalizing the BAS shape descriptor we gain a very important property, shape description in the lack of parametric boundary representation, however we do not loose much.

5.2 The Performance of the Proposed Matching Algorithm

In this section, we investigate the performance of the matching algorithm using the artificial and real edge images.

Figure 5.6: Template Shape Used in the Robustness to Rotation, Reflection, Scaling, Shearing, Gaussian Noise, Speckle Noise, and Occlusion Tests.

Artificial edge images, which are generated from the boundaries of the shapes in the MPEG-7 CE Shape-1 Part-B data set, demonstrate the performance of the matching algorithm under the rotation, scaling, reflection and shearing transformations, occlusion, Gaussian noisy shapes and Speckle noisy images.

Edge detected images are obtained from the MPEG-7 CE Shape-1 Part-C data sets' bream fish images with the addition of Gaussian noise and the application of the Canny edge detection algorithm. These images include the effects of noise, occlusion, smoothing and dislocalization of the edges. Therefore, this experiment demonstrates the performance of the matching algorithm in real images.

All of these experiments show that the matching algorithm can successfully detect template shapes in edge detected images that contain a single object.

5.2.1 Robustness of the Matching Algorithm to Rotation and Reflection

The matching algorithm uses the GBAS Moment functions of the template shape and the edge image as well as the distance between the edge pixels. In chapter 3, we have seen that GBAS Moment functions are invariant to rotation and reflection. The Euclidean distance between the edge pixels that is considered in the matching algorithm, does not affect the rotation and reflection robustness properties. As a result, the matching algorithm is invariant to rotation and reflection.

In figure 5.6 template shape used in robustness to rotation test is shown. It is matched with each of the 6 query images presented in figure 5.7 (a) to (f) and the

results of the matching are given in (g) to (l). Matching points are marked with big dots. Clearly, the template shape is successfully matched with the query images.

The same template shape is used in robustness to reflection test and it is matched with the query images shown in figure 5.8. Matching results are given in figure 5.9. Results indicate that the matching algorithm successfully matches reflected shapes.

5.2.2 Robustness of the Matching Algorithm to Scaling

In chapter 3, we have shown that GBAS is scale invariant. However, the distance of edge pixels depends on scale. Therefore, the matching algorithm searches for the same scaled instances of the template, by default. However, it is possible to search for a specified scaled instance. In this section, we demonstrate that the matching algorithm achieves to detect specified scaled instances of the templates.

The template shape which is shown in figure 5.6, is matched with 2.0, 2.5, 3.0, 3.5, 4.0 and 4.5 times scaled instances presented in figures 5.10 (a) to (f), respectively. Note that the scale factor between the template shape and the query image to be detected determines the α parameter of the matching algorithm. The results of matching, which are shown in figures 5.10 (g) to (l), indicate that the template shape is successfully matched with its scaled instances.

5.2.3 Robustness of the Matching Algorithm to Shearing

We know that GBAS is not invariant to shearing since it is a curvature based shape descriptor. As a result, matching algorithm is not expected to have a good performance in matching shapes that differ by a shearing transformation. However, when the amount of shear is small, then the effects are negligible and it is reasonable to expect good performance.

Sheared instances of the template along the horizontal axis by scale factors of 0.9, and 0.8 are shown in figures 5.11 (a) and (b) and the results of the matching are given in (c) and (d), respectively. Up to scale factor 0.8, the algorithm successfully matches the template shape with query images. However, for larger scale factors that are below 0.8, it fails.

