

Dominant points detection for shape analysis

**A thesis submitted in fulfilment of the requirements
for the degree of Doctor of Mathematics and Computer Science
University of Cagliari, 2011**

Andrea Morgera

Department of Mathematics and Computer Science
University of Cagliari
Italy

*Alla mia maestra,
che si raccomandó di non farmi mai mancare lo studio
e a mia madre,
che le ha dato retta.*

Contents

1	Introduction	1
2	Shape representation and description techniques	5
2.1	Simple Shape Properties	6
2.2	Contour-based methods	8
2.2.1	Shape signatures	8
2.2.2	Correspondece-based shape matching	9
2.2.3	Other descriptors	9
2.2.4	Structural methods	11
2.3	Region based approaches	14
2.3.1	Global methods	14
2.4	Structural methods	16
3	On dominant points	19
3.1	Definition of dominant point	21
3.2	Algorithms for dominant point extraction	22
3.2.1	Direct Methods	22
3.2.2	Polygonal approximation	23
3.3	Polygonal approximation evaluation measures	30
4	Dominant points detection: proposed approaches	33
4.1	Dominant Point Iterative Localization (DPIL)	34
4.1.1	Initial point setup	34
4.1.2	Dominant point detection	34
4.1.3	Refinement	35
4.1.4	Experimental results	36
4.2	Ant Colony Optimization Polygonal Approximation (ACOPA)	45
4.2.1	Ant Colony Optimization algorithms	45
4.2.2	The ACO-based proposed method	46
4.2.3	Experimental results	50
4.3	Genetic Algorithms Polygonal Approximation (GAPA)	55
4.3.1	Genetic algorithms in polygonal approximation	55
4.3.2	The proposed method	55
4.3.3	Experimental results	59
5	Dominant point extraction techniques comparison	65
5.1	Polygonal approximation testing	66
5.2	Affine transformation test	68
5.2.1	Rotation	68
5.2.2	Scale	68
5.2.3	Translation	69
5.3	Noise sensitivity test	71
5.3.1	Noise contour generation	71
5.3.2	Quadratic Error E_2 testing.	72

CONTENTS

5.4	Analysis	75
6	Shape matching by dominant points	77
6.1	Shape outlines matching	78
6.1.1	The mathematical formulations	78
6.1.2	The alignment method	81
6.1.3	Experimental results and conclusions	86
6.2	ACO contour matching	96
6.2.1	Dominant point extraction	96
6.2.2	ACO Shape Matching	96
6.2.3	QAP formulation	97
6.2.4	Our approach	98
6.2.5	Experimental results	99
7	Conclusions	105
A	Image database	107
	Bibliography	111

Acknowledgements

I would like to dedicate this work mainly to my two children, Gaia and Leonardo, whose smile and joy is endless source of satisfaction.

Together with them, my beloved Annalisa whose encouragement and support will never fail.

I would also remember all my family from my mother to my brothers and my grandchildren that have always been very close to me.

Special thanks to Prof. Di Ruberto that, with her great preparation, practice and professionalism, always supported me and guided wisely during this experience.

I would also remind the staff of the Department of Mathematics and Computer Science, University of Cagliari, who gave me a warm welcome and always made me feel part of this academic institution: among them Prof. Nicoletta Dessi, Prof. Marco Gaviano and Prof. Giuseppe Rodriguez.

I also want to thank all those people who can't be listed here but which are an integral part of my daily life, making it happy.

From my heart, thank you all.

Andrea Morgera

List of Figures

1.1	An example of object detection by segmentation.	2
1.2	An example of contour correspondence by dominant points.	3
2.1	Let's denote with S the number of connected components and N the number of the holes. Then $Eul = S - N$. In this case Euler Number is equal to 1, -1 and 0, respectively	7
2.2	Curvature Signature for punch shape. Circles denote local maxima of curvature and squares denote local minima of curvature.	8
2.3	The shape contexts of two different versions of the letter "A" are shown. (a) and (b) are the sampled edge points of the two shapes. (c) is the diagram of the log-polar bins used to compute the shape context. (d) is the shape context for the circle, (e) is that for the diamond, and (f) is that for the triangle. As can be seen, since (d) and (e) are the shape contexts for two closely related points, they are quite similar, while the shape context in (f) is very different. [Source: Wikipedia]	10
2.4	(a) Boundary and its CSS image. (b) Change in orientation causes a circular shift in CSS image. (c) Noise creates small contours in CSS image.	11
2.5	Freeman coding.	12
2.6	A pincher shape partition (a) and its entities with their associated descriptions (b).	13
2.7	(a) An original shape in polar space. (b) polar-raster sampled image plotted in Cartesian space.	15
2.8	(a) Convex hull and its concavities. (b) Concavity tree representation of convex hull.	16
2.9	Construction of the medial axis of a rectangular shape.	17
3.1	Attneave's cat.	21
3.2	Dominant point extraction methods classification.	22
3.3	Example of polygonal approximation (10, 20 and 30 points) on dinosaur shape (550 contour points)	23
3.4	Example of ant colony optimization applied to the travelling salesman problem. 1. An example of solution 2. Each possible connection between cities depending on the visibility 3. Definition of the shortest path: the more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen 4. Having completed its journey, the ant deposits more pheromones on all edges it traversed, if the journey is short. [Source: Wikipedia]	26
3.5	Chromosome example: a) Contour b) Chromosome coding c) Approximating polygon corresponding to (b).	28

LIST OF FIGURES

3.6	E_2 error: the red area indicates the difference between the polygonal approximation and the original contour	30
4.1	A step of dominant point detection. A and B are the current dominant point. The square-enclosed point is the new dominant point.	35
4.2	Chromosome shape (50 points) : (a) Teh-Chin (15 points), (b) Wu BV (16 points), (c) Wu DYN (17 points), (d) our method (15 points), (e) our method (16 points), (f) our method (17 points)	37
4.3	Infinite shape (45 points) : (a) Teh-Chin (13 points), (b) Wu BV (13 points), (c) Wu DYN (13 points), (d) our method (13 points)	38
4.4	Leaf shape (120 points) : (a) Teh-Chin (29 points), (b) Wu BV (24 points), (c) Wu DYN(23 points), (d) our method (23 points), (e) our method (24 points), (f) our method (29 points)	39
4.5	Semicircle shape (102 points) : (a) Teh-Chin (22 points), (b) Wu BV (26 points), (c) Wu DYN (27 points), (d) our method (23 points), (e) our method (27 points)	40
4.6	Shapes used in the experiments.	42
4.7	Shapes used in the experiments.	43
4.8	Africa's map : (a) original shape, (b) 20 points approximation, (c) 30 points approximation, (b) 40 points approximation	43
4.9	Maple leaf : (a) original shape, (b) 40 points approximation, (c) 45 points approximation, (b) 50 points approximation	44
4.10	Example of ants behaviour. At the first step an ant finds food, and traces the path taken (round trip) with the release of pheromone. At the second step the other ants will choose different paths. As you go along, the longer routes will lose the trail of pheromone. In the third step we have created a path with a high concentration of pheromone that will be one of the shortest road to reach the destination. [Source: Wikipedia]	46
4.11	Some real world curves: fish, aircraft, hand, key.	51
4.12	F10 contour: the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying <i>Method_1</i> in (a), (b), (c) and by applying <i>Method_2</i> in (d), (e), (f), respectively.	52
4.13	Chromosome shape : Original (60 points), approximation by 8 segments, approximation by 15 segments, approximation by 22 segments	61
4.14	Semicircle shape : Original (102 points), approximation by 10 segments, approximation by 18 segments, approximation by 30 segments	61
4.15	Noise key shape : Original (269 points), approximation by 8 segments, approximation by 16 segments, approximation by 20 segments	61
4.16	F10 shape : Original (409 points), approximation by 10 segments, approximation by 20 segments, approximation by 35 segments	61
4.17	Hand shape : Original (528 points), approximation by 11 segments, approximation by 28 segments, approximation by 42 segments	61
5.1	Shapes used in experiments.	66

LIST OF FIGURES

5.2	Dominant points extraction for fish shape (325 contour points, 25 dominant points) detected by DPIL, ACOPA and GAPA	66
5.3	Examples of dominant point extraction for rotated hand shape.	69
5.4	Examples of dominant point extraction for scaled shapes.	70
5.5	Examples of noise addition ($\sigma = 0.1$, 5 repetition). The final image is obtained by filling the internal contour pixels.	71
5.6	Difference between original and noisy f10 shape.	72
5.7	E_{2i} trend for the proposed algorithms.	73
6.1	The dark regions contribute to define ind_1	80
6.2	The maximum of the distances of A from I' and B from I gives ind_2	81
6.3	Image contours I and I' with $ind_2(I, I') = 0$ but $\bar{I} \neq \bar{I}'$	82
6.4	A model made up of two image contours.	87
6.5	Plot of the overlapping indices of all images.	87
6.6	Precision-recall curve for algorithm OA_1 and set S^1	93
6.7	Precision-recall curve for algorithm OA_2 and set S^1	93
6.8	Precision-recall curve for algorithm OA_1 and set S^2	93
6.9	Precision-recall curve for algorithm OA_2 and set S^2	94
6.10	Bipartite graph ant crossing example. The path is $i_2 \rightarrow j_1 \rightarrow i_3 \rightarrow j_3 \rightarrow i_1 \rightarrow j_2$. The assignment is given by $\pi(i_1) = j_2, \pi(i_2) = j_1, \pi(i_3) = j_3$	96

List of Tables

4.1.1 Comparative results for the proposed method. Lower E_2/CR is better.	41
4.2.2 F10 Shape : 399 contour points.	50
4.2.3 The initial number of points, N , the number of dominant points, Np , and the approximation error E_2 for the real curves for our methods and Yin's algorithm. In bold the best approximation errors.	51
4.2.4 Comparison in terms of number of dominant points, Np , and E_2 on the benchmark curves.	53
4.2.5 Comparison with other methods on the benchmark curves.	54
4.3.6 Results for Fig.4.13 (60 points)	60
4.3.7 Results for Fig.4.14 (102 points)	60
4.3.8 Results for Fig.4.15 (269 points)	62
4.3.9 Results for Fig.4.16 (409 points)	62
4.3.10 Results for Fig.4.17 (528 points)	62
4.3.11 Comparative results between our method (OUR), Masood' one [49] and Ho-Chen's one [73]	63
4.3.12 Comparative results between our method (OUR), Huang and Sun'one [55] and Yin's one [56]	64
5.1.1 Comparative results for the proposed methods with and without the final refinement process.	67
5.1.2 E_2 comparative results for the proposed method with and without the final refinement process.	67
5.2.3 Rotation invariance test.	69
5.2.4 Scale invariance test.	70
5.3.5 Noise sensitivity test.	73
5.3.6 Dominant points relative position: the reported values are related to the sum of the distance of each dominant point compared to the original dominant point position, i.e. with $\sigma = 0$.	74
5.4.7 PROs & CONs of proposed techniques	75
6.1.1 Results of algorithm OA_1 with models of one image (set S^1).	89
6.1.2 Results of algorithm OA_1 with models of up to two images (set S^1).	89
6.1.3 Results of algorithm OA_2 with models of one image (set S^1).	90
6.1.4 Results of algorithm OA_2 with models of up to two images (set S^1).	90
6.1.5 Results of algorithm OA_1 with models (mod=1) of one image (set S^2).	91
6.1.6 Results of algorithm OA_1 with models (mod=2) of up to two images (set S^2).	91
6.1.7 Results for algorithm OA_2 with models (mod=1) of one image (set S^2).	92
6.1.8 Results for algorithm OA_2 with models (mod=2) of up to two images (set S^2).	92

LIST OF TABLES

6.1.9	Results for algorithm OA_1 with models of one image (set S^1) created automatically.	94
6.1.10	Results for algorithm OA_1 with models of up to two images (set S^1) created automatically.	94
6.1.11	Results for algorithm OA_2 with models of one image (set S^1) created automatically.	94
6.1.12	Results for algorithm OA_2 with models of up to two images (set S^1) created automatically.	95
6.1.13	Results for algorithm OA_1 with models of one image (set S^2) created automatically.	95
6.1.14	Results for algorithm OA_1 with models of up to two images (set S^2) created automatically.	95
6.1.15	Results for algorithm OA_2 with models of one image (set S^2) created automatically.	95
6.1.16	Results for algorithm OA_2 with models of up to two images (set S^2) created automatically.	95
6.2.17	Average accuracy calculated for 35, 40, 45, 50 Dominant Points (DP) and 50 constant sampled points (CS). The experiments are set with 100 iterations, 1 ants and no modification to shape contexts.	100
6.2.18	Average accuracy calculated for 35, 40, 45, 50 Dominant Points (DP) and 50 constant sampled points (CS). The experiments are set with 100 iterations, 1 ants and rotation invariance modification to shape contexts.	100
6.2.19	Methods comparison: accuracy calculated for each query shape. For each image, i.e. for each class, the average accuracy over 10 runs is shown.	102
6.2.20	Image matching for the run of experiment with $DP = 45$, 185 iteration and 1 ant with the best precision. For each query shape, i.e first column image, the ranking of the first 11 images ordered by descending similarity is shown. The achieved accuracy is 0.9192.	103
A.0.1	Image set 1 - the first column represents the query images.	108
A.0.2	Image set 2 - the first column represents the query images.	109

1

Introduction

In the last time the automatic information retrieval from huge databases has attracted a growing interest; this fact is related to the recent advances in computer technology and to the revolution in the way information is processed.

The content extraction and the consequent indexing appear as two of the most analyzed areas of research in a scenario of challenges for effective and efficient retrieval in a multimedia information context.

Nowadays progresses made in the sectors of hardware, telecommunications and image processing make relatively easy the process of acquisition, storage and transmission of a large number of digital images and video. Image repository are used in an extraordinary number of multimedia applications and fields such as entertainment, business, art, engineering and science. Computer imaging has developed as an interdisciplinary research field whose focus is on the acquisition and processing of visual information by computer and has been widely used in object recognition, image matching, target tracking, industrial dimensional inspection, monitoring tasks, etc.

An important goal in image understanding is to detect, track and label objects of interest present in observed images. In the area of machine vision applications, the aim of shape recognition is to identify an object correctly. Image objects can be characterized in many ways according to their colors, textures, shapes, movements and locations (spatial relationships).

The term Content-based image retrieval (CBIR) is commonly referred to any technology helpful in organizing digital pictures archives by their visual content. The CBIR covers different fields, such as, computer vision, machine learning, information retrieval, human-computer interaction, database systems, Web and data mining, information theory, statistics, and psychology.

All the proposed approaches fall into the problem of assigning a semantic similarity starting by a visual similarity because of the semantic gap between low-level content and higher-level concepts.

The task of visual indexing is not “naturally” suitable for an exact matching, unlike to what happens for textual information. The goal is to realize a similarity matching between a given query and images in a database, aside from translation, rotation and scale. For instance in a big-size visual library a user may be interested in find images containing well-defined subjects like cars, people, etc...

CHAPTER 1. INTRODUCTION



Figure 1.1: An example of object detection by segmentation.

In a content-based query processing the first step is the extraction of visual features and/or segmentation for searching similar images in the visual feature space.

The growing interest in region-based visual signatures can't abstract from the step of segmentation which objective is to partition an image into regions (fig. 1.1). This task is addressed in different, more or less successful, ways: thresholding, clustering, compression-based, histogram-based, edge detection, region growing, split-and-merge, graph partitioning, watershed methods and many others. The main problems are noticed in computational complexity, reliability of good segmentation and acceptable segmentation quality assessment. Another crucial step in shape analysis consists in extracting meaningful features from digital objects. To describe a visual property, for the whole image (global) or for a small pixel region (local), a feature extraction process is required.

After the extraction of image features, the main issue regards the way they could be indexed and matched against each other for retrieval. The choice of an appropriate feature representation and a suited similarity measure to rank pictures is fundamental.

A basic factor for content-based image retrieval is the method used for comparing images. Most recognition approaches tend to restrict the representation space of objects by focusing on the main variations in the projected intensity pattern (intensity-based), on the within category variations of shape when restricted to a set of models (model-based), and on the variations of the apparent contour (shape-based). A major feature commonly used attains to the shape of segmented image regions.

Among all aspects underlying visual information, the shape of the objects plays a special role and an emerging opinion in the vision community is that global features such as shapes of contours should also be taken into account for the successful detection and recognition of objects.

Shape matching is a central problem in visual information systems, computer vision, pattern recognition and robotics. Applications of shape matching include industrial inspection, fingerprint matching and content-based retrieval.

Many different kinds of shape matching methods have been proposed so far and their matching rate has been substantially improved during the years. In most cases these approaches yield a pair-wise shape similarity measure. Some distance function is used to measure the difference between two shapes: the smaller the difference, the more similar the two shapes are. Nevertheless this fact should not mislead: practically often shapes belonging to a same class may have large distance because the chosen distance measure cannot capture the intrinsic property of the shape.

The problem of recognition when shapes are encoded by their contours is in-

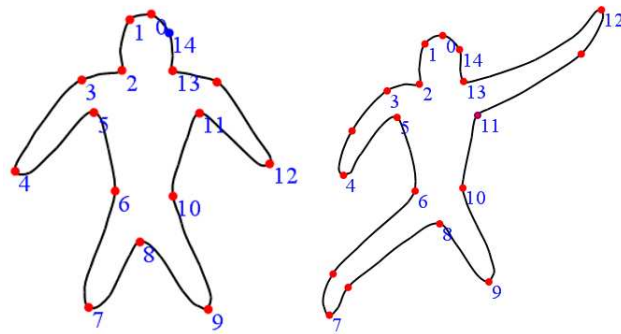


Figure 1.2: An example of contour correspondence by dominant points.

teresting for many reasons: contours are easy to obtain and can be used for shape representation regardless of shape convexity; they are one of the most commonly used shape descriptors; since they may be considered as high order curves, algorithms for contour-based recognition are potentially extensible to more generic shape classes including cursive words and hand-drawn sketches. Due to their semantically rich nature, contours are one of the most commonly used shape descriptors, and various methods for representing the contours of 2D objects have been proposed, each achieves, more or less successfully, the most desirable features of a proper representation, such as data compression, simplicity of coding and decoding, scaling and invariance under rigid motions, etc.

One of the way to address the correspondence problem between two shapes involves the choice of a meaningful mapping from points of first shape to a second shape which minimizes a given objective function in order to obtain a convenient measure to make a similarity ranking in a shape database. Such correspondence may be achieved by matching significant points of the contour also known as *dominant points* instead of taking into account the entire contour of the image (fig. 1.2). With the term dominant points we refer to those points of the boundary having a sufficiently high curvature, containing almost complete information of a contour, that can can suitably describe the curve for both visual perception and recognition. Such kind of matching techniques must consider problems like the order preserving rather than the one-to-one or one-to-many correspondence and, in most cases, lead to an optimization problem often approached by a metaheuristic algorithm. The work of this thesis faces the problem of dominant point detection on closed contours, offering an overview on shape analysis and current state-of-the-art techniques for dominant points detection and polygonal approximation and describing three novel approaches to the problem. The proposed methods are compared with other well known algorithms and their good performance against typical image transformations and noise conditions is also shown. Furthermore the shape matching problem it is proposed by introducing the dominant point approach in a novel alignment technique and in quadratic assignment problem addressed by ant colony optimization.

The thesis is organized as follows:

CHAPTER 1. INTRODUCTION

- In chapter 2 we introduce a classification of object shape features. Talking about binary objects, an “ideal” descriptor must provide important characteristics of shape, must be invariant to affine transformations, must be robust to noise and distortion and must have a low computational cost for indexing and retrieval: in a word it must extract prominent attributes in order to act as the human behaviour in shape recognition processes. An overview on most common used techniques to extract meaningful features from shapes is given, differentiating between simple descriptors (sec. 2.1) and two main classes: *external* methods (sec. 2.2), which are focused on the analysis of contour points of the objects, or *internal* methods (sec. 2.3), which use all the contours pixels and those inside the objects.
- In chapter 3, given the importance of contours, we approach to the problem of dominant points detection by defining them (sec. 3.1) and illustrating a panoramic about extraction techniques and their classification (sec. 3.2), distinguishing corner detection and polygonal approximation approaches. Moreover in sec. 3.3 we describe the main indicators used commonly to assess the efficiency of a polygonal approximation.
- In chapter 4 we describe in detail the three proposed approaches for dominant points detection: an iterative method, called Dominant Point Iterative Localization (*DPIL*) (sec. 4.1), and two heuristics techniques based on Ant Colony Optimization (*ACO*) paradigm, called *ACOPA*, *Ant Colony Optimization Polygonal Approximation* (sec. 4.2) and Genetic Algorithms (*GAs*) paradigm, called *GAPA*, *Genetic Algorithms Polygonal Approximation* (sec. 4.3). We test the performance of the methods by evaluating the polygonal approximation goodness against other common techniques described in literature and used to address this problem.
- In chapter 5 the comparative behaviour of the techniques shown in the previous chapter is evaluated. In sec. 5.1 we compare the polygonal approximation “capability” by measuring the ratio of the integral squared error and the compression ratio. In sec. 5.2 we address the invariance testing to transformations like scale, rotation and translation. In the end (sec. 5.3) we analyse the way in which such methods work on noisy conditions.
- In the last chapter we report two examples that explore one of the many fields in which dominant points can be applied. In practice we adapt two approaches for shape matching described in literature by replacing dominant points instead of manual selection, i.e. “made by human”, and constant contour sampling. In the first case (sec. 6.1) a curve alignment procedure is performed and a drastic computational time reduction without affecting appreciably accuracy results is achieved by using dominant points; in the second (sec. 6.2), where an *ACO* contour correspondence schema is implemented, we improve the already good conclusions of the original work further.
- In the end we describe conclusions and future works in chap. 7.

2

Shape representation and description techniques

In this chapter we discuss about shape analysis models.

Shape representation methods, or shape descriptors, extract effective and perceptually important shape features. In this context it is desirable to preserve important characteristics of shape and, at the same time, to be invariant to transformation like scale, rotation or translation. It is expected that descriptors have a robust behaviour respect to noise affected shapes or distorted, defective or partially occluded shapes as like act the human behaviour in such situations. In addition extracted features have to be suitable for indexing and online retrieval, i.e. with low complexity and computational cost, and performing well not only on certain type of shapes.

These features derive from either shape boundary information or boundary and internal regions content. Based on such difference we can discern into *contour-based* methods and *region-based* methods for feature extraction.

Besides a further classification involve the fact that the contour, in contour-based, or the whole shape, in region-based, is split in sub-parts or not. In the first case we refer to *global* (or *continuous*) approaches and *structural* (or *discrete*) approaches.

Granted this division, in section 2.1 we describe some simple properties. In general such properties are not capable to make fine distinctions and they are suitable to filter the number of possible shapes so that more complicated descriptors can be used for matching in a subset of the original database.

In section 2.2 we describe the most common and important techniques for the extraction of features just by the contour.

Conversely, in section 2.3 we point out the significant shape descriptors developed in the years and obtained by the analysis of the whole shape and not only from boundary information.

CHAPTER 2. SHAPE REPRESENTATION AND DESCRIPTION TECHNIQUES

2.1 Simple Shape Properties

In object retrieval the similarity of shapes is measured by salient features.

A raw comparison, i.e. with large differences, between shapes is possible by evaluating geometric features, even if they are not suitable to be standalone shape descriptors.

Such used measure may be listed in the following:

- **Center of gravity** also known as centroid;
- **Axis of least inertia (ALI)** that represents a unique reference line to preserve the orientation of the shape and is defined as the line for which the integral of the square distance to points on the shape boundary is minimum;
- **Average bending energy** is defined as

$$BE = \frac{1}{N} \sum_{s=0}^{N-1} K(s)^2 \quad (2.1.1)$$

where $K(s)$ is the curvature function, s is the arc length parameter and N is the the number of points on a contour;

- **Eccentricity** is defined as the ratio of the length of the major axis to the length of the minor axis;
- **Circularity** is defined in different ways:
 - ratio of the area of the shape to the area of a circle having the same perimeter as the shape;
 - ratio of the area of the shape to the shape's perimeter square;
 - known as *circle variance*

$$C_{va} = \frac{\sigma_R}{\mu_R} \quad (2.1.2)$$

where μ_R and σ_R are the mean and the standard deviation of the radial distance from the centroid of the shape to the boundary points;

- **Ellipse variance** is defined as the mapping error of a shape to fit the an ellipse that has an equal covariance matrix as the shape;
- **Rectangularity** is defined as the ratio of the area of the shape to the area of the minimum bounding rectangle;
- **Convexity** is defined as the fraction of the region boundary that coincides with the regions convex hull (the convex hull is defined in sec. 2.4);
- **Solidity** is defined as the ratio of the area of the shape to the convex hull area;

2.1. SIMPLE SHAPE PROPERTIES

3 B 9

Figure 2.1: Let's denote with S the number of connected components and N the number of the holes. Then $Eul = S - N$. In this case Euler Number is equal to 1, -1 and 0, respectively

- **Euler Number** describes the relation between the number of connected components and the number of holes in the shape;
- **Profiles**: the projection of the shape to the x -axis and y -axis on a Cartesian coordinate system;
- **Hole Area Ratio**: the ratio of the area of the shape to the total area of the holes in the shape.

2.2 Contour-based methods

2.2.1 Shape signatures

In many application it is useful analysing some features of the shape by studying measures derived from the boundary coordinates transformed in a one-dimensional function. Typically such parameters are used in a preprocessing phase to other feature extraction algorithms.

In [1], for example, a **complex representation** of Cartesian coordinates is done by a parametric curve:

$$z(t) = (x_1 + t(x_2 - x_1)) + j(y_1 + t(y_2 - y_1)), \quad t \in (0, 1), \quad (2.2.3)$$

where $z_1 = x_1 + jy_1$ and $z_2 = x_2 + jy_2$ are the complex numbers obtained by the boundary coordinates (x_i, y_i) .

Another common used function is the **centroid distance** (fig. 2.2) expressed by the distance of the boundary points from the centroid.

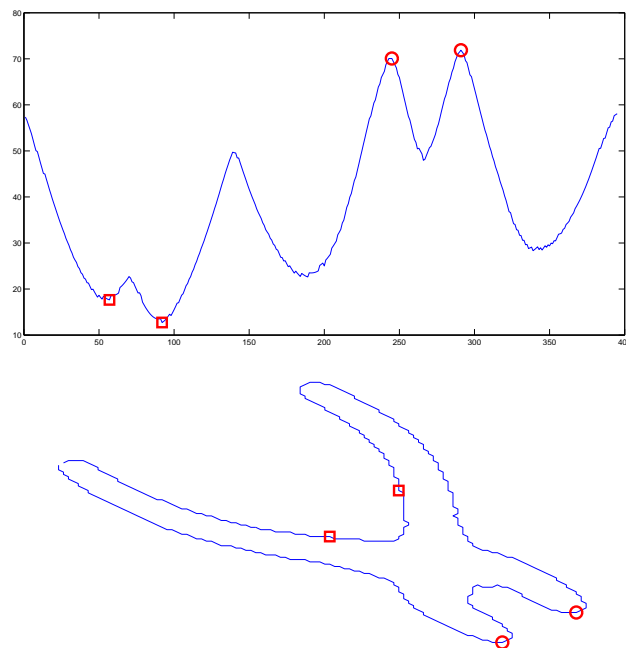


Figure 2.2: Curvature Signature for punch shape. Circles denote local maxima of curvature and squares denote local minima of curvature.

The **tangent angle** function is defined as the tangential direction of a contour at a certain point. It suffers of two main drawbacks: noise sensitivity and discontinuity.

The **contour curvature**, very important for human visual perception, is based

2.2. CONTOUR-BASED METHODS

on the concept of digital curvature:

$$k(n) = \frac{x(n)''y(n)' - x(n)'y(n)''}{(x(n)'^2 + y(n)'^2)^{\frac{3}{2}}} \quad (2.2.4)$$

In [2] the **Area function** is proposed: it computes the area of the triangle formed by two next boundary points and the centroid. In [3] an improved version, Perimeter Area Function (*PAF*), is proposed.

The **Triangle Area (TAR)** signature [4] is computed from the area of the triangles formed by the points on the shape boundary. When the contour is traversed in counter clock-wise direction, positive, negative and zero values of TAR mean convex, concave and straight-line points, respectively.

In [5] the **chord length** function is used for shape matching. Basically it is the shortest distance between two boundary points so that this joining segment is perpendicular to the tangent angle of the first point.

2.2.2 Correspondence-based shape matching

Correspondence-based shape matching works with a point-to-point matching to evaluate similarity measures. **Hausdorff distance** assigns to each point of one set the distance to its closest point in the other set and takes the maximum over all these values. It is used as common metric even if it is sensitive to noise and outliers and is not invariant to translation, scale and rotation. Considering this last fact, the matching may be computationally hard and the shapes need to be compared with different rotation, scale and position.

Another similarity measure based on point-to-point correspondence has been presented in [6] known as **shape context**. For each contour point a histogram map, representing the quantized length and orientation of the vector departing from that point to the others, is created. Such histograms are then put into log-polar space and joined to form the context of the shape (fig. 2.3). The matching between shapes is realized by matching the context maps. In order to reduce computational time rather than taking into account all the contour points a constant sampling selection (see section 6.2) is done.

2.2.3 Other descriptors

Chen [7] modified the **moment** definition proposed by Hu [8] using the shape boundary only:

$$\mu_{pq} = \sum_{(x,y) \in C} (x - \bar{x})^p (y - \bar{y})^q \quad (2.2.5)$$

where (\bar{x}, \bar{y}) are the mass centroid coordinates and C is the digital curve.

