Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης Και Υπολογιστων
Εργαστηριο Εξεργασιας Εικονας Βιντεο και Πολυμεσων

Κατάτμηση Φυσικών Εικόνων με Τεχνικές Αποσύνθεσης Σκελετού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Σπυρίδωνος Μ. Λεονάρδου

Επιβλέπων: Στέφανος Κόλλιας
Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2012

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης Και Υπολογιστων
Εργαστηριο Εξεργασιας Εικονας Βιντεο και Πολυμεσων

Κατάτμηση Φυσικών Εικόνων με Τεχνικές Αποσύνθεσης Σκελετού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Σπυρίδωνος Μ. Λεονάρδου

Επιβλέπων: Στέφανος Κόλλιας
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 19η Μαρτίου 2012.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Στέφανος Κόλλιας	Σταφυλοπάτης Ανδρέας-Γεώργιος	Στάμου Γεώργιος
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Λέκτορας Ε.Μ.Π.

Αθήνα, Μάρτιος 2012

.....

Σπυρίδων Μ. Λεονάρδος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Ευχαριστίες

Η παρούσα διπλωματική εργασία εκπονήθηκε κατά το ακαδημαϊκό έτος 2011-2012 στο Εργαστήριο Ψηφιακής Επεξεργασίας Εικόνας, Βίντεο και Πολυμέσων του Εθνικού Μετσόβιου Πολυτεχνείου. Θα ήθελα να ευχαριστήσω τον επιβλέποντα Καθηγητή κ. Στέφανο Κόλλια για την εμπιστοσύνη που μου έδειξε αναθέτοντάς μου την εργασία αυτή και για τη δυνατότητα που μου έδωσε να ασχοληθώ με το συγκεκριμένο ενδιαφέρον θέμα. Επίσης, ευχαριστώ ιδιαίτερα τον Δρ. Ιωάννη Αβρίθη για την καθοδήγησή του και για την εξαιρετική συνεργασία που είχαμε. Τέλος, θα ήθελα να ευχαριστήσω τους γονείς μου για την καθοδήγηση και την ηθική συμπαράσταση που μου προσέφεραν όλα αυτά τα χρόνια.

Περίληψη

Στην παρούσα διπλωματική εργασία παρουσιάζονται τεχνικές κατάτμησης εικόνων βασισμένες σε μία διαδικασία αποσύνθεσης σκελετού. Ξεκινώντας από το μέτρο της δισδιάστατης κλίσης της εικόνας ή τη γκρίζα εικόνα ακμών της, υπολογίζουμε ένα σταθμισμένο μετασχηματισμό απόστασης και τον αντίστοιχο σταθμισμένο σκελετό μέσω μιας διαδικασίας γραμμικού χρόνου. Εφαρμόζοντας, τώρα, τον ίδιο μεσχηματισμό απόστασης από το σκελετό ανάποδα, λαμβάνουμε δυαδικά μια αρχική κατάτμηση της εικόνας και ένα γράφο που αναπαριστά τη δομής της εικόνας. Αυτό ειναι ισοδύναμο με εφαρμογή μετασχηματισμού watershed στο σταθμισμένο μετασχηματισμό απόστασης. Όμως, είναι πιο αποδοτικό αφού πρώτα αποσυνθέτουμε το σκελετό και ύστερα χρησιμοποιούμε τη διαδικασια γραμμικού χρόνου για τη διάδοση της αποσύνθεσης στην υπόλοιπη επιφάνεια της εικόνας. Αναπαριστώντας την εικόνα ως disjoint-set forest δομή, ενώνουμε γειτονικές περιοχές βάσει κάποιων κριτηρίων. Διάφορα κριτήρια υπολοποιήθηκαν και αξιολογήθηκαν. Κατά το πρώτο από αυτά τα κριτήρια, χρησιμοποιούμε το ύψος των σημείων σέλλας του σκελετού για να εκφράσουμε την ομοιότητα μεταξύ γειτονικών περιοχών. Μια δεύτερη διαφορετική κατεύθυνση που ακολουθούμε ειναι η χρήση ενός μέτρου κλειστότητας των περιοχών της εικόνας για να αποφασίσουμε εάν πρέπει να ενώσουμε δύο γειτονικές περιοχές. Τρίτη και τελευταια κατεύθυνση ειναι η χρήση ενός μέτρου ανομοιομορφίας μεταξύ δύο περιοχών που ικανοποιεί την υπερμετρική ιδιότητα. Με αυτό τον τροπο, υλοποιούμε μια ιεραρχική κατάτμηση εικόνας. Για το μέτρο ανομοιομορφίας μεταξύ περιοχών χρησιμοποιούμε τη μέση τιμή του μέτρου της δισδιάστατης κλίσης στο κοινό σύνορο των δύο περιοχών καθώς και ένα μέτρο κλειστότητας του κοινού συνόρου. Όλες οι παραπάνω τεχνικές αξιολογούνται με βάση τη συλλογή δεδομένων του πανεπιστημίου Berkeley και συγκρίνονται με γνωστές τεχνικές της βιβλιογραφίας. Χωρίς μάθηση, πετυχαίνουμε πολύ καλά αποτελέσματα κοντά στη στάθμη της τεχνικής και με πολύ πρακτικούς χρόνους εκτέλεσης.

Λέξεις Κλειδιά

Όραση υπολογιστών, επεξεργασια εικόνας, κατάτμηση, σκελετός, αποσύνθεση σκελετού, ανίχνευση ακμών, γράφοι γειτνίασης περιοχών, μέθοδοι ταχείας προσπέλασης, σταθμισμένος μετασχηματισμός απόστασης, αλγόριθμος ακριβούς προσπέλασης κατά ομάδες.

Abstract

In the framework of this thesis, we present new image segmentation techniques based on a weighted medial axis decomposition procedure. Starting from image gradient or grayscale contour map, we first compute a weighted distance map and its weighted medial axis by a linear-time process. Now, applying the same distance propagation from the medial axis backwards, we dually obtain an initial image partition and a graph representing image structure. This is equivalent to applying watershed transform on the weighted distance map, hence is both topological and contrast-weighted. However, it is more efficient because we first decompose the medial axis and then use our linear-time process to propagate on the remaining image surface. Using a disjoint-set data structure, we then merge adjacent regions according to some criteria. Several different criteria were examined and tested. First, we use medial axis saddle point height to express similarity between adjacent regions and merge correspondingly. A second distinct direction we follow is to merge adjacent regions according to how fragmented they are. Last but not least, we use ultrametric contour map representation to implement hierarchical segmentation. As inter-region ultrametric dissimilarities, we use mean boundary strength on the common boundary between adjacent regions and inter-region fragmentation. All the above mentioned techniques are evaluated using the Berkeley Segmentation Dataset and compared with some state of the art algorithms. Without learning, we achieve performance near the state of the art with very practical running times.

Keywords

Computer vision, image processing, segmentation, partition, grouping, medial axis, skeleton, medial axis decomposition,weighted distance transform, region adjacency graphs, exact group marching, fast marching methods, contour detection, watershed.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Defining image segmentation

The problem of image segmentation is one of the most fundamental problems in the field of computer vision. Although it has been studied since the early years of computer vision, segmentation still remains a great challenge. Since the time of the Gestalt movement in psychology, it has been known that perceptual grouping plays a powerful role in human visual perception. In late 1930's Wertheiner [54] described perceptual grouping as following: "I stand at the window and see a house, trees, sky. Theoretically, I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees. It is impossible to achieve "327" as such. And yet even though such droll calculation were possible and implied, say, for the house 120, the trees 90, the sky 117 – I should at least have this arrangement and division of the total, and not, say, 127 and 100 and 100; or 150 and 177."

More specifically, image segmentation is described as the process of partitioning an image into disjoint regions, each one being homogeneous and connected with respect to one or multiple cues such as image intensity, texture, color, motion and others. The goal of image segmentation is to cluster pixels into salient image regions, i.e. regions corresponding to individual surfaces, objects or natural parts of objects. Hence, one can easily see that there is no single definition of segmentation. In fact, it is impossible to generally formulate the exact goals of segmentation, as Marr [33] has underlined. Thus, it can be concluded that segmentation is application-dependent and finding a unique solution is, in general, ambiguous.

A formal definition of segmentation is presented by Gonzalez and Woods [23]. Let X represent the spatial domain that is occupied by an image. In this case, image segmentation process can be considered as a procedure which separates space X into n discrete regions $R_1, R_2, .., R_n$ in such a manner that the following properties are satisfied:

(a) R_i is a connected set for all $i = 1, 2, 3, .., n$

(b) $\bigcup_{i=1}^{n} R_i = X$

(c) $R_i \cap R_j = \varnothing$ for all i, j with $i \neq j$

Condition (a) denotes that all elements of a region R_i must be connected under a predefined manner (e.g. 4-connectivity or 8-connectivity). The next condition demands that the process of segmentation must be *complete*, i.e. each image element (pixel) must belong to some region R_i. Moreover, according to the third condition all regions to which the original image is divided into must be disjoint or mutually exclusive. In addition, two more properties related to the termination of the segmentation procedure, must be satisfied. Given a logical predicate $Q(R_i)$ defined for all discrete regions R_i, segmentation procedure must terminate when

(d) $Q(R_i) = \textbf{true}$ for all $i = 1, 2, 3, .., n$

(e) $Q(R_i \cap R_j) = \textbf{false}$ for any two adjacent regions R_i and R_j

where two regions R_i and R_j are *adjacent* if and only if their union is a *connected* set. Condition (d) denotes that predicate Q must be true for each discrete region R_i whereas any two adjacent regions R_i and R_j must differ considering Q. Note that a specific *segmentation* S or *partition* P of an image is defined as the set containing all the discrete subregions $R_1, R_2, .., R_n$ into which the image is divided.

Image segmentation is one of the most popular computer vision problems because it is related to a wide range of applications as numerous visual tasks benefit from the existence of some hundreds or thousands "superpixels" rather than millions of pixels. Segmentation is closely connected to object detection and recognition and there is huge literature on methods combing these two problems [20, 22, 28, 53]. Some other popular applications of segmentation and grouping include topics such as occlusion boundary estimation within motion systems [47], object-based image compression [52], content-based image retrieval [8], medical imaging [41], face recognition [29, 57], iris recognition [17, 21] and fingerprint recognition [36].

1.2 Motivation

Significant psychophysical evidence suggests that when looking at a natural scene the visual brain differentiates the scene into surface regions and boundary contours that delineate possible meaningful object parts. More specifically, two important research results support the explicit use of the medial axis in the human visual system, one using psychophysics and the other through neurophysiological data. First, Kovacs and Julesz [25] showed that the detection of closed curves in an ambiguous scene is much easier than the detection of open curves. Furthermore, they showed that this notion of closure is associated with an enhancement of feature detection inside the figure as opposed to outside the figure. The non-uniformity of this enhancement showed peaks at central loci of the figure, which they correlated very closely to the medial axis of the shape. In addition, Lee et al. [27] confirmed

Lamme's empirical findings of an enhancement inside a figure indicated by texture or motion boundaries [26] and explored the spatial and dynamical aspects of such a response. They showed that the enhancement in response was not spatially uniform but rather showed distinct peaks at the boundary of the figure, and more interestingly, at its medial axis. Thus, the initial local edge contrast perceived by human vision system is expected to improve with feedback from higher areas which have a more abstract and global view of the image. In conclusion, there is strong scientific evidence that the visual brain uses medial axes to assist in scene segmentation and perceptual grouping. Our motivation for a segmentation framework based on medial axis derives from this evidence in conjunction with the fact that medial axis has not been extensively studied as a means towards image partitioning.

The segmentation method presented in the framework of this thesis originates at the *medial feature detector*. Avrithis and Rapantzikos [7] developed a framework for blob-like feature detection. Although this work focused on a different problem, i.e. feature detection, it produces an initial image partition (oversegmentation). They introduced a *medial axis decomposition* based method to obtain this initial partition. This image representation is more descriptive than the classic watershed transform [9] concerning region boundaries. Using the above representation, we do not only overcome contour discontinuities and boundary fragmentation but we can measure the fragmentation of a boundary curve or the fragmentation of an *entire* region.