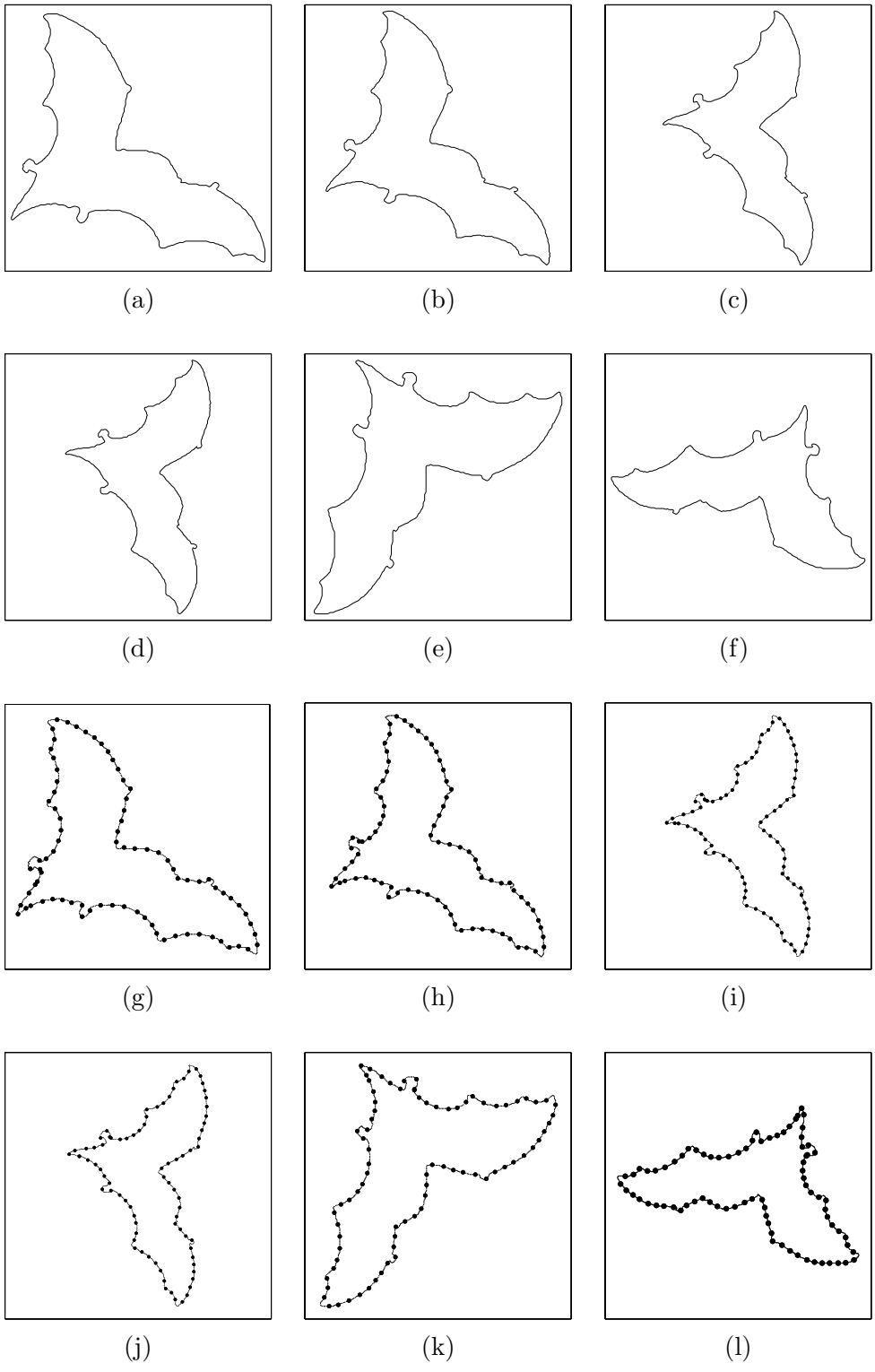


Figure 5.7: Query images of the rotation test are given in (a) to (f). Figures (g) to (l) show the results of matching the query images with the template.

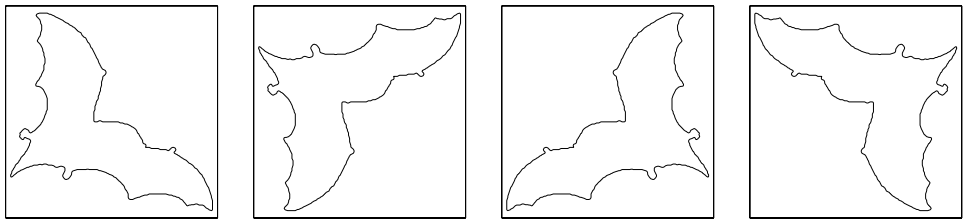


Figure 5.8: Query images used in robustness to reflection test. Original shape in (a) is reflected about the x, y and both of x and y axis and the figures (b), (c) and (d) are obtained, respectively.

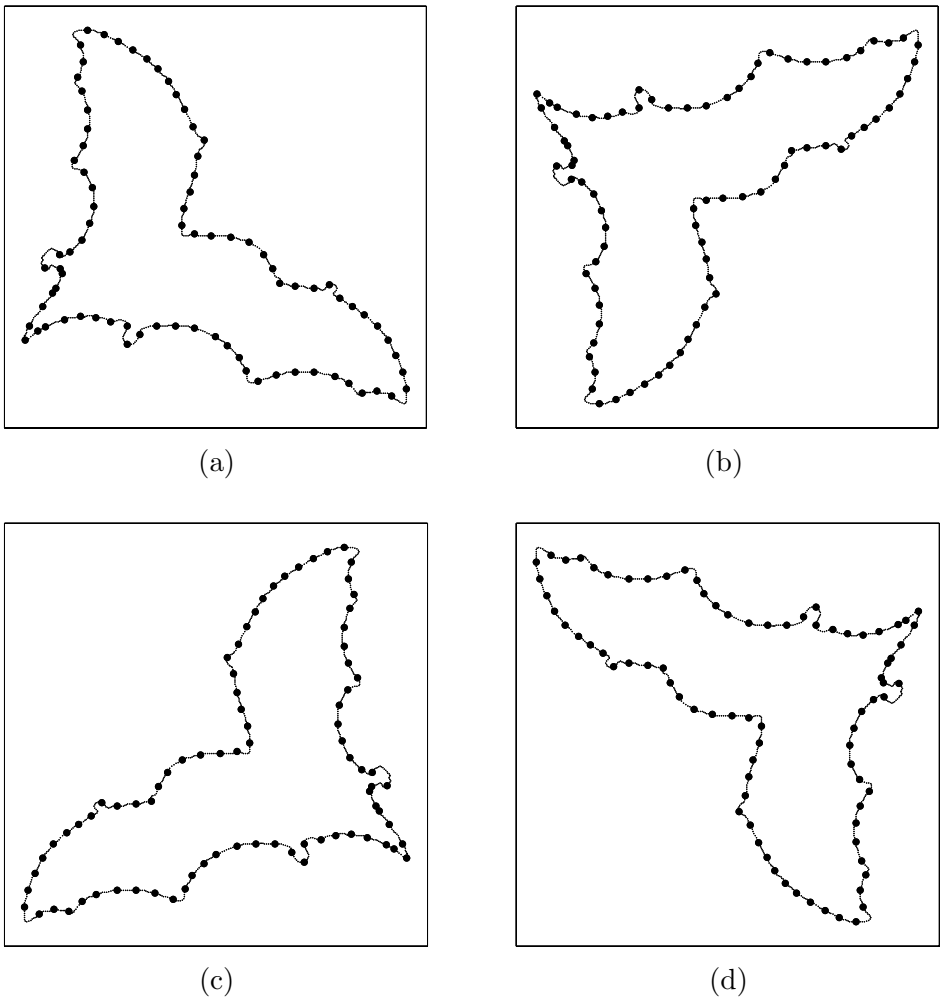


Figure 5.9: Results of matching the template shape with the query images in figure 5.8 (a) to (d) are shown in figures (a) to (d), respectively.