The **elastic matching** [9] uses a deformed template of the shape; the final process leads to minimize a function dependent from five parameters. The parameters (taking into account the bending energy, the overlapping of deformed template and the object image and the curvature function) are classified by a back-propagation

CHAPTER 2. SHAPE REPRESENTATION AND DESCRIPTION TECHNIQUES

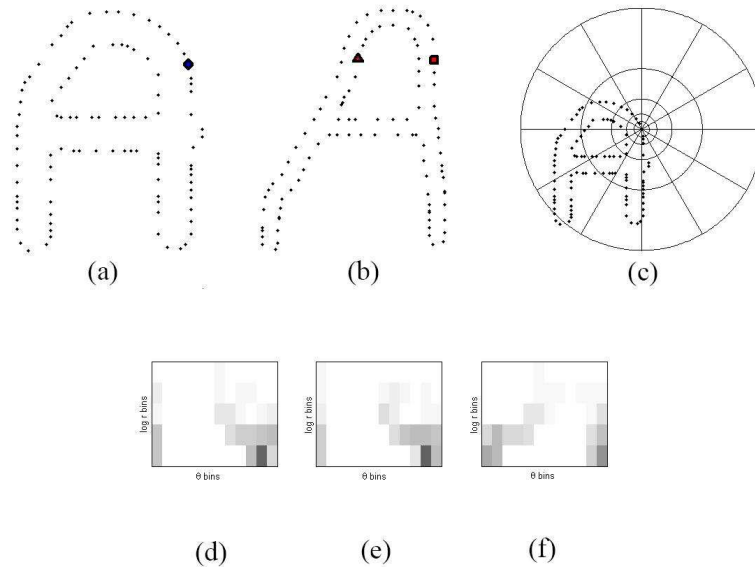


Figure 2.3: The shape contexts of two different versions of the letter “A” are shown. (a) and (b) are the sampled edge points of the two shapes. (c) is the diagram of the log-polar bins used to compute the shape context. (d) is the shape context for the circle, (e) is that for the diamond, and (f) is that for the triangle. As can be seen, since (d) and (e) are the shape contexts for two closely related points, they are quite similar, while the shape context in (f) is very different. [Source: Wikipedia]

neural network and for this reason the approach is not suited for online image retrieval.

Inspired by statistics, **stochastic methods** model a function obtained from the shape as described in section 2.1, interpreted as a stochastic process realization. The estimation resulting parameters make up the set of shape descriptors. Stochastic processes often model a signal derived from time-series; in this case this signal is replaced by the shape boundary analysis.

Carry on monodimensional functions, in **autoregressive AR models** a correlation between the current function value and the previous function values is established in order to predict the future function values.

The **scale space** representation is based on a simple rule: simplify the original curve more and more with a kind of low-pass gaussian filter. In this way small structures on the contour tend to disappear because the contour becomes smoother. In such way it is possible to highlight small details from relevant shape properties. In the large family of methods exploiting this descriptors a main approach is worth reminding: curvature scale space. Proposed by Mokhtarian and Mackworth [10] the **curvature scale space (CSS)** method extracts from the contour a scale space signature. The arc-length position of inflection points (x-axis) on contour on every scale (y-axis) forms a function presenting some peaks in correspondence to the mid

2.2. CONTOUR-BASED METHODS

points of the maximal convex arcs obtained during the curve evolution (fig. 2.4). This method has included in the MPEG-7 standard and has proven to be robust to noise, changes in scale and orientation of objects.

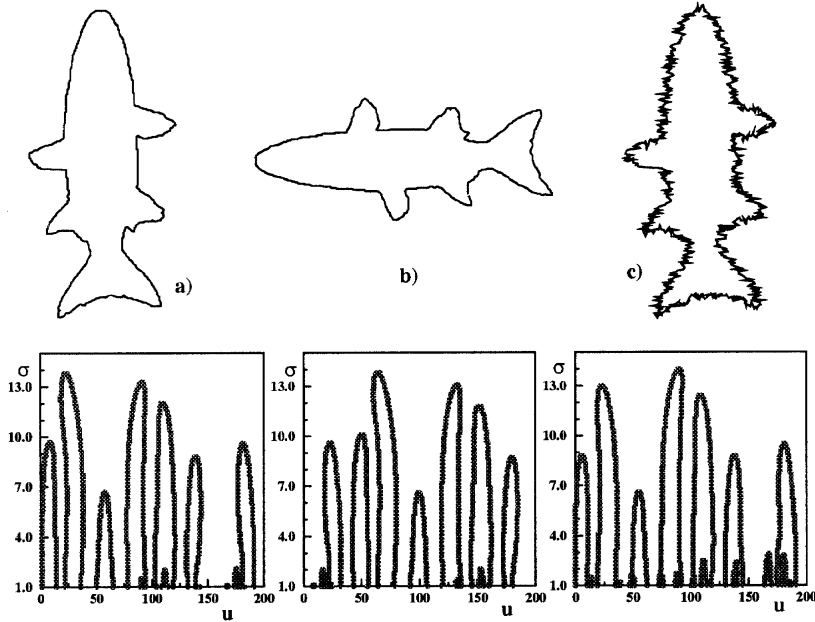


Figure 2.4: (a) Boundary and its CSS image. (b) Change in orientation causes a circular shift in CSS image. (c) Noise creates small contours in CSS image.

In a dual approach to shape description, the transform domain is another representation area. The main advantage of this representation is the low sensitivity to noise and boundary variations. The normalized coefficient of the Fourier transformation are **Fourier descriptors**. This transformation is typically applied to a "signal" from contour samples. In [11] it is shown the better behaviour of the centroid distance respect to other functions. Derived from the wavelet transform, the **wavelet descriptors** has been introduced in [12]. The digital curve is decomposed in components of different scales: the coarsest scale components bring global information approximation properties while the finer scale ones take the local detailed information.

2.2.4 Structural methods

The methods falling in this category are based on a shape breakage into boundary segments called *primitives*. In general such shape element parts are organized into a string form where each symbol corresponds to a single primitive.

Chain codes describes a shape by a set of line segments with a predefined set of possible orientations. Freeman [13] introduced such a concept. Common

CHAPTER 2. SHAPE REPRESENTATION AND DESCRIPTION TECHNIQUES

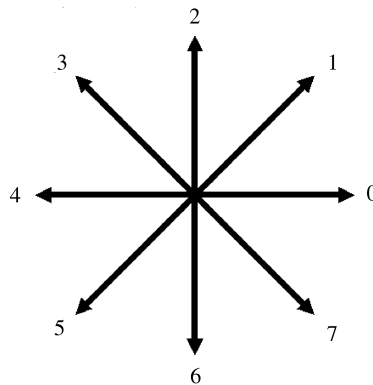


Figure 2.5: Freeman coding.

chain codes are 4-directional or 8-directional. The line segments of an 8-directional chain code can have unit-sized lengths or they can be defined in a square grid. The chain code is a vector of integer numbers where each number represents one of the possible directions. A differential chain code takes the differences between successive numbers in the chain code and take them modulo N (where N is the number of possible directions). A differential chain code which is normalized to its smallest integer number is called *shape number*. Chain codes histograms reflect the probabilities of different directions present in a contour. Used for object recognition, they are translation and scale invariant and partially rotation invariant. **Boundary decomposition** divides the shape contour into line segments. The primitives are, in this case, represented by vertices and the key features are internal angle, distance from the next vertex and (x, y) coordinates. In [14] a transformation invariant adaptation is done. In other works, for example in [15], the boundary is divided into multiple curves called *tokens* and a feature vector is created for each token. In [16] a binary shape is decomposed into entities in correspondence with their protrusions by using the shape skeleton (defined in sec. 2.4). Each entity is then modelled according to a set of perceptually salient attributes, i.e the variation of the curvature, the orientation of each entity compared to the orientation of the shape and the ratio of the entity length to extreme point distances (fig. 2.6). The similarity between shapes is often addressed by computing the Euclidean or the city block norm between the associated feature vectors. In the **syntactic analysis** the primitives are coded as the alphabet letters (this set is call *codebook*), then composed into words (*codewords*) to form phrases and sentences, in order to reproduce the composition of a natural scene. The matching between shapes is realized by evaluating the minimal number of edit operations to convert one string into another. Some methods use the **shape invariants**, i.e. properties which are invariant to a class of transformations [17].

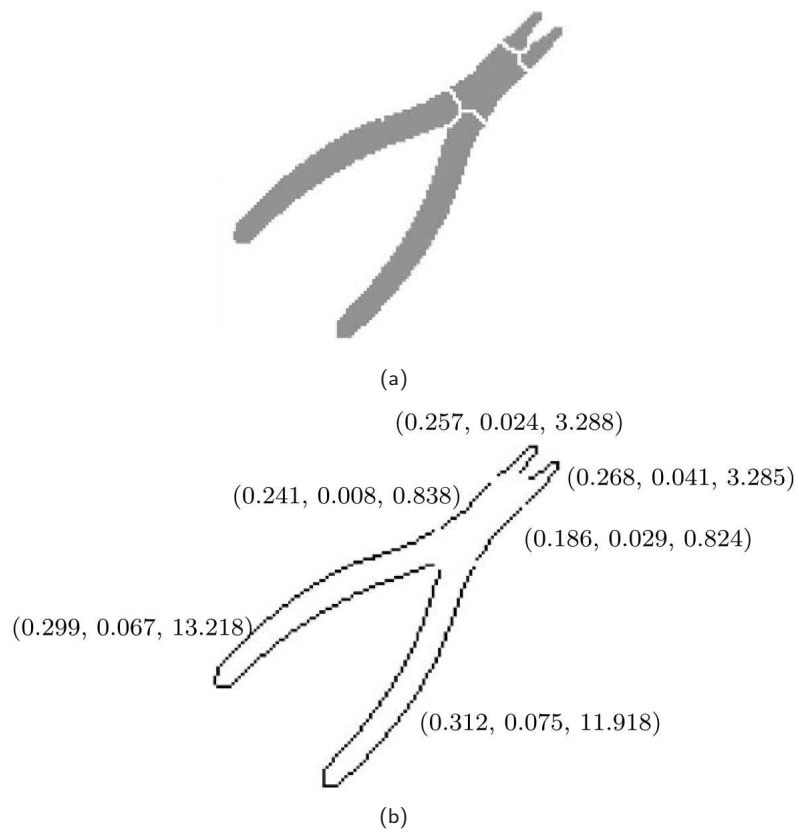


Figure 2.6: A pincher shape partition (a) and its entities with their associated descriptions (b).

2.3 Region based approaches

Region based methods use the whole pixels in a shape region to create shape descriptors, rather than use boundary information as in contour methods.

2.3.1 Global methods

Global methods consider the shape as a whole avoiding its division in sub-parts.

Geometric moments

The *central moment* of order (p, q) of a shape S is given by

$$\mu_{pq}(S) = \sum_{(x,y) \in S} (x - \bar{x})^p (y - \bar{y})^q \quad (2.3.6)$$

where (\bar{x}, \bar{y}) is the center of gravity. We can obtain the normalized central moments, denoted by η_{pq} , by normalizing with μ_{00} moment.

From the second and third-order normalized central moments, a set of seven invariant moments, which are invariant to translation, scale change and rotation, has been derived by Hu [18]. Hu has proved the invariance properties of the seven moments for the case of continuous function.

Flusser and Suk [19] derived **affine moment invariants** which are invariant under general 2-D affine transformations.

Taubin and Cooper [20] defined **algebraic moment invariants** by introducing the concept of covariance matrices. Eight invariants are calculated by eigenvalues and scalar values of this covariance matrices.

Orthogonal moments

In [21] it has been demonstrated that the orthogonal **Zernike moments** offer a set of moments which are highly uncorrelated with little information redundancy. For a discrete image of size $M \times N$ represented by the function $f(x, y)$, the Zernike moment of order (m, n) , Z_{mn} , can be calculated with:

$$Z_{mn} = \frac{m+1}{\pi} \frac{1}{(M-1)} \frac{1}{(N-1)} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} V_{mn}^*(r, \theta) f(x, y) \quad (2.3.7)$$

where $r = (x^2 + y^2)^{1/2} / \sqrt{2MN}$ and $\theta = \tan^{-1}(y/x)$, in order to map the image into the unit disc for Zernike moments calculation. $V_{mn}(r, \theta)$ is the Zernike polynomial (where $|n| \leq m$ and $m+n$ is even). It can then be easily shown that:

$$Z_{mn}^* = Z_{m,-n} \quad (2.3.8)$$

The number of Zernike moments for any order, m , is given by $m+1$, while the number of moments up to order m is $(m/2+1)(m+1)$. In practice, because of the relationship between Z_{mn} and $Z_{m,-n}$ given above, only the moments with $n \geq 0$ need to be known.



Figure 2.7: (a) An original shape in polar space. (b) polar-raster sampled image plotted in Cartesian space.

The kernel of **Legendre moments** is the products of Legendre polynomials defined along rectangular image coordinate axes inside a unit circle. The discrete version for a digital image $f(x, y)$ of size $M \times N$ of the Legendre moments can be written as :

$$\lambda_{pq} = \frac{(2p+1)(2q+1)}{(M-1)(N-1)} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} p_p(x)p_q(y)f(x, y) \quad (2.3.9)$$

where $(p+q)$ is the order, $p, q = 0, 1, 2, 3, \dots, \infty$. The Legendre polynomials, $P_p(x)$ are a complete orthogonal basis set on the interval $[-1,1]$ and are not invariant to rotation.

Generic Fourier Descriptors

Generic Fourier Descriptors (*GFDs*) helped in overcoming Zernike's computational complexity and features domain inconsistency due to the radial components (spatial domain) and circular components (spectral domain). Proposed by [22], GFDs work in this way. At first a polar transformation of an input image $f(x, y)$ is done, obtaining a polar image $f(\rho, \theta)$ by

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2}, \quad \theta = \tan^{-1}(y/x) \quad (2.3.10)$$

where (x_c, y_c) are the coordinates of centroid. Then a transformation of polar raster sampled image in Cartesian space is done (fig. 2.7) . At this point a 2-D Fourier Transform of this polar raster sampled image $f(\rho, \theta)$ is calculated:

$$PF_2(\rho, \phi) = \sum_r \sum_i f(r, \theta_i) \exp \left[j2\pi \left(\frac{r}{R}\rho + \frac{2\pi i}{T}\phi \right) \right] \quad (2.3.11)$$

where $(0 \leq l < R)$ and $\theta_i = i(2\pi/T)$ $(0 \leq i < T), 0 \leq \rho < R, 0 \leq \phi < T$. R and T are the resolution of radial frequency and angular frequency respectively. The normalized Fourier coefficients are the GFDs.

Other global methods

In **grid methods** [23] the shape is represented as a binary feature vector obtained by applying a grid to a binary object, assigning 1 to the cells covered by the shape and

CHAPTER 2. SHAPE REPRESENTATION AND DESCRIPTION TECHNIQUES

0 otherwise, and scanning the grid. The similarity between shapes is then evaluated by comparing such vectors with some metrics, for example the Hamming distance. In [24] Goshtasby proposed the use of a **shape matrix** derived from circular raster sampling technique.

2.4 Structural methods

In structural methods the shape is broken into parts, called *primitives*, based on a certain algorithm. The relationship between such segments are modeled by graphs, trees and strings.

Convex Hull

A region is defined as convex if a line segment between any two points in the region is totally inside the region. The convex hull of a region is the smallest possible convex region which is a superset of the region. A convex deficiency is defined as the difference between a region and its convex hull. Through convex deficiencies it is possible to describe a shape. Each concavity can be characterized by multiple simple properties, for example its area or its maximum curvature. All properties of a concavity and the concavity itself may be represented in a string; by grouping such features a shape descriptor is created. Analyzing the concavities recursively, a region concavity tree can be represented (fig. 2.8): for every concavity the convex hull and the convex deficiencies are computed until the found convex deficiencies are convex.

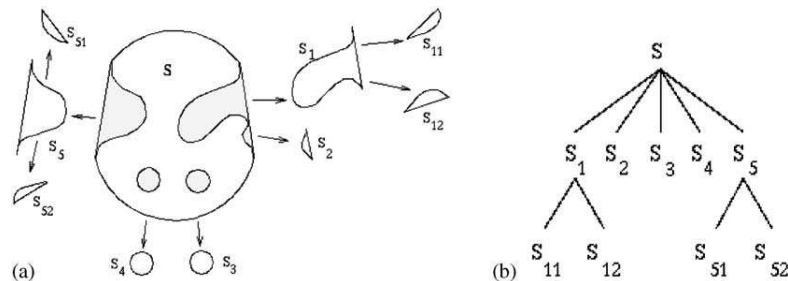


Figure 2.8: (a) Convex hull and its concavities. (b) Concavity tree representation of convex hull.

Medial axis

The region skeleton or medial axis of a shape is defined as the locus of centers of maximal disks that can be inscribed within the shape (see fig. 2.9). This representation of a shape resumes its geometrical and topological features. The original shape can be reconstructed from its skeleton by taking the union of the maximal disks defined by the medial axis. The skeleton can be computed in different ways:

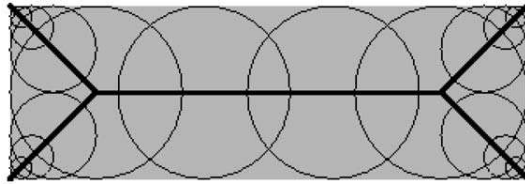


Figure 2.9: Construction of the medial axis of a rectangular shape.

thinning, that removes iteratively boundary pixels, wave propagation, morphological operators and so on. Every point of the skeleton contains the distance to the nearest shape boundary [17]. The skeleton of a shape is very sensitive to noise and variations and shape matching can be done by matching the graphs of individual parts of a skeleton.

In [25] the skeleton is used as basis to attributed skeletal graphs (ASG) to organize in a structured way information about object shape and topology embedded in its thinned representation.

Shock graphs

Shock graphs [26] are a shape representation based on the skeleton of a region. The skeleton is divided into different types of shocks depending on the local variation of the radius function. 1-shocks correspond to protrusions where the radius function varies monotonically. Local minima of the radius function appear at necks or 2-shocks. A constant radius function is defined as a 3-shock, e.g. along a bend with parallel sides. A 4-shock appears at a single point which is also called seed. Shock graphs are created automatically by a set of rewriting rules which control the birth, protrusion, union and death of vertices in the graph. The similarity between two shock graphs can be measured with a largest subgraph isomorphism approach. It is also possible to convert a shock graph into a shock tree which allows significance-based matching where the central shocks are considered first.

3

On dominant points

In the previous chapter we have discussed about methods to extract shape features with some representation techniques.

We think that, rather than using a region-based method, it is better to work on the contour for different reasons:

- due to contour's semantically rich nature, recognition techniques based on contour analysis have broad applicability since they are one of the most commonly used shape descriptors;
- contours are easy to obtain and can be used for shape representation regardless of shape convexity or the lack thereof;
- keeping in mind that contours may be considered as high order curves, algorithms for contour-based recognition are potentially extensible to more generic shape classes including cursive words and hand-drawn sketches;
- contours are easy to manage respect to the whole image: in fact we have to deal with less pixels, i.e. points, than considering internal regions also; as consequence the computational cost in the processing phase is lower;
- moreover a data reduction means, especially in context as computer networks where we need to take into account band constraints, a faster information transmission;
- though we work with few points instead of using all the pixels of shapes, the information is concentrated in the boundary variations.

As result of such considerations, in the continuation of this thesis we choose to work with contours.

After shape representation, another crucial step in shape analysis consists in extracting meaningful features from digital curves. In this chapter we give an overview of the problem of polygonal approximation of a digital closed curve.

The problem can be modelled by the extraction of significant points, also know as *dominant points*, defined in section 3.1. These points, important for visual perception and containing most of the shape curvature information, reduce the computational complexity by simplifying the contour.

CHAPTER 3. ON DOMINANT POINTS

In section 3.2 we provide a panoramic about dominant point extraction techniques and their classification, distinguishing corner detection and polygonal approximation approaches. We emphasize the methods tested or inspired to in the following of the thesis.

In section 3.3 we describe the main indicators used commonly to assess the efficiency of a polygonal approximation. In general the integral square error gives a measure of how the approximating reconstruction differs from the original contour. Such a value is often connected to the number of dominant points discovered.

3.1 Definition of dominant point

In shape analysis an important step consists in extracting meaningful features from digital curves. The term *dominant points (DPs)*, also known as significant points, relevant points, points of interest or corner points, is assigned to points having a sufficiently high curvature on the boundary of a planar object; their detection is a very important aspect in contour methods since information on the shape of a curve is concentrated at dominant points. These points, having curvature extreme, can suitably describe the curve for both visual perception and recognition [27],[28].

Attneave [29] was the first who pointed out that information on a curve is concentrated at the dominant points, by showing an image representing a stylized cat composed by different segments (fig. 3.1).

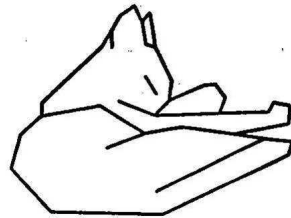


Figure 3.1: Attneave's cat.

Dominant points can be applied to produce a simplified representation of a shape contour for further processing. This representation simplifies the analysis of the images by providing a set of feature points containing almost complete information of a contour. One of the main advantages of a good representation of 2D shapes through dominant points is the high data reduction and its immediate efficiency in feature extraction and shape matching algorithms.

The data reduction leads to a higher speed during execution; furthermore a lower memory usage is required, with immediate benefits for both storage and transmission.

Another important aspect concerns a better quality of data, because, although the reduction of points could suggest a loss of information, one must consider the choice of points of interest, which contain the main features of the curve, discarding the rest. Therefore, an approximated curve will be more suitable for classification and extraction of attributes than the original curve, and less affected by noise.

It is well known also that these points play a dominant role in shape perception by humans.

3.2 Algorithms for dominant point extraction

There are many approaches developed for detecting dominant points. Most of these algorithms require one or more parameters that specify (directly or indirectly) contour regions in order to measure the local properties at each point of the curve. A raw classification of dominant point extraction technique is described in [30] and reported in fig. 3.2.

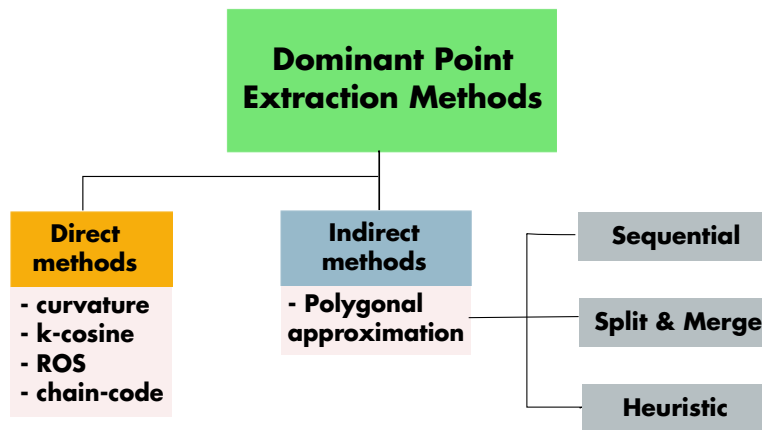


Figure 3.2: Dominant point extraction methods classification.

Basically the methods fall into two main categories: corner detection approaches (*direct methods*) and polygonal approximation approaches (*indirect methods*).

3.2.1 Direct Methods

Most of the methods belonging to this category are based on the concept of digital curvature estimated by:

$$k = \frac{x''y' - x'y''}{(x'^2 + y'^2)^{\frac{3}{2}}} \tag{3.2.1}$$

In [31] the selection of significant curvature maxima and minima is done by falling back upon the so called *k-cosine*, i.e. the angle between vectors representing the *k*-extent arcs sideways a contour point.

In the pioneering paper [32] a “one shot” method that doesn’t require any input parameter is described by Teh and Chin. It uses a procedure that computes a region of support (*ROS*) of size *k* by comparing the length of the chord joining the *k* position far symmetric respect to the selected point and its distance to such chord. After that a non-maxima suppression for points having a *k*-cosine or a *k*-curvature measure (i.e. the difference in mean angular direction of *k*-vectors) under a prefixed threshold is performed.

In [33] the Teh-Chin method is modified: the current region of support is dynamically defined by the previous one and the excess of dummy dominant points is

3.2. ALGORITHMS FOR DOMINANT POINT EXTRACTION

faced with a k -cosine threshold.

Wang et al. [34] proposed a simple method that uses the directions of the forward and backward vectors to find the bending value as the curvature.

In [35] a similar approach is introduced: initially the Freeman chain code [13] is exploited to suppress collinear points. Afterwards the region of support is computed by evaluating the adaptive bending value. The dominant points are extracted by keeping those points having local maximum smoothing bending value also known as *break points*.

In [36] the region of support is evaluated by the sum of the squared perpendicular distances from all the points between the ends of a k -bounded chord subtended to the selected point.

Carmona et al. [27] proposed a new method for corner detection. A normalized measurement is used to compute the estimated curvature and to detect dominant points, and an optimization procedure is proposed to eliminate collinear points.

3.2.2 Polygonal approximation

The phrases dominant points detection and polygonal approximation are used alternatively by some researchers in the area of pattern recognition.

For polygonal approximation approaches, sequential, iterative and optimal algorithms are commonly used. In fig. 3.3 an example of polygonal approximation is shown.

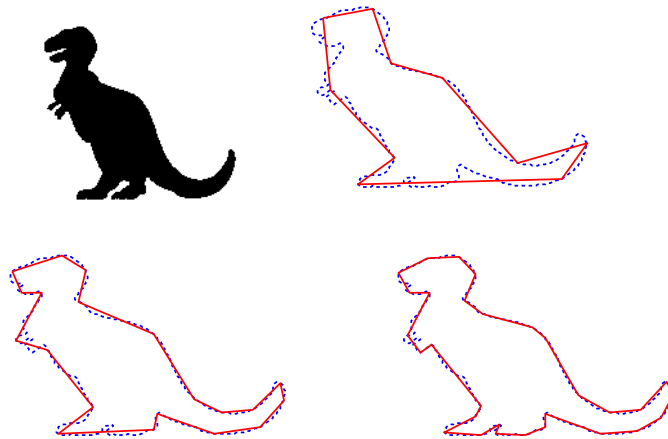


Figure 3.3: Example of polygonal approximation (10, 20 and 30 points) on dinosaur shape (550 contour points)

CHAPTER 3. ON DOMINANT POINTS

Sequential algorithms

Sequential approaches are simple and fast, but the results depend on the location of the point where they start the scan-along process and they can miss important features.

The sequential algorithms obtain the solution of the problem by examining the points that make up the curve in succession. They are characterized by simplicity and high speed computation, but the quality of results depends on the points selected for the initial step of the process. In this category we find the approach proposed by Sklansky and Gonzalez [37] where an algorithm that uses a scan-along procedure that, starting from a point, performs a sequential search to locate the longest line segment.

Ray and Ray [38] have proposed a method which determines the longest segment that has the lowest error.

In [39] a method which determines the longest possible line segments with the minimum possible error is described.

Kurozumi and Davis [40] presented a method in which the approximating segments are derived by minimizing the maximum distance between a given set of points and corresponding segments.

Iterative (or split and merge) algorithms

The iterative approaches split and merge curves iteratively until they meet the pre-set allowances. The solution is reached through iterative steps in which the curves are subject to division (split) and unions (merge). The results get may be far from the optimum if the initial segmentation appears to be unsuitable. If a poor initial segmentation is used, the approximation results of the split-and-merge approaches maybe far from the optimal one. The iterative approaches, in fact, suffer the sensitivity to the selection of the starting points for partitioning curves. For split-and-merge approaches, Ramer [41] estimated the distances from the points to the line segments of two ending points; starting from an initial segmentation, the segments are divided in points that are more distant from the segments themselves. The segment is partitioned at the point with the maximum distance until the maximum distance is not greater than a predefined threshold.

Ansari and Delp [42] proposed another technique which first uses Gaussian smoothing to smooth the boundary and then takes the maximal curvature points as break points to which the split-and-merge process is applied.

Ray and Ray [43] proposed an orientation-invariant and scale-invariant method by introducing the use of ranks of points and the normalized distances.

The method of Sarfraz et al. [44] is a recursive procedure which determines the longest segment that meets a certain threshold. The midpoints are marked as dominant and segments to which they are associated are divided into sub-segments. Everything is then repeated until a stop criterion is reached.

Held et al. [45] propose a technique in which the change of slope is used to divide the segments, which are then assembled by the criteria of perceptual significance.

Wu and Wang [46] proposed the curvature-based polygonal approximation for

3.2. ALGORITHMS FOR DOMINANT POINT EXTRACTION

dominant point detection. It combines the corner detection and polygonal approximation methods and it can detect the dominant points effectively.

Meta-heuristic algorithms

The optimal approaches tend to find the optimal polygonal approximation based on specified criteria and error bound constraints. The idea behind is to approximate a given curve by an optimal polygon with the minimal number of line segments such that the approximation error (described in section 3.3) between the original curve and the corresponding line segments is no more than a pre-specified tolerance.

Local optimal methods are very fast but their results may be very far from the optimal one. However, an exhaustive search for the vertices of the optimal polygon from the given set of data points results in an exponential complexity. Dunham [47] and Sato [48] used dynamic programming to find the optimal approximating polygon. But, when the starting point is not specified, the method requires a worst-case complexity of $O(n^4)$ where n is the number of data points.