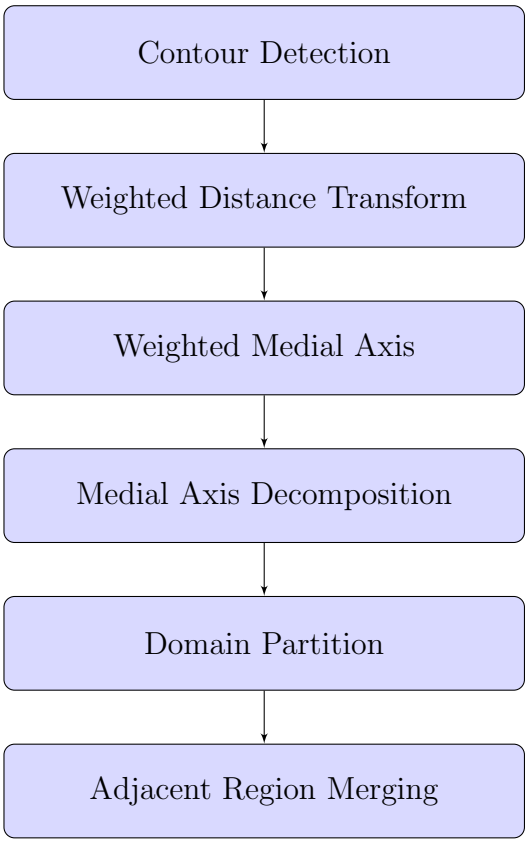
Apart from boundary description, the initial partition of [7] has some advantages over the watershed transform used in mathematical morphology [9]. The latter has the drawback that the initial produced oversegmentation is too fine to be exploited. This problem has been partially dealt with the introduction of marker controlled watershed. On the one hand, the marker controlled watershed seems to function pretty well for blob-like objects subtraction from the background. On the other hand, this approach has not proved to be general enough but is rather image-dependent. The initial partition of [7] is not as fine as classic watershed output but it still preserves boundaries quality. In fact, finesse of the initial partition can be *controlled* by a scale parameter in the weighted distance map computation. Last but not least, it is more efficient because first the medial axis is decomposed and then region labels are propagated on the remaining image surface in linear-time.

Our contribution is the implementation, examination and evaluation of several merging criteria and techniques to process the initial partition at hand, i.e. to turn the oversegmentation into a "proper" partition. As the term "proper partition" is ambiguous we define that, in the framework of this thesis, the "proper" partition we seek is the closest possible partition to human ground-truth segmentations. Human segmentations are provided along with a set of images in the Berkeley Segmentation Dataset [4, 34].

Felzenszwalb and Huttenlocher [18] presented an efficient graph based merging technique. It is of special interest that they define as inter-region dissimilarity as the *minimum* edge weight connecting the two regions. In such a way, the total merging process reduces to a single sorting process and a single pass from each graph edge to

evaluate whether the two corresponding regions should be merged or not. We adopt an analogous efficient merging process in the implemented framework. Furthermore, to include semi-global information from the total area of any two adjacent regions, we propose a second similar efficient merging technique in which adjacent region merging is controlled by an area fragmentation factor. In addition, we propose a method with two variants which exploits information related to region boundaries and uses the notion of *ultrametric contour maps* [2]. Using dissimilarities that satisfy the ultrametric property we manage to implement hierarchical segmentation. The ultrametric dissimilarities we use are based on mean boundary strength between adjacent regions and inter-region fragmentation.

Overall, the segmentation method presented in the framework of this thesis can be summarized in the following six steps :



1.3 Thesis outline

This diploma thesis structure is generally based on the discrete steps of the implemented segmentation framework. **Chapter** 2 includes a literature survey on generic image segmentation methods. Several methods are briefly described, from simplest methods to more sophisticated state-of-art algorithms. Greatest attention

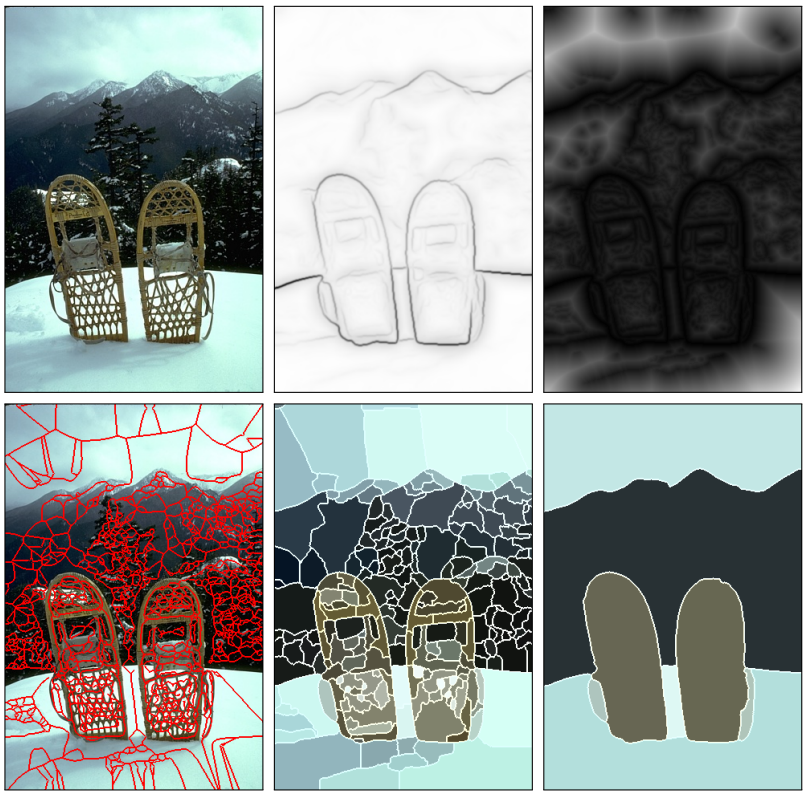


Figure 1.1: **Top row, from left to right**: input image, grayscale contour map and normalized weighted distance transforms. **Bottom row, from left to right**: medial axis(in red), initial partition and final partition after the merging process.

has been given to methods related to the proposed technique. In **Chapter** 3, the two contour detectors whose outputs are used in order to compute the weighted distance transform. The baseline Canny contour detector is first presented and then the state-of-art global probability of boundary detector. In **Chapter** 4, theory and definitions of distance transforms are presented. Then, general algorithms for computing distance transforms such as the fast marching methods are discussed along with the algorithm implemented for the computation of the weighted distance transform, i.e. the linear-time exact group marching algorithm. In **Chapter** 5, the concept of medial axis is briefly discussed and then the algorithm of the computation of the weighted medial axis is presented. In **Chapter** 6, the algorithm for the weighted medial axis decomposition is described along with the way of obtaining an initial image partition by a dual distance propagation. In **Chapter** 7, the merging techniques are presented along with their properties and implementation issues. **Chapter** 8 includes evaluation and comparison of our segmentation techniques using the Berkeley segmentation dataset with some well-know state of the art segmentation algorithms. In addition, we illustrate representative results for the several implemented techniques. In **Chapter** 9, several conclusions derived from this thesis are presented along with directions for future work. In **Appendix** A, several basic proofs of the majority of lemmas and algorithms are included.

Chapter 2

Literature survey on image segmentation methods

2.1 Generic image segmentation methods

Image segmentation methods can be classified into two broad categories: *boundary-based* approaches, which generate an edge image which delineates the segments of the image, and *region-based* approaches, which group image pixels based on the homogeneity of spatially localized features.

The simplest form of segmentation is *thresholding*. A threshold is defined and then every pixel in an image is compared with this threshold. If the pixel lies above the threshold it will be marked as foreground. Otherwise, it will be marked as background. The threshold will most often be an intensity or colour value. Other forms of thresholding exist where the threshold is allowed to vary across the image, but thresholding is a primitive technique, and will only work for very simple segmentation tasks. The key of this method is to select the threshold value. Several popular methods can be used for this purpose including the maximum entropy method and Otsu's method [40] (maximum variance).

Compression based methods postulate that the optimal segmentation is the one that minimizes, over all possible segmentations, the coding length of the data. Mobahi et al. [37] describe each segment by its texture and boundary shape. Each of these components is represented by a probability distribution function and its coding length is computed as follows: the boundary encoding leverages the fact that regions in natural images tend to have a smooth contour. This prior is used by Huffman coding to encode the difference chain code of the contours in an image. Thus, the smoother a boundary is, the shorter coding length it attains. Texture is encoded by lossy compression in a way similar to *minimum description length* (MDL) principle, but here the length of the data given the model is approximated by the number of samples times the entropy of the model. The texture in each region is represented by a multivariate normal distribution whose entropy has closed form expression. For

any given segmentation of an image, this scheme yields the number of bits required to encode that image based on the given segmentation. Thus, among all possible segmentations of an image, the goal is to find the segmentation which produces the shortest coding length. This can be achieved by a simple *agglomerative clustering* method. The distortion in the lossy compression determines the coarseness of the segmentation. It has the drawback that distortion optimal value may differ for each image.

Another broad family of segmentation techniques includes *clustering* methods. One basic representative of this category is the k-means algorithm, an iterative technique that is used to partition an image into K clusters. The basic algorithm proceeds as following:

1. Initialize the K cluster centers.
2. Assign each pixel in the image to the cluster that minimizes the distance between the pixel and the cluster center.
3. Recompute cluster centers by averaging all of the pixels in the cluster.
4. Repeat steps 2 and 3 until convergence.

In this case, distance is the squared or absolute difference between a pixel and a cluster center. The difference is typically based on pixel color, intensity, texture, location or a weighted combination of these. k-means is guaranteed to converge, but it may not return the optimal solution. The quality of the solution depends on initializations and on the value of K. It is similar to the *expectation-maximization* algorithm for mixtures of Gaussians in that they both attempt to find the centers of natural clusters in the data. The objective it tries to achieve is to minimize total intra-cluster variance, or, the squared error function. In terms of performance the algorithm is not guaranteed to return a global optimum. The quality of the final solution depends largely on the initial set of clusters, and may, in practice, be much poorer than the global optimum. A drawback of the k-means algorithm is that the number of clusters K is an input parameter. An inappropriate choice of K may yield poor results. The algorithm also assumes that the variance is an appropriate measure of cluster scatter.

The *mean shift* algorithm [15] offers an alternative *clustering* framework. Pixels are represented in the joint spatial-range domain by concatenating their spatial coordinates and color values into a single vector. Applying mean shift filtering in this domain yields a convergence point for each pixel. Regions are formed by grouping together all pixels whose convergence points are closer than h_s in the spatial domain and h_r in the range domain, where h_s and h_r are respective bandwidth parameters. Additional merging can also be performed to enforce a constraint on minimum region area.

Histogram-based methods are very efficient when compared to other image segmentation methods because they typically require only one pass through the pixels. In this technique, a histogram is computed from all of the pixels in the image, and

the peaks and valleys in the histogram are used to locate the clusters in the image [46]. Color or intensity can be used as the measure. One disadvantage of the histogram-seeking method is that it may be difficult to identify significant peaks and valleys in the image. In this technique of image classification distance metric and integrated region matching are familiar.

Graph partitioning methods can effectively be used for image segmentation. In these methods, the image is viewed as a weighted, undirected graph. Usually a pixel or a group of pixels are associated with nodes and edge weights define the similarity or dissimilarity between adjacent regions or pixels. The graph (image) is then partitioned according to a specific criterion. Each partition of the nodes (pixels) output from these algorithms are considered an object segment in the image. Some popular algorithms of this category are the normalized cuts [44], the minimum cut algorithm [55] and minimum spanning tree-based segmentation [18].

Spectral graph theory [14] and in particular the *normalized cuts* criterion [44] provide a method of integrating image global information into the grouping process. In the normalized cuts framework the feature space the image is represented as an undirected weigthed graph. Vertices represent (pixels) and edges exist between all pairs of vertices. Edge weights measure the similarity between the two corresponding vertices. A graph can be partitioned in two disjoint sets A, B by simply removing edges connecting these two parts. The degree of dissimilarity between these two sets can be computed as the sum of weights of all the edges that have been removed. This quantity is called the *cut*. The *optimal* bipartitioning of a graph is the one that minimizes this *cut* value. However, using the minimum cut criterion results in cutting small sets of isolated nodes in the graph. Wu and Leahy [55] proposed a clustering method based on this minimum cut criterion and noticed the above mentioned problem in their work. To overcome this problem, Shi and Malik [44] introduced a normalized measure of disassociation between two groups taking into account the total edge connections to all nodes in the graphs. This measure, called the *normalized cut (Ncut)*, is defined as:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} \tag{2.1}$$

where $assoc(A, V)$ is the total connection from nodes in A to all nodes in the graph. Shi and Malik proved that finding the optimal graph bipartition that minimizes the *Ncut* value reduces to a generalized eigenvalue problem. The image is bipartitioned using the eigenvector solution with the second smallest eigenvalue. Then, the segmented parts are recursively bipartitioned if *Ncut* is below a prespecified value. Alternatively, k-means clustering is applied to obtain the desired image partition. However, this approach often breaks uniform regions where the eigenvectors have smooth gradients. Recently, a *multiscale* approach of normalized cuts for image segmentation was introduced by Cour et al. [16]. The fact that the affinity matrix must be sparse, in order to avoid a prohibitively expensive computation, limits the naive implementation to using only local pixel affinities. Cour et al. [16] solve this limitation by computing sparse affinity matrices at multiple scales, setting up

cross-scale constraints and deriving a new eigenvector problem for this constrained multiscale cut.

Many approaches to image segmentation fall into a different category than those covered so far, relying on the formulation of the problem in a variational framework. An example is the model proposed by Mumford and Shah [38], where the segmentation of an observed image u_0 is given by the minimization of the functional:

$$\mathcal{F}(u, C) = \int_{\Omega} (u - u_0)^2 dx + \mu \int_{\Omega \backslash C} |\nabla u|^2 dx + \nu \oint_{C} ds \tag{2.2}$$

where C denotes the smooth and closed segmenting curve, u denotes the piecewise smooth approximation to u_0 with discontinuities only along C, Ω denotes the image domain and μ, ν are weighting parameters where. Several algorithms have been developed to minimize the energy (2.2) or its simplified version, where u is piecewise constant in $\Omega \backslash C$.