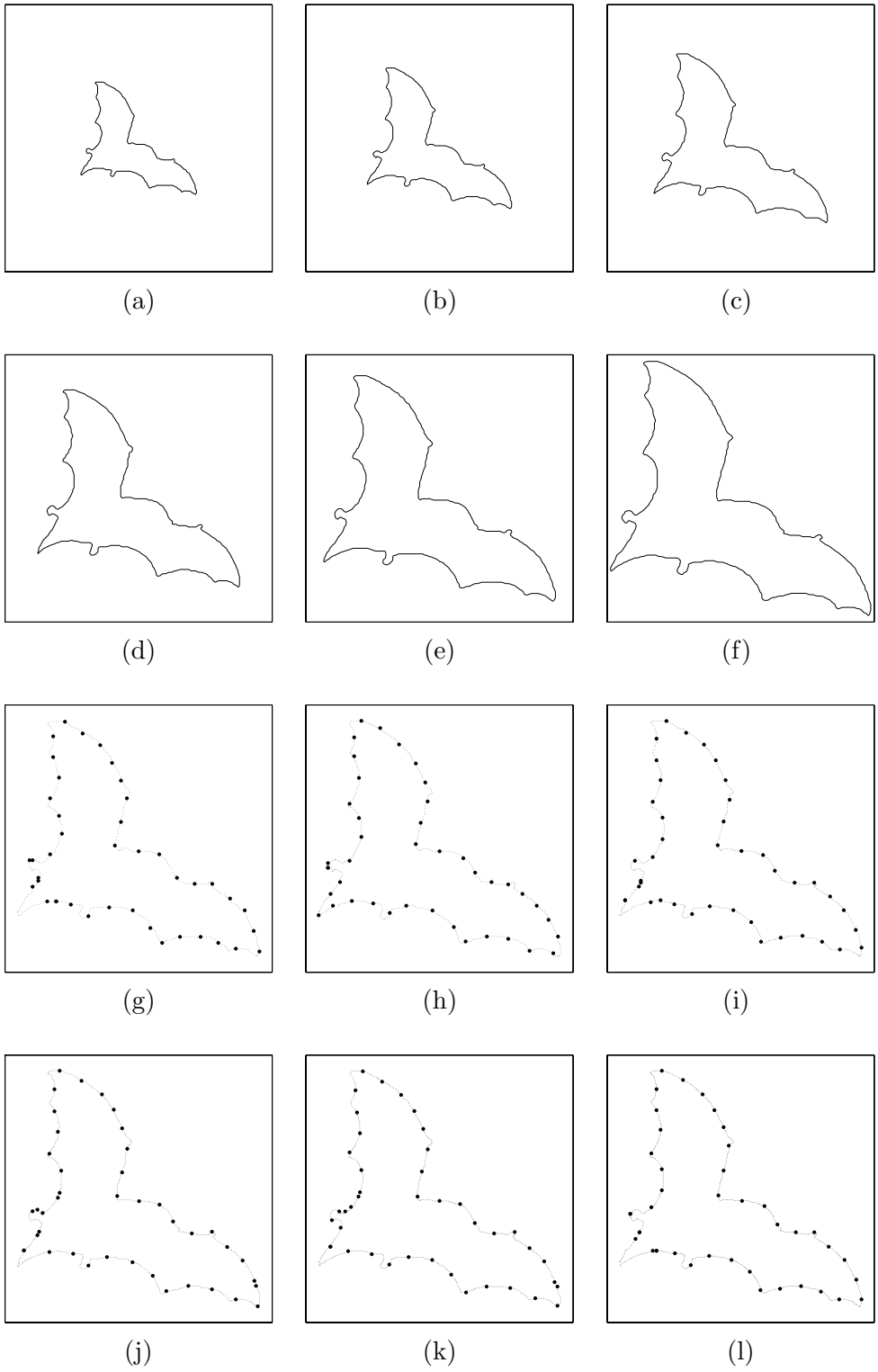
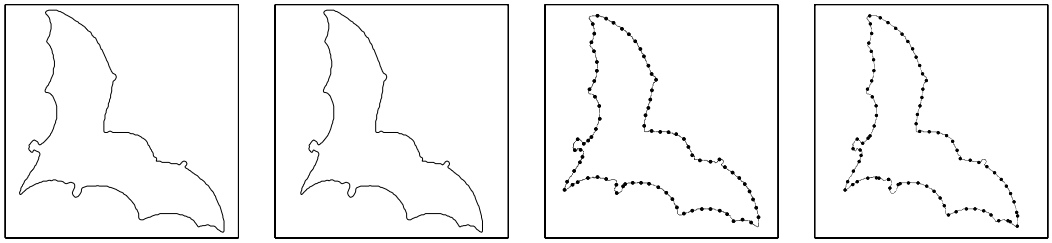


Figure 5.10: Query images used in the scaling test are given in (a) to (f). These are the scaled instances of the template presented in figure 5.6 having scale factors 2.0, 2.5, 3.0, 3.5, 4.0 and 4.5, respectively. Results of matching the template with the query images are shown in figures (g) to (l), respectively.



(a) (b) (c) (d)

Figure 5.11: Query images (a) and (b) are obtained by shearing the template with scale factors of 0.9 and 0.8 along the horizontal axis, respectively. Figures (c) and (d) show the results of matching the template with the query images.

5.2.4 Robustness of the Matching Algorithm to Gaussian Noise

In figure 5.12 (a), template shape used to demonstrate the robustness of the Matching algorithm to Gaussian noise is shown. Gaussian noise having zero mean and various variances (2.0, 4.0, 6.0, 8.0 and 10.0) is added to the template. Figures 5.12 (b) to (f) show these images. Noise is taken as the magnitude in the displacement of shape's boundary points. Direction of the displacement is taken as the direction of the gradient. To obtain noisy images, every boundary point of the shape is displaced with the amount of noise in the direction of the gradient. Then, the template is matched with the noisy images. Results are shown in figures 5.12 (g) to (l), respectively. Comparing the results of matching the template with itself (see figure 5.12 (a)) and the noisy images, one can conclude that the matching algorithm is robust to Gaussian noise.

5.2.5 Robustness of the Matching Algorithm to Speckle Noise

In order to create speckle noise, we add edge pixels randomly to the edge image. Coordinates of the noisy edge pixels are generated from a uniform distribution. Speckle noise, 20%, 30%, 40%, 50%, 60% and 75% of the number of true edge pixels is added to the template which are shown in figures 5.13 (a) to (f), respectively. Then, the template shape is matched with the query images. Results that are shown in figures 5.13 (g) to (l) indicate that the template shape can be detected successfully in speckle noisy images that have noise up to 75% of the true edges.