Masood [49] presented an algorithm for polygonal approximation based on dominant point deletion that combines local and global optimization. Dominant points are eliminated depending upon the error associated with each of them. The algorithm selects an initial set of DPs and starts eliminating them one by one depending upon the error associated with each DP. A local optimization of few neighbouring points is performed after each deletion.

To solve complex combinatorial optimization problems metaheuristic techniques have been introduced.

These algorithms do not guarantee finding the optimal solution, but are able to provide a good solution (close to the optimum) in a reasonable time and obtain better results than most of the local optimal methods. Fred Glover [50] first coined the term metaheuristic as a strategy that guides another heuristic to search beyond the local optimality such that the search will not get trapped in local optima.

Meta-heuristic algorithms perform a search for "wizard" within the whole search space of solutions, trying to maintain a balance between exploration and exploitation. Exploration and exploitation, respectively, indicate the search for full (or at least a good part) of the search space and exploitation of solutions found from time to time to direct the search. The exploration strategy searches for new regions, and once it finds a good region the exploitation heuristic further intensifies the search for this area. However, the balance between exploration and exploitation is not easily accessible and this could lead to high execution time or solutions far from the optimum.

For the polygonal approximation there are four commonly used meta-heuristics: tabu-search [51], ant colony [52], [53], particle swarm optimization [54] and genetic algorithms [55], [56].

Yin [51] focused in the computation efforts and proposed the tabu search technique to reduce the computational cost and memory in the polygonal approximation.

Hong [57] proposed a dynamic programming approach to improve the fitting quality of polygonal approximation by combining the dominant point detection and the dynamic programming.

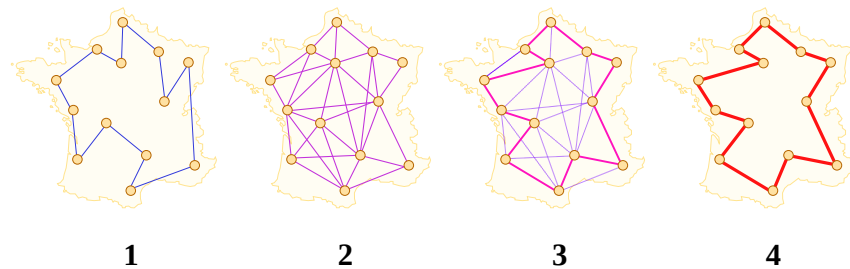


Figure 3.4: Example of ant colony optimization applied to the travelling salesman problem. 1. An example of solution 2. Each possible connection between cities depending on the visibility 3. Definition of the shortest path: the more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen 4. Having completed its journey, the ant deposits more pheromones on all edges it traversed, if the journey is short. [Source: Wikipedia]

Generally, the quality of the approximation result depends upon the initial condition where the heuristics take place and the metric used to measure the curvature.

Yin [58] focuses on the more recently proposed metaheuristics and give their comparative evaluations.

Most of the central metaheuristic methods have been applied to the polygonal approximation problems and obtained promising results.

Ant colony

An ant colony optimization algorithm is a probabilistic technique for problems that can be related to the location of a good path within a graph. Ant Colony Optimization firstly proposed by Dorigo in [59] is based on a computational paradigm inspired by the way real ant colonies function.

It tends to reproduce the behaviour and the communication system of ants when from their nest they are directed to an area where there is food. In fact, in its walking each ant issues a substance called pheromone, tracing the path that actually takes in order to allow the other ants to locate it and to choose a route based on the amount of pheromone they warn.

This behaviour is important not only to locate a path that leads to food, but also to determine a short path to reach the destination. In fact, the pheromone evaporates after a certain period of time, so the longer routes (which require more time to be flown) will tend to have a few tracks and then have little chance of being chosen. Conversely, in the shortest paths, which allow to reach your destination more quickly, the concentration of pheromone will be higher and consequently have more chance of being chosen by ants.

The ACO algorithms have been applied to many different discrete optimization problems, like the travelling salesman problem and the quadratic assignment problem. Recent applications cover problems like vehicle routing, sequential ordering, graph colouring, routing in communications networks, and so on.

3.2. ALGORITHMS FOR DOMINANT POINT EXTRACTION

In the case of applications for polygonal approximations, the steps of the algorithm can be summarized in four main points:

1. Construction of the graph of the problem and random placement of the ants within nodes.
2. Movement of ants led by a probabilistic rule transaction. On the way ants release the pheromone.
3. Updating of traces of pheromone at each iteration cycle.
4. Termination of the algorithm if the stop criterion is reached (i.e. maximum number of cycles or elapsed time).

A deep discussion on ACO polygonal approximation algorithm is reported in section 4.2.

Tabu search

The tabu search is an evolution of the method of descent that implements some tricks to avoid the convergence towards a local optimum. In fact, while the methods of descent only allow the choice of directions which could achieve an improvement in the partial solution, the tabu search allows to make "worsening" moves that lead to partial solutions qualitatively less than the previous one.

In addition, to avoid the risk of falling in local optimum, a set of forbidden moves (*tabu*), which prevents the algorithm to retrace their steps so as not to undo the progress achieved, is made up. However, there are variations and criteria for aspiration, allowing to choose a move if this tabu leads to a "good solution". The concept of a good solution is not absolute and may depend on the type of problem and the implementation choices.

In general, the pattern followed by the tabu search can be summarized as follows. An initial solution, called configuration, is selected and evaluated. Therefore, a certain set of movements candidates, called neighbourhood of the current solution, is done. If the best move is not a tabu, or if it is a tabu but falls within the criteria of aspiration, then it is chosen and the solution to which it leads is set as the current solution. Everything is repeated until a criterion of sleep is reached. The solution is given by the best solution achieved during processing. In the polygonal approximation field, we can cite the study by Zhang and Guo [60] which has led to a method summarized in four steps.

Let G_c , G_b , G_t be, respectively, the current solution and better candidates, and J_c , J_b and J_t the values of their objective functions.

1. Randomly generate a solution G_c and set $G_b = G_c$ e $J_b = J_c$. After that, select the tabu list L , the threshold probability P and the maximum number of iterations N . The k value, representing the current iteration, is initialized to 1;
2. Starting from G_c , is generated G_t (calculating, in addition, the value of its objective function) around the neighbourhood $NB(G_c)$ of G_c . After that,

CHAPTER 3. ON DOMINANT POINTS

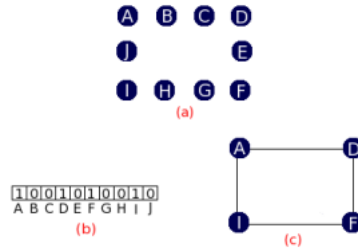


Figure 3.5: Chromosome example: a) Contour b) Chromosome coding c) Approximating polygon corresponding to (b).

we pass to the addition, removal and exchange of G_c with a probability P obtained from G_t ;

3. We move toward the best temporary solution G_t^b in $\text{NB}(G_c)$ which is not tabu, or it is tabu with $J(G_t^b) < J_b$, and set $G_c = G_t^b$ and $J_c = J(G_t^b)$. If all solutions of $\text{NB}(G_c)$ are tabu, return to step 2;
4. Insert G_c in the bottom of the tabu list. If the tabu list is full, remove the first element of the list. If $J_c < J_b$, set $G_b = G_c$ and $J_b = J_c$. If $k = N$, the algorithm terminates, otherwise set to $k = k + 1$ and return to 2.

Genetic Algorithms

Genetic algorithms are optimization techniques, proposed in 1975 by J.H. Holland [61], inspired by the principle of natural selection by Charles Darwin, in which a population evolves through several generations, so that individuals with better features have more opportunities to pass their genetic information to the next populations. Holland and his colleagues developed the genetic algorithms (GAs), with the aim of studying the phenomenon of evolution in nature and create artificial systems with properties similar to those of natural systems.

The basic idea is to encode the solution of a problem in the form of a string of values, using a population of constant size, where each individual, representing a possible solution is encoded by a string differently. The evolution of the population, according to the mechanisms of biological evolution, leads to obtain a solution close to the optimum. Leaving unchanged the terminology used in biology, the string is called chromosome, while its values are called genes.

In the problem of polygonal approximation, the chromosome coincides with an approximating polygon and is represented by a binary string. Each bit, called gene, represents a curve point where dominant points have 1-value (fig. 3.5).

Another concept common to the biological theory and the theory of GAs is the ability or fitness, or a function associated with vitality, i.e. the probability that the individual lives long enough to reproduce, or fertility seen as many descendants that an individual is able to generate. In our context the fitness function takes into

3.2. ALGORITHMS FOR DOMINANT POINT EXTRACTION

account the quadratic error in approximating the original contour with the found solution.

The main operations of the GAs performed on the population of chromosomes in order to create new generations are selection, crossover and mutation. These operations have the task of balancing the exploration of the search space with the use of information obtained from the partial results. In the selection phase the "best", i.e. with the highest fitness, individuals are chosen. Then, in the crossover step, some of the selected chromosomes are "broken" and recombined, assuming that the fitness function may be improved. During the mutation some parts of genes with low fitness are randomly changed in order to enhance their fitness value; it is used to maintain genetic diversity from one generation of a population of algorithm chromosomes to the next one.

The creation of an initial population, in which the chromosomes that compose it are subject to an assessment in order to determine the fitness, is followed by a sequential application of the operators of selection, crossover and mutation. This phase leads to the creation of a new population, to which apply the same procedure in the event the stop criterion is not reached, otherwise the algorithm will terminate. The result will be represented by the best chromosome found in various populations.

In section 4.3 we provide a close examination of the problem.

3.3 Polygonal approximation evaluation measures

The performance of most dominant point-detection methods depends on the accuracy of the curvature evaluation. The evaluation of each dominant point extraction technique efficiency relies on the difference between the original curve and the approximating polygon.

Consider a curve $C = \{p_1, p_2, \dots, p_i, \dots, p_j, \dots, p_n\}$ consisting of n ordered points and identify with $\widehat{P_i P_j}$ and $\overline{p_i p_j}$, respectively, the arc and the chord of extremes p_i and p_j . The error obtained by approximating the arc $\widehat{P_x P_y}$ with the segment $\overline{p_x p_y}$ may be defined as

$$L_{x,y} = \sum_{p_i \in \widehat{P_x P_y}} d^2(p_i, \overline{p_x p_y}) \quad (3.3.2)$$

where $d(p_i, \overline{p_x p_y})$ represents the distance of point p_i from the chord $\overline{p_x p_y}$, which can be understood as the length of the perpendicular to the segment led from p_i to r , where r represents the line through the points p_x and p_y .

Letting $C' = \{v_1, v_2, \dots, v_i, \dots, v_j, \dots, v_m\}$ with $m \leq n$, the list of vertices forming the polygonal approximation of C , the quantity

$$E_2(C, C') = L_{1,2} + L_{2,3} + \dots + L_{m-1,m} + L_{m,1} \quad (3.3.3)$$

is defined as the sum of squared error (integral square error or *ISE*). In figure 3.6 the red area denotes the committed error in contour approximation.

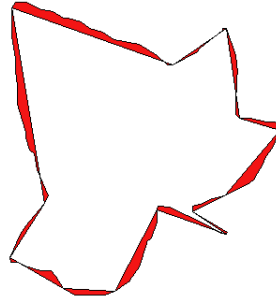


Figure 3.6: E_2 error: the red area indicates the difference between the polygonal approximation and the original contour

In the next chapters of this thesis we refer to this measure every time we will compare dominant point extraction techniques.

The E_2 error depends strongly from the dominant points placement. It is clear, even if not obvious, that, in general, the larger the dominant point number the smaller the error E_2 produced by the approximation. The “quality” of the approximation needs to be referred to the number of dominant points also and for this reason it is typically introduced the following ratio (also known as *figure of merit*)

3.3. POLYGONAL APPROXIMATION EVALUATION MEASURES

(FOM) [62]):

$$FOM = \frac{CR}{E_2} \quad (3.3.4)$$

where E_2 is the same defined in 3.3.3 while CR is the compression ratio defined as:

$$CR = \frac{m}{n} \quad (3.3.5)$$

where m is the dominant point number and n is the contour points number, according to the previous definitions.

In other works [63, 64] the approximation is connected with the error incurred by the optimal algorithm E_{opt} . In particular Rosin [63] defined

$$Fidelity = \frac{E_{opt}}{E_2} \quad (3.3.6)$$

and

$$Efficiency = \frac{M_{opt}}{M_2} \quad (3.3.7)$$

where M_2 is the number of lines in the approximating polygon produced by the suboptimal algorithm and M_{opt} is the number of lines that the optimal algorithm requires to produce E_2 . In [65] the approximation algorithms are tested under variations in their scale parameters introducing a monotonicity index. In [66] and in section 5.2 we introduce some measures to evaluate the affine transformation and noise invariance of studied dominant point approaches.

4

Dominant points detection: proposed approaches

This chapter describes the three approaches proposed for the detection of dominant points. A deterministic method based on an iterative algorithm and two heuristic methods are examined.

The first, renamed Dominant Point Iterative Localization (*DPIL*) looks for dominant points, dividing the contour into “segments” until a certain stop criterion is reached.

The second one, Ant Colony Optimization Polygonal Approximation (*ACOPA*) borrows the Ant Colony Optimization technique, based on the typical behaviour of ants in their search for food; even in this case we obtain the solution of the problem after a predetermined number of steps or after satisfying a condition about E_2 error tolerance.

The third algorithm, Genetic Algorithms Polygonal Approximation (*GAPA*), incorporates the theory of Genetic Algorithms: in practice the problem is traced to the representation of the contour points as genes of a chromosome. Through a series of steps we proceed to a combination and selection of “stronger” individuals among which the suboptimal solution of the problem is found.

In the case of DPIL we also made a final improvement by the suppression of points that are near enough. In other cases, an additional refinement step of the solution to improve it according to the minimum E_2 error criterion, described in section 3.3, is done.

In chapter 5 a comparison between the three described algorithm will be presented.

4.1 Dominant Point Iterative Localization (DPIL)

The method is based on a iterative approach. The first step is to assign an initial set of points and after detect other possible dominant points, i.e. relevant points, applying an iterative selection based on a particular distance criterion.

The results we obtain are then refined by suppressing some overmuch points in critical regions of the shape contour.

4.1.1 Initial point setup

In order to reach a complete set of dominant points of a given a contour, we need of a initial group of starting points. The four vertex theorem [67] states that a simple closed curve in the plane, other than a circle, must have at least four “vertices”, that is, at least four points where the curvature has a local maximum or a local minimum. By the result of this theorem we choose to locate four initial points by searching for local maxima and minima in the signature of the shape.

At first we calculate the centroid of the contour defined as:

$$x_c = \frac{1}{n} \sum x_i \quad , \quad y_c = \frac{1}{n} \sum y_i \quad (4.1.1)$$

where x_i, y_i are the coordinates of the n contour points.

Then we calculate the signature of the contour obtained as :

$$s_i = \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2}, i = 1, 2, \dots, n \quad (4.1.2)$$

Given the signature profile of the contour we calculate local maxima and local minima of this function. The values are computed by the following criterion : a point is considered a maximum peak if it has the maximal value, and was preceded (to the left) by a value lower by Δ . In our case we use $\Delta = 0.5$. In such way, searching the minimal values, we find minimum peaks. We use the first and second maximum and minimum peaks so we have four starting points for our method.

4.1.2 Dominant point detection

After the initial points setup, the method builds a set of dominant points in the following way:

1. Let $D = \{d_i, \quad i = 1, \dots, m\}$ the initial dominant points set. In the first iteration $m = 4$ according to the starting points setup. For each pair of points $d_i d_{i+1}$, $i = 1, 2, \dots, m$ where d_{i+1} is a neighbour of d_i (modulus m) it is possible to calculate the sum of distance from each contour point p_j ($j = 1, \dots, n$) between d_i and d_{i+1} and the line connecting the two points:

$$dist_j = \frac{|(x_{i+1} - x_i)(y_i - y_j) - (x_i - x_j)(y_{i+1} - y_i)|}{\sqrt{((x_{i+1} - x_i))^2 + ((y_{i+1} - y_i))^2}} \quad (4.1.3)$$

4.1. DOMINANT POINT ITERATIVE LOCALIZATION (DPIL)

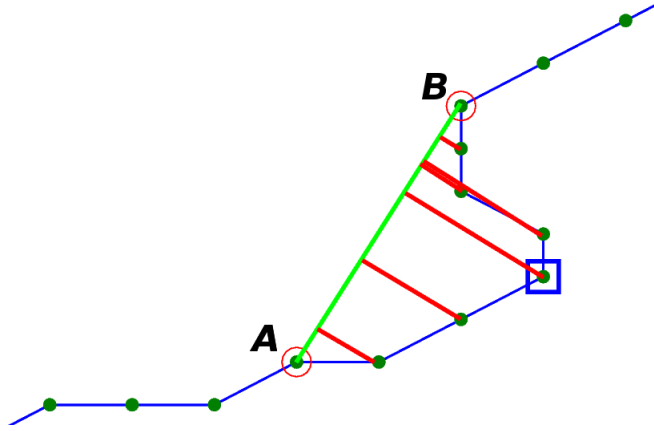


Figure 4.1: A step of dominant point detection. A and B are the current dominant point. The square-enclosed point is the new dominant point.

where (x_{i+1}, y_{i+1}) , (x_i, y_i) and (x_j, y_j) are coordinates of d_{i+1} , d_i and p_j respectively. A global distance is calculated as :

$$T = \sum_j dist_j \quad , \quad j = 1, \dots, n \quad (4.1.4)$$

2. In order to find dominant points the value T of the previous step is compared to a threshold s defined as $s = p * l$, where p is an input parameter of the algorithm and l is the number of points of the line that connects d_i and d_{i+1} . If $T > s$ we add an intermediate point between d_i and d_{i+1} looking for the point with maximum distance to the chord. The current set of points and the new dominant points added are the set of points for the next iteration.
3. The algorithm stops when, after two following iterations, the number of the current set of points is equal to the number of the previous set, i.e. each distance T is less or equal to the corresponding threshold s .

In fig. 4.1 the process of dominant points detection is depicted. We can suppose that A and B are two dominant points at a certain step. The green dots are the contour points. The green line represent the \overline{AB} segment joining A and B . The red lines are the distance between the segment \overline{AB} and each contour point between A and B . In this case, assuming that the sum of the length of red line is more than the method input threshold, we add an intermediate point looking for the point with maximum distance from the chord (represented by the point enclosed by the square), updating the set of dominant points.

4.1.3 Refinement

Given a dominant points set D we adopt a refinement technique to suppress dominant points which are near enough between them. Let $l_{i,j}$ the distance between

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

d_i and d_j and τ a given threshold, if $l_{i,j} < \tau$ then we substitute d_i and d_{i+1} with their medium dominant point :

$$\overline{d_k} \equiv \frac{d_{i+1} + d_i}{2} \quad (4.1.5)$$

For example if $i = 3$ and $i + 1 = 6$ are the index of dominant points meeting the refinement condition we can replace such dominant points with the point having index $k = 5$ (because we round the value 4.5 to 5). A reasonable value of $\tau = 0.007$ is set as default.

4.1.4 Experimental results

In order to test the effectiveness of our method we perform some experiments, both on typical shapes (introduced by [31] and [32] and commonly used in many studies) and on biological and common tools shapes.

In particular we test four contour commonly used synthetic curves : chromosome, infinity, leaf and semicircles curves. Their contour and the set of dominant points, obtained by each method and highlighted with a small circle, are shown in figs.4.2-4.5. Our method is compared with others [32, 35, 33] by showing results with the same or nearest number of dominant points.

The proposed method is compared with the other methods and results are shown in table 4.1.1. We report some meaningful measures for each tested method: the number of dominant points m , the compression ratio (CR), the E_2 error and the E_2/CR ratio. CR is defined as the ratio between the number of dominant points m and the number of contour points n .

We report also the value of threshold s chosen. The results clearly demonstrate better performance of our method respect to other techniques. In fact the E_2 error (and subsequently the E_2/CR value) is always lower keeping the same number of dominant points of other methods. Vice versa if we set an acceptable value of E_2 error, our method gives the lowest number of dominant points, i.e. well-located points, compared to the other.

We test our method and compare it to the other dominant points extraction techniques on common shapes (taken by [68] and [69]) also.

In fig. 4.6 and 4.7 it's possible to see some shapes we used in experiments. These images are processed by thresholding camera taken objects and their contour is affected by noise. For each image the graph which shows the E_2 error trend against the dominant point number is plotted. We use the log scale for the error E_2 . We obtain such values by varying the input parameter for each method except for the Teh-Chin method which does not require any. These graphs confirm the effectiveness of our method.

The polygonal approximation for some real images like Africa's map and maple leaf (taken by [70]) are shown in fig.4.8 and 4.9, respectively. We work on a Linux 64 bit operating system running on AMD64 6000.

The Africa's map contour is composed by 1364 points: we calculate the approximation with 20, 30 and 40 points respectively (the dotted line is the original

4.1. DOMINANT POINT ITERATIVE LOCALIZATION (DPIL)

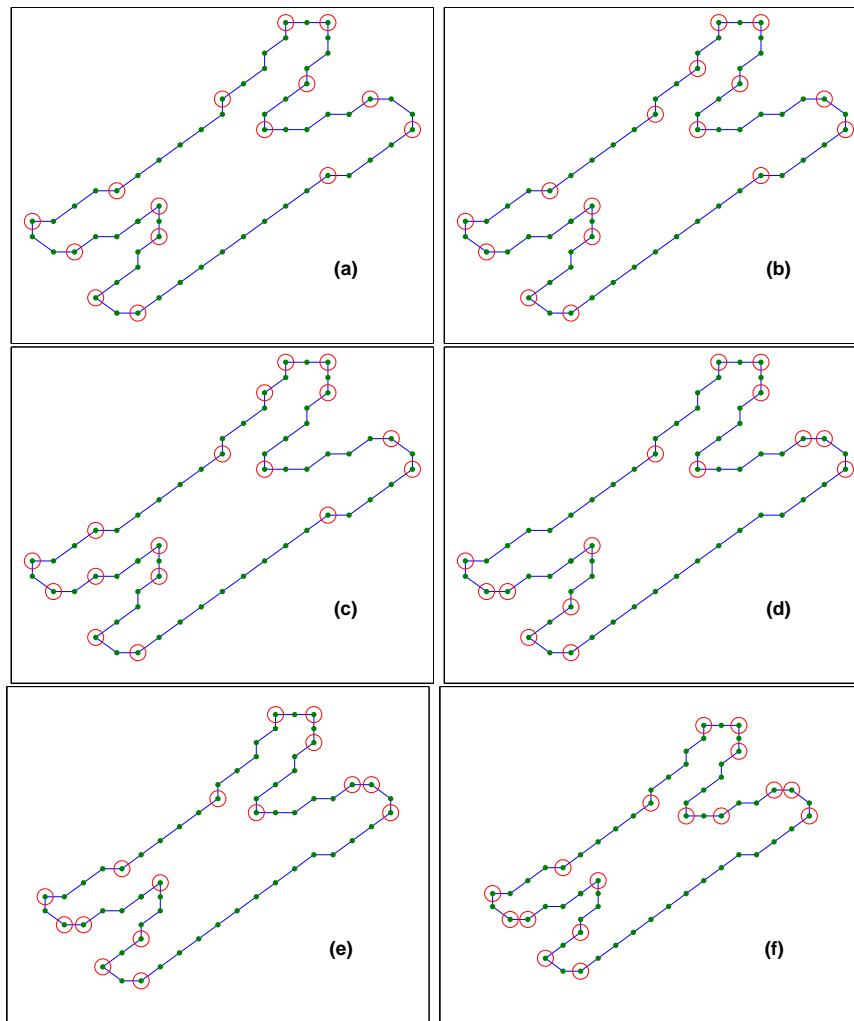


Figure 4.2: Chromosome shape (50 points) : (a) Teh-Chin (15 points), (b) Wu BV (16 points), (c) Wu DYN (17 points), (d) our method (15 points), (e) our method (16 points), (f) our method (17 points)

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

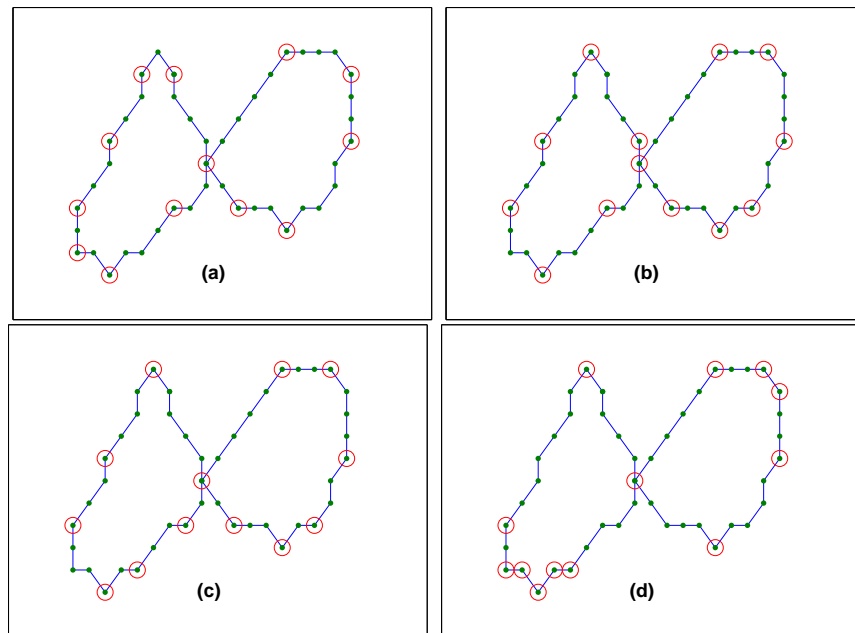


Figure 4.3: Infinite shape (45 points) : (a) Teh-Chin (13 points), (b) Wu BV (13 points), (c) Wu DYN (13 points), (d) our method (13 points)

contour). The resulting computational time is 0.0756 sec., 0.0731 sec and 0.0659 sec.

The maple leaf shape is composed by 1630 points : we calculate the approximation with 40, 45 and 50 points respectively (the dotted line is the original contour). The resulting computational time is 0.1343 sec., 0.1449 sec and 0.1895 sec., respectively.

The proposed method not only has a very low computational time and robustness to noise but also produces a good polygonal approximation while keeping low the E_2 error. By empirical evidence we notice that a threshold value in a range around 0.5 give the best E_2/CR value.