The *watershed transform* considers the gradient magnitude of an image as a topographic surface. Pixels having the highest gradient magnitude correspond to watershed lines, which represent the region boundaries. Water placed on any pixel enclosed by a common watershed line flows downhill to a common local intensity minimum. Pixels draining to a common minimum form a catchment basin, which represents a segment. One of the principal applications of watershed segmentation is in the extraction of nearly uniform (bloblike) objects from the background. As the method implemented in the framework of this thesis is related to the watershed transform, a more detailed description of this method is discussed in section 2.2.1.

2.2 Related Work

2.2.1 Watershed Transform

A brief presentation of the *watershed transform* as described by [9, 23] follows. The basic concept of watersheds is based on visualizing an image in three dimensions: two spatial coordinates versus gray levels. In such a "topographic" interpretation, three types of points are considered: (a) points belonging to regional minima; (b) points at which a drop of water, if placed at the location of any of those points, would fall with certainty to a single minimum and (c) points at which water would be equally likely to fall to more than on such minimum. The points satisfying condition (c) form crest lines on the topographic surface and are termed *divide lines* or *watershed lines* or *watershed arcs*.

The principal objective of segmentation algorithms based on these concepts is to find the watersheds lines. The basic idea is simple: suppose that a hole is punched in each regional minimum and that the entire topography is flooded from below by letting water rise through the holes at a uniform rate. When the rising water in

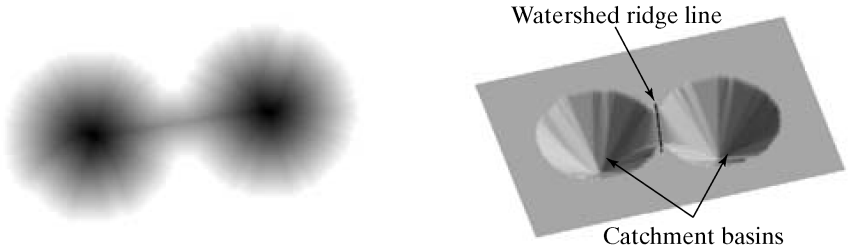


Figure 2.1: **left**: input image, **right**: its topographic surface with corresponding catchment basins and a watershed ridge line between them.

distinct catchment basins is about to merge, a dam is built to prevent the merging. The flooding will eventually reach a stage when only the tops of the dams are visible above the water line. These dam boundaries correspond to the divide lines of the watersheds. Therefore, they are the continuous boundaries extracted by a watershed segmentation algorithm.

Regions characterized by small variations in gray levels have small gradient values. Thus, in practise, watershed segmentation is most often applied to the gradient of an image, rather to the image itself. In this formulation, the regional minima of catchment basins correlate nicely with the small value of the gradient corresponding to the objects of interest.

The dam construction is based on *binary morphological dilation*. Initially, the set of regional minima corresponds to value 1 while all other pixels have zero value. In each subsequent step, the $3D$ topography is flooded from below and the pixels covered by the rising water are 1s and others 0s. At each step of the algorithm, the binary image in obtained by a binary morphological dilation. The dam is constructed by the points on which the dilation would cause the sets being dilated to merge, resulting one-pixel thick connected path.

Let M_1, M_2, ..., M_R denote the regional minima of an (gradient) image $g(x, y)$, $C(M_i)$ the sets of the points in the catchment basin associated with regional minimum M_i and the minimum and maximum gray levels of $g(x, y)$ as $minlevel$ and $maxlevel$. $T[n]$ is defined as:

$$T[n] = \{(s, t) : g(s, t) < n\} \tag{2.3}$$

The topography will be flooded in *integer* flood increments, from $n = minlevel+1$ to $n = maxlevel + 1$. If $C_n(M_i)$ denote the sets of the points in the catchment basin associated with regional minimum M_i and flooded at step n one can write:

$$C_n(M_i) = C(M_i) \cap T[n] \tag{2.4}$$

Let $C[n]$ denote as the union of the flooded catchment basin portions at stage n. Then $C[n] = \bigcup_{i=1}^{R} C_n(M_i)$ and obviously $C[maxlevel+1] = \bigcup_{i=1}^{R} C(M_i)$. The algorithm

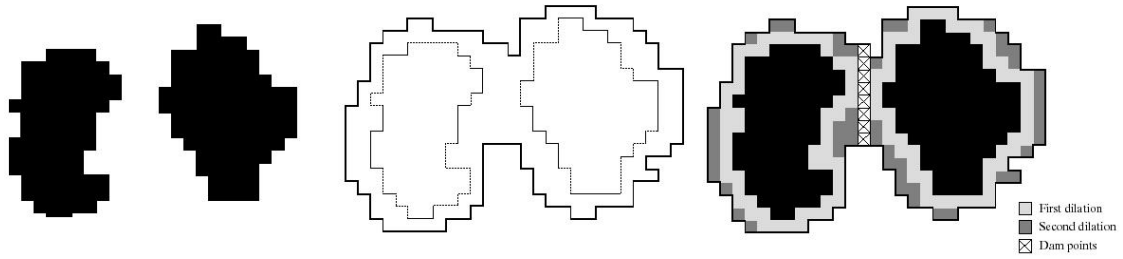


Figure 2.2: **Left**: two partially flooded catchment basins at stage $n-1$ of flooding, **Middle**: flooding at stage n, showing that water has spilled between basins, **Right**: result of dilation and dam construction (from [23])

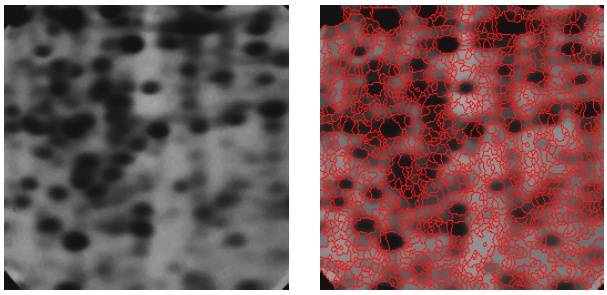


Figure 2.3: **Left**: input image. **Right**: oversegmentation produced by applying watershed transform directly to image gradient.

is initialized with $C[minlevel+1] = T[minlevel+1]$. The algorithm at each step constructs $C[n]$ given $C[n-1]$. Denote $Q[n]$ the set of connected components in $T[n]$. For each connected component $q \in Q[n]$, there are three possibilities:

1. $q \cap C[n-1]$ is empty. Then, a new minimum is encountered and q is incorporated into $C[n-1]$ to form $C[n]$.
2. $q \cap C[n-1]$ contains one connected component of $C[n-1]$. Then, q is incorporated into $C[n-1]$ to form $C[n]$.
3. $q \cap C[n-1]$ contains more than one connected components of $C[n-1]$. Then, a ridge separating two or more catchment basins has been encountered and a dam has to be built within q to prevent overflow between the catchment basins .

The above procedure is repeated until $n = maxlevel + 1$

The direct application of the watershed transformation to a gradient image usually leads to oversegmentation similar to the result shown in Fig. 2.3.

To prevent this phenomenon, a marker-controlled watershed transform is introduced. A major enhancement of the watershed transformation consists in flooding the topographic surface from a previously defined set of markers. An example of marker extraction is the generalized *maxima/minima extraction* or *dome/basin* extraction [50]. For the domes, the principle is to subtract an arbitrary constant h

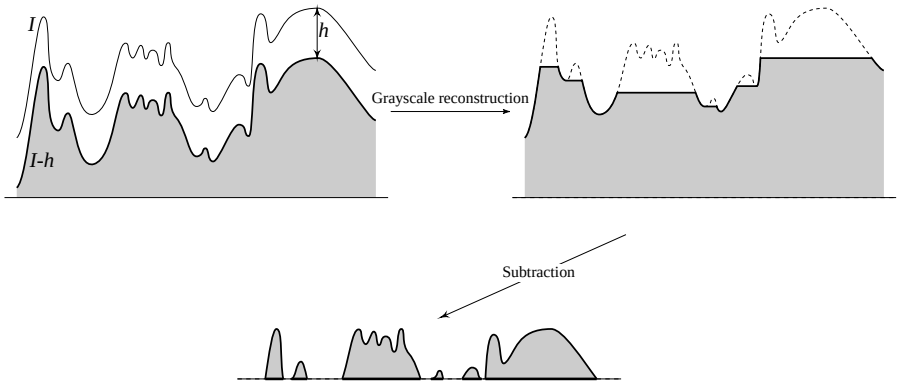


Figure 2.4: Determination of the h-domes of grayscale image I (from [50]).

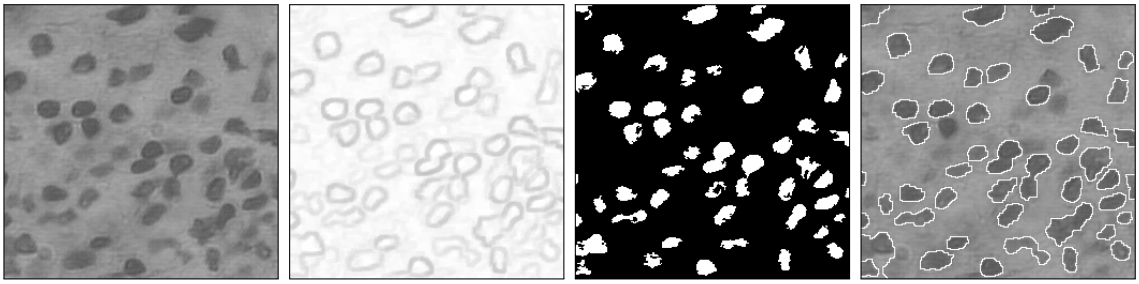


Figure 2.5: **From left to right**: input image, morphological gradient, markers of domes (in black), final segmentation.

from the original image I and to perform a grayscale *reconstruction opening* of I from $I - h$. The reconstructed image is then subtracted from the original one, thus yielding a grayscale image J of *all* the domes and crest lines of I. From J it easy to extract a binary picture of the most important domes by a simple thresholding operation. The dual process can be used to extract the basins and valleys of I.

2.2.2 Oriented watershed transform and ultrametric contour map

A recent version of the classical watershed transform which includes information about boundaries strength and orientation is the *oriented watershed transform*. Arbelaez et al. [4] introduces the *oriented watershed transform*(OWT). As input to this algorithm can be used any contour detector output $E(x, y, \theta)$ which predicts the probability of an image boundary at location (x, y) with orientation θ. Regional minima of $E(x, y) = \max_\theta E(x, y, \theta)$ are considered as seed locations for homogeneous segments and then the classical watershed transform described in section 2.2.1 is applied on the topographic surface of $E(x, y)$. The catchment basins of the minima ($\mathcal{P}_0$) provide the regions of the finest partition and the corresponding watershed arcs ($\mathcal{K}_0$) the possible locations of the boundaries.

Next a adjacent region dissimilarity based on boundary strength is defined. Simply weighting each arc by the mean value of $E(x, y)$ can introduce artifacts. To correct this problem, consistency is enforced between the strength of the boundaries of $\mathcal{K}_0$ and the underlying E signal. The watershed arcs are approximated with line segments. Then, each pixel (x, y) is assigned the orientation $o(x, y) \in [0, \pi)$ of the corresponding line segment. Boundary strength at pixel (x, y) is now $E(x, y, o(x, y))$ instead of $E(x, y, \theta)$. In addition, this dissimilarity satisfies the ultrametric property as discussed in [2]. A more detailed description of ultrametric contour maps notion is included in section 7.3.

Because of the existence of closed, weighted and non-self-intersecting contours the method used in the next step is the *ultrametric contour map* (UCM) . UCM produces an hierarchy on regions based on contours with the latter properties. The hierarchy of region is constructed by a greedy graph-based region merging algorithm. An initial graph $G = (\mathcal{P}_0, \mathcal{K}_0)$ is defined where the nodes are the regions $\mathcal{P}_0$, the links are the arcs $\mathcal{K}_0$ separating adjacent regions, and the weights $W : \mathcal{K}_0 \to \mathbb{R}_+$ are a measure of dissimilarity between regions. The algorithm proceeds by sorting the links by similarity and iteratively merging the most similar regions.

1) Select minimum weight contour: $C^* = \arg\min_{C \in \mathcal{K}_0} W(C)$
2) Let $R_1, R_2 \in \mathcal{P}_0$ be the regions separated by C^*
3) Set $R = R_1 \cup R_2$ and update: $\mathcal{P}_0 \leftarrow \mathcal{P}_0 \backslash \{R_1, R_2\} \cup R$ and $\mathcal{K}_0 \leftarrow \mathcal{K}_0 \backslash \{C^*\}$
4) Stop if $\mathcal{K}_0$ is empty. Otherwise, *update weights* $W(\mathcal{K}_0)$ and repeat.