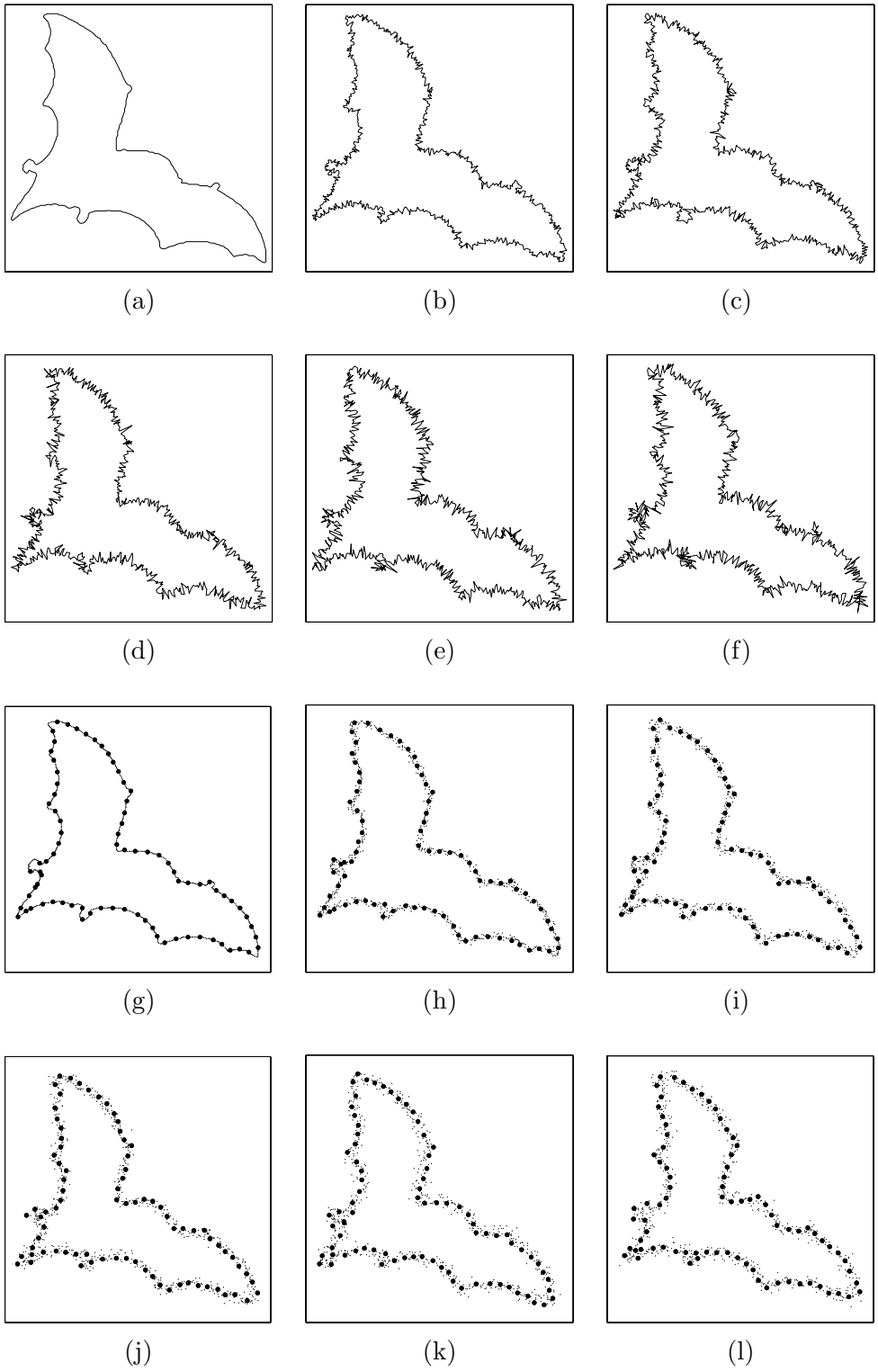


Figure 5.12: Template shape used in the Gaussian noise test is shown in (a). Query images generated from the template with the addition of Gaussian noise having 2.0, 4.0, 6.0, 8.0 and 10.0 variances are given in (b) to (f), respectively. The results of matching the template with the query images are shown in figures (g) to (l).