4.1. DOMINANT POINT ITERATIVE LOCALIZATION (DPIL)

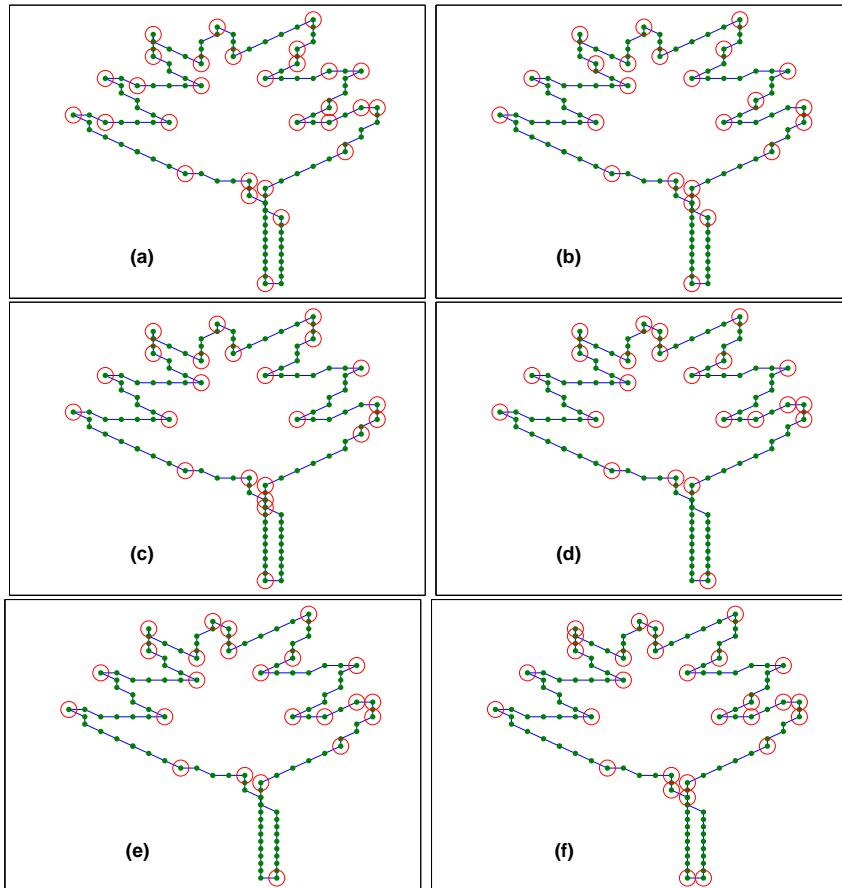


Figure 4.4: Leaf shape (120 points) : (a) Teh-Chin (29 points), (b) Wu BV (24 points), (c) Wu DYN(23 points), (d) our method (23 points), (e) our method (29 points), (f) our method (29 points)

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

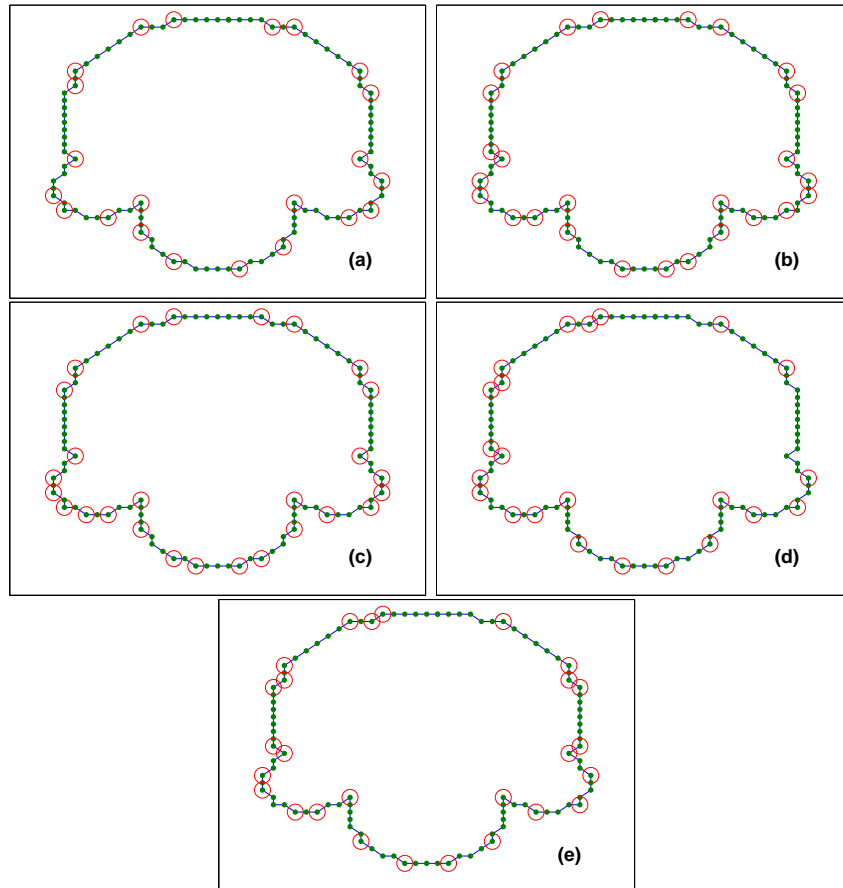


Figure 4.5: Semicircle shape (102 points) : (a) Teh-Chin (22 points), (b) Wu BV (26 points), (c) Wu DYN (27 points), (d) our method (23 points), (e) our method (27 points)

4.1. DOMINANT POINT ITERATIVE LOCALIZATION (DPIL)

Contour	Method	m	CR	E_2	E_2/CR
Chromosome (<i>points</i> =60)	Our method	15	4	6.5	1.64
	Our method	16	3.75	4.62	1.23
	Our method	17	3.53	3.74	1.06
	Teh-Chin	15	4	7.19	1.8
	Wu_bv	16	3.75	4.70	1.25
	Wu_dyn	17	3.53	5.01	1.42
Leaf (<i>points</i> =120)	Our method	23	5.22	14.32	2.74
	Our method	24	5	12.73	2.55
	Our method	29	4.14	7.90	1.91
	Teh-Chin	29	4.14	14.96	3.62
	Wu_bv	24	5	17.40	3.48
	Wu_dyn	23	5.22	22.44	4.3
Semicircle (<i>points</i> =102)	Our method	23	4.43	11.65	2.37
	Our method	27	3.78	6.85	1.51
	Teh-Chin	22	4.64	21.86	4.45
	Wu_bv	26	3.92	9.04	2.18
	Wu_dyn	27	3.78	9.92	2.49
Infinite (<i>points</i> =45)	Our method	13	3.46	3.65	1.05
	Teh-Chin	13	3.46	5.93	1.71
	Wu_bv	13	3.46	5.40	1.56
	Wu_dyn	13	3.46	5.60	1.62

Table 4.1.1: Comparative results for the proposed method. Lower E_2/CR is better.

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

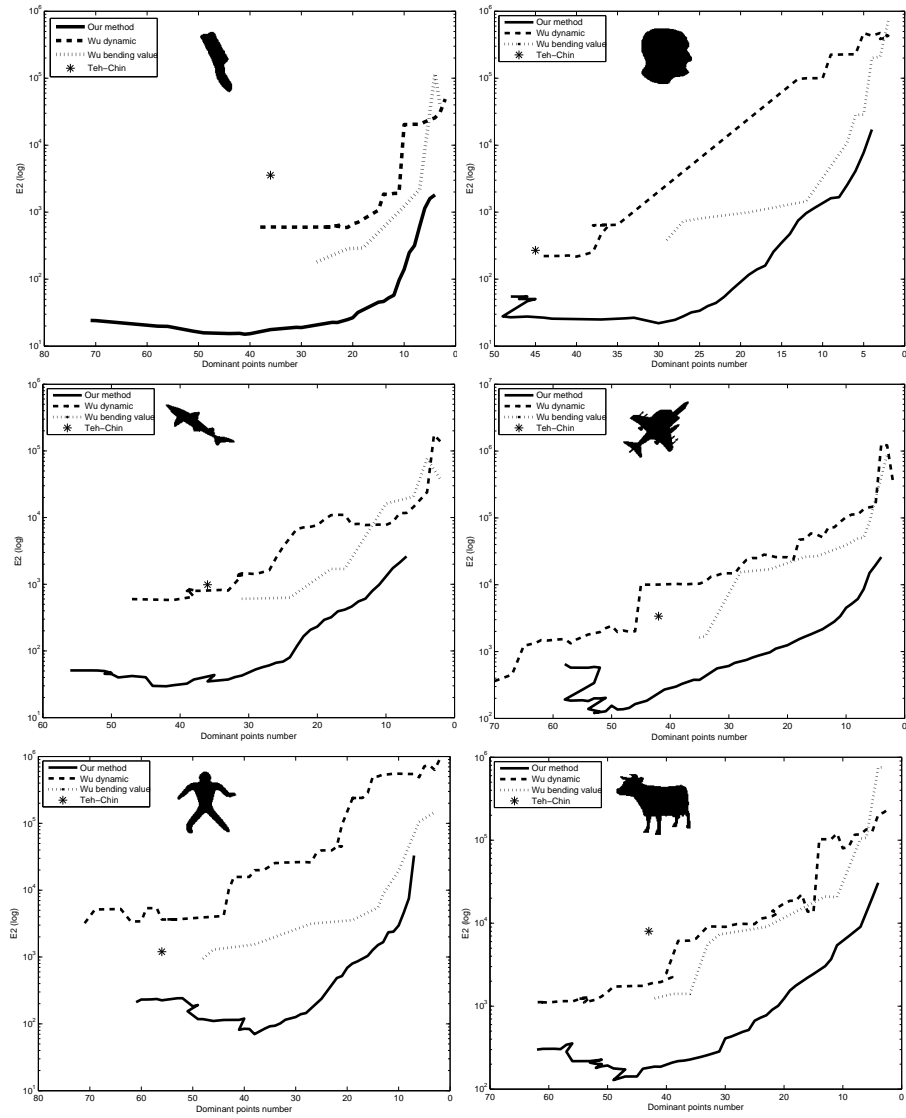


Figure 4.6: Shapes used in the experiments.

4.1. DOMINANT POINT ITERATIVE LOCALIZATION (DPIL)

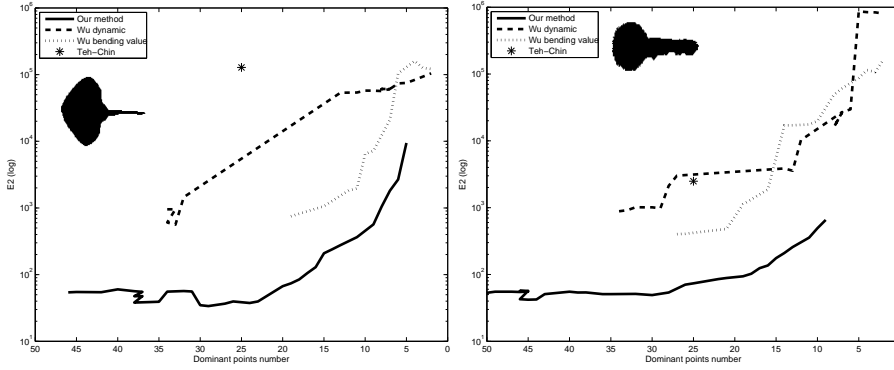


Figure 4.7: Shapes used in the experiments.

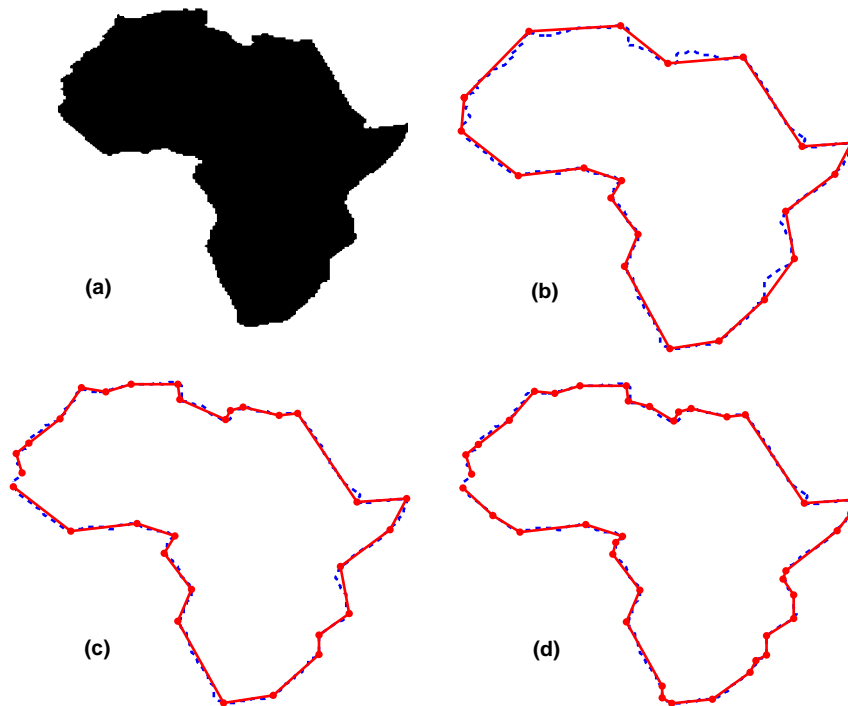


Figure 4.8: Africa's map : (a) original shape, (b) 20 points approximation, (c) 30 points approximation, (d) 40 points approximation

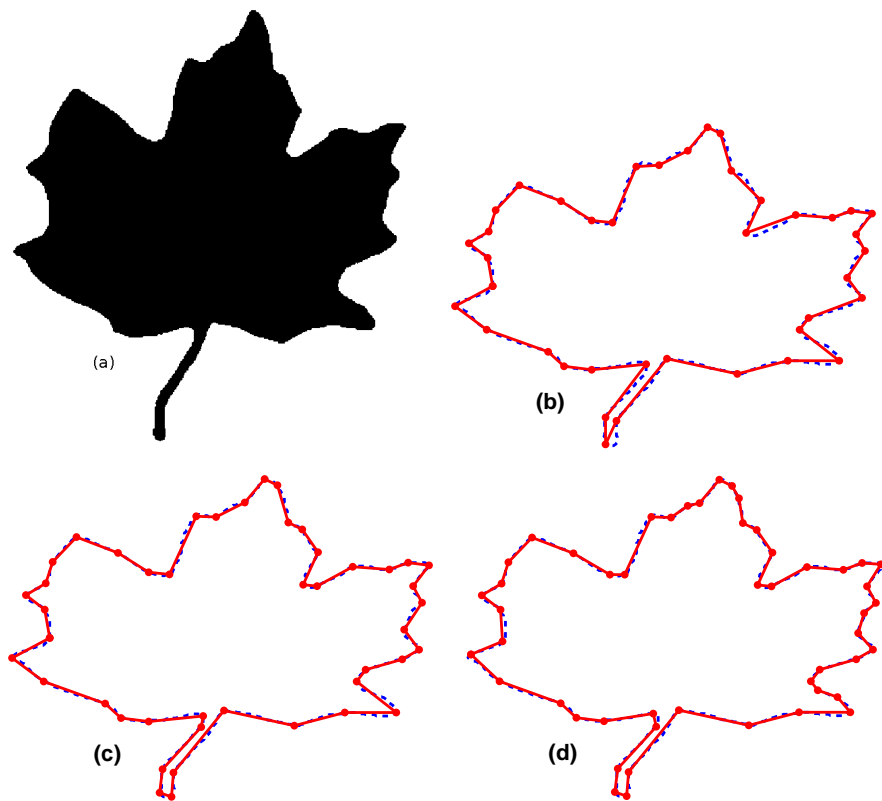


Figure 4.9: Maple leaf : (a) original shape, (b) 40 points approximation, (c) 45 points approximation, (d) 50 points approximation

4.2 Ant Colony Optimization Polygonal Approximation (ACOPA)

In this section we discuss a method that combines the dominant point detection and the ant colony optimization search.

The method is inspired by the ant colony search suggested in [53] but it results in a much more efficient and effective approximation algorithm.

The excellent results have been compared to works using an optimal search approach and to works based on exact approximation strategy.

4.2.1 Ant Colony Optimization algorithms

An Ant Colony Optimization (ACO) algorithm is a probabilistic technique for problems that can be related to the location of a good path within a graph.

Ant Colony Optimization firstly proposed by Dorigo in [59] is based on a computational paradigm inspired by the way real ant colonies function. This algorithm tends to reproduce the behaviour and the communication system of ants when from their nest they are directed to an area where there is food.

In fact, in its walking each ant issues a substance called pheromone, tracing the path that actually takes in order to allow the other ants to locate it and to choose a route based on the amount of pheromone they warn. This behaviour is important not only to locate a path that leads to food, but also to determine a short path to reach the destination. Indeed, the pheromone evaporates after a certain period of time, so the longer routes (which require more time to be flown) will tend to have a few tracks and then have little chance of being chosen. Conversely, in the shortest paths, which allow to reach your destination more quickly, the concentration of pheromone will be higher and consequently have more chance of being chosen by ants (see fig. 4.10).

While an ant moves practically at random, it encounters a previously laid trail and can decide, with high probability, to follow it, thus reinforcing the trail with its own pheromone. The process is characterized by a positive feedback loop, where the probability with which an ant chooses a path increases with the number of ants that previously chose the same path.

The ACO paradigm is inspired by this process. Artificial ants build solutions (tours) of a problem by moving on the problem graph from one node to another. The algorithm executes N_{max} iterations. During each iteration m ants build a tour in which a probabilistic decision rule is applied. If an ant in node i chooses to move to node j , the edge (i, j) is added to the tour under construction. This step is repeated until the ant has completed its tour. After ants have built their tours, each ant deposits pheromone on the visited edges to make them more desirable for future ants. The amount of pheromone trail τ_{ij} associated to edge (i, j) is intended to represent the learned desirability of choosing node j when in node i . The pheromone trail information is changed during problem solution to reflect the experience acquired by ants during problem solving. Ants deposit an amount of pheromone proportional to the quality of the solutions they produced: the shorter

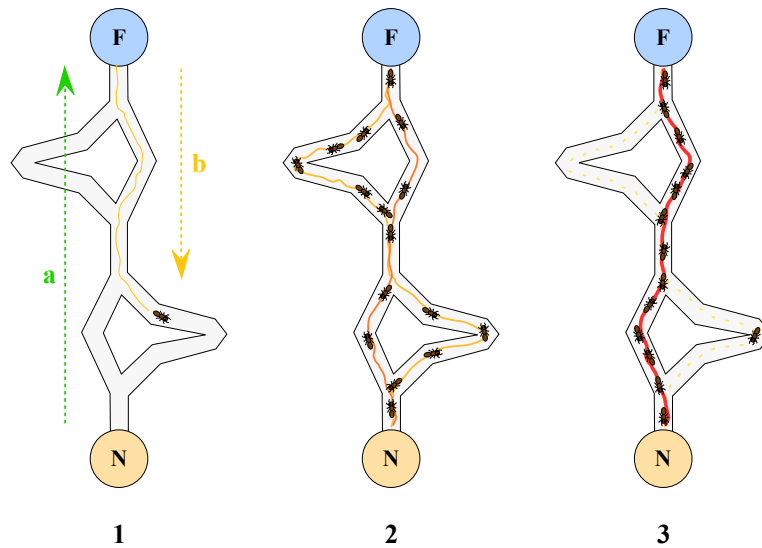


Figure 4.10: Example of ants behaviour. At the first step an ant finds food, and traces the path taken (round trip) with the release of pheromone. At the second step the other ants will choose different paths. As you go along, the longer routes will lose the trail of pheromone. In the third step we have created a path with a high concentration of pheromone that will be one of the shortest road to reach the destination. [Source: Wikipedia]

the tour generated by an ant, the greater the amount of pheromone it deposits on the edges which it used to generate the tour. This choice helps to direct search towards good solutions.

4.2.2 The ACO-based proposed method

We first define the problem of polygonal approximation in terms of optimization problem and then we describe how we use the ACO algorithm to solve it.

Problem formulation and graph representation

Given a digital curve represented by a set of N clockwise-ordered points, $C = \{c_1, c_2, \dots, c_N\}$ where $c_{(i+1) \bmod N}$ is the succeeding point of c_i , we define arc $\widehat{c_i c_j}$ as the set of points between c_i and c_j , and chord $\overline{c_i c_j}$ as the line segment connecting c_i and c_j . In approximating the arc $\widehat{c_i c_j}$ by the chord $\overline{c_i c_j}$, we make an error, denoted by $e(\widehat{c_i c_j}, \overline{c_i c_j})$ that can be measured by a distance norm as described in section 3.3.

The objective is to approximate a given curve by an optimal polygon with the minimal number of line segments such that the approximation error between the original curve and the corresponding line segments is less than a pre-specified tol-

4.2. ANT COLONY OPTIMIZATION POLYGONAL APPROXIMATION (ACOPA)

erance. Formally, the aim is to find the ordered set $T = \{c_{p_1}, c_{p_2}, \dots, c_{p_M}\}$ where $T \subset C$ and $M \leq N$ such that M is minimal and the set of M line segments, $P = \{\overline{c_{p_1}c_{p_2}}, \overline{c_{p_2}c_{p_3}}, \dots, \overline{c_{p_{M-1}}c_{p_M}}, \overline{c_{p_M}c_{p_1}}\}$, composes an approximating polygon to the point set C with an error norm between C and P no more than a pre-specified tolerance, ϵ .

To apply the ACO paradigm, we need to represent our problem in terms of a graph, $G = (V, E)$. For the polygonal approximation problem, each point on the curve should be represented as a node of the graph, i.e. $V = C$. The edge set E is generated as follows. Initially, the set E is created as an empty set. Then, for every node $c_i \in C$, we examine each of the remaining nodes, $c_j \in C$, in clockwise order. If the approximation error between the arc $\widehat{c_i c_j}$ and the line segment $\overline{c_i c_j}$ is no more than ϵ , then the directed edge $\overrightarrow{c_i c_j}$ is added to the edge set E . Thus, we have:

$$E = \{\overrightarrow{c_i c_j} | e(\widehat{c_i c_j}, \overline{c_i c_j}) \leq \epsilon\}. \quad (4.2.6)$$

This means that E contains edges the error of which is smaller or equal to ϵ . The edge is directed to avoid the ants walking backward. Then, the problem of polygonal approximation is equivalent to finding the shortest closed circuit on the directed graph $G = \langle V, E \rangle$ such that $E_2 \leq \epsilon$. In the following, the closed circuit completed by the ant k will be denoted by $tour_k$, its length will be $|tour_k|$ and the approximation error between the original curve C and the approximating polygon corresponding to $tour_k$ will be $E_2(C, tour_k)$.

Starting node initialization and selection

During each cycle an ant chooses a starting node and sequentially constructs a closed path to finish its tour. To find the shortest closed tour, it is convenient to place the ants at the nodes with a high probability of finding such a tour, instead of a randomly distribution. For the selection of the starting nodes we propose two alternative strategies, we call *Selection_1* and *Selection_2*. They differ with respect to the way the initial set of starting nodes is defined. Let's describe the two algorithms in detail.

Selection_1.

The number of signature extrema and the relative boundary points guide us to determine automatically the number of distributed ants and to select the nodes on the graph they choose to start their tour.

In this algorithm the number m of distributed ants is equal to the number of the signature extrema and the m relative boundary points represent the m starting nodes at the beginning of the first cycle. In the next cycles, the ants are placed at the nodes with higher probability to find the shortest closed tour.

We thus create a selection list for the starting nodes $T_i, i = 1, 2, \dots, N$. Initially,

$$T_i = \begin{cases} 1 & \text{if } i \text{ is a starting node} \\ 0 & \text{otherwise} \end{cases} \quad (4.2.7)$$

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

The probability with which the node i is chosen as a starting node in the next cycles, denoted $Choose_i$, is estimated as the value T_i normalized by the sum of all values,

$$Choose_i = \frac{T_i}{\sum_{j=1}^N T_j} \quad (4.2.8)$$

At the end of each cycle, the value of $T_i, i = 1, 2, \dots, N$ is updated.

Let's denote by $Start_Node_i$ the set of ants which start with the node i at the current cycle and by $|Start_Node_i|$ its size. We update the value T_i making a trade-off between the average quality of current solutions constructed by the ants in $Start_Node_i$ and the value of $Choose_i$ derived from older cycles. Thus, we let

$$T_i = \begin{cases} \frac{1}{|Start_Node_i|} \sum_{j \in Start_Node_i} \frac{1}{|tour_j|} + Choose_i & \text{if } i \text{ is a start node} \\ & \text{at current cycle} \\ T_i & \text{otherwise} \end{cases} \quad (4.2.9)$$

During the process, in the early iterations the ants will prefer the nodes which have not been chosen yet as starting nodes, by exploring new solutions. After all the nodes have been tested, they will tend to choose the starting nodes which have more experiences of constructing shorter tours and enforce an exploitation search to the neighborhood of better solutions.

Selection_2.

For each node $i, i = 1, \dots, N$, of the graph we evaluate the greatest approximation error among all the directed edges departing from i . We generate a list of the N greatest errors sorted in ascending order. We select the first D edges, where D is a percentage on N (in all the experiments $D = 15$) and detect the nodes where these edges depart from.

The list of such nodes, after eliminating duplications, represents the set of starting nodes at the beginning of the first cycle and its size is the number of the distributed ants. The selection phase continues as described previously.

Node transition and pheromone updating rules

According to ACO paradigm, the probability with which an ant k chooses to move from node i to node j is determined by the pheromone intensity τ_{ij} and the visibility value η_{ij} of the corresponding edge.

In the proposed method, τ_{ij} is equally initialized to $1/N$ and is updated at the end of each cycle according to the average quality of the solutions that involve this edge. The value of η_{ij} is determined by a greedy heuristic which encourages the ants to walk to the farthest accessible node in order to construct the longest possible line segment in a hope that an approximating polygon with fewer vertices is obtained eventually. This can be accomplished by setting $\eta_{ij} = |\widehat{c_i c_j}|$, where $|\widehat{c_i c_j}|$ is the number of points on $\widehat{c_i c_j}$. The value of η_{ij} is fixed during all the cycles since

4.2. ANT COLONY OPTIMIZATION POLYGONAL APPROXIMATION (ACOPA)

it considers local information only. The transition probability from node i to node j through directed edge $\overrightarrow{c_i c_j}$ is defined as

$$p_{ij} = \frac{\tau_{ij} \eta_{ij}}{\sum_{\text{for all } \overrightarrow{c_i c_h} \text{ from } c_i} \tau_{ih} \eta_{ih}}. \quad (4.2.10)$$

At the end of each cycle we update the intensity of pheromone trails of an edge by the average quality of the solutions that use this edge. At the end of each cycle, the pheromone intensity at directed edge $\overrightarrow{c_i c_j}$ is updated by combining the pheromone evaporation and deposition as follows:

$$\tau_{ij} = \rho \tau_{ij} + \max\left(\sum_{k=1}^m \Delta_{ij}^k, 0\right) \quad (4.2.11)$$

where $\rho \in (0, 1)$ is the persistence rate of previous pheromone trails and Δ_{ij}^k is the quantity of new trails left by the ant k and it is computed by

$$\Delta_{ij}^k = \begin{cases} \frac{1}{|tour_k|} & \text{if the } \overrightarrow{c_i c_j} \in tour_k \text{ and } E_2(C, tour_k) \leq \epsilon \\ -\frac{E_2(C, tour_k)}{\epsilon N} & \text{if the } \overrightarrow{c_i c_j} \in tour_k \text{ and } E_2(C, tour_k) > \epsilon \\ 0 & \text{otherwise.} \end{cases} \quad (4.2.12)$$

Acting upon the ACO paradigm, more quantities of pheromone trails will be laid at the edges along which most ants have constructed shorter feasible tours. As consequence, the proposed rule will guide the ants to explore better tours corresponding to high quality solutions.

Refinement of polygonal approximation

On the detected dominant points we apply an enhancement process to refine their localization according to a minimal distance criterion.

Let $T = \{c_{p_1}, c_{p_2}, \dots, c_{p_M}\}$ the ordered set of detected dominant points. Considering the couple c_{p_i} and $c_{p_{i+2}}$, $i = 1, \dots, M$, the refinement process consists in moving the intermediate point $c_{p_{i+1}}$ between c_{p_i} and $c_{p_{i+2}}$ in a new point by a local distance minimization. For all the points c_k on the arc $\widehat{c_{p_i}, c_{p_{i+2}}}$ we evaluate the sum of the approximation errors, $e(\widehat{c_{p_i}, k}, \overline{c_{p_i}, k}) + e(\widehat{k, c_{p_{i+2}}}, \overline{k, c_{p_{i+2}}})$. The new position for the point $c_{p_{i+1}}$ is chosen as follows:

$$c_{p_{i+1}} = \min_{c_k} e(\widehat{c_{p_i}, c_k}, \overline{c_{p_i}, c_k}) + e(\widehat{c_k, c_{p_{i+2}}}, \overline{c_k, c_{p_{i+2}}}) \quad (4.2.13)$$

The new point $c_{p_{i+1}}$ between c_{p_i} and $c_{p_{i+2}}$ is then used in the rest of the process. The refinement step is repeated for each pair $(c_{p_i}, c_{p_{i+2}})$, $i = 1, \dots, M$.

In table 4.2.2 the great improvement driven from this step in other methods also is shown.

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

Method	d.p. #	E_2	
		w/o refinement	with refinement
ACOPA	32	192.48	134.28
Yin [53]	32	208.02	146.23
Teh-Chin [32]	64	1488.6	183.61
Wu-bv [35]	35	1977.36	750.03
Wu-dyn [33]	32	19629.77	1687.33

Table 4.2.2: F10 Shape : 399 contour points.

Summary of the proposed method

We summarize the proposed algorithm as follows:

Input:

$C = \{c_1, c_2, \dots, c_N\}$: a set of clockwise ordered points.

N_{\max} : the maximal number of running cycles.

1. Initialization:

Construct the directed graph $G = \langle V, E \rangle$ as described in Subsection 4.2.2.

Determine the number m of ants as described in Subsection 4.2.2.

Set $\tau_{ij} = 1/N$ for every directed edge $\overrightarrow{c_i c_j}$.

Initialize T_i for $i = 1, \dots, N$ as described in Subsection 4.2.2.

Set $NC=1$, where NC is the cycle counter.

Set $best_tour = \{\overrightarrow{c_1 c_2}, \overrightarrow{c_2 c_3}, \dots, \overrightarrow{c_{N-1} c_N}, \overrightarrow{c_N c_1}\}$.

2. For every ant do

Select a starting node as described in Subsection 4.2.2.

Repeat

Move to next node according to the node transition rule using Eq. 4.2.10.
until a closed tour is completed.

3. Find out the shortest feasible tour, say $current_tour$, among the m tours obtained in step 2.

4. If ($|current_tour| < |tour_best|$) or
($|current_tour| = |best_tour|$ and $E_2(C, current_tour) < E_2(C, best_tour)$)
then $best_tour = current_tour$.

5. For every directed edge $\overrightarrow{c_i c_j}$ do

Update the pheromone intensity using Eqs. 4.2.11 and 4.2.12.

6. For every selection entry T_i do

Update the entry value using Eq. 4.2.7.

7. If ($NC < N_{\max}$) $NC=NC+1$ and go to Step 2;

8. Refine the approximation polygon as described in Subsection 4.2.2.

4.2.3 Experimental results

We present now the experimental results of the proposed approximation algorithm.

In the following, we call by *Method_1* and by *Method_2* the methods based on *Selection_1* and *Selection_2* algorithms, respectively. The methods have been first

4.2. ANT COLONY OPTIMIZATION POLYGONAL APPROXIMATION (ACOPA)

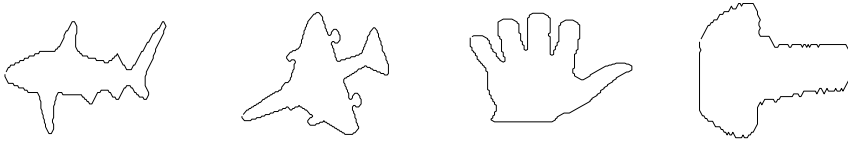


Figure 4.11: Some real world curves: fish, aircraft, hand, key.

tested on some real contours, showed in Fig.4.11. In Fig.4.12 we show the initial starting nodes, the approximating polygon before and after the refinement step by applying *Method_1* and *Method_2* for the aircraft curve.

The performances have been compared to those of the method proposed by Yin [53] and presented in table 4.2.3. We have evaluated the initial number of points, N , the number of detected dominant points, Np , and the approximation error, E_2 . For all the curves, our methods show a much better efficacy, both before and after the refining step. Also, our algorithms are less parameters dependent than Yin's one.