2.2.3 Efficient graph-based image segmentation

The graph based region merging algorithm advocated by Felzenszwalb and Huttenlocher [18] attempts to partition image pixels into components such that the resulting segmentation is neither too coarse nor too fine. In this work, definitions of the terms "too fine", "proper refinement of a segmentation" and "too coarse" are given as following:

Definition 2.2.1 *A segmentation S is too fine if there is some pair of regions $R_1, R_2 \in S$ for which there is no evidence for a boundary between them.*

Definition 2.2.2 *Given two segmentations S and T of the same base set, T is a refinement of S when each component of T is contained in (or equal to) some component of S. T is a proper refinement of S if and only $T \neq S$.*

Definition 2.2.3 *A segmentation S is too coarse if there exists a proper refinement of S that is not too fine.*

In this approach, an image is represented as an undirected graph $G = (V, E)$. Vertices $v_i \in V$ are the set of elements to be segmented, i.e. pixels, and edges

$(v_i, v_j) \in E$ correspond to pairs of neighboring vertices. Each edge $(v_i, v_j) \in E$ has a corresponding weight $w(v_i, v_j)$, which is a non-negative measure of dissimilarity between neighboring vertices (e.g. intensity or color differences).

The internal difference of a region (component) $R \subseteq V$ is defined as the largest weight in the minimum spanning tree of the component $MST(R, E)$. That is:

$$\text{Int}(R) = \max_{e \in MST(R,E)} w(e) \tag{2.5}$$

The difference between two regions $R_1, R_2 \subseteq V$ is defined as the minimum edge connecting the two regions. That is:

$$\text{Dif}(R_1, R_2) = \min \{w(v_i, v_j) : v_i \in R_1, v_j \in R_2, (v_i, v_j) \in E\} \tag{2.6}$$

If there is no edge connecting R_1 and R_2 then let $\text{Dif}(R_1, R_2) = \infty$

A region comparison predicate is introduced to evaluate evidence for a boundary between a pair of regions by checking if the difference between the regions, $\text{Dif}(R_1, R_2)$, is large compared to the internal difference within at least one of the regions, $Int(R_1)$ and $Int(R_2)$. For small components, $\text{Int}(R)$ is not a good estimate. Therefore, a threshold function $\tau(R)$, based on region size, is used to control the degree to which the difference between the regions must be larger than the minimum internal difference. That is, for small regions a stronger evidence of boundary is required. The pairwise comparison predicate is defined as:

$$\text{D}(R_1, R_2) = \begin{cases} \text{true} , & \text{if Dif}(R_1, R_2) > \text{MInt}(R_1, R_2) \\ \text{false} , & \text{otherwise} \end{cases} \tag{2.7}$$

where the minimum internal difference, MInt, is defined as

$$\text{MInt}(R_1, R_2) = \min (\text{Int}(R_1) + \tau(R_1), \text{Int}(R_2) + \tau(R_2)) \tag{2.8}$$

According to the main merging algorithm each node is initially placed in its own component (region). Considering edges in non-decreasing order by weight, each step of the algorithm merges regions R_1 and R_2 connected by the current edge if $\text{D}(R_1, R_2)$ is false. The running time of the above segmentation algorithm is $O(n\text{log}n)$, where n is the total number of image pixels. For grayscale images, the edge weight function is based on the absolute intensity difference between the pixels connected by an edge. For color images, the above algorithm is performed three times, one for each color channel, and then the three sets of components are intersected. An important characteristic of the method is its ability to *preserve* detail in *low-variability* image regions while *ignoring* detail in *high-variability* regions.

Chapter 3

Contour detection

3.1 Baseline Canny contour detection

As a baseline to the contour detection step of the segmentation framework, a grayscale version of the Canny edge detector [13] is implemented. In this section, the basic steps of the Canny edge detector are briefly presented and then it is explained how the basic algorithm is modified to produce *grayscale* and *not binary* contour maps.

The first step of the Canny edge detection is a convolution of the grayscaled input image with a Gaussian filter to achieve noise reduction. Then, first order gaussian derivatives are used to compute the blurred image gradient. This derivative procedure returns a value for the first derivative in the horizontal direction (G_x) and the vertical direction (G_y). From this, the gradient magnitude G and direction Θ can be determined:

$$G = \sqrt{G_x^2 + G_y^2} \tag{3.1}$$

$$\Theta = arctan\left(\frac{G_y}{G_x}\right) \tag{3.2}$$

The gradient direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals (0, 45, 90 and 135 degrees for example). Given estimates of the image gradients, a search is then carried out to determine if the gradient magnitude assumes a local maximum in the gradient direction. This is called the *non-maximum suppression* step and produces thin lines in the output contour image.

The last step is the *hysteresis thresholding* operation. Large intensity gradients are more likely to correspond to edges than small intensity gradients. In most cases, it is impossible to specify a threshold at which a given intensity gradient switches from corresponding to an edge into not doing so. Therefore Canny uses thresholding with hysteresis. Thresholding with hysteresis requires two thresholds

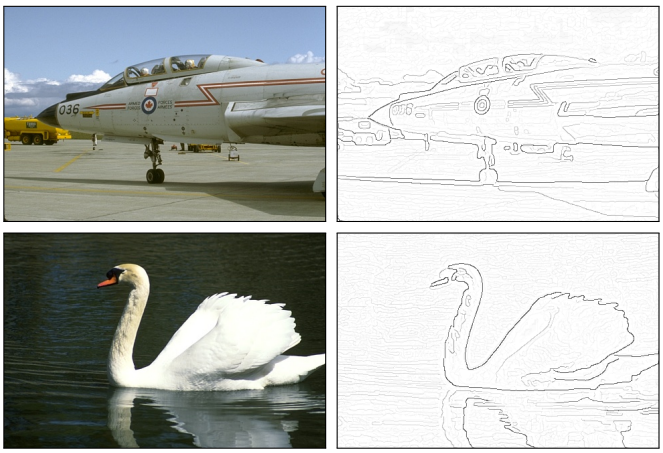


Figure 3.1: **left**: input images, **right:** grayscale contour images produced by the modified Canny contour detection algorithm.

– high and low. Making the assumption that important edges should be along continuous curves in the image allows to follow a faint section of a given line and to discard a few noisy pixels that do not constitute a line but have produced large gradients. Therefore, a high threshold is first applied. This marks out the edges that are probably genuine. Starting from these, using the directional information derived earlier, edges can be traced through the image. While tracing an edge, the lower threshold (which is usually a fraction of the high threshold) is applied, allowing to trace faint sections of edges as long as we find a starting point. Once this process is complete, a binary image where each pixel is marked as either an edge pixel or a non-edge pixel is obtained. Using multiple values for the high threshold and weighting accordingly results in the desired grayscale contour image. Examples are illustrated in Fig. 3.1. The implementation used in the framework of this thesis is based on the implementation of D. R. Martin, which is publicly available at `http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/code/Detectors/`.

3.2 Globalized probability of boundary

At this point there will be a brief description of the second contour detection input to the implemented segmentation framework. That is the *globalized probability of boundary* (gPb) detector [3]. This detector combines multiple local cues such as brightness, color, texture in multiple scales and combines these cues with global information. It is the state-of-art algorithm in the field of contour detection and produces results closest to human results than any other algorithm.

To begin with, Martin et al. [35] define a function $Pb(x, y, \theta)$ to describe the posterior probability of a boundary with orientation θ at each image pixel (x, y) by measuring the difference in local image brightness, color, and texture channels. Arbelaez et al. [3] introduce the *globalized probability of boundary* (gPb). The globalized

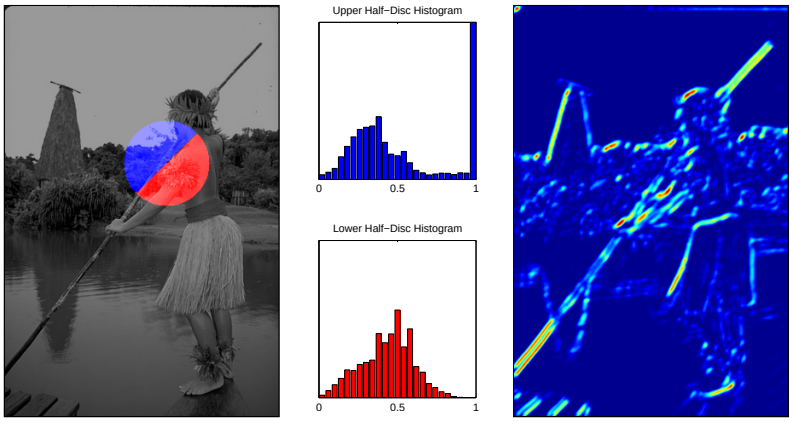


Figure 3.2: **Left**: input image with a circular disc with orientation $\theta = \pi/4$ centered at an arbitrary pixel. The disc is bigger for illustrative purposes. **Middle**: the blue and red distributions are the histograms of the pixel brightness values in the blue and red regions, respectively, in the input image. **Right**: result after a second order Savitzky-Golay smoothing filter is applied to the raw histogram difference output.

probability of boundary includes a multiscale version of the previous Pb detector plus an additional globalization step based on spectral clustering.

The basic building block of the Pb contour detector is the computation of an oriented gradient signal $G(x, y, \theta)$ from an intensity image I. Its computation proceeds as following: a circular disc is placed at location (x, y) and split into two half-discs by a diameter at angle θ. The gradient magnitude G at location (x, y) is defined by the χ^2 distance between the two half-disc histograms g and h:

$$\chi^2 = \frac{1}{2} \sum_i \frac{(g(i) - h(i))^2}{g(i) - h(i)} \tag{3.3}$$

Then, a second-order Savitzky-Golay filtering is applied to enhance local maxima and smooth out multiple detection peaks in the direction orthogonal to θ. An example of the above procedure is illustrated in Fig. 3.2. In total, four channels are used to combine cues. The first three are the channels of the CIELab colorspace (L, a, b) and the fourth is the texture channel. This first step of gPb detector is illustrated in Fig. 3.2.

The process of computing the texture gradient is of special interest. First, the grayscale version of the input image is convolved with the 17 filters of Fig. 3.3. Then, each pixel is associated with a 17-dimensional vector of responses and these vectors are clustered using k-means. The cluster centers define a set of image-specific *textons*. Finally, each pixel is assigned the integer id in $[1, K]$ of the closest center and the texture gradient is computed with the histograms same as before.

All the above cues consist the Pb detector. To detect coarse as well as fine structures, gradients are computed at three scales (radii of disks): $[\frac{\sigma}{2}, \sigma, 2\sigma]$. The

Figure 3.3: The filterbank for creating textons consists of 8 oriented even- and odd-symmetric Gaussian derivative filters and a center-surround (difference of Gaussians) filter.

multiscale oriented gradient signal is computed as:

$$mPb(x, y, \theta) = \sum_{s} \sum_{i} a_{i,s} G_{i,\sigma(i,s)}(x, y, \theta) \tag{3.4}$$

Orientation θ is sampled at 8 equally spaced orientations in the interval $[0, \pi)$. The weights $a_{i,s}$ are learned by gradient ascent on the F-measure using the Berkeley Segmentation Dataset [3].

On top of the multiscale implementation of the Pb detector there is a globalization procedure. As input to this stage, a sparse affinity matrix $\mathbf{W}$ is constructed using the maximal value of mPb along the line segment $\overline{ij}$ connecting two pixels i,j. All pixels i and j within a fix radius r are connected with affinity:

$$\mathbf{W}_{ij} = \exp(-\max_{p \in \overline{ij}} \{mPb(p)\}/\rho) \tag{3.5}$$

where ρ is a constant. In order to introduce global information, a similar globalization procedure as in the Normalized Cuts framework [44] is adopted. A diagonal matrix $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$ is defined and the corresponding generalized eigenvalues problem is solved:

$$(\mathbf{D} - \mathbf{W})\mathbf{v} = \lambda \mathbf{D}\mathbf{v} \tag{3.6}$$

Eigenvectors carry themselves contour information as one can easily see in Fig. 3.5. Based on this observation, each eigenvector $\mathbf{v_k}$ is treated as a image and convolved with Gaussian directional filters at multiple orientations θ. In this way, oriented signals $\{\nabla_\theta \mathbf{v_k}(x, y)\}$ are obtained and the spectral component of the boundary detector is formed:

$$sPb(x, y, \theta) = \sum_{k=1}^{n} \frac{1}{\sqrt{\lambda_k}} \mathbf{v_k}(x, y) \tag{3.7}$$

Finally, the *globalized probability of boundary* is a linear combination of the multiscale Pb and the above mentioned spectral component sPb:

$$gPb(x, y, \theta) = \sum_{s} \sum_{i} \beta_{i,s} G_{i,\sigma(i,s)}(x, y, \theta) + \gamma \cdot sPb(x, y, \theta) \tag{3.8}$$

The weights $\beta_{i,s}$ and γ are learned by gradient ascent on the F-measure using the Berkeley Segmentation Dataset [3]. An example of the gPb detector output is illustrated in Fig. 3.6.