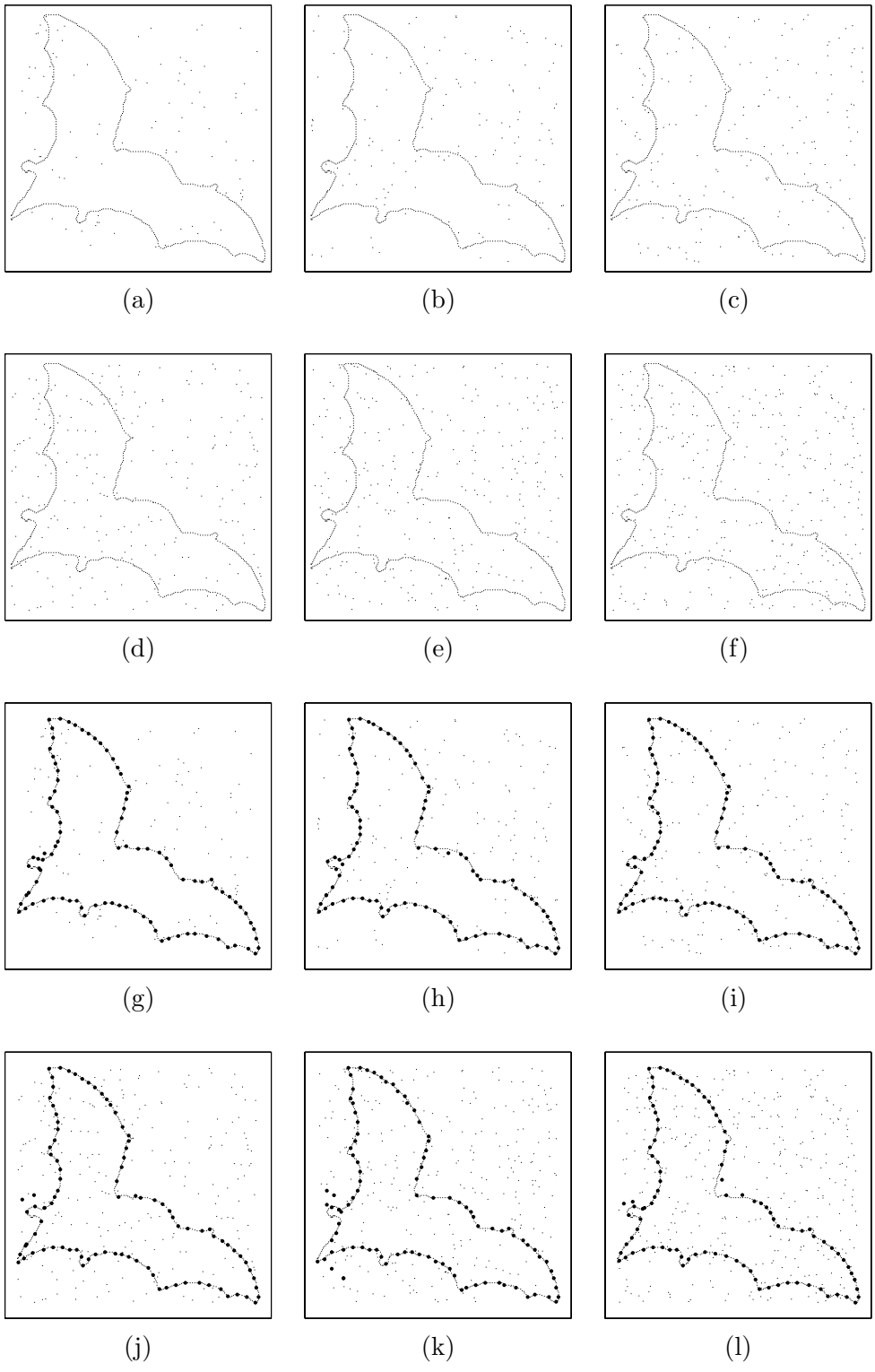


Figure 5.13: Figures (a) to (f) show query images generated from the template shape by adding 20%, 30%, 40%, 50%, 60% and 75% of speckle noise, respectively. The results of matching the template with the query images are shown in (g) to (l).

5.2.6 Robustness of the Matching Algorithm to Occlusion

Figures 5.14 (a) to (f) show query images generated by occluding the template 5%, 10%, 15%, 20%, 25% and 30% of the boundary pixels, respectively. Then, the template is matched with the query images. Results that are shown in figures 5.14 (g) to (l) indicate that the matching algorithm successfully matches images occluded up to 30% of the number of edge pixels.

5.2.7 Overall Performance

In the previous sections, we have shown tests that are applied on a single shape for the demonstration of the performance of the matching algorithm. In this section, we report the results obtained with the application of the tests to a larger class of shapes. The test class is generated by taking a sample shape from each of the 70 MPEG-7 CE Shape-1 Part-B data set classes. Therefore, a test case of 70 shapes is obtained. Then, reflected, rotated, scaled, sheared, occluded, Gaussian and speckle noisy instances of these shapes are generated and the template shapes are matched with them.

The results of matching the template shapes with themselves and rotated, reflected and scaled instances are shown in table 5.2. Success is presented in three scales: Perfect, Good and Un-matched which correspond to the matching of more than 95%, 70% and less than 70% of the template shapes' selected pixels and the edges of the query images. Accordingly, 92.86% of the templates are matched in the good scale for all tests. Moreover, 77.14%, 77.14%, 75.71% and 74.29% of the templates are matched in the perfect scale with themselves and rotated, reflected and scaled instances, respectively. However, 7.14% shapes could not be matched at all. These results indicate that most of the template shapes are matched successfully under rotation, reflection and scaling transformations. Additionally, 7.14% failure in the self-matching test show that, the matching algorithm is not suitable for the matching of some classes of shapes. We explain this fact as follows. The matching algorithm uses a polynomial time dynamic programming algorithm for the solution of the NP matching problem. Therefore, it finds a sub-optimal solution for the matching-problem. As a result, this solution does not correspond to the optimum solution or a successful matching for some classes of shapes.

The results of matching the template shapes with 0.9 and 0.8 factor sheared in-

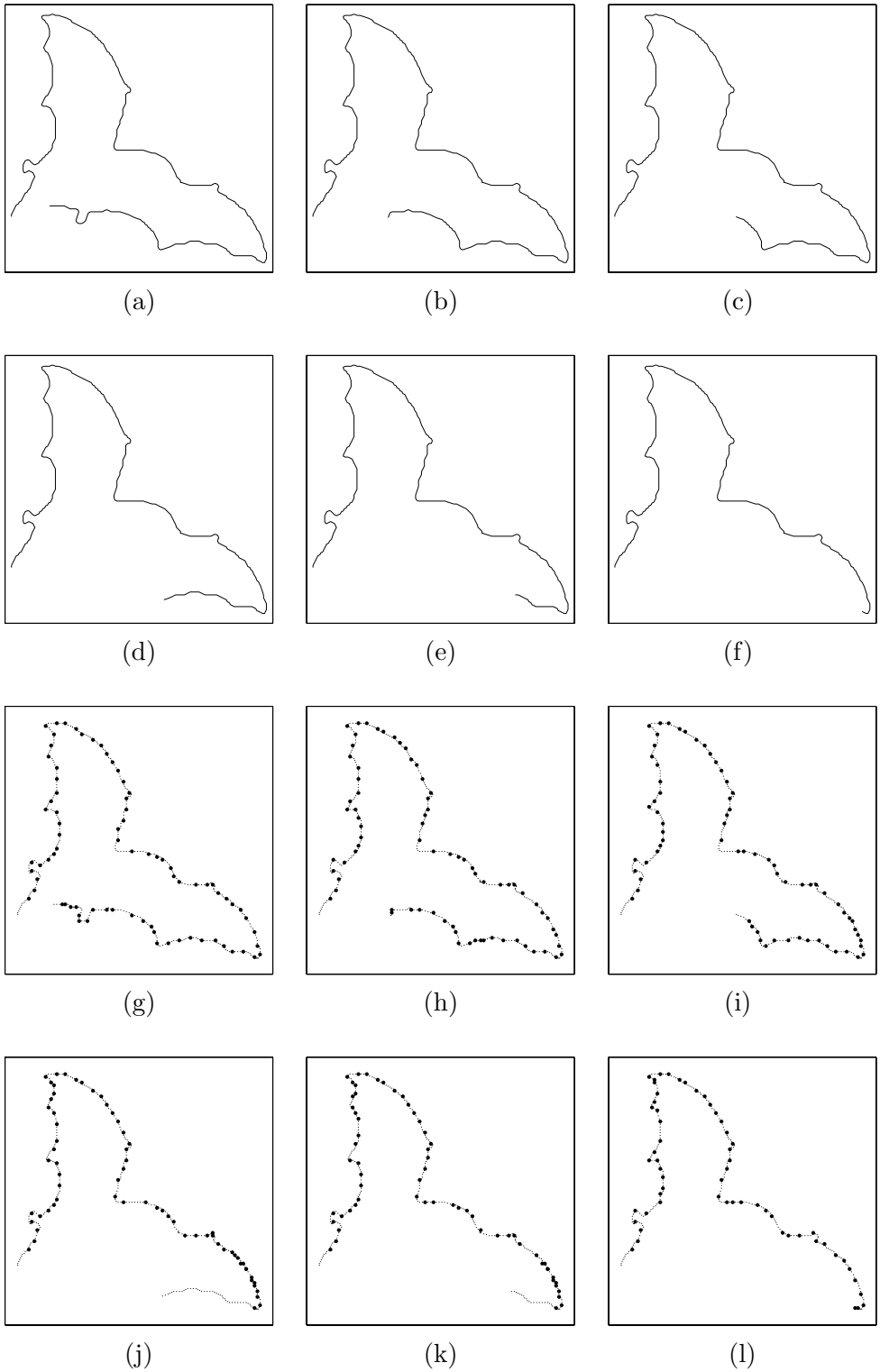


Figure 5.14: Figures (a) to (f) show query images generated from the template by occluding 5%, 10%, 15%, 20%, 25% and 30% of the boundary pixels, respectively. The results of matching the template with the query images are shown in figures (g) to (l).