Apart the ϵ -bound constraint and the number of the maximum cycles, Yin's method uses other five parameters. The detected approximating polygon is strongly dependent on the chosen parameter values. Our methods automatically choose the number of distributed ants and do not need any other parameter. We only define the maximum number of iterations and the ϵ -bound.

Moreover, since the approach is non deterministic, different runs can lead to different solutions. We have approached this problem by choosing the initial set of starting nodes automatically and by introducing a refinement step to reduce the approximation error. Such two aspects are not present in Yin's approach.

Finally, the approximation error levels off after only few iterations of the iterative process (in case of F10 curve after 3 iterations we achieve the best value $E_2 = 193.03$ and in case of hand curve just one iteration is needed to get $E_2 = 210.63$). It means that our methods are computationally more efficient, too.

Our algorithms have been, also, tested on four commonly used curves (leaf, chromosome, semicircle and infinite, sec. 4.1.4). We have, first, compared our approach to other methods based on global search heuristic, like ACS_Poly method [53], the GA-based method [71] and the TS-based method [51]. The results confirm the superiority of our method both in effectiveness and in efficiency.

	N	Np	<i>Method_1</i> before refine	<i>Method_1</i> after refine	<i>Method_2</i> before refine	<i>Method_2</i> after refine	Yin [53]
Fish	325	25	172.45	102.03	171.10	101.83	176.19
F10	399	32	193.03	135.52	192.48	134.28	208.02
Hand	513	31	210.63	149.56	210.63	149.56	218.28
Key	257	16	116.14	80.96	117.85	82.79	121.25

Table 4.2.3: The initial number of points, N , the number of dominant points, Np , and the approximation error E_2 for the real curves for our methods and Yin's algorithm. In bold the best approximation errors.

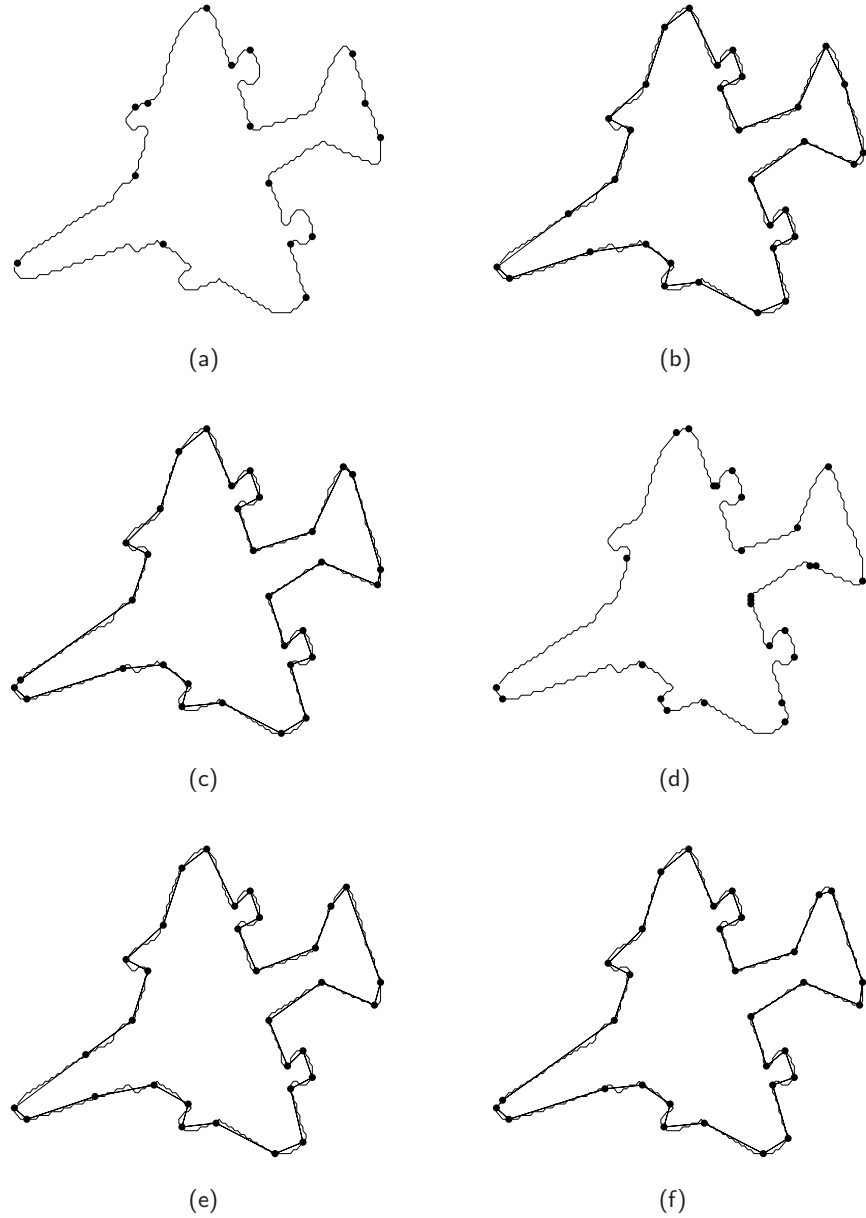


Figure 4.12: F10 contour: the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying *Method_1* in (a), (b), (c) and by applying *Method_2* in (d), (e), (f), respectively.

4.2. ANT COLONY OPTIMIZATION POLYGONAL APPROXIMATION (ACOPA)

Curves	E_2	GA-based	TS-based	ACO_Poly	Method_1	Method_2
Leaf $N = 120$	150	17	11	13	9	9
	100	16	14	13	11	11
	90	17	15	14	12	11
	30	21	20	19	17	16
	15	23	23	23	19	19
Chromosome $N = 60$	30	7	6	6	6	6
	20	8	8	8	7	7
	10	10	11	11	11	9
	8	12	12	11	11	11
	6	15	14	14	13	12
Semicircle $N = 102$	60	13	11	10	10	10
	30	14	14	13	12	11
	25	17	15	13	12	12
	20	19	16	16	15	15
	15	23	18	18	17	16
Infinite $N = 45$	30	N/A	N/A	6	5	5
	20	N/A	N/A	7	6	6
	15	N/A	N/A	7	6	6
	10	N/A	N/A	7	7	7
	6	N/A	N/A	9	8	8
	3	N/A	N/A	17	10	10

Table 4.2.4: Comparison in terms of number of dominant points, N_p , and E_2 on the benchmark curves.

The experimental results have been, finally, compared to many existing methods for dominant point detection. In table 4.2.3 for each algorithm we show the number of detected point, N_p , the approximation error, E_2 , the compression ratio, $CR = N/N_p$, as suggested in [27] and [36] to measure the efficiency of the dominant point detectors and to compare them. Analyzing the table, we can affirm that the proposed approach is superior to many existing algorithms based on exact search methodology, too.

Curves	Method	N_p	E_2	$CR = N/N_p$	E_2/CR	E_2/CR^2	E_2/CR^3
Leaf $N = 120$	Teh_Chin [32]	28	15.43	4.286	3.600	0.840	0.196
	Cornic [28]	N/A	N/A	N/A	N/A	N/A	N/A
	Ray-Ray [43]	26	16.43	4.615	3.560	0.771	0.167
	Wu [35]	24	15.93	5.000	3.186	0.637	0.127
	Marji-Siy [36]	17	28.67	7.059	4.062	0.575	0.082
	Carmona [27]	23	15.63	5.217	2.996	0.574	0.110
	Yin [53]	24	13.73	5.000	2.746	0.549	0.110
	Our Method_1	24	10.62	5.000	2.124	0.425	0.085
	Our Method_2	24	10.62	5.000	2.124	0.425	0.085
Chromosome $N = 60$	Teh_Chin	16	6.40	3.750	1.707	0.455	0.121
	Cornic	17	5.54	3.529	1.570	0.445	0.126
	Ray-Ray	14	7.67	4.286	1.790	0.418	0.097
	Wu	16	4.70	3.750	1.253	0.334	0.089
	Marji-Siy	10	10.01	6.000	1.668	0.278	0.046
	Carmona	14	4.93	4.286	1.150	0.268	0.063
	Yin	14	6.41	4.286	1.496	0.349	0.081
	Our Method_1	14	5.39	4.286	1.258	0.293	0.068
	Our Method_2	14	5.39	4.286	1.258	0.293	0.068
Semicircle $N = 102$	Teh_Chin	22	20.61	4.636	4.445	0.959	0.207
	Cornic	30	9.19	3.400	2.703	0.795	0.234
	Ray-Ray	19	16.33	5.368	3.042	0.567	0.106
	Wu	26	9.04	3.923	2.304	0.587	0.150
	Marji-Siy	15	22.70	6.800	3.338	0.491	0.072
	Carmona	24	9.88	4.250	2.325	0.547	0.129
	Yin	23	10.50	4.435	2.368	0.534	0.120
	Our Method_1	23	6.28	4.435	1.416	0.319	0.072
	Our Method_2	23	6.28	4.435	1.416	0.319	0.072
Infinite $N = 45$	Teh_Chin	13	3.46	3.462	1.000	0.289	0.083
	Cornic	10	4.30	4.500	0.956	0.212	0.047
	Ray-Ray	12	3.75	3.750	1.000	0.267	0.071
	Wu	13	5.78	3.462	1.670	0.482	0.139
	Marji-Siy	N/A	N/A	N/A	N/A	N/A	N/A
	Carmona	10	5.56	4.500	1.236	0.275	0.061
	Yin	11	4.44	4.091	1.085	0.265	0.065
	Our Method_1	11	3.44	4.091	0.841	0.206	0.050
	Our Method_2	11	3.44	4.091	0.841	0.206	0.050

Table 4.2.5: Comparison with other methods on the benchmark curves.

4.3 Genetic Algorithms Polygonal Approximation (GAPA)

In this section we present a method based on a genetic approach.

The chromosome coincides with an approximating polygon and is represented by a binary string. Each bit, called gene, represents a curve point where dominant points have 1-value.

The proposed algorithm enhances the selection and mutation phase avoiding the premature convergence issue. It is compared to other similar approaches and its efficiency is clearly demonstrated by experimental results giving a better approximation by lowering the error norm with respect to the original curves.

4.3.1 Genetic algorithms in polygonal approximation

In the last decade, GAs have found large space in the field of polygonal approximation.

The starting point of GAs polygonal approximation is the modeling of the problem, which defines the structure of chromosomes to be used and the measure of comparison to use to quantify the goodness of the solutions. Generally, the selected chromosomes are binary, while the most used measure is the error of approximation.

Given a closed curve consisting of n points and a value of v representing the number of points that the approximating polygon must have, each chromosome will be composed of n genes that can take the following values:

$$\text{gene} = \begin{cases} 0 & \text{the point is not part of the approximating polygon} \\ 1 & \text{the point is part of the approximating polygon} \end{cases} \quad (4.3.14)$$

From this we can see that the algebraic sum of the genes that make up a chromosome representing a permissible solution must be equal to v . Moreover, the points that make up the curve (and hence also those that make up the polygon) must be ordered (clockwise or counterclockwise) so that for every point you can determine its predecessor and its successor.

In [55] a chromosome is used to represent a polygon and is represented by a binary string. A data reduction process is firstly applied to prevent unnecessary computation in the evolutionary process. The objective function is defined as the error E_2 (section 3.3), between the given curve and the approximating polygon. Three genetic operators namely selection, crossover and mutation, have been constructed for this specific problem. The method proposed in [56] is different from the standard GAs in cross-over and mutation operations in order to solve the specific application. It has good approximation results among several existing methods, but the computational load is relatively high because of the parallel search of GA.

4.3.2 The proposed method

In order to speed up the computational time, especially in the case of complex curves, we use the same technique described in [55]: in the preprocessing phase

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

a matrix L , containing the values of quadratic errors of every possible segment, is created. In this way we can compute quickly every single generation even if a time to build the matrix L is required.

Taking into account that GAs tend to balance the exploration of the space research it follows that a considerable part of the values contained in the array is never used during algorithm execution. So we decided initially to create L empty and update it only during the steps demanding fitness control. Hence when the algorithm needs to know the fitness value first check it: if the standard error of the required segment is present in the matrix we use it, otherwise the fitness computed value is stored in the matrix. By following this procedure the first generations are computed slow while the speed is greatly increased as the matrix L is being filled.

The selection phase has been implemented, rather than with the roulette wheel technique, using the method of *simulated annealing* [72].

As for selection, even for the crossover was decided to use a completely different method than the one proposed by [55]. First of all we leave the elitism, because, after the introduction of Boltzmann selection, it wasn't considered influent in searching for solutions.

We also eliminate the probability of crossover, preferring to process all the chromosomes with this operator; in fact we can notice that the competition between parents and children is enough to keep an adequate number of chromosomes (in terms of balance) within the new population respect to the previous generation. Besides we decide to increase the diversity of children by parents, through a decomposition of chromosomes characterized by a variable number of sub-strings.

To achieve this goal we use a method inspired by the crossover adopted by [73]. Given two chromosomes X and Y involved in crossover process and represented as:

$$X = x_1x_2x_3\dots x_k \quad (4.3.15)$$

$$Y = y_1y_2y_3\dots y_k \quad (4.3.16)$$

starting from the first gene we compute the value d so that:

$$\sum_{i=1}^d x_i = \sum_{i=1}^d y_i \quad (4.3.17)$$

This allows to determine a sub-string that exchanged between the two genes permit to maintain two eligible solutions. After identifying the first group of genes, we repeat the procedure applying it from time to time to the portion of the chromosomes that have not been examined yet; the process ends when the last gene is reached.

We decide to introduce an additional constraint: each sub-string should have a length not less than two. At this point the method described in [73] uses an orthogonal array to determine the best combination of sub-strings suitable to generate the two best children. It implies that the second child differs from the first only for a sub-string; this could facilitate a low diversification of the population and lead to premature convergence. We therefore decide to build every time eight children randomly, where each i -th sub-string has a 50% chance of being taken

4.3. GENETIC ALGORITHMS POLYGONAL APPROXIMATION (GAPA)

from the first parent and 50% to be extracted from the second parent. This value is chosen because it was observed that by using the L matrix, containing the errors of the segments, the construction of additional children does not involve significant worsening over time used; furthermore we find that values greater than eight do not produce significant enhancements. Thereafter eight children and two parents will compete: the two chromosomes with higher fitness are included in the next population.

During our experiments we have to deal with the premature convergence problem. We decide to observe in more detail the evolution of the population at different generations. This phase is characterized by analytical methods common to *numerical taxonomy*, a technique used in biology to classify organisms on the basis of similarity, without taking into account the evolutionary relations. Premature convergence is due to the presence of chromosomes characterized by a value of fitness very high compared to other sectors of the population; this situation in turn leads to true direct cause of the problem, namely the reduction of diversity and consequently uniformity of the resulting chromosomes. So, using the numerical taxonomy, we compare the various chromosomes to identify the moment at which the features of some chromosomes begin to spread too intensely.

The instrument used for the purposes of comparison is known as *index*, a statistical technique that allows to quantify the similarity between elements of a set. By considering the analysis of the population, in order to compare all possible pairs of chromosomes at the beginning of each generation the Jaccard index [74] is used.

By experimental results we also notice that if in the population had been placed two or more chromosomes identical with relatively high fitness values, convergence is reached very quickly, because in each generation the similar chromosomes increase exponentially. We decide then to modify identical chromosomes before applying the operators of selection, crossover and mutation. We choose to apply a sort of deep and random mutation to the chromosomes in question, in order to increase the diversification of the population. For each set H_i of identical chromosomes, consisting of n_i elements, with $n_i > 1$, $n_i - 1$ are taken and each is modified by a random shift of their genes. This modification brings a significant improvement in performance by reducing the variance between the values obtained in various tests on the same case and without an aggravation of the best results.

Moreover, since the information offered by Jaccard index allows to identify not only the equal chromosomes but also those which are most similar to others, we decided to exploit them for further improvements. The similarity values are stored in a matrix where the rows and columns represent chromosomes. The row (or column) with the highest sum corresponds to the chromosome most similar to all others and therefore are more inclined to have children alike.

By performing different actions in this direction, it is noted that the application of the shift to the first two random chromosomes, more similar to the others, results in additional improved performance.

Since the application of the shift can be seen as a random mutation that can improve the spread of research within the space of solutions, we thought of making more profound operator intervention to the original mutation which, as written previously, tend to focus research toward specific pathways.

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

Indeed it is considered that the two approaches could offset each other and thus improve the behavior of the algorithm in such situations where we are close to a global optimum.

Initially we increase the value of the probability of mutation, from 0.15 to 0.25; we also decide to use more operators into the final stages of processing.

As mentioned above the threshold t_e , representing the number of successive generations needed before the algorithm is stopped, is set, in the method of [55], to 10. Hence we set $t_e = 18$ and apply the mutation, with a probability equal to 80%, to a portion of the population when the number of consecutive generations without improvement is greater than $t_e/2$. The number of chromosomes affected by this procedure must be equal to

$$\frac{z}{t_e} - g_c - g_b + 2 \quad (4.3.18)$$

where z is the population size, g_c is the current generation number and g_b the number of generation in which the best chromosome has been discovered.

As expected the introduction of these changes has led in some cases to obtain better results than the previous ones, further reducing their variance. Initially we decided to leave the pre-processing method proposed in [55] for the removal of collinear points. But if on the one hand, this produces a significant reduction in the workload of the algorithm, with benefits on execution speed, on the other side we observed that in the final results of more complex curves the positioning of some vertices was not entirely satisfactory. For example, there have been frequent cases where two (or more in the worst cases) vertices were close to each other, when it would be more appropriate to locate one of them in a different position.

By eliminating the pre-processing phase the computational time is longer, but the results improved considerably. By the tests carried out we infer that the removal of collinear points does not lead to significant losses of information only when we deal with simple digital curves; when we need to analyze more complex curves, it is preferable not to reduce points. For these reasons, we choose not to include this feature in the final version of the algorithm. We point out however that, despite keeping the collinear point suppression step in pre-processing phase, global results obtained are better than those obtained by [55].

The proposed algorithm can be resumed as follows:

- **Input:** the original curve C and the number of segments m of the approximating polygon;
- **Output:** List of vertices that approximate the given curve C .

Steps:

1. Random generation of a population made by 60 chromosomes each representing possible polygonal approximation.
2. Localization of the best chromosome S_b ; $g_b=0;g_c=1;t_e=18$.
3. Generation computing. Random mutation through shift of any identical chromosome in the population and the three chromosome most similar to the others.
4. Boltzmann selection and mating pool construction.

4.3. GENETIC ALGORITHMS POLYGONAL APPROXIMATION (GAPA)

5. Crossover : couple generation on chromosomes randomly chosen by the mating pool; for every couple random generation of eight children; competition between parents and children; insertion of the two best chromosomes into the new population.
6. New population chromosomes mutation with 25% of mutation probability for each gene.
7. New population best chromosome (S_{cb}) localization;
Let F be the fitness function defined as $F(S) = 1/E(C, R(S))$ where $E(C, R(S))$ is the error defined in eq. 3.3.3 between the curve C and the approximating polygon $R(S)$
if $F(S_{cb}) > F(S_c)$ **then** $S_b = S_{cb}$, $g_b = g_c$; **goto** 3
else if $(g_c - g_b) = t_e$ **then goto** 7;
else if $(g_c - g_b) > t_e/2$ **then** mutation of a new population chromosome part: we consider only those which have 80% probability of gene mutation; **goto** 3;
8. The algorithm returns S_b , representing the vertices list of the best C approximation found.

4.3.3 Experimental results

We show now the results obtained by applying the algorithm to some images (Fig.4.13-Fig.4.17). In the following tables we indicate the parameter k , representing the number of segments of the polygonal approximation, the lowest and the average integral square error (Avg E_2), its variance (Var) and the average computational time expressed in seconds (Avg time). The average value is referred to 20 times execution tests. Other well known polygonal approximation techniques are tested and compared to our algorithm.

From the results obtained, we can see that the algorithm implemented allows to obtain better results than other methods analyzed exploiting GAs. From Table 4.3.11 we can also notice that for the examined cases the results appear to be equal or close to those reported in [49], using a method based on the elimination of dominant points which ensure to obtain the best result in 96% of cases.

As for the computational time we can make some observations. It has been said that the pre-processing phase for the collinear points suppression was not included because, although it allows to reduce considerably the execution time, in most cases leads to greater errors on the polygonal approximations. However, we must remember that the results obtained, though lower quality, can be useful in many fields of application (e.g. in automated comparison of figures) that require not particularly close to the original curve figures. So in these cases the pre-processing can be inserted to obtain a reduction of workload and hence the processing time.

Another aspect that emerges from the tables 4.3.6-4.3.10 is the processing time reduction as the number of segments k increases. This phenomenon is caused by the operator of mutation. Please note in fact that, if h_i is the gene to be changed and h_u e h_v are the previous and the next gene respectively (both equal to 1), then the algorithm will process all the positions between (h_u, h_i) e (h_i, h_v) in order to find the best placement of h_i . As the number of polygons approximating segments increases, the space separating two consecutive genes equal to 1 within

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

k	Lowest E_2	Avg E_2	Var	Avg time
8	13.43360	13.43360	0.00	1.91
9	12.07770	12.13250	0.01	1.86
10	8.06804	8.08098	0.00	1.98
12	5.81803	5.87695	0.03	1.90
14	4.16713	4.25055	0.02	2.01
15	3.79610	3.85910	0.01	1.99
17	3.12945	3.18453	0.00	1.94
18	2.82600	2.88708	0.00	2.11
20	2.33252	2.37767	0.00	2.08
22	1.93252	1.96881	0.00	2.16

Table 4.3.6: Results for Fig.4.13 (60 points)

k	Lowest E_2	Avg E_2	Var	Avg time
10	38.9200	39.8390	16.89	2.82
12	26.0045	26.1159	0.10	2.69
14	17.3855	17.3855	0.00	2.85
15	14.3994	14.3994	0.00	2.79
17	12.2179	12.3572	0.04	2.97
18	11.1902	11.3016	0.01	2.85
19	10.0364	10.1940	0.04	2.86
20	9.0087	9.10742	0.03	3.07
22	7.0112	7.23467	0.02	3.22
25	4.6248	4.85342	0.06	3.18
27	3.7013	3.91912	0.04	3.08
30	2.6425	2.92136	0.02	3.04

Table 4.3.7: Results for Fig.4.14 (102 points)

the chromosome reduces; therefore the number of operations that the mutation operator must perform decreases also.

In order to speed up the algorithm we can consider to reduce the mutation probability; this is not suitable for two main reasons: this would lead to a worsening of final solutions and there would be a slowdown of convergence. This fact increases the number of generations to be produced and leads to the opposite effect to that intended, increasing the processing time required by the algorithm.

4.3. GENETIC ALGORITHMS POLYGONAL APPROXIMATION (GAPA)

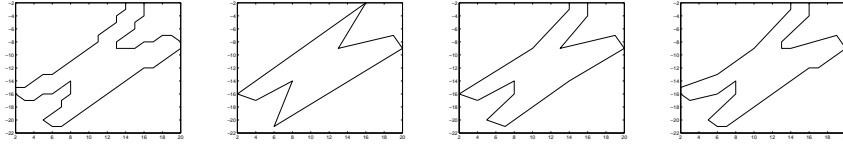


Figure 4.13: Chromosome shape : Original (60 points), approximation by 8 segments, approximation by 15 segments, approximation by 22 segments

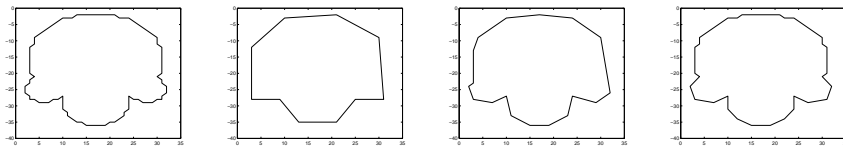


Figure 4.14: Semicircle shape : Original (102 points), approximation by 10 segments, approximation by 18 segments, approximation by 30 segments

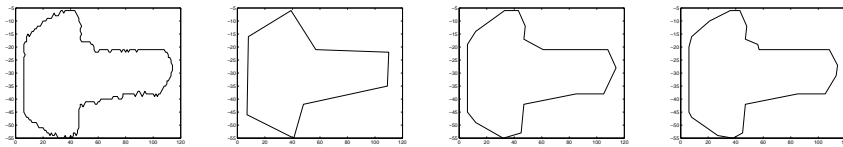


Figure 4.15: Noise key shape : Original (269 points), approximation by 8 segments, approximation by 16 segments, approximation by 20 segments

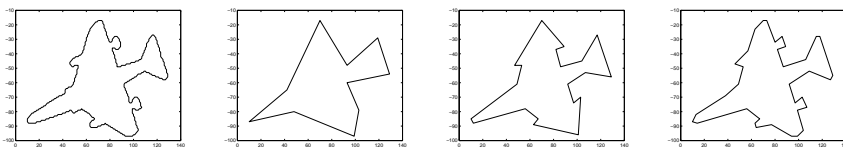


Figure 4.16: F10 shape : Original (409 points), approximation by 10 segments, approximation by 20 segments, approximation by 35 segments

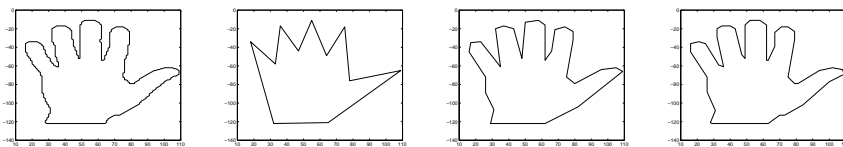


Figure 4.17: Hand shape : Original (528 points), approximation by 11 segments, approximation by 28 segments, approximation by 42 segments

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

k	Lowest E_2	Avg E_2	Var	Avg time
8	550.1150	550.1150	0.00	16.83
10	237.1970	237.1970	0.00	14.09
13	107.7740	108.6160	2.40	11.71
15	83.3271	84.3572	1.15	11.57
16	73.3830	74.3279	0.45	11.20
18	57.7161	59.6446	1.64	11.00
20	51.2445	52.6971	3.67	10.42

Table 4.3.8: Results for Fig.4.15 (269 points)

k	Lowest E_2	Avg E_2	Var	Avg time
10	2603.6600	2603.6600	0.00	37.42
20	628.25200	639.6410	482.33	22.73
30	143.19200	162.8280	703.49	19.07
35	94.9054	98.9640	14.09	18.58

Table 4.3.9: Results for Fig.4.16 (409 points)

k	Lowest E_2	Avg E_2	Var	Avg time
11	3177.9200	3177.9200	0.00	57.91
19	402.6030	405.5330	171.72	37.18
28	148.6800	154.5570	71.83	29.45
35	88.5785	93.6038	17.38	27.87
42	51.3637	55.7249	12.74	26.63

Table 4.3.10: Results for Fig.4.17 (528 points)

4.3. GENETIC ALGORITHMS POLYGONAL APPROXIMATION (GAPA)

k	OUR		Masood	Ho-Chen	
	Best ISE	Avg ISE	ISE	Best ISE	Avg ISE
Chromosome					
8	13.43360	13.43360	-	13.43	15.51
9	12.07770	12.13250	-	12.08	13.49
10	8.06804	8.08098	8.07	-	-
12	5.81803	5.81695	5.82	5.82	6.79
14	4.16713	4.25055	-	4.17	5.11
15	3.79610	3.85910	3.88	3.80	4.32
17	3.12945	3.18453	3.14	3.13	3.55
18	2.82600	2.88708	2.83	2.83	3.04
Semicircles					
10	38.9200	39.8390	-	38.92	44.07
12	26.0045	26.1159	-	26.00	29.53
14	17.3855	17.3855	-	17.39	20.14
15	14.3855	14.3994	14.40	-	-
17	12.2179	12.3572	-	12.22	14.58
18	11.1902	11.3016	-	11.34	12.86
19	10.0364	10.1940	10.04	10.04	11.52
22	7.0112	7.23467	7.01	7.19	8.52
27	3.7013	3.91912	3.70	3.74	5.03
30	2.6425	2.92136	2.64	2.84	3.57

Table 4.3.11: Comparative results between our method (OUR), Masood' one [49] and Ho-Chen's one [73]

CHAPTER 4. DOMINANT POINTS DETECTION: PROPOSED APPROACHES

k	OUR		Huang-Sun		Yin	
	Avg ISE	Avg time	Avg ISE	Avg time	Avg ISE	Avg time
Chromosome (fig. 4.13)						
8	13.43360	1.91	13.9809	0.68	32.7134	8.61
12	5.81803	1.90	6.0960	0.64	20.7801	8.56
15	3.85910	1.99	4.3262	0.66	12.3058	8.63
18	2.88708	2.11	3.5422	0.68	7.71227	8.72
Semicircles (fig. 4.14)						
10	39.8390	2.82	38.9200	1.18	93.4187	13.59
14	17.3855	2.85	21.8944	1.31	45.7951	13.36
19	10.1940	2.86	12.0241	1.35	27.9310	13.52
27	3.91912	3.08	6.5279	1.37	15.0895	13.65
Key (fig. 4.15)						
8	550.1150	16.83	628.616	14.24	1590.360	33.03
15	84.3572	11.57	125.144	14.62	387.210	33.11
18	59.6446	11.00	83.3598	14.71	300.167	33.21
20	52.6971	10.42	70.0745	14.73	246.844	33.19
F10 (fig. 4.16)						
10	2603.6600	37.42	2746.4595	114.16	7616.95	49.86
20	639.6410	22.73	841.5713	114.06	2755.90	50.06
30	168.8280	19.07	322.9759	114.99	1460.79	50.36
Hand (fig. 4.17)						
11	3177.9200	57.91	7694.6000	25.51	14532.80	64.65
28	154.5570	29.45	257.235	26.06	1824.25	65.18
35	93.6038	27.87	169.99	26.38	1163.73	65.29
42	55.7249	26.63	124.4370	26.28	907.716	65.65

Table 4.3.12: Comparative results between our method (OUR), Huang and Sun's one [55] and Yin's one [56]

5

Dominant point extraction techniques comparison

In this chapter we propose a comparison between the three dominant point extraction techniques described in chapter 4, i.e. the two heuristic techniques, based on Ant Colony Optimization (*ACOPA*) and based on Genetic Algorithm (*GAPA*) and the method based on an iterative approach, Dominant Points Iterative Localization (*DPIL*).