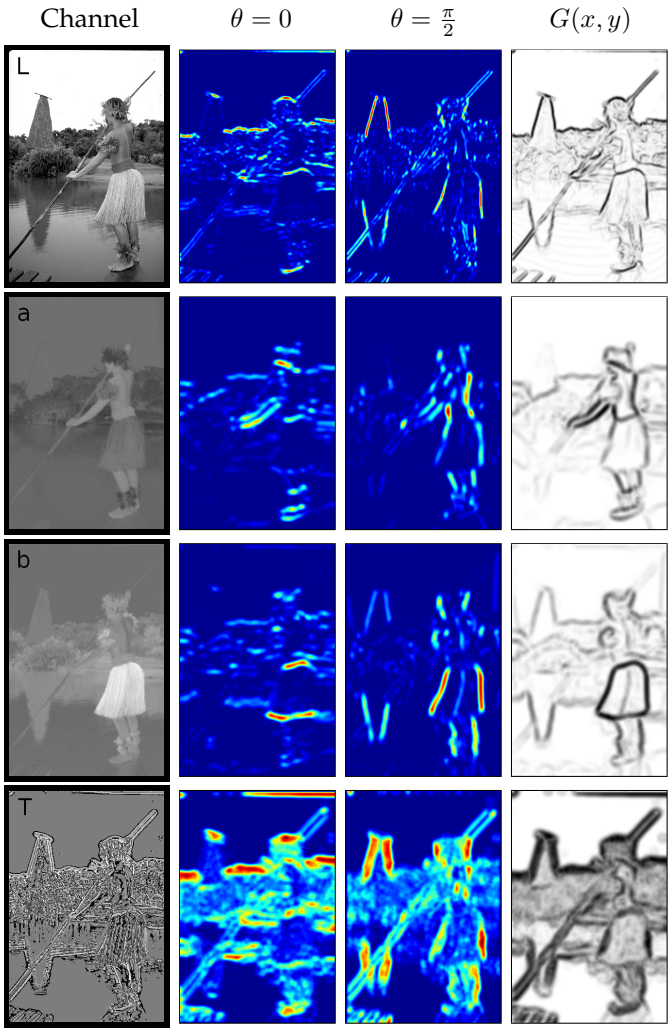


Figure 3.4: **Left, from top to bottom**: the brightness, color a, color b channels of Lab color space and the texton channel. **Rows, from left to right**: next to each channel, oriented gradient of histograms for $\theta = 0$ and $\theta = \pi/2$ (horizontal and vertical) and maximum response over eight orientations in $[0, \pi)$.

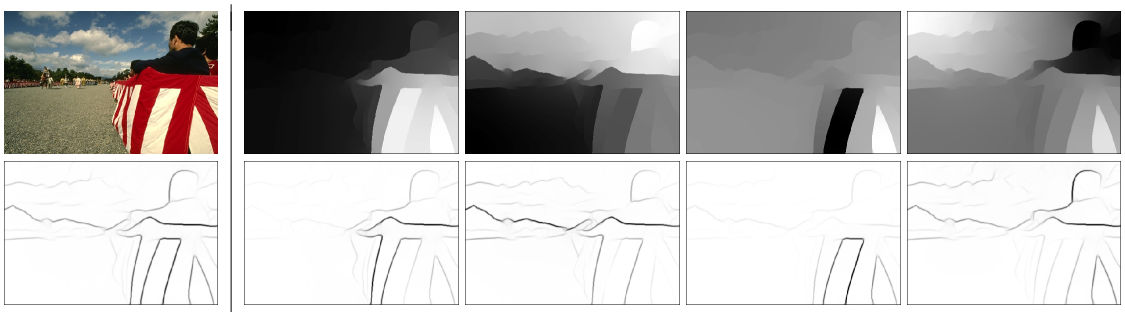


Figure 3.5: **Left**: Image and maximum response of spectral Pb over orientations, $sPb(x, y) = \max_\theta\{sPb(x, y, \theta)\}$. **Right top**: first four generalized eigenvectors, $\mathbf{v_1}, \ldots, \mathbf{v_4}$ used in creating sPb. **Right bottom**: maximum gradient response over orientations,$\max_\theta\{\nabla_\theta \mathbf{v_k}\,(x, y)\}$, for each eigenvector.

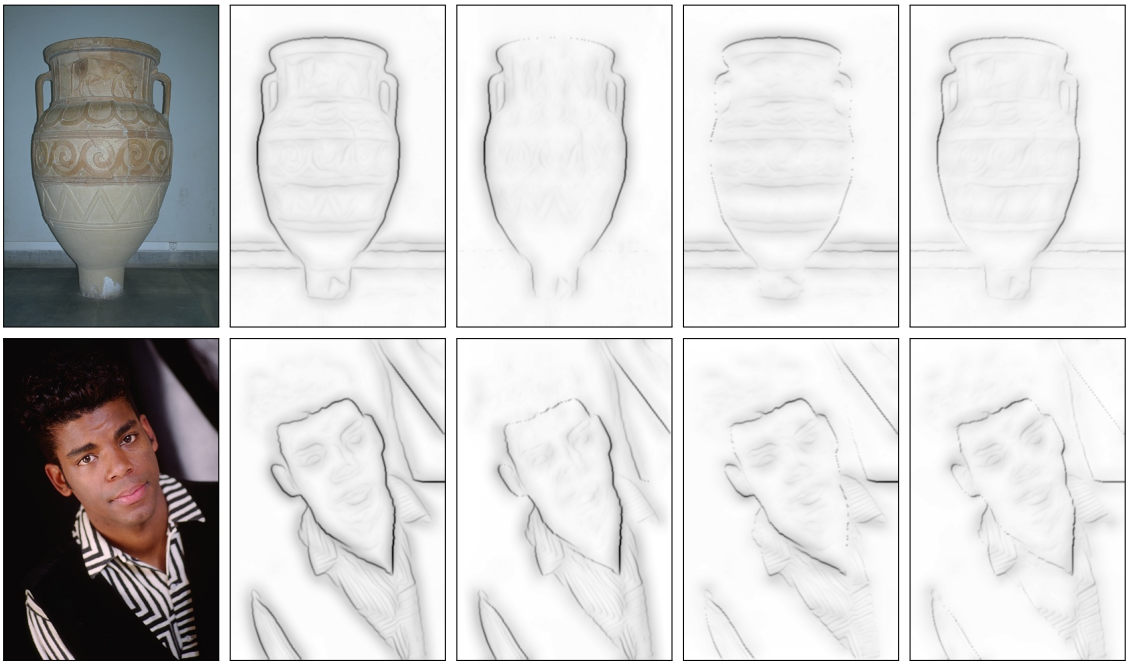


Figure 3.6: **From left to right**: input image, maximum response of gPb over all eight orientations, gPb for $\theta = \pi/2$, $\theta = 0$ and $\theta = \pi/4$ respectively.

Chapter 4

Distance transforms

4.1 Distance transform

4.1.1 Definitions

Before proceeding to definitions concerning distance transforms, there is a short description of the representations that will be used from this point and on. These representations are the same as in [7] in order to retain consistency with the work we are based on. To begin with, 2D images are represented by functions $f : \mathbb{X} \to \mathbb{V}$. As *range* $\mathbb{V}$ will be used the extended real line $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ and as *domain* $\mathbb{X}$ the continuous (discrete) space $\mathbb{R}^2$ ($\mathbb{Z}^2$).The space of all such functions is denoted by $\mathbb{F}$. In practice, only a bounded subset $X \subseteq \mathbb{X}$ is used. In the *discrete* domain, X is identified with the set of vertices V of a *grid* (graph) $G = \{V, E\}$ and its edges $E \subseteq V \times V$ are defined as the set of vertex pairs $e = (u, v)$ such that $u, v \in V$ are *connected*. We use 4- or 8-connectivity, and write $u \diamond v$ ($u \ast v$) iff u, v are 4- (8-) connected.

A function g from $V \times V \to \overline{\mathbb{R}}_+$ is called:

(a) *Positive definite:* if $g(u, v) = 0 \Leftrightarrow u = v$, $\forall u, v \in V$

(b) *Symmetric:* if $g(u, v) = g(v, u)$, $\forall u, v \in V$

(c) *Triangular:* if $g(u, w) \leq g(u, v) + g(v, w)$, $\forall u, v, w \in V$

If g satisfies (a)-(c), it is called a *distance function* or a *metric* [42].

A distance transform is a special distance function that will be shortly introduced. Consider first of all a binary image consisting of feature and non-feature points, respectively foreground and background pixels, which can be single points, edges or entire objects. A distance transform is an operation that converts a binary image to a grayscale image where all points have a value corresponding to the distance to the nearest feature value. The image foreground set is denoted as $S \subseteq X$

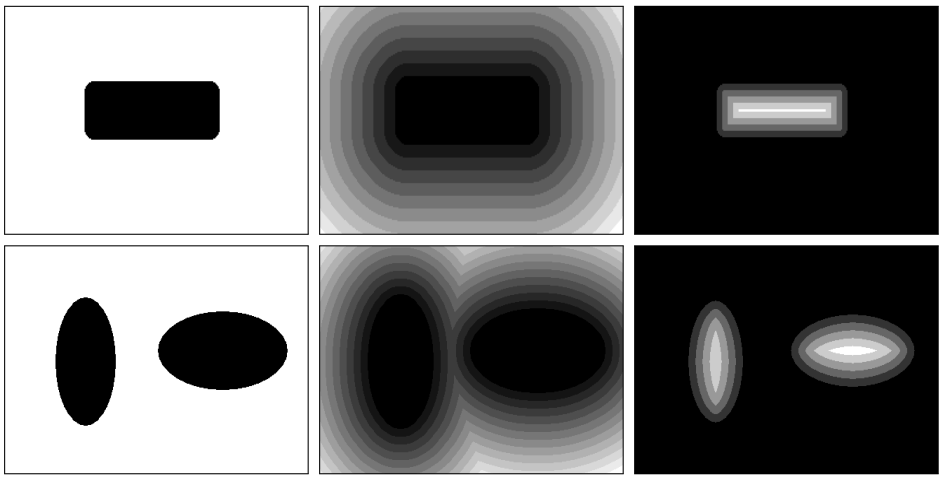


Figure 4.1: **From left to right**: binary source set, outer distance transform, inner distance transform.

and its background set $S^c \subseteq X$, where S^c denotes the complement $X \backslash S$ of set S. Given a metric d in $\mathbb{X}$, the *distance transform* of S is defined as the nonnegative function $D_d(S)$ whose value at each point $x \in X$ is the (infimum) distance between x and the foreground S:

$$D_d(S)(x) \triangleq \bigwedge_{y \in S} d(x - y) \tag{4.1}$$

where $\bigwedge$ stands for infimum (or minimum in the discrete case). Obviously, $D_d(S)(x)$ is zero for all $x \in S$. Thus, the foreground set S can be seen as a set of *sources* emanating a distance wave propagating away from the foreground and into the background. The distance transform as a wave propagation will be analyzed in section 4.1.2. From this point of view, definition (4.1) consists an *outer* distance transform and is useful when we are concerned about the geometrical structure and the morphology of the background set S^c. In case we are interested in the shape of the foreground set S, for instance when S represents an object, it is preferable to consider the *inner distance transform* $D_d(S^c)$ which uses the foreground set S to measure distances from the background set S^c. Two examples of euclidean distance transforms are illustrated in Fig. 4.1 and Fig. 4.2.

The distance transform has a wide range of applications in image processing, computer vision and robotics problems like smoothing, skeletonization, segmentation, size distributions, shape description, object detection and recognition, motion planning, shortest paths and even pathfinding.

4.1.2 Distance transform as wave propagation

Maragos [31] describes how a distance transform can be viewed as wave propagation. This formulation has been adopted in this section. Using Huygen's construction [12], the boundaries of multiscale dilations (erosions) by some convex struc-

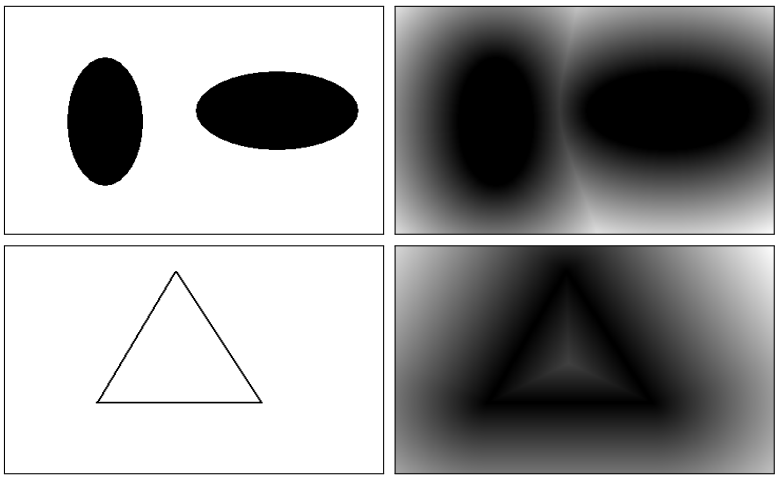


Figure 4.2: **From left to right**: binary source set, normalized (outer) distance transform.

turing elements can also be viewed as the wavefronts of a wave initiating from the original image boundary and propagating outwards. In case of euclidean distance, the structuring elements are disks and the wave propagates with constant normal speed, i.e. in a homogeneous medium.