Table 5.2: Performance of the matching algorithm under Self, Rotation, Reflection and Scaling Transformations.

Test	Perfect	Good	Un-matched
Self	77.14	92.86	7.14
Rotation	77.14	92.86	7.14
Reflection	75.71	92.86	7.14
Scaling	74.29	92.86	7.14

Table 5.3: Performance of the matching algorithm under Shearing.

Test	Perfect	Good	Un-matched
Sheared(0.9)	70	90	10
Sheared(0.8)	58.57	78.57	21.43

stances along the horizontal axis are presented in table 5.3. Accordingly, up to shear factor of 0.8, for 78.57% of the shapes matching is satisfactory.

The results of matching the template shapes with occluded shapes 5%, 10% and 15% of the boundary points are presented in table 5.4. Consequently, up to occlusion factor of 10% results are satisfactory for 70% of the shapes. For 15% occlusion, only 57.14% of the shapes can be matched satisfactorily.

The results of matching the template shapes with Gaussian noisy shapes having variances 1, 3, 5, 7 and 9 are presented in table 5.5. Experiments show that up to Gaussian noise having variance 7, more than 71.43% of the shapes can be matched successfully.

The results of matching the template shapes with Speckle noisy images having noise 25%, 50% and 75% of the number of true edge pixels are shown in table 5.6. Experiments show that up to Speckle noise 75% of the number of true edge pixels, more than 71.43% of the shapes can be successfully matched.

Table 5.4: Performance of the matching algorithm under Occlusion.

Test	Perfect	Good	Un-matched
Occluded(5%)	4.29	87.14	12.86
Occluded(10%)	0	70	30
Occluded(15%)	0	57.14	42.56

Table 5.5: Performance of the matching algorithm under Gaussian Noise.

Test	Perfect	Good	Un-matched
Gaussian Noise(0,1)	67.14	85.71	14.29
Gaussian Noise(0,3)	52.86	81.43	18.57
Gaussian Noise(0,5)	42.86	80	20
Gaussian Noise(0,7)	48.57	71.43	28.57
Gaussian Noise(0,9)	27.14	60	40

Table 5.6: Performance of the matching algorithm under Speckle Noise.

Test	Perfect	Good	Un-matched
Speckle Noise(25%)	57.14	90	10
Speckle Noise(50%)	55.71	78.57	21.43
Speckle Noise(75%)	45.51	71.43	28.57

5.2.8 Experiments performed on Edge Detected Images

In this experiment, we apply the matching algorithm to the edge detected images. For this purpose, we use the binary bream fish images of MPEG-7 CE Shape-1 Part-C data set. We add Gaussian noise to the binary images in order to introduce imperfections such as dislocalizations, missing and excessive edges. Then, we detect edges using the Canny edge detection algorithm. In figure 5.15, formation of the edge detected images are summarized. A binary bream fish image of the Part-C data set and its Gaussian noisy instances having variances 0.2, 0.4, 0.6, 0.8 and 1.0 are shown in figures (a) and (b) to (f), respectively. Figures (g) to (l) show the results of applying the Canny edge detection algorithm to the noisy images. Note that the intensities of the binary images are in the range 0 to 1. Observe that, as the variance of Gaussian noise increases, then the deformations introduce in the edge detected images increases as well. As a result, noisy edge detected images having dislocalizations, missing and excessive edges are obtained.

In the experiments, we match the query image (see figure 5.3) of the Part-C data set with the edge detected images. Table 5.7 summarizes the results of the experiments. The rows of the table correspond to the Gaussian noisy test sets generated from the Part-C data set with the addition of noise having different variances. Recall that there exists 200 motion pictures of a bream fish in the Part-C data set. Therefore, in each test case there are 200 edge detected images. The columns of the table show the percentage of perfect, good and un-matched matching cases. Recall also that a perfect,