Our aim is to understand how the polygonal approximation obtained by presented approaches acts if we apply some realistic transformations to the analyzed images compared the “original” ones.

In section 5.1, we make some experiments to test the the effectiveness of the proposed method in terms of “best” polygonal approximation, i.e. the lowest quadratic error E_2 produced.

We assess the extent to which the affine transformations invariance affect the dominant point detection also. In section 5.2 we probe each algorithm respect to rotation, scale and translation by showing numeric results and describing the implemented methodology to estimate the invariance degree respect to affine transformations.

Further, in order to complete the analysis, we consider the noise sensitivity by making different runs based on image affected by growing noise in section 5.3.

5.1 Polygonal approximation testing

In order to test the effectiveness of the three methods we perform some experiments on four contour of common shapes - fish, f10, head, and hand (fig. 5.1) - taken by [68] and [69]. We choose such images because of their irregular, in the case of the fish and the f10, or articulated, in the case of the hand, contour opposed to the more smoothed head silhouette.



Figure 5.1: Shapes used in experiments.

In fig. 5.1 we report their contour and a sample set of dominant points, plotted for each method and highlighted with a small circle, is shown in fig.5.2.

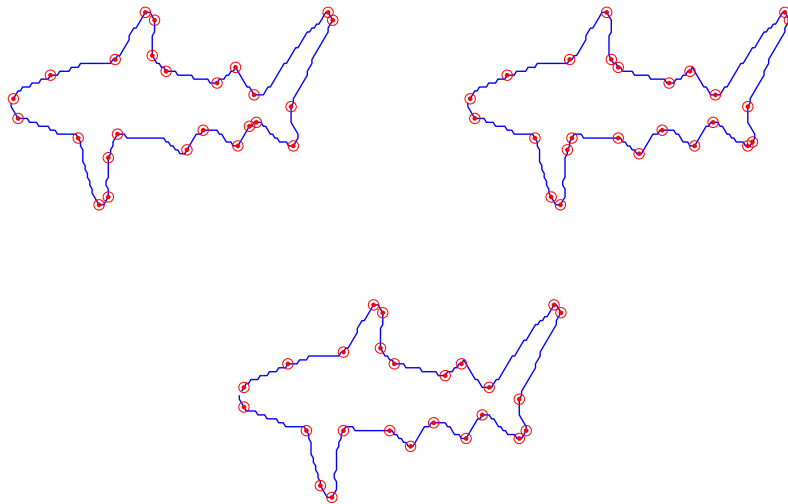


Figure 5.2: Dominant points extraction for fish shape (325 contour points, 25 dominant points) detected by DPIL, ACOPA and GAPA

5.1. POLYGONAL APPROXIMATION TESTING

All methods are compared by showing numeric results obtained using the same number of dominant points in table 5.1.2. The three algorithms run on 64-bit linux environment and AMD Athlon 64 6000+ platform. We report the E_2 error before and after the refinement step and the elapsed time to process the contour for each tested method. We point out the great performance improvement due to the refinement effect.

F10 - 399 points / 32 dominant points			
Method	E_2	ref. E_2	elps. time (sec.)
DPIL	285.17	221.25	0.0069
ACOPA	192.49	134.28	91.12
GAPA	116.94	116.94	10.48
Hand - 513 points / 31 dominant points			
Method	E_2	ref. E_2	elps. time (sec.)
DPIL	287.00	205.59	0.0092
ACOPA	208.84	149.56	140.99
GAPA	126.24	126.24	17.91
Head - 266 points / 19 dominant points			
Method	E_2	ref. E_2	elps. time (sec.)
DPIL	116.24	97.12	0.0046
ACOPA	122.13	78.24	41.29
GAPA	62.80	62.80	6.58
Fish - 325 points / 25 dominant points			
Method	E_2	ref. E_2	elps. time (sec.)
DPIL	208.66	131.69	0.0067
ACOPA	172.46	102.03	36.51
GAPA	85.71	85.71	9.82

Table 5.1.1: Comparative results for the proposed methods with and without the final refinement process.

Method	Contour	# points	# dom. points	E_2	ref. E_2	elps. time (sec.)
DPIL	F10	399	32	285.17	221.25	0.0069
ACOPA	F10	399	32	192.49	134.28	91.12
GAPA	F10	399	32	116.94	116.94	10.48
DPIL	Hand	513	31	287.00	205.59	0.0092
ACOPA	Hand	513	31	208.84	149.56	140.99
GAPA	Hand	513	31	126.24	126.24	17.91
DPIL	Head	266	19	116.24	97.12	0.0046
ACOPA	Head	266	19	122.13	78.24	41.29
GAPA	Head	266	19	62.80	62.80	6.58
DPIL	Fish	325	25	208.66	131.69	0.0067
ACOPA	Fish	325	25	172.46	102.03	36.51
GAPA	Fish	325	25	85.71	85.71	9.82

Table 5.1.2: E_2 comparative results for the proposed method with and without the final refinement process.

5.2 Affine transformation test

We test the three proposed approaches respect to affine transformation invariance considering rotation, scaling and translation by applying operators directly on object contours Cartesian coordinates.

5.2.1 Rotation

After extracting contour from binary images we mapped each point by applying a rotation matrix to the coordinates array. In this way we obtain a rotated contour by a prefixed angle θ respect to the mass center. Let:

$$T(i) = (x_i, y_i) \quad i = 1, \dots, N \quad (5.2.1)$$

the array containing the Cartesian coordinates of each contour point of the original shape,

$$c = \left(\frac{\sum_{i=1}^N x_i}{N}, \frac{\sum_{i=1}^N y_i}{N} \right) \equiv (x_c, y_c) \quad (5.2.2)$$

the array containing the coordinates of the mass center,

$$A = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \quad (5.2.3)$$

the rotation matrix then the transformation from T to the rotated shape contour T^R is:

$$T^R = A(T - c) + c \quad (5.2.4)$$

In our experiments θ is set to 5 degrees getting 72 rotations in all. In fig. 5.3 we show some examples of dominant point extraction for rotated shapes.

We achieve the same number of the original contour dominant points (M_0) for almost all rotations. This fact is due to some limitations of DPIL and ACO methods, related to the output dominant points number and depending on the input threshold; in some case we couldn't find a suitable parameter to obtain this value. Hence we report a percentage of occurrence of the same number of M_0 for all the rotations.

The base strategy to weigh the rotation invariance of the proposed techniques is to evaluate the displacement between the position of dominant points of M_0 and the position of the points extracted after the j -th rotation M_j . We compute finally an index I_R by considering the mean variance in points displacement. In table 5.2.3 we show such results.

5.2.2 Scale

The scale invariance is studied similarly to sec.5.2.1. The original (i.e size = 1.0) contour F_0 of the object is shrunk and enlarged. In our experiment the coordinates

5.2. AFFINE TRANSFORMATION TEST

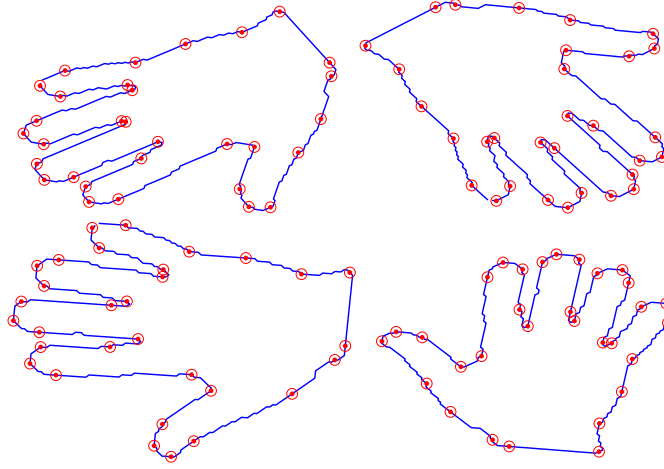


Figure 5.3: Examples of dominant point extraction for rotated hand shape.

	F10		Hand		Head		Fish	
	% occ.	I_R	% occ.	I_R	% occ.	I_R	% occ.	I_R
DPIL	37.5	0.25	50	1.3	50	5.82	66.67	8.21
ACO	100	0.04	100	0.2	100	0.09	100	0.13
GAs	100	9.93	100	12.05	100	5.86	100	1.46

Table 5.2.3: Rotation invariance test.

respect to the mass center of each contour point are altered by applying a transformation matrix depending on a scaling parameter λ . According to the same notation used before the transformation from the original shape contour T to the scaled one T^S is:

$$T^S = \lambda(T - c) + c \quad (5.2.5)$$

All methods are tested with a variable λ parameter in the range [0.5 - 1.5] obtaining 11 transformations in all. Some examples are illustrated in fig. 5.4.

As mentioned above, DPIL and ACOPA couldn't produce the same expected number of dominant points of the base scale shape so the percentage of occurrence of the F_0 model is reported.

Also in this case the displacement between the F_0 dominant points position and the j -th rescaling F_j is evaluated by computing the index I_S that is the mean variance in points displacement. Table 5.2.4 reports scaling results.

5.2.3 Translation

We make some experiments by moving the mass center of each shape, then calculating dominant points for each moved shape and finally comparing the displacement

CHAPTER 5. DOMINANT POINT EXTRACTION TECHNIQUES COMPARISON

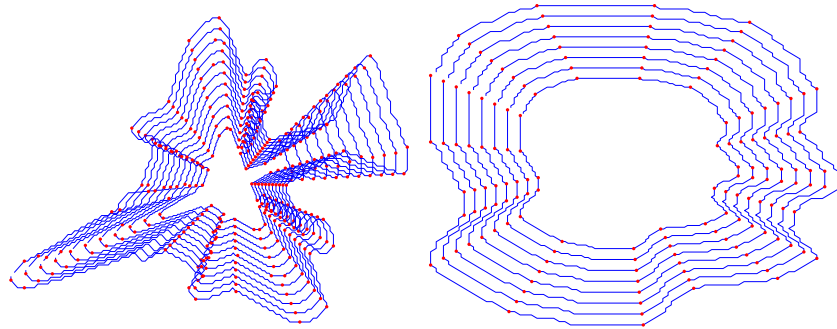


Figure 5.4: Examples of dominant point extraction for scaled shapes.

	F10		Hand		Head		Fish	
	% occ.	I_S	% occ.	I_S	% occ.	I_S	% occ.	I_S
DPIL	81.82	1.91	90.91	1.51	72.73	0.00	81.82	1.16
ACO	90.91	0.23	100	0.40	100	0.20	100	3.98
GAs	100	0.05	100	12.74	100	0.25	100	5.98

Table 5.2.4: Scale invariance test.

of dominant points index are made. Any particular variation in dominant points placement is observed. In fact in the case of DPIL the position of each dominant point is always the same, keeping in mind its deterministic behavior; in the case of GAPA and ACOPA we have to deal with heuristic methods, that means that in every test it is possible to obtain different results, even if the mean variance of many executions is very low (about 0.15 positions). This results is more observable than in other transformations - rotation and scaling - because the three methods work with the relative position of contour points.

5.3 Noise sensitivity test

In this section we evaluate the behavior of polygonal approximation obtained with dominant points extracted by contours in presence of gaussian noise; we test both the E_2 error trend and the relative position of dominant points in noisy contours compared to the original ones, i.e. without noise.

5.3.1 Noise contour generation

We generate automatically noisy images. To accomplish this task we add gaussian noise to the original image by applying a filter characterized from zero mean and a standard deviation σ so that $0 \leq \sigma \leq 1$. By adding random noise we try to obtain a certain number of noisy points connected to the original contour of the current examined image. For this reason we repeat the noise adding operation for several times until an high number of "spurious" points are generated. For example in fig. 5.5 we show the original image and the resulting images obtained by adding repeatedly for 5 times a gaussian noise with $\sigma = 0.1$. The final image is obtained by filling the internal contour pixels.

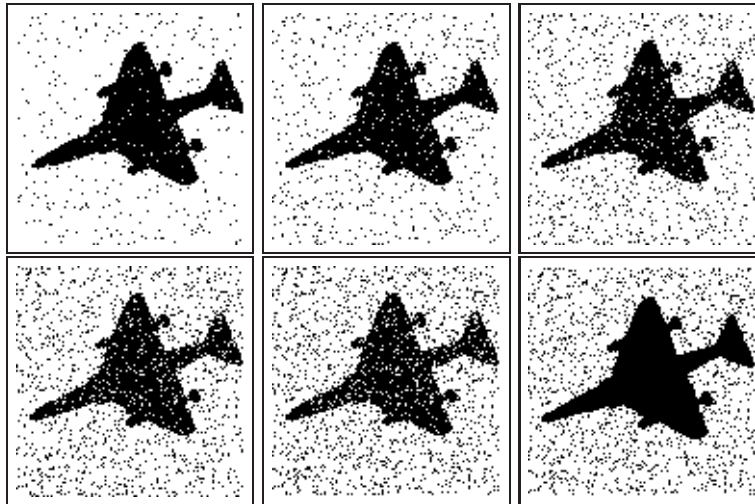


Figure 5.5: Examples of noise addition ($\sigma = 0.1$, 5 repetition). The final image is obtained by filling the internal contour pixels.

We make some manipulations to the noisy image so that we get a noisy contour. In particular we extract the original image contour and we locate the new 4-adjacent neighbors (see the shape difference in fig. 5.6). This new set of points replace the original ones so that our algorithms work on the new noisy contour.



Figure 5.6: Difference between original and noisy f10 shape.

5.3.2 Quadratic Error E_2 testing.

Taken an original, i.e. without additive noise, image, we compute dominant points storing their position in the contour. On the same image we get the E_2 error also. After this, we apply an additive noise, as described above, with a σ varying between 0.0025 to 0.20 with a step of 0.0025 for a total of 80 samples, adding it as described in sec. 5.3.1. Lets denote with N_σ this number. For every iteration we consider the polygonal approximation error and the dominant points position. Taking into account the random placement of noise additive points every test is repeated several times depending on the computational speed of each technique. Let N_R the number of these repetitions. According to the previous notations, we use the following indicator:

$$E_{2A} = \frac{1}{N_\sigma} \frac{1}{N_R} \sum_{i=1}^{N_\sigma} \sum_{k=1}^{N_R} \left(\frac{E_{2i}}{A_i} * 100 \right) \quad (5.3.6)$$

where E_{2i} is the quadratic error and A_i is the area of the image; both values are referred to the experiment made with σ_i for each of the N_R repetitions.

1. DPIL

The DPIL algorithm runs with a fixed input parameter set to 0.5. Due to the random placement of noisy points in the “new” contour, the number of dominant points found may change in every run. We set $N_R = 50$ times, computing the E_{2A} value and the mean dominant point number produced.

2. DPIL fixed number

In order to test the dominant point extraction techniques in the same condition we use a modified version of DPIL that attempts to find a given number of dominant points, by applying a fine variation of the input parameter so that we can obtain the desired, or at least the nearest, dominant point number. We set 30 as dominant point number. In most cases we obtain the exact or a very near value. In some cases, to be exact in 13 experiments, this value is unacceptable; we are talking about a reasonable percentage of $13/80 * 100 = 16.25\%$. In any case, such experiments are excluded from the mean calculation. As for the experiment described we calculate the average E_{2A} error and the number of dominant points obtained.

5.3. NOISE SENSITIVITY TEST

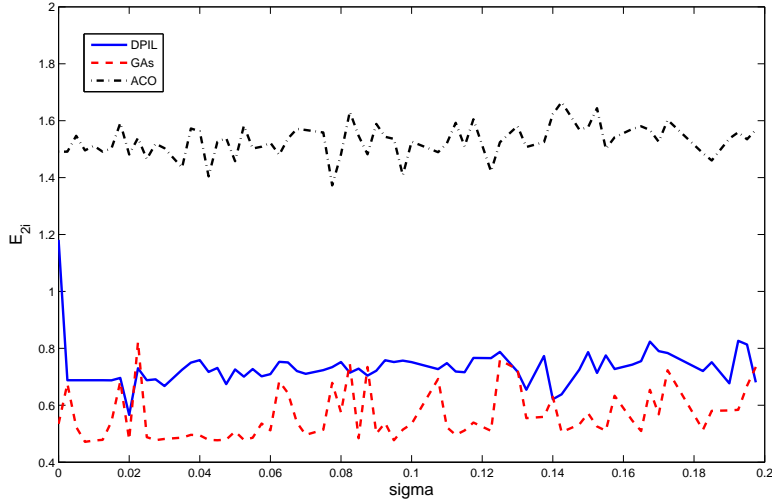


Figure 5.7: E_{2i} trend for the proposed algorithms.

3. ACOPA and GAPA

We repeat the same test made in the case of the DPIL. Considering that GAs give always the desired number of dominant points we compute the average E_{2A} error only; in the case of ACO we set the input parameter to 10 getting, more or less, the required number of dominant points. We set, for computational reasons, 8 repetitions.

In fig. 5.7 the plot of the error E_{2A} is shown, while experimental results are summarized in table 5.3.5.

		DPIL, 0.5	DPIL, 30 dp	ACOPA	GAPA
E_{2A}	avg.	0.2850	0.7318	1.5292	0.5623
	var.	0.0051	0.0032	0.0031	0.0079
# of dp.	$\sigma = 0$	43	30	32	30
	avg.	46.6880	29.6817	31.6866	30
	var.	1.5042	0.5884	0.4306	0

Table 5.3.5: Noise sensitivity test.

It's clearly visible that GAs is the best method, both considering that we can obtain the desired input dominant point number and evaluating the E_{2A} error. Besides the previous testing we also compute relative position of the dominant points located in noisy image with respect to their position on the original image; in table 5.3.6 each row reports the average sum of the distances of each detected

CHAPTER 5. DOMINANT POINT EXTRACTION TECHNIQUES COMPARISON

		DPIL	ACOPA	GAPA
sum of moved	avg.	2.3207	7.9714	6.9705
dp. distance	var.	5.1733	10.3515	24.0278

Table 5.3.6: Dominant points relative position: the reported values are related to the sum of the distance of each dominant point compared to the original dominant point position, i.e. with $\sigma = 0$.

dominant point respect to its position in the original, i.e. with $\sigma = 0$, contour, evaluated in the variation range of σ . The test is made with 30 dominant points.

From the analysis of results, it is possible to notice the excellent behavior of DPIL compared to other methods.

5.4. ANALYSIS

	PROs	CONs
DPIL	Very fast Deterministic Lowest Noise Sensitivity Translation invariance	Lowest Accuracy in polyg. approx. Threshold dependent d.p. #
ACOPA	Good Accuracy in polyg. approx. Error bound limit Rotation Invariance	Very slow Non deterministic
GAPA	Desired d.p.# Highest accuracy in polyg. approx. (*)	Slow Non Deterministic
(*) even in noisy conditions.		

Table 5.4.7: PROs & CONs of proposed techniques

5.4 Analysis

The performance of the three presented methods are satisfactory. ACOPA and GAPA are characterized by a lower global error than DPIL. Indeed DPIL is an iterative method regulated by a stop criterion while ACOPA and GAPA tend to sub-optimal solutions achievement.

This fact leads necessarily to an high computational time, especially in the case of ACOPA. Considering each technique features DPIL may be suitable for real time application, in which is more important the reactivity and less the precision, while ACOPA and GAPA are more appropriate for an off-line work.

By experimental results we proved a good invariance respect to affine transformation of the three proposed techniques. If we compare the mean variance in dominant points position respect to the number of contour points we reach a 1% medium displacement for rotation and a 0.5% medium displacement for scaling. A similar behavior is noted in the case of noisy images also.

Based on described experiments a quick resume of the analyzed dominant point extraction techniques, showing strong and weakness points, is presented in table 5.4.7.

6

Shape matching by dominant points

In previous chapters we described the studied methods for dominant point extraction.

As told in the introduction there are many fields in which dominant points can be applied. In this chapter we propose useful applications of dominant points to the field of image retrieval by present two different approaches to the use of dominant points.

In section 6.1 a novel method of image comparison, overlapping the curves and calculating the degree of alignment as a measure of similarity, is introduced. We compare both “manual” dominant point extraction and “automatic”, i.e. with the algorithm proposed in section 4.1, showing that the performance are even better than human point localization.

In section 6.2 we adapt a technique described in [75] by using dominant points instead of the proposed constant sampling. Also in this case the performance are greatly improved.

6.1 Shape outlines matching

Our approach to find the optimal correspondence between 2D curves relies on using the intrinsic properties of the curves in an energy minimization framework by overlapping the curves and calculating the degree of alignment as a measure of similarity of the curves.

6.1.1 The mathematical formulations

In this section we write down the mathematical results that will provide the guidelines of the method described in section 6.1.2.

Definition 6.1.1 (Image contour or curve). *A set I of n points*

$$I : \{P_i \equiv (x_i, y_i) \in R^2, i = 1, \dots, n \mid P_1 = P_n;$$

$$P_i \neq P_j, i = 1, \dots, n-2, j = i+1, \dots, n-1\}$$

is said to be an image contour or curve if, for any line segment with endpoints two successive points in I , the following

$$\begin{aligned} \overline{P_i P_{i+1}} \cap \overline{P_j P_{j+1}} &= \emptyset, i = 1, \dots, n-3, \\ j = i+2, \dots, n-1 \\ \overline{P_i P_{i+1}} \cap \overline{P_{i+1} P_{i+2}} &= P_{i+1}, i = 1, \dots, n-2 \end{aligned}$$

holds.

We can state that an *image contour* or *curve* I is the set of the successive vertices of a closed polygon that is not self intersecting. In the sequel we denote by \bar{I} the closed polygon associated with I .

Definition 6.1.2 (Translation). *Given $(x_t, y_t) \in R^2$, a translation mapping $T : R^2 \rightarrow R^2$ is defined by*

$$T : \begin{cases} x' = x + x_t \\ y' = y + y_t. \end{cases} \quad (6.1.1)$$

Definition 6.1.3 (Rotation). *Given $(x_r, y_r) \in R^2$ and $\theta \in [0, \pi]$, a rotation mapping $R : R^2 \rightarrow R^2$ through the angle θ and the rotation center (x_r, y_r) is defined by*

$$R : \begin{cases} x' = \cos(\theta) * (x - x_r) - \sin(\theta) * (y - y_r) + x_r \\ y' = \sin(\theta) * (x - x_r) + \cos(\theta) * (y - y_r) + y_r. \end{cases}$$

Definition 6.1.4 (Scaling). *Given two real numbers s_1 and s_2 , a scaling mapping $S : R^2 \rightarrow R^2$ is defined by*

$$S : \begin{cases} x' = s_1 * x \\ y' = s_2 * y. \end{cases} \quad (6.1.2)$$

6.1. SHAPE OUTLINES MATCHING

Proposition 6.1.5 Let I' be a curve obtained from a curve I by a chain application $S * R * T$ of a translation T , a rotation R and a scaling S with parameters (x_t, y_t) , (θ, x_r, y_r) and $(s_1 = s_2 = s)$, respectively. Let I^b be the image contour obtained by the chain application to I' of the mapping map_1

$$map_1 : \begin{cases} x' &= x + x_c - x'_c \\ y' &= y + y_c - y'_c \end{cases} \quad (6.1.3)$$

where $C \equiv (x_c, y_c)$ and $C' \equiv (x'_c, y'_c)$ denote the centroid of I and I' , respectively. Further, let I'' be obtained by the application to I^b of the mappings map_2 and map_3 given by

$$map_2 : \begin{cases} x' &= \alpha * (x - x_c) - \beta * (y - y_c) + x_c \\ y' &= \beta * (x - x_c) + \alpha * (y - y_c) + y_c \end{cases} \quad (6.1.4)$$

$$map_3 : \begin{cases} x' &= \bar{s} * (x - x_c) + x_c \\ y' &= \bar{s} * (y - y_c) + y_c \end{cases} \quad (6.1.5)$$

where

$$\alpha = \frac{(x_1 - x_c)(x_0 - x_c) + (y_1 - y_c)(y_0 - y_c)}{((x_1 - x_c)^2 + (y_1 - y_c)^2)^{0.5}((x_0 - x_c)^2 + (y_0 - y_c)^2)^{0.5}}$$

$$\beta = \frac{(y_1 - y_c)(x_0 - x_c) - (x_1 - x_c)(y_0 - y_c)}{((x_1 - x_c)^2 + (y_1 - y_c)^2)^{0.5}((x_0 - x_c)^2 + (y_0 - y_c)^2)^{0.5}}$$

$$\bar{s} = \frac{((x_1 - x_c)^2 + (y_1 - y_c)^2)^{0.5}}{((x_0 - x_c)^2 + (y_0 - y_c)^2)^{0.5}}$$

and $P_{max} \equiv (x_1, y_1)$ and $Q_{max} \equiv (x_0, y_0)$ the points in I and I^b , respectively, with maximum distance from (x_c, y_c) , then $I'' = I$.

Proof. The chain application of $S * R * T$ maps the centroid C of I into the centroid C' of I' and maps P_{max} into P'_{max} with P'_{max} the point in I' , with maximum distance from C' . The application of map_1 maps, through a translation, C' into C and the application of map_2 cancels the rotation done in R . Finally, map_3 maps P'_{max} into P_{max} by a scaling. Hence, we can state $I = I''$. \square

Proposition 6.1.6 Let I' be a curve obtained from a curve I by a chain application $S * R * T$ of a translation T , a rotation R and a scaling S with parameters (x_t, y_t) , (θ, x_r, y_r) and (s_1, s_2) , respectively. Let I^b be the curve obtained by the application to I' of the mapping map_1

$$map_1 : \begin{cases} x' &= x + x_c - x'_c \\ y' &= y + y_c - y'_c \end{cases} \quad (6.1.6)$$

where $C \equiv (x_c, y_c)$ and $C' \equiv (x'_c, y'_c)$ denote the centroid of I and I' , respectively. Further, let I^{bb} be obtained by the application to I^b of the mapping map_2

$$map_2 : \begin{cases} x'(\theta) &= \cos \theta * (x - x_c) \\ &- \sin \theta * (y - y_c) + x_c \\ y'(\theta) &= \sin \theta * (x - x_c) \\ &+ \cos \theta * (y - y_c) + y_c \end{cases} \quad (6.1.7)$$

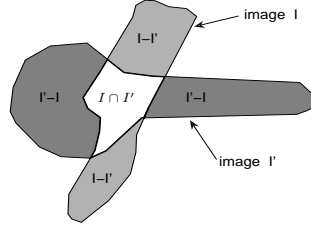


Figure 6.1: The dark regions contribute to define ind_1

with $\theta \in [0, 2\pi]$. Further, let I'' be obtained by the application to I^{bb} of the mapping map_3 given by

$$map_3 : \begin{cases} x'(\theta) &= \bar{s}_1 * (x(\theta) - x_c) + x_c \\ y'(\theta) &= \bar{s}_2 * (y(\theta) - y_c) + y_c \end{cases} \quad (6.1.8)$$

where

$$\bar{s}_1 = \frac{x_0}{x_1}, \quad \bar{s}_2 = \frac{y_0}{y_1}$$

with $P_x \equiv (x_1, y_c)$ and $P_y \equiv (x_c, y_1)$ the intersection points, with the smallest distance from (x_c, y_c) , of the boundary of \bar{I} with the straight lines $y = y_c$ and $x = x_c$, respectively, and $Q_x \equiv (x_0, y_c)$, $Q_y \equiv (x_c, y_0)$ the intersection points of the boundary of \bar{I}^{bb} with the same straight lines. Then, there exists $\theta^* \in [0, 2\pi]$ such that $I'' = I$.

Proof. The chain application of $S * R * T$ maps the centroid C of I into the centroid C' of I' . The application of $T * R$ can be rewritten as an application of a rotation R' through a suitable angle θ' with the centroid of I as rotation center and of a suitable translation T' . The transformation map_1 maps C' into C . The application of map_2 with angle $-\theta'$ and the scaling in map_3 moves I'' to the original curve. \square

In order to compare shapes, we give now a definition that will provide an overlapping index between two curves I and I' . Such an index measures the degree of alignment or overlapping of two curves and it does represent a measure of similarity between two shapes.

Definition 6.1.7 (Overlapping index 1). Let $I \equiv (P_i, i = 1, \dots, n)$ and $I' \equiv (P'_j, j = 1, \dots, m)$ be two curves and \bar{I} and \bar{I}' the associated polygons with vertices I and I' , respectively. Let

$$ind_1(I, I') = \max \left(\frac{\text{area}(\bar{I} - \bar{I}')}{\text{area}(\bar{I})}, \frac{\text{area}(\bar{I}' - \bar{I})}{\text{area}(\bar{I}')} \right)$$

where $\text{area}(\bar{I})$ stands for the area of the set \bar{I} , then $ind_1(I, I')$ will evaluate the overlapping degree between I and I' .