To quantify this, assume the boundary $C(0) = \partial S$ of the set S is a smooth simple closed curve and let $\overrightarrow{C}_0(s)$ be a parametrization of the initial curve. Further, let $\overrightarrow{C}(s,t)$ represent the evolving boundary curve, i.e. wavefront, $\overrightarrow{C}(t)$ of the multiscale dilation or erosion of S by some structuring element $B(\lambda)$ of size λ. After dilation (erosion) of $\overrightarrow{C}(s,t)$ with the above structuring element, the boundary curve will be evolved to a new curve $C(s,t+\lambda)$. This evolution can be represented as [48]

$$\overrightarrow{C}(s,t+\lambda) - \overrightarrow{C}(s,t) = \Delta(s,t,\lambda)\overrightarrow{N}(s,t) \tag{4.2}$$

where $\overrightarrow{N}$ is the outward normal vector at each point of the curve and $\Delta(s,t,\lambda)$ is the distance along $\overrightarrow{N}$ that a point on the boundary moves in a dilation operation with a structuring element of size λ. Since $\Delta(s,t,0) = 0$:

$$\frac{\partial C}{\partial t}(s,t) = \lim_{\lambda\to 0}\frac{\Delta(s,t,\lambda) - \Delta(s,t,0)}{\lambda}\overrightarrow{N} = \left.\frac{\partial\Delta(s,t,\lambda)}{\partial\lambda}\right|_{\lambda=0}\overrightarrow{N} = \beta(s,t)\overrightarrow{N} \tag{4.3}$$

According to [5], the amount of differential deformation, β, of shape S at a point $x \in X$ due to dilation (erosion) with a convex structuring element B, is the maximal (minimal) projection of B onto the normal $\overrightarrow{N}$ of the boundary at x, so that β can only depend on the orientation of the target to the shape $\overrightarrow{T}(s,t)$, leading to the following differential evolution law:

$$\begin{aligned}\frac{\partial\overrightarrow{C}(s,t)}{\partial t} &= \beta(\overrightarrow{T}(s,t))\overrightarrow{N}(s,t)\\ \overrightarrow{C}(s,0) &= \overrightarrow{C}_0(s)\end{aligned} \tag{4.4}$$

If $\beta(\overrightarrow{T}(s,t)) = 1$, we obtain the dilations and the outer distance transform:

$$D_d(S)(x) = \inf\{t \geq 0 : x \in C(t)\} \tag{4.5}$$

Thus, by equating scale (distance) with time, the distance function has a minimum-of-arrival interpretation and its isolevel contours coincide with those of the wave phase function. By using a negative normal velocity $\beta(\overrightarrow{T}(s,t)) = -1$ in (4.4) the distance wavefront propagates inwards and creates the boundaries of multiscale erosions. According to Blum's *grassfire propagation* principle [11], points where these wavefronts intersect and extinguish themselves are the points of the Euclidean *medial axis* of S [10, 11]. Overall, the euclidean distance function of $D_2(S)$ is the weak solution of the following nonlinear partial differential equation:

$$\begin{aligned} \|\nabla u(x)\| &= 1 \quad x \in S^c \\ u(x) &= 0 \quad x \in \partial S \end{aligned} \tag{4.6}$$

This is a special case of the *eikonal PDE* which corresponds to wave propagation in heterogeneous media and whose solution u is a weighted distance function, where the weights $F(x)$ are inversely proportional to the varying propagation speed:

$$\begin{aligned} \|\nabla u(x)\|_2 &= F(x) \text{ in } \Omega, \quad F(x) > 0 \\ u(x) &= g(x) \text{ on } \Gamma \end{aligned} \tag{4.7}$$

where Ω is a domain in $\mathbb{R}^2$ or $\mathbb{R}^3$ and $F(x)$ is typically supplied as known input to the equation, as is the boundary condition that u equal a known function $g(x)$ given along a prescribed curve or surface Γ in X.

4.1.3 Distance transform as infimal convolution

The *infimal convolution* $f \star g$ of the two-dimensional signals f, g is defined as:

$$(f \star g)(x) \triangleq \bigwedge_{y \in X} g(x-y) + f(y), \quad x \in X \tag{4.8}$$

The distance transform of a function is closely related to the infimum convolution operation. One can easily observe that when $g(x-y) = \|x-y\|_d$ for a given metric d in $\mathbb{X}$, the distance transform of f is *exactly* the infimum convolution of f and g. Analytically, recalling the definition of distance transform of the foreground set S we have:

$$\begin{aligned} D_d(S)(x) &= \bigwedge_{y \in S} d(x-y), \quad x \in X \Rightarrow \\ D_d(S)(x) &= \bigwedge_{y \in X} d(x-y) + 1_S(y), \quad x \in X \Rightarrow \\ D_d(S)(x) &= (1_S \star d)(x) \end{aligned}$$

where $1_S(x)$ is the $0/\infty$ indicator function for membership in the set S:

$$1_S(x) = \begin{cases} 0, & x \in S \\ +\infty, & \text{otherwise} \end{cases}$$

4.2 Solving the eikonal equation

4.2.1 Fast marching methods

Fast marching methods are computational techniques introduced by Sethian [43] that approximate the solution to nonlinear eikonal equations of the form:

$$\begin{aligned} \|\nabla u(x)\| = F(x) \text{ in } \Omega, \quad F(x) > 0 \\ u = g(x) \text{ on } \Gamma \end{aligned} \tag{4.9}$$

where Ω is a domain in $\mathbb{R}^2$ or $\mathbb{R}^3$. Here, the right-hand side, $F(x) > 0$, is typically supplied as known input to the equation, as is the boundary condition that u equal a known function $g(x)$ given along a prescribed curve or surface Γ in Ω.

Equation (4.9) is part of a broader class of Hamilton-Jacobi equations of the form:

$$H(u_x, u_y, u_z, x, y, z) = 0 \tag{4.10}$$

In the case of the eikonal equation, the function H reduces to $H = |\nabla u(x)| - F(x)$.

First of all, the finite difference approximation that Sethian [43] uses in its formulation should be mentioned:

$$u_x^2 \approx \left(\max\left(D_i^{+x} u, 0\right)^2 + \min\left(D_i^{-x} u, 0\right)^2 \right) \tag{4.11}$$

where the standard finite difference notation has been used:

$$D_i^{-x} u = \frac{u_i - u_{i-1}}{h}, \quad D_i^{+x} u = \frac{u_{i+1} - u_i}{h} \tag{4.12}$$

Here, u_i is the value of u on a grid at the point ih with grid spacing h.

Extending the previous approximations for the gradient to multiple dimensions, results in the following scheme:

$$|\nabla u| \approx \begin{bmatrix} \max\left(D_{ij}^{-x} u, 0\right)^2 + \min\left(D_{ij}^{+x} u, 0\right)^2 \\ + \max\left(D_{ij}^{-y} u, 0\right)^2 + \min\left(D_{ij}^{+y} u, 0\right)^2 \end{bmatrix}^{1/2} = F_{ij} \tag{4.13}$$

or equivalently:

$$\begin{bmatrix} \max\left(D_{ij}^{-x} u, -D_{ij}^{+x} u, 0\right)^2 \\ + \max\left(D_{ij}^{-y} u, -D_{ij}^{+y} u, 0\right)^2 \end{bmatrix}^{1/2} = F_{ij} \tag{4.14}$$

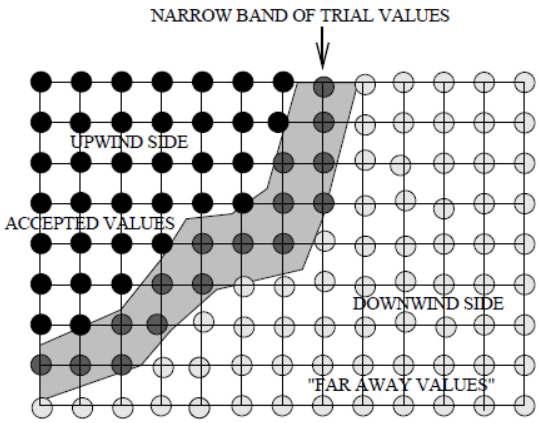


Figure 4.3: Upwind construction of accepted values (from [43])

One way to solve (4.14) is through iteration as it is a piecewise quadratic equation for u_{ij}, assuming that the neighboring grid values for u are given. Assuming $N \times N$ grid and N iterations until convergence, the computational complexity of the above method is $O(N^3)$, which is computationally expensive. Fast marching methods reduce this complexity to $O(N^2 \log N)$.

The central idea behind fast marching methods is to systematically construct the solution in a "downwind" fashion to produce the solution u. The upwind difference structure of (4.14) means that information propagates "one way", that is, from smaller values of u to larger values. Hence, the fast marching algorithm rests on "solving" (4.14) by building the solution outwards from the smallest u value. The algorithm is made fast by confining the "building zone" to a narrow band around the front. The idea is to sweep the front ahead in a downwind fashion by considering a set of points in a narrow band around the existing front, and to march this narrow band forward, freezing the values of existing points and bringing new ones into the narrow band structure (see Fig. 4.2.1). The key is in the selection of which grid point in the narrow band to update.

Another way to look at this scheme is that each minimum trial value begins an application of the Huygens principle, and the expanding wavefront touches and updates all others. The speed of the algorithm comes from a heapsort technique to efficiently locate the smallest element in the set *Trial*. Thus, the fast marching method is as follows: First, tag points in the initial conditions as *Alive*. Then tag as *Close* all points one grid point away. Finally, tag as *Far* all other grid points. Then the loop is as follows:

1. Begin Loop: let *Trial* be the point in *Close* with the smallest value of u.
2. Tag as *Close* all neighbors of *Trial* that are not *Alive*. If the neighbor is in *Far*, remove it from that list and add it to the set *Close*.
3. Recompute the values of u at all *Close* neighbors of *Trial* by solving the piecewise quadratic equation according to 4.14.

4. Add the point *Trial* to *Alive*; remove it from *Close*.
5. Return to top of Loop.

The key to an efficient version of the above technique lies in a fast way of locating the grid point in the narrow band with the smallest value for u. This is accomplished by the use of a min-heap data structure. In an abstract sense, a min-heap is a "complete binary tree" with a property that the value at any given node is less than or equal to the values at its children. Since the total work in changing the value of one element of the heap and bubbling its value upwards is $O(\log M)$, where M is the size of the heap, this produces a total operation cost of $O(M\log M)$ for the fast marching method on a grid of M total points. Thus, for a two-dimensional grid of $N \times N$ points, the fast marching method reduces the computational complexity from $O(N^3)$ to $O(N^2\log N)$; essentially, each grid point is visited once to compute its value.

4.3 Weighted distance transform and the exact group marching algorithm

In this section, the *additively weighted distance transform* will be presented along with an efficient linear time algorithm for its computation. To help readers understanding, definitions and main algorithm are analytically presented as in [7]. To begin with, given a metric d in $\mathbb{X}$ the *additively weighted distance transform* $D_d(f)$ of image f is defined as:

$$D_d(f)(x) = \bigwedge_{y \in X} d(x,y) + f(y), \quad x \in X, \tag{4.15}$$

Most often, one can use a metric induced by a norm $\|\cdot\|$, that is, $d(x,y) = \|x-y\|$ for $x, y \in \mathbb{X}$. Also, d can be omitted and one can simply write $\mathcal{D}(f)$ instead. Although (4.15) applies to arbitrary functions f, if the problem at hand is for instance image segmentation or contour detection, a function that is related to *boundaries*, like *gradient* or *contour map*, should be used.

The *weighted distance transform* has been studied primarily as a solution to the *eikonal equation*, in problems like shading from shape [49]. A given function specifies the *refractive index* on the plane, while a set of *source points* specifies boundary conditions. A given source map is *not* required here. The weighting mechanism is more similar to [19] where the distance map is obtained by an *infimal convolution* operation, equivalent to *weighted erosion* [32].

The *exact group marching* algorithm is a variant of *group marching* (GMM) [24], a linear-time fast marching method that selects a number of points on the propagating front to move as a group, thus avoiding the cost of sorting. *all* points are moved of the front as a group using a constant-time *priority queue* on *quantized* distance. This is more similar to [56], and in the binary case it would reduce to

the *two-queue* scheme of [32]. However, due to the Euclidean assumption and a bidirectional update, the entire computation is *exact*.

With a careful look at equations (4.15) and (4.8), it can be immediately concluded that the weighted distance transform is equivalent to the infimal convolution of a norm-induced metric d with the image f. Thus, the algorithm described here for distance transforms of sampled functions can be seen as a minimum convolution algorithm.

Now, given an image function f, the *minimal set* $\hat{S}(x)$ is defined for each point $x \in X$ as the set of points $y \in X$ for which quantity $d(x, y) + f(y)$ is minimized:

$$\hat{S}(x) = \{y \in X \,:\, d(x, y) + f(y) = \mathcal{D}(f)(x)\} \tag{4.16}$$

for $x \in X$. If $y \in \hat{S}(x)$, one can equivalently write $y \succcurlyeq x$. The *source set* $S(x)$ of x is defined as the subset of its minimal set such that no two points $y, z \in S(x)$ are related by $y \succcurlyeq z$:

$$S(x) = \{y \in \hat{S}(x) \,:\, \nexists z (y \succcurlyeq z \succcurlyeq x)\}. \tag{4.17}$$

A point y is a *source of* x, or equivalently $y \succ x$, iff $y \in S(x)$. More generally, $y \in X$ is a *source* iff $y \succ x$ for some $x \in X$, even itself. In this work, it is assumed that each $x \in X$ has at least one source: $y \succ x$ for some $y \in X$. This is always true in the discrete domain.