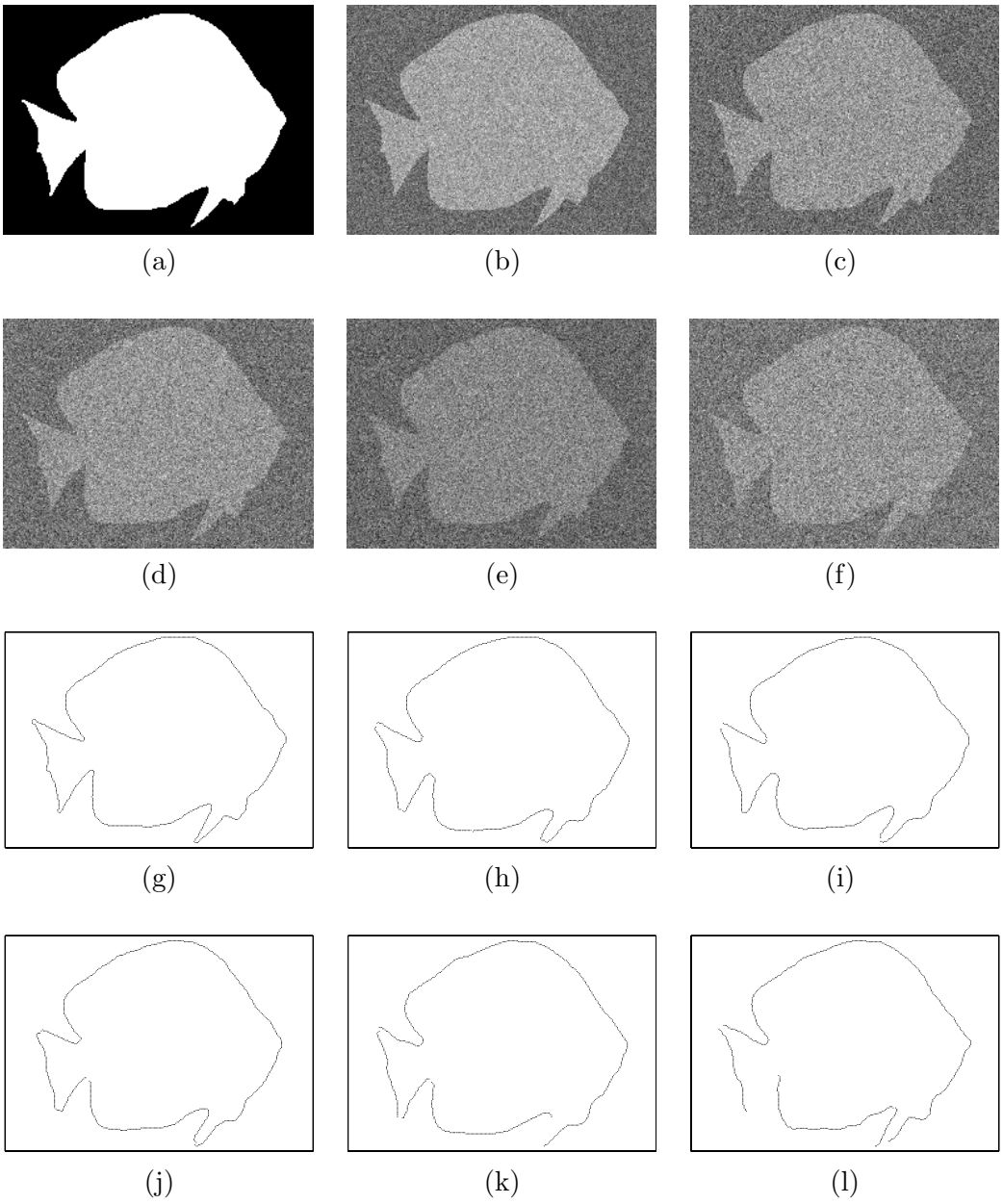


Figure 5.15: Original image is shown in figure (a). Images in (b) to (f) are the Gaussian noisy instances of the original image having zero mean and variances 0.2, 0.4, 0.6, 0.8 and 1.0, respectively. Edge detected images produced from the noisy images by the Canny edge detection algorithm are given in figures (g) to (l).

Table 5.7: Performance of the Matching Algorithm in Edge Detected Images.

Test	Perfect	Good	Un-matched
Gaussian Noise (0,0.2)	80	94	6
Gaussian Noise (0,0.4)	80	85.5	14.5
Gaussian Noise (0,0.6)	75	85	15
Gaussian Noise (0,0.8)	73	83.5	16.5
Gaussian Noise (0,1.0)	55.5	74.5	25.5

good and un-matched case refers to the matching in which more than 95%, 70% and less than 70% of the sampled points of the template are matched with the edges of the query image correctly, respectively. In this set of experiments it is observed that up to Gaussian noise having variance 1.0, 74.5% of the shapes can be matched successfully. As a result, we can say that the proposed shape descriptor and the matching algorithm can be used to detect a given template shape in edge detected images that contain a single object satisfactorily.

CHAPTER 6

CONCLUSION

In this thesis, we introduce a new method for the detection of shapes in edge detected images. This task is achieved by defining a shape descriptor, Generalized Beam Angle Statistics, GBAS and proposing a template based matching algorithm.

GBAS is obtained with the generalization of the Beam Angle Statistics (BAS), [5] for the purpose of object detection. It improves BAS so that the computation of the feature vectors of boundary points can be performed in the lack of parametric boundary representation. In other words, GBAS does not require tedious boundary extraction process. This property makes GBAS a suitable shape descriptor that can be used right after an edge detection algorithm that lead to edge map which may include missing or excessive edges. Therefore, GBAS can be applied on complex images where the boundaries of objects are not available.

Inspite of its ability to extract shapes from incomplete boundary information, GBAS preserves all the superiorities of BAS. First of all, GBAS assures the consistency to the Human Visual System by preserving all the convexities and concavities of the shapes. Also, it is invariant to translation, rotation, scale and reflection because none of these transformations change the set of generalized beam angles of boundary points significantly. An immediate consequence of this fact is invariance to more complex transformations that can be obtained with a combination of translation, rotation, scaling and reflection such as Euclidean motion, rigid body and similarity. Moreover, GBAS is robust to shearing, occlusion and noise. It has improved statistical stability because it uses $O(N^2)$ beam angles as opposed to BAS which uses $O(N)$ beam angles

for the description of the boundary points. Although GBAS uses more beam angles for description, its complexity is the same as the complexity of BAS, $O(N^2)$. This efficiency is achieved by the Fast GBAS algorithm that allows the computation of GBAS feature vectors without computing all beam angles.

The proposed matching algorithm locates the template shape within the edge map of the query image. It is translation, rotation and reflection invariant because the GBAS feature vectors and the Euclidean distance of edges, which are used by the algorithm, are invariant to these transformations. Although the matching algorithm is not invariant to scaling, it can be used to locate scaled instances of the template shape provided that the expected range of the scale factor is given.

Noise robustness is one of the most important properties of the matching algorithm. This is due to the averaging process performed during the computation of moments. Tests performed on artificial edge images show that the algorithm is robust to Gaussian noisy shapes until 10.0 standard deviation. The method is also robust to Speckle noisy images. The tests performed on real images show that it is also robust to the effects of smoothing, dislocalizations, motion and non-rigid deformations.

The experiments indicate that the proposed matching algorithm handles shearing successfully when the amount of shear is less than 0.2. However, larger amounts of shear deteriorate the performance of the algorithm. Employment of a method for the online discovery of shear can be quite useful in the matching of sheared shapes.

Discontinuities up to 30% of the entire shape boundary do not affect the matching performance significantly. However, the percentage length of the discontinuity to be tolerated depends on the characteristics of the boundary. In order to improve the matching performance, the algorithm can be extended to skip some of the boundary points without matching and continue the matching process.

Most important problem of the matching algorithm is the sub-optimal solution generated by the dynamic programming approach. As a result, some type of shapes cannot be matched. Fortunately, only for a small percentage of the shapes this led to the failure of the algorithm in the experiments. The class of shapes that the algorithm does not work must be further investigated thoroughly in the future studies.