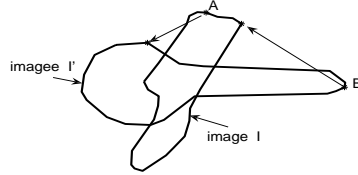


Figure 6.2: The maximum of the distances of A from I' and B from I gives ind_2 .

Fig.6.1 explains the definition, the slightly dark region and the darker region contribute to define ind_1 . That value goes to zero as the areas of both regions reduce to zero, i.e. the regions are identical. In the sequel we shall make use of a second definition of the *overlapping index* that will turn out to be computationally less expensive.

Definition 6.1.8 (Overlapping index 2). Let $I \equiv (P_i, i = 1, \dots, n)$ and $I' \equiv (P'_j, j = 1, \dots, m)$ be two image contours and \bar{I} and \bar{I}' the associated polygons. The overlapping index ind_2 is defined as

$$ind_2(I, I') = \max\left(\max_{i=1, \dots, n} (\text{distance}(P_i, \mathfrak{F}(\bar{I}'))), \max_{j=1, \dots, m} (\text{distance}(P_j, \mathfrak{F}(\bar{I})))\right)$$

where $\mathfrak{F}(S)$ denotes the boundary of the set S and *distance* refers to the euclidean distance.

In Fig.6.2 $A \in I'$ and $B \in I$ are the points with the maximum distance from I and I' respectively. The maximum of these distances gives ind_2 . Note that given two image contours I and I' such that $ind_1(I, I') = 0$ then the associated polygons fully overlap, i.e., $\bar{I} = \bar{I}'$; that is not true if $ind_2(I, I') = 0$. The following example

$$\begin{aligned} I &\equiv \{(-1, 1), (0, 2), (0, 0.5), (1, 1), (1, -1), (0, -2), \\ &\quad (0, -0.5), (-1, -1), (-1, 1)\} \\ I' &\equiv \{(-1, 1), (0, 0.5), (0, 2), (1, 1), (1, -1), (0, -2), \\ &\quad (0, -0.5), (-1, -1), (-1, 1)\} \end{aligned}$$

shows that. Whenever we don't make reference to a specific definition, by $ind(I, I')$ we shall denote any overlapping index between I and I' .

6.1.2 The alignment method

The mathematical definitions given in section 6.1.1 suggest the main features of our method: to compare two shapes I and I' we assume that I' is derived from I

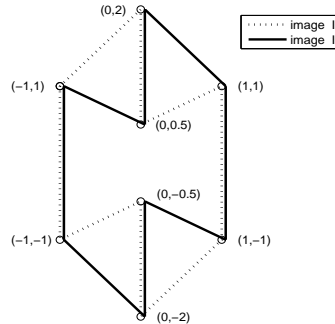


Figure 6.3: Image contours I and I' with $ind_2(I, I') = 0$ but $\bar{I} \neq \bar{I}'$

by some linear transformations. That is in order to compare two curves I and I' we proceed as follows:

1. overlap by a translation I and I' so that $C = C'$, with C and C' being the centroids of I and I' , respectively;
2. carry out a rotation on I' with rotation center C ;
3. perform a scaling;
4. calculate the overlapping index $ind(I, I')$.

We assume that two cases can take place: I and I' may differ either (a) by a equal scaling along both axis, either (b) by non equal scaling along the axis.

To handle cases (a) and (b) we shall consider two algorithms OA_1 and OA_2 ; in the first case we rewrite the four steps sketched above as

Algorithm OA_1 (single scaling)

1. find centroids C and C' and translate I and I' so that their centroids are the axis origin $O \equiv (0, 0)$;
2. find points P_{max} and P'_{max} in I and I' , respectively, whose distance from O is maximum;
3. rotate I by the angle $(P_{max} - O)^T(P'_{max} - O) / (\|P_{max}\| \|P'_{max}\|)$;
4. scale I by $(\|P_{max}\| / \|P'_{max}\|)$.
5. calculate $ind(I, I')$.

In the second case we rewrite the four steps as follows:

Algorithm OA_2 (double scaling)

1. find centroids C and C' and translate I and I' so that their centroids are the axis origin ($O \equiv (0, 0)$); let I^a and I^b be the new curves;
2. let $\theta(i)$ $i = 1, \dots, n$ a vector of n points equally spaced in $[0, 2\pi]$;

6.1. SHAPE OUTLINES MATCHING

3. find $\theta(i^*)$ such that

$$ind(I^a, I''(\theta(i^*))) = \min_i ind(I^a, I''(\theta(i)))$$

with $I''(\theta)$ obtained as result of a chain application of 6.1.7, and 6.1.8 to I^b .

Shape representation and modeling

In our approach shapes are represented by their contours.

Given a set S of assigned shapes in order to find all the similar shapes in S or that share a similar contour, we consider a model M which all the shapes of S must be compared to.

Such a model is made up of one or more curves $m_i, i = 1 \dots m$ and we measure the shape similarity of an element I of S to M by calculating the overlapping indices between I and each element m_i of M .

The smallest value of these indices is assumed as similarity measure of I to the model and is denoted by $ind(I, M)$:

$$ind(I, M) \equiv \min_{i=1 \dots m} ind(I, m_i)$$

The shape models have been drawn from real image contours by detecting the most significant points of the contours and joining them by straight lines.

Shape matching and timing analysis

We now give a remark allowing to reduce substantially the computation of the overlapping index 2.

In comparing an element I of S to all elements in M , the calculation of $ind_2(I, M)$ has to be carried out by calculating first $ind_2(I, m_1)$ and then $ind_2(I, m_i), i = 2, \dots m$.

Once a $ind_2(I, m_j), j \in \{1, \dots, m\}$ has been carried out we can start to compute $ind_2(I, m_i), i > j$. However, if for a point $(\tilde{x}, \tilde{y}) \in I$ the overlapping index $ind_2((\tilde{x}, \tilde{y}), \mathfrak{F}(m_i))$ is greater than or equal to $ind_2(I, m_j)$ then $ind_2(I, m_i)$ will be greater or equal to $ind_2(I, m_j)$. As a consequence we don't need to complete the calculation of $ind_2(I, m_i)$. This feature will be present in the implementation of our algorithms.

Finally, for algorithm *OA_1* we shall consider an option that carries out a sort of suboptimization procedure. In comparing contour I to the contour of each element m_j of M we perform rotations of m_j through angles $\theta(i)$ and calculate the best $ind(I, m_j)$.

We proceed much the same as in algorithm *OA_2*.

The *overlapping* algorithm *OA_1* is now written in *pseudo-code* language. It calls four procedures: *find_centroid*, *find_max_point*, *rotate*, and *find_ind* and, eventually, it uses *option_A*

Algorithm, single scaling (OA_1) $m \equiv$ number of contours in model M .

input_model: [x_s, y_s, n_s, m];

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

```

[xb,yb]=find_centroid (x_s,y_s,n_s,m);
translate model centroids to (0,0)
for j = 1 : m
    x_s(j,:)=x_s(j,:)-xb(j); y_s(j,:)=y_s(j,:)-yb(j);
end
[x1,y1,n1]=input_data I ∈ S;
[xb1,yb1]=find_centroid(x1,y1,n1,1);
translate image centroid to (0,0)
x1=x1-xb1; y1=y1-yb1
nn1=find_max_point(x1,y1);
best=∞;
for i_m=1:m
    n=n_s(i_m);
    x=x_s(i_m,1:n); y=y_s(i_m,1:n);
    best_buf=find_ind(x,y,n,x1,y1,n1,best)
    if best_buf < best best=best_buf; end
    nn=find_max_point(x,y);
    [xx1,yy1,sca]= rotate_scale(x1,y1,x1(nn1),
        y1(nn1),x(nn),y(nn))
    best_buf=find_ind(x,y,n,sca*xx1,sca*yy1,n1,best);
    if best_buf < best best=best_buf; end
    Option A
end
output: best.

```

We give the main features of the procedures called by the *OA-1* algorithm.

Find_centroid. This function reads the contours of the m images that made up the model. Then it calculates and returns their centroids values. The centroid (x_c, y_c) of $I \equiv \{P_i \equiv (x_i, y_i), i = 1, \dots, n\}$ is given by

$$\begin{aligned}
 x_c &= \frac{1}{6A} \sum_{i=0}^{n-1} (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \\
 y_c &= \frac{1}{6A} \sum_{i=0}^{n-1} (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \\
 A &= \frac{1}{2} \sum_{i=0}^{n-1} (x_i y_{i+1} - x_{i+1} y_i).
 \end{aligned}$$

Find_max_point. This function reads the coordinates of a contour I and, then, finds and returns the index of the element of I whose distance from the $(0, 0)$ is the largest one.

Rotate_scale. This function reads the coordinates of a contour I , then applies

6.1. SHAPE OUTLINES MATCHING

to I a transformation so that it is rotated by the angle

$$\theta = \frac{(x(nn), y(nn))^T (x1(nn1), y1(nn1))}{\|(x(nn), y(nn))\| \|(x1(nn1), y1(nn1))\|}$$

and scaled by

$$sca = \frac{\|(x(nn), y(nn))\|}{\|(x1(nn1), y1(nn1))\|}.$$

It returns the transformed I .

Find_ind. This function reads the coordinates of a contours I and I' and a real value $best$, then calculates and returns the overlapping index $ind(I, I')$ between the two contours. If $ind(I, I')$ is $ind_2(I, I')$ the procedure is stopped as soon as any point P of I has a distance from $\mathfrak{F}I'$ greater than $best$ either any point P of I' has a distance from $\mathfrak{F}I$ greater than $best$.

Option_A for OA_1. It consists of the following program segment:

```
% option: find the best rotation with rot_no a pos integer
teta=[0 : 2 * pi / (rot_no - 1) : 2 * pi];
x_start=x; y_start=y;
for j= 1: rot_no
    x=cos(teta(j))*x_start-sin(teta(j))*y_start;
    y=sin(teta(j))*x_start+cos(teta(j))*y_start;
    best_buf=find_ind(x, y, n, scal * x1,
                    scal * y1, n1, best);
    if best > best_buf best=best_buf; end
end
```

The *overlapping* algorithm OA_2 is now written in *pseudo-code* language. It calls procedures already introduced for OA_1 and $find_zero$.

Algorithm, double scaling (OA_2)

```
input_model: [x_s, y_s, n_s, m];
[xb, yb]=find_centroid(x_s, y_s, n_s, m);
translate model centroids to (0,0)
for j=1:m
    x_s(j,:)=x_s(j,:)-xb(j); y_s(j,:)=y_s(j,:)-yb(j);
end
[x1, y1, n1]=input_data I ∈ S;
[xb1, yb1]=find_centroid(x1, y1, n1, 1);
translate image centroid to (0,0)
x1=x1-xb1; y1=y1-yb1
best=∞;
for i_m=1:m
    n=n_s(i_m);
    x=x_s(i_m, 1:n); y=y_s(i_m, 1:n);
```

```

best_buf=find_ind(x,y,n,x1,y1,n1,best)
  if best_buf < best best=best_buf; end
x1_0=find_zero(x1,y1,n1);
y1_0=find_zero(y1,x1,n1);
  find the best rotation
teta=[0:2*pi/(rot_no-1):2*pi];
x_start=x; y_start=y;
for j= 1:rot_no
  x=cos(teta(j))*x_start-sin(teta(j))*y_start;
  y=sin(teta(j))*x_start+cos(teta(j))*y_start;
  x_0=find_zero(x,y,n);
  y_0=find_zero(y,x,n);
  scal1=abs(x_0/x1_0);
  scal2=abs(y_0/y1_0);
  best_buf=find_ind(x,y,n,scal*x1,scal*y1,n1,best);
  if best > best_buf best=best_buf; end
end
end
output: best.

```

Find_zero. This reads the coordinates of a contour $I \equiv (x, y)$ then calculates and returns the smallest distance from $O \equiv (0, 0)$ to the intersection of the boundary of I with the x axis.

6.1.3 Experimental results and conclusions

In this section we illustrate our shape recognition approach with some experiments.

We have evaluated the matching ability of the proposed method by indexing two databases of shapes, both constructed by Kimia's group [76, 77].

The two proposed algorithms, OA_1 and OA_2 , have been first tested on a set S^1 of 216 images $I_i, i = \dots, 216$ (see table A.0.1 in appendix A).

The database consists of twelve shapes each from eighteen families (or classes) for a total of 216 shapes.

In order to simplify the presentation of our procedure, we first assume that we want to identify the shapes of just one family $F \subset S^1$; for instance, we want to identify all screwdrivers. We assume F made up of $n_F = n_2 - n_1 + 1$ images and $F \equiv \{I_j, j = n_1, \dots, n_2\}$, where I_j is the j^{th} image in S^1 .

The indexing process is achieved by first defining a model M for the family and then by comparing each element of S with the model and calculating the overlapping indices $ind(I_j, M), j = 1, \dots, 216$.

The overlapping indices are then sorted in ascending order and stored as a vector named res_sort . The algorithm is considered fully effective if the overlapping indices of the images in F are in the first n_F positions of res_sort . Further, whenever this does not take place, we consider how much the positions of the overlapping indices of the images in F which are not in the first n_F positions are far from the n_F

6.1. SHAPE OUTLINES MATCHING

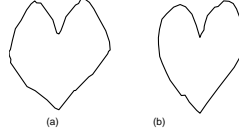


Figure 6.4: A model made up of two image contours.

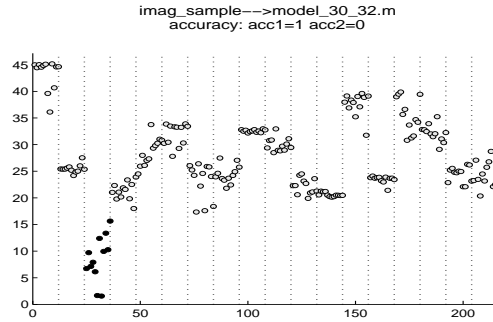


Figure 6.5: Plot of the overlapping indices of all images.

position. Hence, we evaluate the effectiveness of the algorithm by defining two accuracy values acc_1 and acc_2 .

Given

$$\bar{F} = \{j | n_1 \leq j \leq n_2, pos(j) \leq n_F\},$$

$$\underline{F} = \{j | n_1 \leq j \leq n_2, pos(j) > n_F\},$$

$$acc_1 \equiv \frac{card(\bar{F})}{n_F},$$

$$acc_2 \equiv \frac{\sum_{j \in \underline{F}} (pos(j) - n_F)}{card(\underline{F})}$$

where by $pos(j)$ and $card(\bar{F})$ we denote the position of image I_j in res_sort and the cardinality of \bar{F} respectively. In Fig.6.5 we show the results obtained by comparing the set S^1 with the model made up of the two images given in Fig.6.4.

The filled circles represent contours in the family to be identified; the empty ones represent those not in the family. In such a case all the contours in the family are identified ($acc_1=0, acc_2=0$).

We now present the overall results of our experiments.

The set S^1 is partitioned into 18 families $F_i, i = 1, \dots, 18$: bones, glasses, and so on.

Since our task is to identify the shapes of each family (or class), we define four sets $M_{1,1}, M_{1,2}, M_{2,1}$, and $M_{2,2}$ each made up of 18 models, one for each family of images to identify.

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

For each family, both the models in $M_{1,1}$ and $M_{1,2}$ consist of one image model; each model in $M_{1,1}$ is an approximated contour chosen such that its mean overlapping ind_1 calculated with respect to all the images of the family to be identified provides the minimum value. In the same way we have chosen the models in $M_{1,2}$, but calculating ind_2 . That is, only one image model has been selected as the most representative approximated contour of a family with respect to ind_1 or to ind_2 .

The models in $M_{2,1}$ and $M_{2,2}$ consist of from one to two images. They have been chosen trying to minimize the overlapping indices ind_1 and ind_2 . In our case we have

$$\begin{aligned} M_{1,1} &\equiv \{3, 13, 25, 44, 57, 67, 79, 89, 103, 109, 129, \\ &\quad 142, 145, 168, 173, 187, 194, 207\} \\ M_{1,2} &\equiv \{1, 13, 25, 37, 57, 67, 84, 89, 102, 109, 123, \\ &\quad 135, 145, 168, 173, 187, 195, 207\} \end{aligned}$$

and

$$\begin{aligned} M_{2,1} &\equiv \{3, 13, 25, 44, 54_56, 67_68, 75_82, 89, 103, \\ &\quad 109, 128_129, 142, 146_152, 168, 173_180, \\ &\quad 186_190, 193_197, 208_216\} \\ M_{2,2} &\equiv \{1, 13, 30_32, 38_42, 49_52, 67, 75_84, \\ &\quad 89, 102, 109, 123_125, 135, 147_155, 168, \\ &\quad 171_175, 187, 195_201, 205_207\}. \end{aligned}$$

Each element in the model sets corresponds with the shape number in the image database. We run algorithms OA_1 and OA_2 with various settings:

1. the overlapping index can be ind_1 or ind_2 ;
2. the contour image of S may be transformed by a chain application of the map $R * S * T$ with R a rotation through an angle θ chosen at random in the interval $[0, \pi]$, S a scaling and T a translation;
3. the option A in OA_1 may be or not be present; whenever is present $rot_no = 40$;
4. OA_2 is run with $rot_no = 40$.

The scaling S is assumed equal along both axis for OA_1 , while is not equal in the case of OA_2 .

The results of the numerical experiments referring to OA_1 are summarized in tables 6.1.1 and 6.1.2; while those referring to OA_2 are in 6.1.3 and 6.1.4. The acc_1 and acc_2 appearing in the tables are mean values of the values calculated for each family.

We now report the results gathered by testing algorithms OA_1 and OA_2 for another set of images S^2 , again created by Kimia's group [76, 77](see table A.0.2 in appendix A). There are 99 shapes in this database from nine families, which were

6.1. SHAPE OUTLINES MATCHING

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	0	$M_{1,1}$	0.83	8.23	23
0	1	40	$M_{1,1}$	0.86	8.18	596
0	2	0	$M_{1,2}$	0.86	6.64	3
0	2	40	$M_{1,2}$	0.89	6.30	18
1	1	0	$M_{1,1}$	0.71	15.47	21
1	1	40	$M_{1,1}$	0.79	10.28	587
1	2	0	$M_{1,2}$	0.75	16.12	2
1	2	40	$M_{1,2}$	0.87	5.84	18

Table 6.1.1: Results of algorithm OA_1 with models of one image (set S^1).

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	0	$M_{2,1}$	0.93	6.92	35
0	1	40	$M_{2,1}$	0.89	7.37	926
0	2	0	$M_{2,2}$	0.97	10.83	4
0	2	40	$M_{2,2}$	0.92	3.96	25
1	1	0	$M_{2,1}$	0.79	13.71	32
1	1	40	$M_{2,1}$	0.86	10.07	919
1	2	0	$M_{2,2}$	0.80	21.67	3
1	2	40	$M_{2,2}$	0.92	8.11	24

Table 6.1.2: Results of algorithm OA_1 with models of up to two images (set S^1).

collected from a variety of sources: fish, and sea-animals from Farzin Mokhtarian, "greebles" from Mike Tarr, and tools from Stan Sclaroff. Eleven shapes are included from each family so that shapes have variations in form, and as well transformations such as occlusion, articulation and missing parts.

The numerical experiments have been carried out as for the set S^1 . In such a case the model set are given by

$$M_{1,1} \equiv \{3, 17, 30, 39, 51, 56, 72, 80, 99\}$$

$$M_{1,2} \equiv \{3, 17, 32, 38, 53, 56, 76, 87, 98\}$$

and

$$M_{2,1} \equiv \{1_8, 17, 30_31, 34_44, 48_54, 58_66, 72, 84_88, 93_97\}$$

$$M_{2,2} \equiv \{1_8, 17_21, 32, 38, 48_53, 57_58, 76, 84_86, 91_98\}.$$

An overall analysis of the results given in the tables allows to state that the overlap-

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	40	$M_{1,1}$	0.80	24.20	637
0	2	40	$M_{1,2}$	0.84	14.61	24
1	1	40	$M_{1,1}$	0.69	39.90	637
1	2	40	$M_{1,2}$	0.79	29.48	19

Table 6.1.3: Results of algorithm OA_2 with models of one image (set S^1).

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	40	$M_{2,1}$	0.83	20.31	917
0	2	40	$M_{2,2}$	0.89	8.85	25
1	1	40	$M_{2,1}$	0.72	44.60	989
1	2	40	$M_{2,2}$	0.80	25.50	25

Table 6.1.4: Results of algorithm OA_2 with models of up to two images (set S^1).

ping algorithm is effective in identifying the images of a each family. Furthermore, the results highlight the features of each setting taken into account. We summarize the main points in the following remarks

1. the use of overlapping $index_1$ does produce any clear advantage with respect to that of $index_2$, but the computer execution time is dramatically favorable to the use of $index_2$;
2. option A gives usually best results but is computationally expensive;
3. the use of models made up of more images improves considerably the algorithm efficiency.

Precision-Recall

In order to compare the performance of the algorithms the *precision recall curve* has been used.

Precision and recall are two classical performance evaluation metrics in the field of information retrieval. Precision is a measure of the ability of a system to present only relevant items. Recall is a measure of the ability of a system to present all relevant items.

As described above results of the comparison of shapes are sorted in ascending order and stored in a vector. We have obtained a ranking of database images based on similarity with a given model; a particular item in the ranking is considered as relevant if the image belongs to the class of the model, otherwise it is not relevant.

Building a precision-recall curve requires another important parameter, called *scope*. It is defined as the number of retrieved items from the top of the ranking list.

We have set a scope size value equal to 12 for the set S^1 and 11 for the set S^2 . This choice is due to the fact that in [76] the summary of rank-ordered results

6.1. SHAPE OUTLINES MATCHING

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	0	$M_{1,1}$	0.84	6.33	5
0	1	40	$M_{1,1}$	0.87	7.50	131
0	2	0	$M_{1,2}$	0.86	8.78	0
0	2	40	$M_{1,2}$	0.83	13.80	5
1	1	0	$M_{1,1}$	0.71	10.03	4
1	1	40	$M_{1,1}$	0.84	9.57	131
1	2	0	$M_{1,2}$	0.65	11.14	0
1	2	40	$M_{1,2}$	0.82	13.40	4

Table 6.1.5: Results of algorithm OA_1 with models (mod=1) of one image (set S^2).

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	0	$M_{2,1}$	0.97	4.44	8
0	1	40	$M_{2,1}$	0.94	5.33	232
0	2	0	$M_{2,2}$	0.96	8.44	0
0	2	40	$M_{2,2}$	0.85	15.58	8
1	1	0	$M_{2,1}$	0.86	6.07	8
1	1	40	$M_{2,1}$	0.92	4.72	232
1	2	0	$M_{2,2}$	0.80	11.91	0
1	2	40	$M_{2,2}$	0.84	14.00	6

Table 6.1.6: Results of algorithm OA_1 with models (mod=2) of up to two images (set S^2).

is given only for the first 12 and 11 shapes, respectively. For each family, given a sample shape and the relative rank results within the fixed scope size, precision and recall values are calculated.

Generally recall reaches the value of 1 if all relevant items are located within the s first positions, where s is the scope size value. In our performance test s is set to the class size of each sample; it follows that recall can reach 1 only if all items of the class are found in the rank-ordered results. Otherwise, recall never reaches 1 but it stops to a value $t < 1$. In this case the precision is set to 0 for those interpolated recall values between t and 1.

At the end of the process, for each family, we have interpolated precision values at standard recall values.

By mediating this values we have obtained a final interpolated precision set of values for each method.

In Fig.6.6 and Fig.6.7 the comparative precision recall curves for the set S^1 are shown. Each image is referred to OA_1 and OA_2 algorithm, respectively. Both figures display the comparison between performance of [76], our best result and our worst result. In the same way (Fig.6.8 and Fig.6.9) the precision-recall curves are

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	40	$M_{1,1}$	0.76	13.92	135
0	2	40	$M_{1,2}$	0.73	15.83	4
1	1	40	$M_{1,1}$	0.63	22.33	138
1	2	40	$M_{1,2}$	0.64	29.00	4

Table 6.1.7: Results for algorithm OA_2 with models (mod=1) of one image (set S^2).

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	1	40	$M_{2,1}$	0.84	7.83	240
0	2	40	$M_{2,2}$	0.82	10.24	6
1	1	40	$M_{2,1}$	0.70	16.83	242
1	2	40	$M_{2,2}$	0.71	20.86	8

Table 6.1.8: Results for algorithm OA_2 with models (mod=2) of up to two images (set S^2).

plotted for the set S^2 .

The experimental results show that the proposed method has given satisfying results in terms of accuracy, although a reasonable performance degradation in the OA_2 variation, compared to similar approaches. Such results justify the computational effort required to shapes matching.

Automatic dominant points extraction

In our experiments we have detected the dominant points automatically also. Among the proposed dominant point extraction methods, we use the fastest technique, i.e DPIL (section 4.1). We have repeated the experiment described in the previous section starting by dominant points automatically located. The numerical results are reported in tables 6.1.9-6.1.12 for the first database (S^1) and in tables 6.1.13-6.1.16 for the second database (S^1). The overlapping index is ind_2 .

By matching the results shown in the tables, it is possible to notice that, using the automatic method for creating models, we achieve a very impressive improvement in terms of required computation in spite of a negligible lack in accuracy in few circumstances.

6.1. SHAPE OUTLINES MATCHING

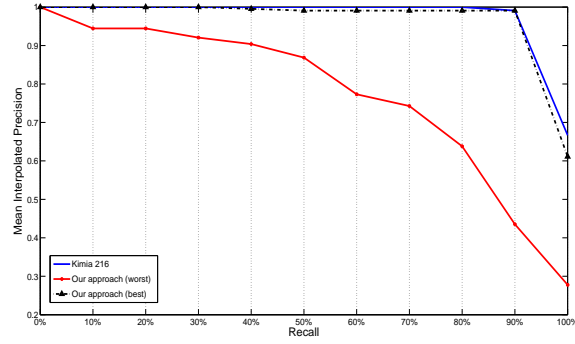


Figure 6.6: Precision-recall curve for algorithm OA_1 and set S^1 .

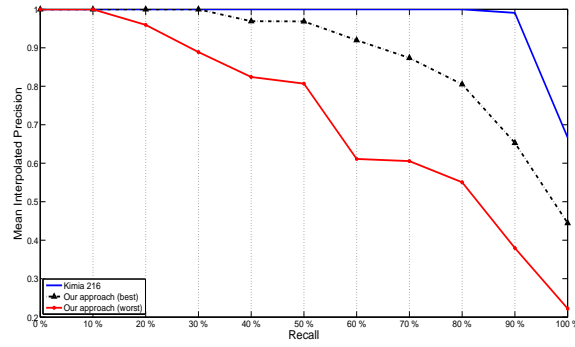


Figure 6.7: Precision-recall curve for algorithm OA_2 and set S^1 .

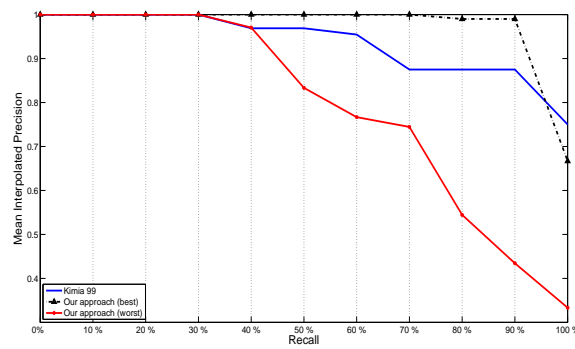


Figure 6.8: Precision-recall curve for algorithm OA_1 and set S^2 .

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

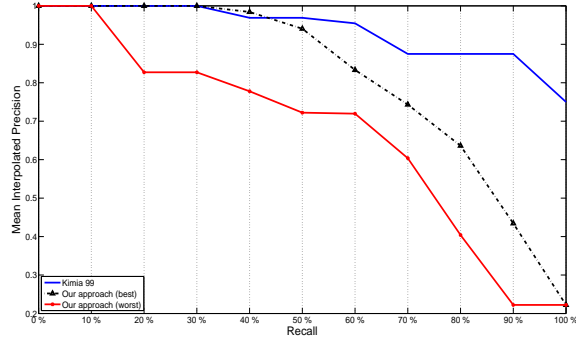


Figure 6.9: Precision-recall curve for algorithm OA_2 and set S^2 .

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	0	$M_{1,2}$	0.86	6.54	0.29
0	2	40	$M_{1,2}$	0.88	6.44	3.00
1	2	0	$M_{1,2}$	0.77	16.71	0.31
1	2	40	$M_{1,2}$	0.87	5.54	3.6

Table 6.1.9: Results for algorithm OA_1 with models of one image (set S^1) created automatically.

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	0	$M_{2,2}$	0.96	8.22	0.35
0	2	40	$M_{2,2}$	0.94	6.64	4.11
1	2	0	$M_{2,2}$	0.85	18.71	0.37
1	2	40	$M_{2,2}$	0.93	5.87	4.59

Table 6.1.10: Results for algorithm OA_1 with models of up to two images (set S^1) created automatically.

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	40	$M_{1,2}$	0.85	17.07	3.37
1	2	40	$M_{1,2}$	0.77	38.35	3.43

Table 6.1.11: Results for algorithm OA_2 with models of one image (set S^1) created automatically.

6.1. SHAPE OUTLINES MATCHING

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	40	$M_{2,2}$	0.87	9.76	5.15
1	2	40	$M_{2,2}$	0.82	19.92	5.34

Table 6.1.12: Results for algorithm OA_2 with models of up to two images (set S^1) created automatically.