Lemma 4.3.1 *Given $y \in X$, the following are equivalent:*

(a) y is a source,

(b) $y \succcurlyeq y$.

(c) $\mathcal{D}(f)(y) = f(y)$,

(d) $S(y) = \{y\}$,

(e) $y \succ y$.

The *source set* $S(f)$ of f is defined as the set of all sources $y \in X$. It follows that $S(f) = \{x \in X \,:\, x \succ x\}$. This makes it easy to detect sources. By $s(x), x \in X$ is denote the source of x if it is unique, otherwise any representative of $S(x)$. Function $s : X \to X$ is called a *source map*.

Lemma 4.3.2 *The distance map $\mathcal{D}_d(f)$ is uniquely determined by the restriction $f|_{S(f)}$ of f on its source set.*

This is a generalization of an analogous observation on the binary distance map, which, for a binary input $B \subseteq \mathbb{X}$, is uniquely determined by its boundary ∂B. Source

sets are then closely related to region boundaries. Accordingly, the *interior set* of f is defined as $I(f) = X \setminus S(f)$.

Given an image f, we use the *exact group marching* (EGM) algorithm to compute the distance map $h = \mathcal{D}(f)$ according to (4.15) and the source map s in the discrete domain, using the Euclidean metric. EGM is outlined in algorithm 1. Propagation is initialized at the *source seed set* $S_+(f)$, defined as

$$S_+(f) = \{x \in X : f(x) < \min_{y \diamond x} f(y) + 1\}. \tag{4.18}$$

Because $d(x, y) = 1$ for $y \diamond x$, it can be shown that $S_+(f)$ is a superset of the source set $S(f)$.

Algorithm 1 Exact Group Marching

```
 1: procedure EGM(image f)
 2:     initialize q, h, s; construct seed S_+ as in (4.18)
 3:     for all x ∈ S_+ do {s(x) ← x; PROP(x, x);}
 4:     for all x ∈ X \ S_+ do { label x as far;}
 5:     while ¬ q.EMPTY( ) do
 6:         x ← q.POP( ); label x as done
 7:         for y◇x, y near do UPDATE(y, x)          ▷ incoming
 8:         for y◇x, y near do UPDATE(x, y)          ▷ outgoing
 9:         for y◇x, y far do PROP(x, y)
10:     end while
11:     return distance map h, source map s
12: end procedure
13:
14: procedure PROP(point x, point y)
15:     h(y) ← d(y, s(x)) + f(s(x));
16:     s(y) ← s(x);
17:     q.PUSH(y, ⌊h(y)⌋);
18:     label y as near;
19: end procedure
20:
21: procedure UPDATE(point x, point y)
22:     h_0 ← d(y, s(x)) + f(s(x));
23:     if h_0 ≥ h(y) return
24:     h(y) ← h_0;
25:     s(y) ← s(x);
26: end procedure
```

At the heart of propagation lies a *priority queue* with discrete priority levels, implemented as an array of internal FIFO queues. Points are labelled as `far`, `near`, or `done`. The queue holds points that are `near`, that is, points on the propagation front. Points are processed in *groups*: each point x is processed according to its level $\lfloor h(x) \rfloor$ and points with the same level at random order. Neighbors y that are

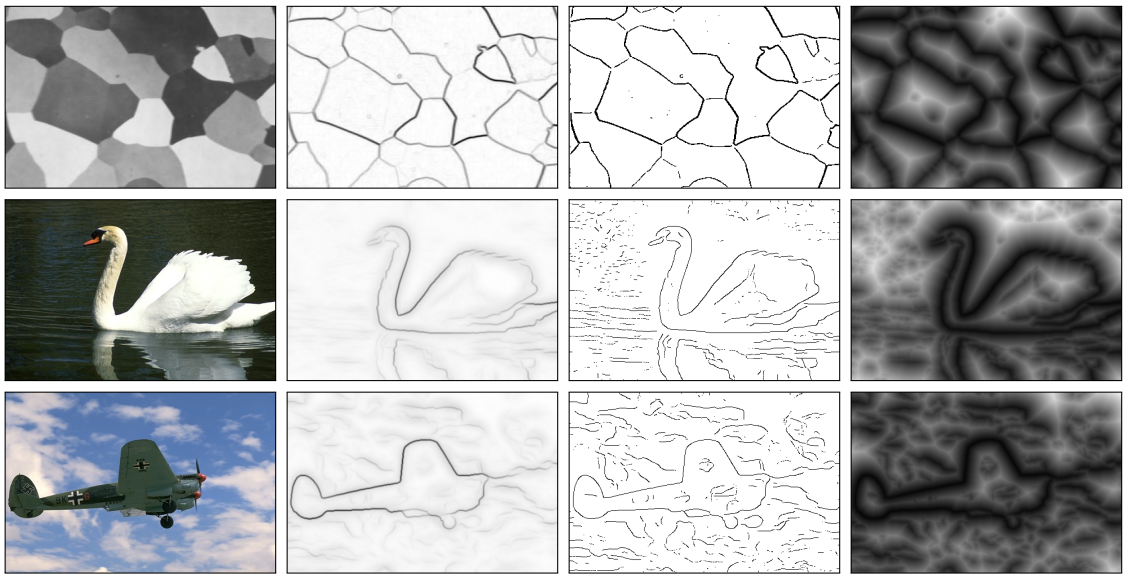


Figure 4.4: **From left to right**: input images, corresponding contour maps, binary source sets (in black) and normalized weighted distance transforms computed by the exact group marching algorithm.

`far` PROPagate the front; `near` ones participate in an UPDATE process *twice*, first in an incoming and then in an outgoing direction x. The computation is *exact*, despite the random processing order.

Proposition 4.3.3 *(a) EGM computes the exact distance $\mathcal{D}(f)(x)$ as defined in 4.15 and the correct source point $s(x)$ for each $x \in X$. (b) The **while** loop processes each x exactly once. (c) Its time complexity is $O(n)$, where $n = |X|$.*

What remains undefined so far is the function *height function* f one should use for distance computation. Here, we start from a contour detector output g and then $f(x) = \sigma/g(x)$ for $x \in X$, where σ is a *scale* parameter. This is a generalization of the $0/\infty$ *indicator function* used in binary distance transform. The contour detectors we used are the Canny detector and the gPb detector as described in the previous chapter.

In Fig. 5.4 some examples of the weighted distance transform using euclidean metric are illustrated. Computation is done by our implementation of the exact group marching algorithm. One can easily see that sources lie close to true image edges where images contour maps have high values.

Chapter 5

Medial axis

5.1 Introduction

The *medial axis transformation* is a technique first proposed by Blum [10] as a means to describe a figure and has been extensively used for shape representation and description [11]. It is formally defined as follows: given an object represented, say by a simple polygon G, the medial axis $A(G)$ is the set of points q internal to G such that there are at least two points on the object's boundary that are equidistant from q and are closest to q. Because of its shape, the medial axis of a figure is also called the *skeleton* or the *symmetric axis* of the figure. Associated with the medial axis is a radius function R, which defines for each point on the axis its distance to the boundary of the object (see Fig. 5.1). With the axis and the radius function one can reconstruct the figure by taking the union of all circles centered on the points comprising the axis, each with a radius given by the radius function.

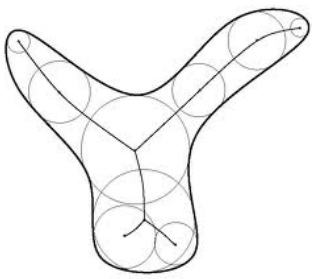


Figure 5.1: Illustration of a binary shape and its medial axis. Radii of drawn circles correspond to the distance of each medial axis point to the boundary.

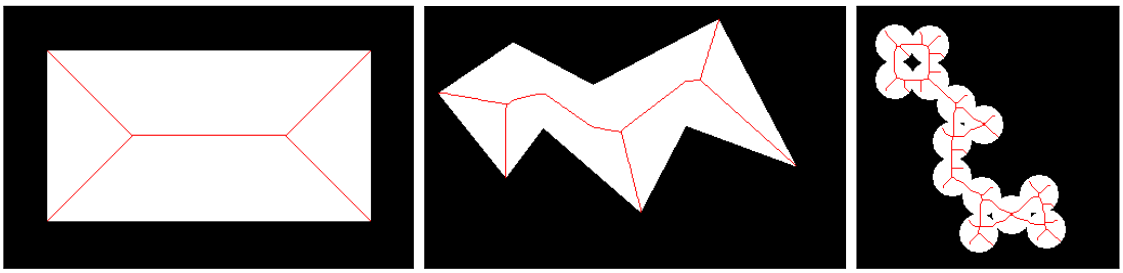


Figure 5.2: Binary shapes (in white) and corresponding medial axes (in red).

5.2 Weighted medial axis

Recently, Avrithis and Rapantzikos [7] studied the medial axis on a distance map weighted by infimal convolution. Rather than working on PDE's like [45], they use a *residue* criterion based on proximity of source points along boundaries. This is naturally connected to the definition of the medial axis and guarantees connectedness. They extend it from binary shapes to arbitrary functions in the plane and compute it with a similar *constant-time* operation. This approach is adopted here and will be analytically described.

Given the definitions of sources in a weighted distance map in section 4.3, $x \in X$ is a *medial point* of f if it has at least two distinct sources. The *weighted medial axis* or simply *medial axis* $A(f)$ is the set of all such points:

$$A(f) = \{x \in X \,:\, |S_f(x)| > 1\} \tag{5.1}$$

Lemma 5.2.1 *The source set and the medial axis of an image f are mutually exclusive: $S(f) \cap A(f) = \emptyset$. Hence the medial axis is contained in the interior set, $A(f) \subseteq I(f)$.*

The *medial axis transform* or *medial axis function* $\mathcal{A}(f)$ is defined as the restriction of the distance map $\mathcal{D}(f)$ on the medial axis: $\mathcal{A}(f) = \mathcal{D}(f)|_{A(f)}$. It is a subset of the (3D) *product space* $\mathbb{E} = \mathbb{X} \times \mathbb{V}$. The definitions above make sense only in the continuous domain. In the discrete domain, the following properties are applicable:

Lemma 5.2.2 *Let A be the medial axis of f in a Euclidean space, and let $x \in A$ and $y \in S(x)$.*
(a) Construct a parametrized, open line segment *from x to y. Then each point z on the segment has a unique source $s(z) = y$.*
(b) A has zero thickness, i.e. $A \subseteq \partial A$.

Given two neighboring points $x \diamond y$ with $s(x) \neq s(y)$, lemma 5.2.2(b) suggests there is a medial point m with $S(m) = \{s(x), s(y)\}$ on the line segment between

x, y. Therefore, *pair* (x, y) is labeled as medial. To deal with singularities in the distance map in the discrete domain, an extension of the *chord residue* criterion [39] is used.

The weighted medial axis (WMA) algorithm computes the medial axis $A(f)$ of image f given its weighted distance map $h = \mathcal{D}(f)$ and its source map s. Propagation starts with the *medial seed set* defined as:

$$A_+(f) = \{x \in X \,:\, h(x) \geq \max_{y \diamond x} h(y)\}, \tag{5.2}$$

and continue propagating downwards along $A(f)$ using a FIFO queue q. For each point x being processed, we SCAN 4-connected neighbors $y \diamond x$ to decide if (x, y) is a medial pair. We only PROPagate to x's 8-connected neighbors if x is found medial after SCANning. "Medialness" is recorded by means of *residue* $r(x) = \max_{y \diamond x} \mathrm{res}(x, y)$ for $x \in X$ and the medial axis is given by (**??**). Residue function res is discussed below.

$$A(f) = \{x \in X \,:\, r(x) > 0\} \tag{5.3}$$

Ogniewicz and Kübler [39] define chord residue for binary shapes only, as the difference between the length of a boundary curve segment and corresponding chord length of a circle that is contained in the shape and bitangent to the boundary curve at the two endpoints of the segment. The generalized distance map (4.15) is used and the distance value is seen as a third dimension, or *height*. Recalling lemma 4.3.2, Avrithis and Rapantzikos [7] define *source function* $\mathcal{S}(f)$ of f as the restriction of $\mathcal{D}(f)$ on the source set: $\mathcal{S}(f) = \mathcal{D}(f)|_{S(f)} = f|_{S(f)}$. Dually to the medial axis function, $\mathcal{S}(f) \subseteq \mathbb{E}$ is associated to *local minima* and *valleys* of the distance map. Circles are generalized to *cones* lying below and bitangent to $\mathcal{S}(f)$, and 2D curve segments in $\mathbb{X}$ to *3D paths* along $\mathcal{S}(f)$ in $\mathbb{E}$. Distances are measured with the *product metric* δ formed by the Euclidean metric d of 2D space $\mathbb{X}$ and the absolute difference of $1D$ space $\mathbb{V}$:

$$\delta(u, v) = d(u, v) + |h(u) - h(v)|, \quad u, v \in \mathbb{X}. \tag{5.4}$$

Now, given two points $x, y \in X$ with sources $u = s(x), v = s(y)$, the *chord residue* is generalized as:

$$\mathrm{res}(x, y) = \ell(u, v) - \delta(u, v) \tag{5.5}$$

The *length function* ℓ generalizes the *potential function* of [39] as the length of the shortest path (geodesic) connecting points $(u, f(u))$ and $(v, f(v))$ along the surface of the source function $\mathcal{S}(f)$ in space $\mathbb{E}$. Its computation is facilitated by the following.