One major constraint of the proposed method is the detection of a single object within an image. In other words, the proposed method assumes that there is a single

object to be extracted from a predefined region of the image. Therefore, the proposed method can work with the single-object-images or require a coarse segmentation as a preprocessing step to partition the image into parts with single object.

Symmetric shapes constitute another problem for the matching algorithm. For some of them, the algorithm matches the same symmetric part more than once leaving other parts unmatched. This is especially observed when one of the symmetric parts of the shape has some sort of noise. Different strategies can be employed for the solution of this problem including the removal of the already matched parts of shapes from further processing.

The computational complexity of the matching algorithm is prohibitive without the implementation of the proposed improvement methods. Multi-dimensional indexing mechanisms and the turn-angle constraint can provide significant performance gains. However, these methods complicate the coding of the algorithm.

In conclusion, the proposed shape descriptor, GBAS, together with the matching algorithm can be used to detect the template shapes in edge detected images which contain a single object. The method works under a variety of transformations and noise. Future studies will be directed for the detection of the template shapes in images that contain more than one object.

REFERENCES

[1] Y. Zhong A. K. Jain and S. Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):267–278, 1996.

[2] S. Theodoridis A. Pikrakis and D. Kamarotos. Recognition of isolated musical patterns using context dependent dynamic time warping. *IEEE Transactions on Speech and Audio Processing*, 11(3):175–183, 2003.

[3] C. Suen A. Taza. Discrimination of planar shapes using shape matrices. *IEEE Transactions on Systems, Man and Cybernetics*, 19:1281–1289, 1989.

[4] N. Arıca. *Shape: Representation, Description, Similarity and Recognition*. PhD thesis, Middle East Technical University, 2003.

[5] N. Arıca and F. T. Yarman Vural. Bas: A perceptual shape descriptor based on the beam angle statistics. *Pattern Recognition Letters*, 24:1627–1639, 2003.

[6] H. Asada and M. Brandy. The curvature primal sketch. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):2–14, 1986.

[7] C. Bahlmann and H. Burkhardt. The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):299–310, 2004.

[8] M. Basu. Gaussian-based edge-detection methods - a survey. *IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews*, 32(3):252–260, 2002.

[9] S. Berchtold C. Bhm and D. A. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys*, 33(3):322–373, 2001.

[10] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8 No.6:679–698, 1986.

[11] P. C. Chang and B.H. Juang. Discriminative training of dynamic programming based speech recognizers. *IEEE Transactions on Speech and Audio Processing*, 1(2):135–143, 1993.

[12] Luciano da Fontoura Costa and Roberto Marcondes Cesar Jr. *Shape Analysis and Classification: Theory and Practice*. CRC Press LLC, New York, 2001.

[13] G. Dudek and J. K. Tsotsos. Shape representation and recognition from multiscale curvature. *Computer Visian and Image Understanding*, 68(2):170–189, 1997.

[14] Nick Efford. *Digital Image Processing: a practical introduction using Java*. Harrow, England ; New York : Addison-Wesley, 2000.

[15] A. Mackworth F. Mokhtarian. Scale based description and recognition of planar curves and two-dimensional shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):34–43, 1986.

[16] S. Abbasi F. Mokhtrian and J. Kittler. Robust and efficient shape indexing through curvature scale space. In *Proceedings of the British Machine Vision Conference, Edinburgh, UK*, 1996.

[17] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Publishing Company, Inc., 1993.

[18] A. L. Ratan W. E. L. Grimson and W.M. Wells III. Object detection and localization by dynamic template warping. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1998 Santa Barbara, CA USA*, pages 634–640, 1998.

[19] J. Illingworth and J. Kittler. A survey of the hough transform. *Computer Vision, Graphics and Image Processing*, 44:87–116, 1988.

[20] C. S. Chen J. H. Chen and Y. S. Chen. Fast algorithm for robust template matching with m-estimators. *IEEE Transactions on Medical Imaging*, 51(1):230–243, 2003.

[21] S.Belongie J. Malik and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[22] H. K. Kim and J. D. Kim. Region-based shape descriptor invariant to rotation, scale and translation. *Signal Processing: Image Communication*, 16:87–93, 2000.

[23] K. F. Lai and R. T. Chin. Deformable contours: Modeling and extraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1084–1090, 1995.

[24] V. F. Leavers. Survey which hough transform? *CVGIP: Image Understanding*, 58(2):250–264, 1993.

[25] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31:983–1001, 1998.

[26] V. Hlavac M. Sonka and R. Boyle. *Image Processing, Analysis and Machine Vision*. Brooks/Cole Publishing Company A division of International Thomson Publishing Inc., 1998.

[27] David Marr. *A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman and Company, New York, 1982.

[28] E. Paquet M. Rioux A. Murching T. Naveen and A. Tabatabai. Description of shape information for 2-d and 3-d objects. *Signal Processing: Image Communication*, 16:103–122, 2000.

[29] J. Ponce R. Bajcsy D. Metaxas T. O. Binford D. A. Forsyth M. Hebert K. Ikeuchi A. C. Kak L. Shapiro S. Scarloff A. Pentland and G. C. Stockman. Panel on object representation for object recognition. *IEEE Conference on Computer Vision and Pattern Recognition, Seattle, Washington, June 1994*, 68(2):147–152, 1994.

[30] L. R. Rabiner and C. E. Schmidt. Application of dynamic time warping to connected digit recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28(4):377–388, 1980.

[31] F. Mokhtarian S. Abbasi and J. Kittler. Curvature scale space image in shape similarity retrieval. *Multimedia Systems*, 7(6):467–476, 1999.

[32] Y. W. Kim S. N. Kim, I. C. Hwang and S. W. Kim. A vlsi chip for isolated speech recognition system. *IEEE Transactions on Consumer Electronics*, 42(3):458–467, 1996.

[33] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

[34] D. H. Tagare. Deformable 2-d template matching using orthogonal curves. *IEEE Transactions on Medical Imaging*, 16(1):108–117, 1997.

[35] Q. M. Tieng and W. W. Boles. Recognition of 2d object contours using the wavelet transform zero-crossing representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8):910–916, 1997.

[36] Y. S. Kim W. Y. Kim. A region-based shape descriptor using zernike moments. *Signal Processing: Image Communication*, 16:95–102, 2000.

[37] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37:1–19, 2004.