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	0	$M_{1,2}$	0.82	9.56	0.8
0	2	40	$M_{1,2}$	0.77	9.74	0.47
1	2	0	$M_{1,2}$	0.74	11.55	0.7
1	2	40	$M_{1,2}$	0.77	9.89	0.49

Table 6.1.13: Results for algorithm OA_1 with models of one image (set S^2) created automatically.

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	0	$M_{2,2}$	0.91	6.89	0.9
0	2	40	$M_{2,2}$	0.89	9.91	1.11
1	2	0	$M_{2,2}$	0.87	6.56	0.9
1	2	40	$M_{2,2}$	0.89	6.66	1.13

Table 6.1.14: Results for algorithm OA_1 with models of up to two images (set S^2) created automatically.

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	40	$M_{1,2}$	0.80	12.30	0.57
1	2	40	$M_{1,2}$	0.66	24.07	1.0

Table 6.1.15: Results for algorithm OA_2 with models of one image (set S^2) created automatically.

transf	overlap. index	rot_no Opt. A	model set	acc_1	acc_2	time min.
0	2	40	$M_{2,2}$	0.77	9.55	1.30
1	2	40	$M_{2,2}$	0.69	22.45	1.52

Table 6.1.16: Results for algorithm OA_2 with models of up to two images (set S^2) created automatically.

6.2 ACO contour matching

While analyzing contour matching we can figure out the points on the edges as the vertices of a bipartite graph and the problem structured as Quadratic Assignment (*QA*); the key idea is to compute a mapping between input and model vertices, minimizing a global dissimilarity measure between the attributes from the two images. In many cases, the problem can be formulated as a quadratic assignment problem and we seek a correspondence which minimizes a particular objective function.

In this section we address this problem by using the original contour matching method described in [75] and replacing the commonly used constant sampling of the contour with dominant points.

Results are improved by applying a variation when calculating the used shape descriptors as described in [6].

We'll show how few dominant points performs better than an image contour uniform sampled points distribution, improving the accuracy and reduce computational time.

6.2.1 Dominant point extraction

We compute dominant points with GA [66] rather than other techniques. The main reason is related to the excellent tradeoff between approximation error and computational time compared to other dominant point extraction techniques. Furthermore GA permits to obtain the desired input number of dominant points.

6.2.2 ACO Shape Matching

Given two figure contours, the sets of points to compare are generally modeled with a graph $G = (V, E)$ that is to be traversed by the ants as a complete directed bipartite graph; the solution is searched through this graph. Each crossing of an ant corresponds to a feasible solution that is evaluated using the objective function of the initial problem. Every time an ant traverses the graph (fig. 6.10), a path is created and a particular substance, called pheromone, is released.

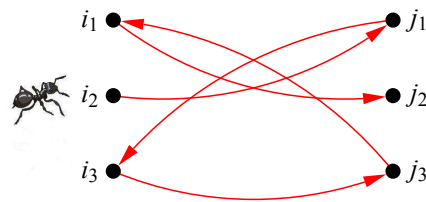


Figure 6.10: Bipartite graph ant crossing example. The path is $i_2 \rightarrow j_1 \rightarrow i_3 \rightarrow j_2 \rightarrow i_1 \rightarrow j_2$. The assignment is given by $\pi(i_1) = j_2, \pi(i_2) = j_1, \pi(i_3) = j_3$.

These paths lead to a correspondence between the points of two images: paths with greater amount of pheromone represent the sub-optimal solutions. When

6.2. ACO CONTOUR MATCHING

traversing from a vertex $i \in I$ to a vertex $j \in J$ the probability p_{ij}^k of an ant k choosing the edge that connects to vertex $j \in J$ is given by:

$$p_{ij}^k = \frac{\alpha\tau_{ij} + (1 - \alpha)\eta_{ij}}{\sum_{l \in N_i} [\alpha\tau_{il} + (1 - \alpha)\eta_{il}]}$$

where τ_{ij} quantifies the pheromones accumulated on the edge (i, j) , η_{ij} indicates the desirability (or probability) of traversing (i, j) based on heuristic information, and $N_i = \{l \in J : (i, l) \in E\}$ is the immediate neighborhood of vertex i . The parameter $0 \leq \alpha \leq 1$ regulates the influence of pheromones over heuristic information. While moving from J to I an edge pointing to any vertex of I has not been visited yet can be chosen. Each edge from j to I has the same probability of being selected. After $m \geq 1$ ants have traversed the graph, completing an ACO iteration, the corresponding solution is evaluated and the pheromones are updated with the following rules:

$$\text{Pheromone evaporation : } \tau_{ij} \leftarrow (1 - \rho)\tau_{ij}, \quad 0 \leq \rho \leq 1$$

where ρ is the *pheromone evaporation rate* and

$$\text{Pheromone deposition : } \tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^m \Delta\tau_{ij}^k$$

where $\Delta\tau_{ij}^k$ is the amount of pheromone that an ant k has deposited on the edge (i, j) . After each iteration the cost of solutions is computed the pheromones amounts are updated.

6.2.3 QAP formulation

Given two shapes represented as set of points I and J , the mapping between them can be reached by minimizing a given objective function. Let OBJ is the objective or cost function which evaluates the matching π in relation to the shapes characterized by I and J , and π is a mapping such that $\forall i \in I, \exists j \in J : \pi(i) = j$, then we are looking for $\hat{\pi}$ such that:

$$\hat{\pi} = \underset{\pi}{\operatorname{argmin}}(\operatorname{OBJ}(\pi, I, J))$$

The typical approach is to assign each point to a shape descriptor then a distance measure between two shape descriptors has to be defined. The assignment problem (AP) seeks for a correspondence which minimizes the sum of the distances between descriptors of points on one set and the descriptors of the corresponding points on the other set:

$$AP(\pi, R, I, J) = \sum_{i \in I} \mathcal{D}_R(R_i, R_{\pi(i)})$$

where R_i denotes the descriptor at point i and D_R is the shape descriptor distance measure. This kind of problems have a $O(n^3)$ computational complexity where n is the greatest between points number of I and J .

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

Taking into account the proximity information, i.e. if point i from shape I and point j from shape J are matched then a close-by neighbor i' of i should be matched with a point j' on J that is close to j , we obtain the general objective function of QAP:

$$QAP(\pi, R, I, J) = (1 - \nu)S(\pi, R, I, J) + \nu\mathcal{X}(\pi, I, J) \quad (6.2.9)$$

where $0 \leq \nu \leq 1$ is a control parameter. The similarity function S is defined as

$$S(\pi, R, I, J) = 1 - \frac{1}{|I|} \sum_{i \in I} e^{\frac{-\mathcal{D}_R(R_i, R_{\pi(i)})^2}{\sigma_R}}$$

while the proximity term \mathcal{X} is defined as:

$$\mathcal{X}(\pi, I, J) = \frac{\sum_{i \in I} \sum_{i' \neq i \in I} e^{\frac{-\mathcal{D}_I(i, i')^2}{\sigma_I}} |\mathcal{D}_I(i, i') - \mathcal{D}_J(\pi(i), \pi(i'))|}{|I|(|I| - 1)/2}$$

where \mathcal{D}_I and \mathcal{D}_J are distances, normalized to $[0,1]$, between two points from I and J respectively.

6.2.4 Our approach

Let M be the number of query shapes, N the number of test shapes and C the number of items of each query image class in the test set, i.e. cardinality, so that $N = M \cdot C$.

The matching algorithm consists of the following steps:

- 1: Set ACC as empty array
- 2: **for** $i = 1$ to M **do**
- 3: Set BC as empty bidimensional array
- 4: Extract image contour S_i
- 5: Determine the dominant points set D_i with GAPA extraction technique
- 6: **or**
- 7: Sample the contour with a constant step obtaining a point set F_i
- 8: **for** $j = 1$ to N **do**
- 9: Extract image contour S_j
- 10: Determine the dominant points set D_j with GAPA extraction technique
- 11: **or**
- 12: Sample the contour with a constant step obtaining a point set F_j
- 13: Compute the best cost similarity measure bc between D_i and D_j
- 14: **or**
- 15: Compute the best cost similarity measure bc between F_i and F_j
- 16: $BC[j, 1] = bc$
- 17: $BC[j, 2] = j$
- 18: **end for**
- 19: Sort BC by column 1 (ascending order)
- 20: $ACC[i] = 0$
- 21: **for** $k = 1$ to C **do**

6.2. ACO CONTOUR MATCHING

```
22:   if  $BC[k, 2] \in class(S_i)$  then
23:        $ACC[i] = ACC[i] + 1$ 
24:   end if
25: end for
26:  $ACC[i] = ACC[i]/C$ 
27: end for
28: Compute AVG_ACC as average of  $ACC$ 
```

We test the algorithm by replacing D_i and D_j in lines 5, 8, 9 with new sets of points F_i and F_j derived by a constant contour sampling. In our experiments we use shape contexts as shape descriptors and chi-square distance as metric.

Shape Context Rotation Invariance

Shape contexts are local descriptors widely used to compare two images [6]. The shape contexts are obtained by sampling points from both contours and generate histograms based on the angular view of each point to others. The histograms, now representing the points, are used to match the most suited point from a first drawing and a second drawing, and are originally made to find a perfect match.

In order to improve the accuracy of recognition by modifying the shape context extraction we apply a change to achieve rotational invariance according to [6].

In the calculation of the shape context of a given point, it is required to compute the angle and the length of the line between that point and all the other contour points.

Thanks to this property a rotation of the whole image is not required; in fact relation between tangent angles stays the same as points rotate and it is just necessary a modification on angle computation.

In particular, we calculate the angles between pairs of points and subtract the angle of the vector tangent to the point of interest. Let p_i be the current point; we choose to approximate the tangent vector with the segment joining p_{i-1} and p_{i+1} estimating the angle defined between such vector and the x-axis. In order to reduce the approximation error we work on the original image contour and selecting then the corresponding angle correction on the sampled or dominant points.

6.2.5 Experimental results

Our aim is to test the effectiveness of the shape matching algorithm comparing the dominant points and a uniform sampled points distribution of an image contour.

In our experiments we use the 216 image database S^1 (table A.0.1 appendix A).

We adopt shape contexts as shape descriptors and chi-square distance as metric, rather than euclidean, because of their weighting factors used to equalize differences. As highlighted in previous chapters ACO optimization techniques are not deterministic, i.e. they are subjected to different results in different runs. For this reason we repeat matching experiments for 10 runs and we consider the average accuracy results and the standard deviation. We set as experiments parameters 1 as number of ants and 100 as number of iterations each ants make in optimization process.

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

Exp	Points Type	# points	mean accuracy	mean accuracy standard deviation	execution time (sec. per matching)
A	DP	35	0.8298	4.16E-03	0.4080
B	DP	40	0.8429	9.05E-03	0.5097
C	DP	45	0.8333	6.30E-03	0.6048
D	DP	50	0.8258	3.57E-03	0.7055
E	CS	50	0.8328	5.02E-03	0.7023

Table 6.2.17: Average accuracy calculated for 35, 40, 45, 50 Dominant Points (*DP*) and 50 constant sampled points (*CS*). The experiments are set with 100 iterations, 1 ants and no modification to shape contexts.

Exp	Points Type	# points	mean accuracy	mean accuracy standard deviation	execution time (sec. per matching)
F	DP	35	0.8540	4.42E-03	0.4977
G	DP	40	0.8561	1.12E-02	0.6245
H	DP	45	0.8778	6.65E-03	0.7482
I	DP	50	0.8535	1.38E-02	0.8797
J	CS	50	0.8854	1.35E-02	0.8804

Table 6.2.18: Average accuracy calculated for 35, 40, 45, 50 Dominant Points (*DP*) and 50 constant sampled points (*CS*). The experiments are set with 100 iterations, 1 ants and rotation invariance modification to shape contexts.

We choose this configuration firstly in order to reduce computational time during tryouts.

In table 6.2.17 we show the experiments made with 35, 40, 45, 50 dominant points (respectively exp. *A-D*) and the comparison with 50 uniform sampled points (exp. *E*).

In addition we indicate the standard deviation of the mean accuracy computed over the 10 runs and the computational time of each matching between two figures. The experiments are executed on a linux 64 bit operating system running on a AMD Phenom II X6 1090T@3.2 GHz.

It is possible to notice the good performance offered by dominant points compared to constant sampling despite the lower number of points. The lower accuracy as the number of dominant point grows up depends on an overfitted distribution of dominant points along the contour corner.

We introduce the roto-invariance version of shape context (as described in sec. 6.2.4). In table 6.2.18 the results of experiments made with the same parameters of table 6.2.17 are shown. It is possible to notice the general performance increase. This is related to the matching condition enhancement due to a kind of figures alignment derived from modified shape contexts. The better result obtained by constant points is probably due to the more stable position of points obtained with uniform sampling during the rotational invariance process than dominant points. As well dominant points have a good behaviour taking into account the lower number of points.

6.2. ACO CONTOUR MATCHING

Gathering from table 6.2.17 and table 6.2.18 that a lower number points leads to a lower computational time, we consider to increase the number of ants or the number of iterations in the experiments involving less dominant points while having the same computational time of the experiments with the highest number of points. In other words, if we indicate with t^* the computational time needed for the experiment J we set the number of ants of experiment H or its number of iterations so that the execution time it is approximatively equal to t^* .

To evaluate the effect of such variation we run 100 iterations computing the average execution time and the average best cost obtained by matching some random figures of the same category. We observe, in general, that a lower best cost is obtained by increasing the number of iterations rather than raising the number of ants.

So, we decide to run the experiments getting the iteration from 100 to 185 obtaining an execution time nearly equal to t^* . In table 6.2.19 we show a resume of best average accuracy experiments. For each image, i.e. for each class, the average accuracy over 10 runs is shown. In table 6.2.20 we show the distribution of correct image samples detected choosing the run of experiment L with the best accuracy (0.9192). The first column reports the query shape for each class while the others represent the items ordered by similarity, i.e. lower best cost, to the query shape.

CHAPTER 6. SHAPE MATCHING BY DOMINANT POINTS

Exp.	K	L	J	
Points type	DP	DP	CS	
# points	40	45	50	
# iterations	185	185	100	
Average accuracy for each class	Bird	0.9545	0.9455	0.6909
	Bone	1.0000	1.0000	0.9273
	Brick	0.8727	0.8091	0.8818
	Camel	0.7364	0.7091	0.7273
	Car	1.0000	1.0000	0.9909
	Children	1.0000	1.0000	1.0000
	Classic	0.9909	1.0000	0.9727
	Elephant	0.8818	0.8909	0.9455
	Face	1.0000	1.0000	1.0000
	Fork	0.8000	0.9455	0.9455
	Fountain	1.0000	1.0000	1.0000
	Glass	1.0000	0.9818	1.0000
	Hammer	0.5636	0.5545	0.6818
	Heart	0.9909	1.0000	1.0000
	Key	0.8818	0.9636	0.8545
	Misk	1.0000	1.0000	1.0000
Ray	0.7182	0.7545	0.7545	
Turtle	0.7000	0.7545	0.5636	
Global Average Accuracy	0.8939	0.9061	0.8854	
Execution time (sec. per matching)	0.8634	0.8993	0.8804	

Table 6.2.19: Methods comparison: accuracy calculated for each query shape. For each image, i.e. for each class, the average accuracy over 10 runs is shown.

6.2. ACO CONTOUR MATCHING

Table 6.2.20: Image matching for the run of experiment with $DP = 45$, 185 iteration and 1 ant with the best precision. For each query shape, i.e first column image, the ranking of the first 11 images ordered by descending similarity is shown. The achieved accuracy is 0.9192.

7

Conclusions

In this thesis the problem of polygonal approximation of closed digital curves is presented.

After an examination in chap. 2 of the commonly used descriptors for the extraction of binary objects features we highlight the importance of contours in shape analysis.

Along a digital curve it is also possible locating salient points with high curvature that can be used to approximate, without a significant loss of information, the original contour. Among the current trends in dominant point detection (chap. 3) we propose three novel approaches, picking out their outstanding results in term of accuracy compared to other methods and taking into account the ratio between integral square error and compression ratio as performance indicator (chap. 4). Furthermore we propose a fast technique suitable for on-line processing despite a minor precision.

In chap. 5 we evaluate also the robustness of the proposed approaches both with affine transformations and noisy conditions, demonstrating the substantial invariance to such kind of alterations and perturbations of the original contours.

We describe the application of dominant points to shape matching in chap. 6. In particular, we introduce a novel approach to curve alignment by showing the almost identical matching rate of the proposed method with an automatically contour points selection, reducing considerably the computational time. We report also the experimentation made by adapting a quadratic assignment problem addressed with the Ant Colony Optimization paradigm by replacing a constant contour sampling with a dominant points configuration showing the excellent results achieved.

Although the large application fields of polygonal approximation in the future we will focus on the following issues:

- study the problem of the correlation between shape features and the DPIL threshold so that it is possible to make it self independent, i.e. to adopt an input parameter in order to reach the desired input number of dominant points based on shape descriptors;
- speed up the meta-heuristics methods by implementing a parallel computation: in fact both ACOPA and GAPA provide several “break points” in their algorithm formulation to fork the process into multiple concurrent instances;

CHAPTER 7. CONCLUSIONS

- apply the polygonal approximation to improve the shape features extraction process and matching algorithms: working on a “reconstructed” contour allows to reduce the influence of the noise and to uptake of stylized model easy to manage.



Image database













































































































											
											
											
											
											
											
											
											
											

Table A.0.2: Image set 2 - the first column represents the query images.

Bibliography

Bibliography

- [1] T. W. Rauber and A. S. Steiger-Garcia, "2-D Form Descriptors Based on a Normalized Parametric Polar Transform (UNL Transform)", Proc. of MVA 1992 IAPR Workshop on Machine Vision Applications (1992), 466-469
- [2] D. Zhang and G.J. Lu, "Study and evaluation of different Fourier methods for image retrieval", Image Vis. Comput., 23 (1), (2005), 33
- [3] B. Wang, "Shape retrieval using combined Fourier features", Optics Communications, 284 (14), (2011), 3504 - 3508
- [4] N. Alajlan, I. E. Rube M. Kameland G. Freeman, "Shape retrieval using triangle-area representation and dynamic space warping", Patt. Recogn, 40, (2007), 19111920.
- [5] B. Wang and C. Shi, "Shape Matching Using Chord-Length Function", Intelligent Data Engineering and Automated Learning - IDEAL, (2006), 746-753
- [6] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell., 24 (4) , (2002), 509522
- [7] CC. Chen, "Improved moment invariants for shape discrimination", Pattern Recognition, 26, (1993), 683-686
- [8] M. K. Hu, "Visual pattern recognition by moment invariants", IEEE Trans. Inf. Theory, 12, (1962), 179-187
- [9] A. Del Bimbo and P. Pala, "Visual image retrieval by elastic matching of user sketches", Pattern Analysis and Machine Intelligence, 19 (2), (1997), 121 - 132
- [10] F. Mokhtarian and A. Mackworth. "Scale-based description and recognition of planar curves and two-dimensional shapes". IEEE Transactions on Pattern Analysis and Machine Intelligence, 8(1), (1986), 34-43.
- [11] D. Zhang and G. Lu, "Evaluation of MPEG-7 shape descriptors against other shape descriptors", Multimedia Systems 9, (2003), 1530
- [12] G. C.-H. Chuang and C.-C. J. Kuo, "Wavelet descriptor of planar curves: Theory and applications", IEEE Trans. Image Processing, 5 (1), (1996), 56-70
- [13] H. Freeman. "On the encoding of arbitrary geometric configurations", IRE Tans. Electron. Comput. 10, (1961), 260-268
- [14] R. Merhotra and J. Gary, "Similar-shape retrieval in shape data management", Computer, 28 (9), (1995), 57-62
- [15] S. Berretti, A. Del Bimbo and P. Pala, "Retrieval by shape similarity with perceptual distance and effective indexing". IEEE Trans. on Multimedia, 2 (4), (2000), 225-239

BIBLIOGRAPHY

- [16] C. Di Ruberto, L. Cinque, "Decomposition of two-dimensional shapes for efficient retrieval", *Image and Vision Computing*, 27 (8), 2008, 1097-1107
- [17] M. Sonka, V. Hlavac and R. Boyle, "Image Processing, Analysis and Machine Vision", Thomson-Engineering, (2007)
- [18] M. K. Hu, "Pattern recognition by moment invariants", *Proc.IRE* , 49, (1961)
- [19] J. F. T. Suk, "Pattern recognition by affine moment invariants", *Pattern Recognition*, 26, (1993), 167-174
- [20] G. Taubin and D. Cooper, "Geometric invariance in computer vision", MIT Press, ch. "Object recognition based on moment (or algebraic) invariants", (1992), 375-397.
- [21] C. H. Teh and R. Chin., "On image analysis by the method of moments", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10 (4), (1988), 496-513
- [22] D. Zhang, "Image retrieval based on shape" Ph.D. dissertation, Monash University, March 2002
- [23] G. J. Lu and A. Sajjanhar, "Region-based shape representation and similarity measure suitable for content-based image retrieval", *Multimedia Systems*, 7 (2), (1999), 165-174
- [24] A. Goshtasby, "Description and discrimination of planar shape matrices", *IEEE Trans. Pattern Anal. Mach. Intell.*, 7, (1985), 738-743
- [25] C. Di Ruberto, "Recognition of shapes by attributed skeletal graphs", *Pattern Recognition*, 37 (1), 2004, 21-31
- [26] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson and S. W. Zucker, "Shock graphs and shape matching". *International Journal of Computer Vision*, 35 (1), (1999)
- [27] A. Carmona-Poyato, N.L. Fernández-García, R. Medina-Carnicer and F.J. Madrid-Cuevas, "Dominant point detection: A new proposal", *Image and Vision Computing*, 23, (2005), 1226-1236
- [28] P. Cornic, "Another look at dominant point detection of digital curves", *Pattern Recognition Letters*, 18, (1997), 13-25
- [29] E. Attneave, "Some informational aspects of visual perception", *Psychol. Rev*, 61, (1954), 183-193
- [30] T.P. Nguyen and I. Debled-Rennesson, "A discrete geometry approach for dominant point detectio'n", *Pattern Recognition*, 44, (2011), 3244
- [31] A. Rosenfeld and E. Johnston, "Angle detection on digital curves", *IEEE Trans. Comput.*, C-22, (1973), 875-878

BIBLIOGRAPHY

- [32] C.H. Teh, R.T. Chin, "On the detection of dominant points on digital curves", IEEE Trans. on Pattern Analysis and Machine Intelligence, 11 (8), (1989), 859-871
- [33] W.Y. Wu, "A dynamic method for dominant point detection", Graphical Models, 64, (2003), 304-315
- [34] M.J. Wang, W.Y. Wu, L.K. Huang and D.M. Wang, "Corner detection using bending value", Pattern Recognition Letters, 16, (1995), 575-583
- [35] W.Y. Wu, "Dominant point detection using adaptive bending value", Image and Vision Computing, 21, (2003), 517-525
- [36] M. Marji and P. Siy, "A new algorithm for dominant points detection and polygonization of digital curves", Pattern Rec., 36, (2003), 2239-2251
- [37] J. Sklansky and V. Gonzalez, "Fast polygonal approximation of digitized curves", Pattern Recognition, 12, (1980), 327-331
- [38] B.K. Ray and S. Ray, "Detection of Significant Points and Polygonal Approximation of Digitized Curves", Pattern Recognition Letters, 13, (1992), 443-452
- [39] B.K. Ray and K.S. Ray, "Determination of optimal polygon from digital curve using L1 norm", Pattern Recognition, 26, (1993), 505-509
- [40] Y. Kurozumi and W.A. Davis, "Polygonal Approximation by Minimax Method", Computer Graphics and Image Processing, 19, (1982), 248-264
- [41] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves", Comput. Graphics Image Process., 1, (1972), 244-256
- [42] N. Ansari, E.J. Delp, "On detecting dominant points", Pattern Recognition, 24, (1991), 441-450
- [43] B.K. Ray, K.S. Ray, "A new split-and-merge technique for polygonal approximation of chain coded curves", Pattern Recognition Letters, 16, (1995), 161-169
- [44] Z. Habib, M. Sakai and M. Sarfraz, "Interactive Shape control with Rational Cubic Splines", Computer-Aided Design & Applications, 1(1-4), (2004), 709-718
- [45] A. Held, K. Abe and C. Arcelli, "Towards a Hierarchical Contour Description via Dominant Point Detection", IEEE Trans. on Systems Man and Cybernetics, 24, (1994), 942-949
- [46] W.Y. Wu and M.J. Wang, "Detecting the dominant points by the curvature-based polygonal approximation", CVGIP: Graphical Model Image Process., 55, (1993), 79-88

BIBLIOGRAPHY

- [47] J.G. Dunham, "Optimum uniform piecewise linear approximation of planar curves", *IEEE Trans. Pattern Anal. Mach. Intell.*, 8, (1986), 67-75
- [48] Y. Sato, "Piecewise linear approximation of plane curves by perimeter optimization", *Pattern Recognition*, 25, (1992), 1535-1543
- [49] A. Masood, "Optimized Polygonal Approximation by Dominant Point Deletion", *Pattern Recognition*, 41, (2008), 227-239
- [50] F. Glover, "Future Paths for Integer Programming and Links to Artificial Intelligence", *Comput. & Ops. Res.*, 13 (5), (1986), 533-549
- [51] P.Y. Yin, "A tabu search approach to the polygonal approximation of digital curves", *Int. J. Pattern Recognition Artif. Intell.*, 14, (2000), 243-255
- [52] C. Di Ruberto and A. Morgera, "A new algorithm for polygonal approximation based on Ant Colony System", *ICIAP 2009 - 15th International Conference on Image Analysis and Processing*, (2009), 633-641
- [53] P.Y. Yin, "Ant colony search algorithms for optimal polygonal approximation of plane curves", *Pattern Recognition*, 36(8), (2003), 1783-1797
- [54] Yin, P.Y., "Genetic particle swarm optimization for polygonal approximation of digital curves", *Pattern Recognition and Image Analysis*, 16 (2), (2006), 223-233
- [55] S.C. Huang and Y.-N. Sun, "Polygonal Approximation Using Genetic Algorithms", *Pattern Recognition*, 32, (1999), 1409-1420
- [56] P.-Y. Yin, "A New Method for Polygonal Approximation Using Genetic Algorithms", *Pattern Recognition Letters*, 19, (1998), 1017-1026
- [57] J.H. Horng, "Improving fitting quality of polygonal approximation by using the dynamic programming technique", *Pattern Recognition Letters*, 23, (2002), 1657-1673
- [58] P.Y. Yin, "Polygonal approximation of digital curves using the state-of-the-art metaheuristics", *Vision Systems: Segmentation and Pattern Recognition*, (2007), 451-464
- [59] M. Dorigo, "Optimization, learning, and natural algorithms", Ph.D. Thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy, (1992)
- [60] H. Zhang and J. Guo, "Optimal Polygonal Approximation of Digital planar Curves Using Meta Heuristics", *Pattern Recognition*, 34, (2001), 1429-1436
- [61] J.H. Holland, "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor (1975)
- [62] D. Sarkar, "A simple algorithm for detection of significant vertices for polygonal approximation of chain-coded curves", *Pattern Recognition Letters*, 14 (12), (1993) 959-964

BIBLIOGRAPHY

- [63] P. L. Rosin, "Techniques for assessing polygonal approximations of curves", *IEEE Trans. Pattern Anal. Mach. Intell.*, 19 (6), (1997), 659-666
- [64] J. A. Ventura and J. M. Chen, "Segmentation of two-dimensional curve contours.", *Pattern Recognition*, 25 (10), (1992), 1129-1140
- [65] P. L. Rosin, "Assessing the behaviour of polygonal approximation algorithms", *Pattern Recognition*, 36, (2003), 505-518.
- [66] C. Di Ruberto and A. Morgera, "Dominant Points Detection on Digital Curves: A Comparison Between Optimal and Exact Approaches.", *Proceedings of the International Conference on Artificial Intelligence, ICAI (2010)*, 570-575
- [67] D. DeTurck, H. Gluck, D. Pomerleano and D. S. Vick, "The four vertex theorem and its converse", *Notice of the AMS*, 54 (2), (2007), 192-207
- [68] <http://www.cs.rutgers.edu/pub/sven/rutgers-tools/>
- [69] <http://www.lems.brown.edu/vision/software/>
- [70] A. Masood, "Dominant point detection by reverse polygonization of digital curves", *Image and Vision Comp.*, 26, (2008), 702-715
- [71] P.Y. Yin, "Genetic algorithms for polygonal approximation of digital curves", *Int. J. Pattern Recognition Artif. Intell.*, 13, (2009), 1-22
- [72] T. D. Mavridou and P. M. Pardalos, "Simulated Annealing and Genetic Algorithms for the Facility Layout Problem: A Survey", *Computational Optimization and Applications*, 7, (1997), 111-126.
- [73] S.-Y. Ho and Y-C Chen, "An Efficient Evolutionary Algorithm for Accurate Polygonal Approximation", *Pattern Recognition*, 34, (2001), 2305-2317
- [74] R. Real and J. M. Vargas, "The Probabilistic Basis of Jaccard's Index of Similarity", *Systematic Biology*, 245, (1996), 380-385
- [75] O. van Kaick, G. Hamarneh, H. Zhang, and P. Wighton. "Contour correspondence via ant colony optimization", *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, (2007), 271-280
- [76] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "Recognition of Shapes by Editing Shock Graphs", *Proceedings of the Eighth International Conference on Computer Vision*, (2001) 1755-1762
- [77] T.B. Sebastian and B.B. Kimia, "Curves vs. Skeletons in Object Recognition", *Signal Processing*, 85 (2), (2005), 247-263