Lemma 5.2.3 *The medial axis $A(f)$ is uniquely determined by the restriction $f|_{\partial S(f)}$ of function f on the* boundary *of its source set.*

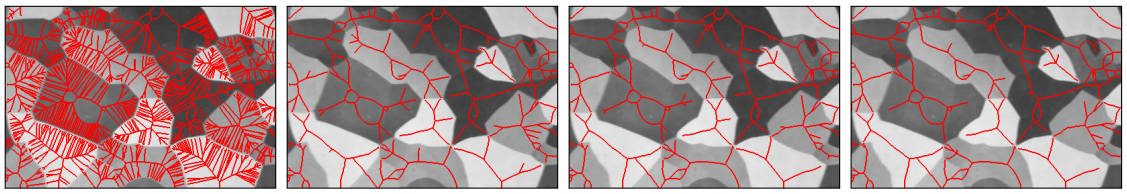


Figure 5.3: **From left to right**: medial axis (in red) computed with the WMA algorithm for *scale* $= 0, 1, 2, 5$.

In the discrete domain, the source set $S(f) = \{x \in X \,:\, x \succ x\}$ is first computed and follows the discrete boundary of the source set $S(f)$ w.r.t. 4-connectivity as

$$\partial S(f) = \{x \in S(f) \,:\, \exists y (y \diamond x \wedge y \in I(f))\}. \tag{5.6}$$

A weighted graph H is constructed as a subgraph of grid G with vertex set $V(H) = \partial S(f)$, and weight function $w(e) = \delta(u, v)$ for edge $e = (u, v) \in E(H)$. Its *components* and the *faces* of each component are computed. Then, seeing each face c as a cycle with start vertex v_0, we compute for each vertex v of c the *weight* $w_c(v)$ of path $(v_0, \ldots, v)$. Each vertex $v \in V(H)$ may belong to up to four faces. If $C(v)$ denotes the set of faces containing v, intersection $C(u, v) = C(u) \cap C(v)$ is either empty (if u, v belong to distinct components, in which case we define $\ell(u, v) = +\infty$) or contain exactly one common face c, associated to the component of $I(f)$ containing x, y. In the latter case:

$$\ell(u, v) = \min(\ell_c(u, v), w(c) - \ell_c(u, v)) \tag{5.7}$$

where $\ell_c(u, v) = |w_c(u) - w_c(v)|$ and $w(c)$ is the total weight of face c. This is a *constant-time* operation.

Lemma 5.2.4 *(a) Given point pairs (x, y), (x', y') in the same component of interior set $I(f)$ with source pairs (u, v), (u', v'), respectively, define paths $\pi = (u, \ldots, v)$, $\pi' = (u', \ldots, v')$. If $\pi \subset \pi'$, then $\mathrm{res}(x, y) < \mathrm{res}(x', y')$. (b) WMA generates exactly one component of $A(f)$ for each component of $I(f)$. (c) Its complexity excluding initialization is $O(k)$, where $k = |A(f)|$.*

Hence the residue function is increasing w.r.t. *inward* moves along the medial axis, and pruning is as simple as thresholding with parameter scale. Typically scale = 2 (pixels). In Fig. 5.3 the medial axis is illustrated for multiple values of scale. One can observe that minor changes occur for values above the default value of scale = 2.

In addition, we define for each point $x \in X$ its *medial pair* set $P(x)$ as the set of points that maximize the residue $r(x)$, or equivalently:

$$P(x) = \{y \diamond x : r(x) = \mathrm{res}(x, y)\} \tag{5.8}$$

and its computation is done in SCAN procedure of the WMA algorithm.

Algorithm 2 Weighted Medial Axis

```
1: procedure MEDIAL(distance map h, source map s)
2:     initialize q, r;
3:     construct A_+ as in (5.2);
4:     for x ∈ X do r(x) ← ∅; label x as far;
5:     for x ∈ X do if x ≻ x then label x as done;
6:     for x ∈ A_+ do PROP(x)
7:     while ¬ q.EMPTY( ) do
8:         x ← q.POP( );
9:         label x as done;
10:        for y◇x, ¬y done do SCAN(x, y)
11:        if r(x) ≠ 0 then for y∗x, y far do PROP(y);
12:    end while
13:    return residue r
14: end procedure
15:
16: procedure PROP(point x)
17:    q.PUSH(x);
18:    label x as near;
19: end procedure
20:
21: procedure SCAN(point x, point y)
22:    ρ ← res(x, y);
23:    if s(x) = s(y) ∨ ρ < scale then return
24:    if ρ > r(y) ∧ y far then PROP(y);
25:    r(x) ← max(r(x), ρ);
26:    r(y) ← max(r(y), ρ);
27: end procedure
```

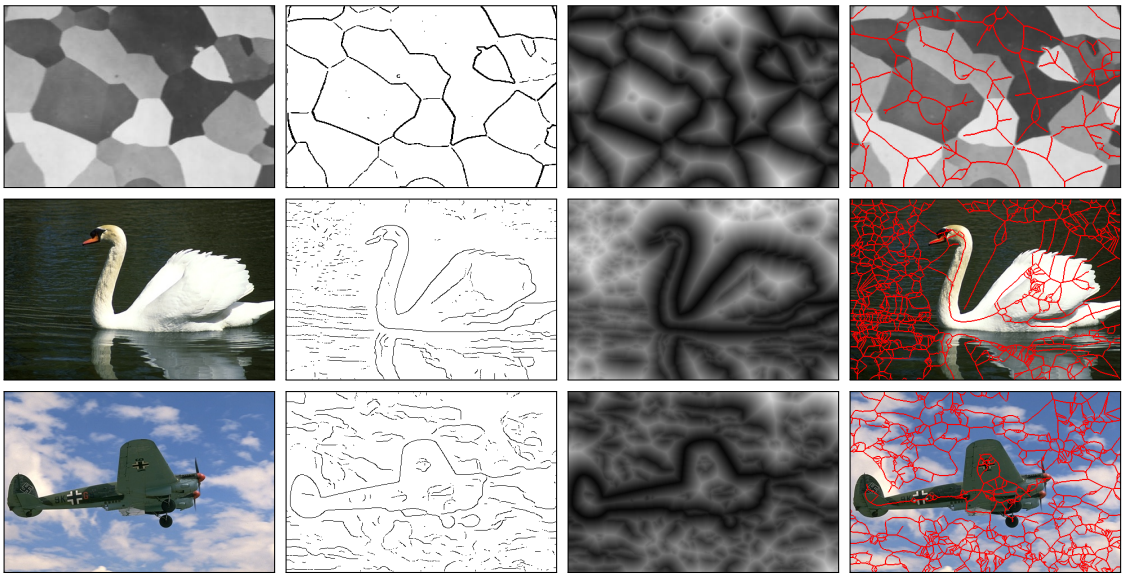


Figure 5.4: **From left to right**: input images, corresponding binary source sets (in black), normalized weighted distance transforms and medial axes(in red).

Chapter 6

Image partition

6.1 Medial Axis Decomposition

Medial axis decomposition methods are most often found in problems of computational geometry like domain decomposition [30] in binary images. The approach adopted in the framework of this thesis in order to decompose the weighted medial axis is the decomposition methodology developed by Avrithis and Rapantzikos [7] and it will be shortly presented in this section. It is closest to *watershed* segmentation applied to the distance map of binary regions [51], but using the *weighted distance map* of the gray-level input instead. The partitioning is fundamentally different from gray-level watershed, in that the latter is guided by image gradient.

While most work in the literature uses the medial axis to represent the shape of single object or image region, its usage, here, is to represent the *structure* of an entire image in terms of regions. Medial axis is decomposed into components and a corresponding weighted graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is constructed:

(a) Vertices $\mathcal{V}$ correspond to local maxima (peaks) of the distance map.

(b) Edges $\mathcal{E}$ correspond to local minima along the medial axis (i.e., along ridges), therefore to saddle points of the distance map.

(c) Edge weights $w(\mathcal{G}) : \mathcal{E} \rightarrow \mathbb{R}$ are given by a function of the height at saddle points.

As function f is related to image gradient or to grayscale contour map, peaks of the distance map correspond to the interior of image regions, and saddle points to adjacent region pairs, like mountain passes. Referring to Fig. 6.1, red components correspond to regions, each contains a peak, and each is represented in $\mathcal{G}$ as a vertex. Similarly, black points correspond to saddle points, and are each represented as an edge of graph $\mathcal{G}$. The *medial axis decomposition* (MAD) algorithm [6] constructs graph $\mathcal{G}$ given a distance map $h = \mathcal{D}(f)$ and the associated medial axis $A(f)$. Start

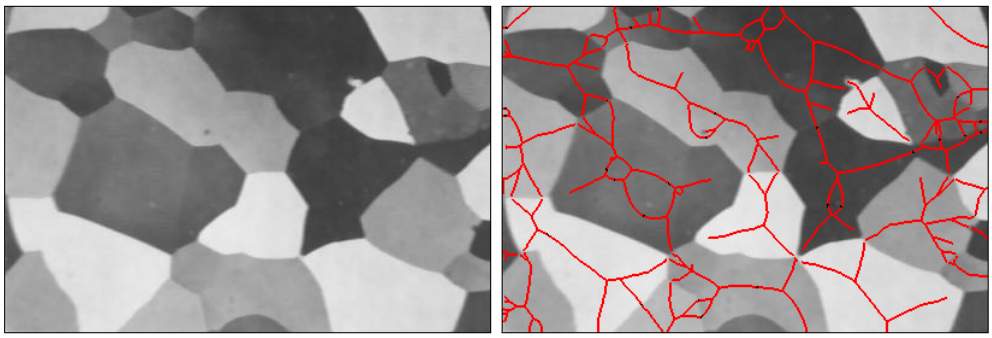


Figure 6.1: **Left**: input image. **Right**: each red component contains a peak and is represented in $\mathcal{G}$ as a vertex. Black points correspond to saddle points and are each represented as an edge of graph $\mathcal{G}$.

points are the distance peaks on the medial axis:

$$\hat{A}_+(f) = A_+(f) \cap A(f) \tag{6.1}$$

and propagation continues downwards as outlined in algorithm 3. A *priority queue* q is used again and propagation is performed to 8-connected neighbors according to height, as in EGM algorithm. However, the priority level is now *negated* in PROP, because of the downward direction. A component label $\kappa(x)$ is assigned to each $x \in X$, represented by a vertex of graph $\mathcal{G}$. Then, $\mathcal{G}$ is build by gradually inserting a VERTEX whenever a peak with unlabelled neighbors is visited for the first time and an EDGE whenever two fronts with distinct labels meet.

Propagation and component labelling in MAD is equivalent to applying watershed segmentation to the negated distance map restricted to the medial axis (i.e. on $A(f)$) with peaks as markers. However: (a) due to group marching, complexity is linear in k, where $k = |A(f)|$. (b) A single point per marker is ensured, even in flat areas (plateaus), in which case this point is chosen at random; effectively, the connected components of the markers are build in parallel to propagation. (c) The graph $\mathcal{G}$ is constructed again in parallel. (d) What is not shown in outline algorithm 3, is that an edge $e = (u, v)$ is contracted or equivalently u is identified with v whenever $|h(u) - h(x(e))| \leq 1$ or $|h(v) - h(x(e))| \leq 1$ in order to remove discretization effects along ridges while retaining true peaks, where $x(e)$ is the corresponding saddle point.

6.2 Image partition

Next, the entire image is partitioned via a reconstruction operation. A *duality* property, which reduces this operation to EGM algorithm, is exploited. Recall that the distance map $\mathcal{D}(f)$ applies to functions f defined on domain X whereas the medial axis function $\mathcal{A}(f)$ is restricted to subset $A(f) \subset X$. Given any function $f : U \to \mathbb{V}$, the *extension* operator is defined as $f|^X = f \cup ((X \setminus U) \times \{-\infty\})$, which

Algorithm 3 Medial Axis Decomposition

```
 1: procedure MAD(distance map h, medial axis A)
 2:     initialize q, 𝒢;
 3:     construct Â₊
 4:     for x ∈ A do κ(x) ← ∅; label x as far
 5:     for x ∈ Â₊ do PROP(x)
 6:     while ¬ q.EMPTY( ) do
 7:         x ← q.POP( );
 8:         label x as done;
 9:         for y∗x, y ∈ A do SCAN(x, y)
10:         if κ(x) = ∅ then κ(x) ← 𝒢.VERTEX(x)
11:     end while
12:     return graph 𝒢
13: end procedure
14:
15: procedure PROP(point x)
16:     q.PUSH(x, ⌊−h(y)⌋);
17:     label x as near;
18: end procedure
19:
20: procedure SCAN(point x, point y)
21:     if y far then PROP(y)
22:     if κ(y) = ∅ return
23:     if κ(x) = ∅ then κ(x) ← κ(y); return
24:     if κ(x) ≠ κ(y) then 𝒢.EDGE(κ(x), κ(y), w(x))
25: end procedure
```

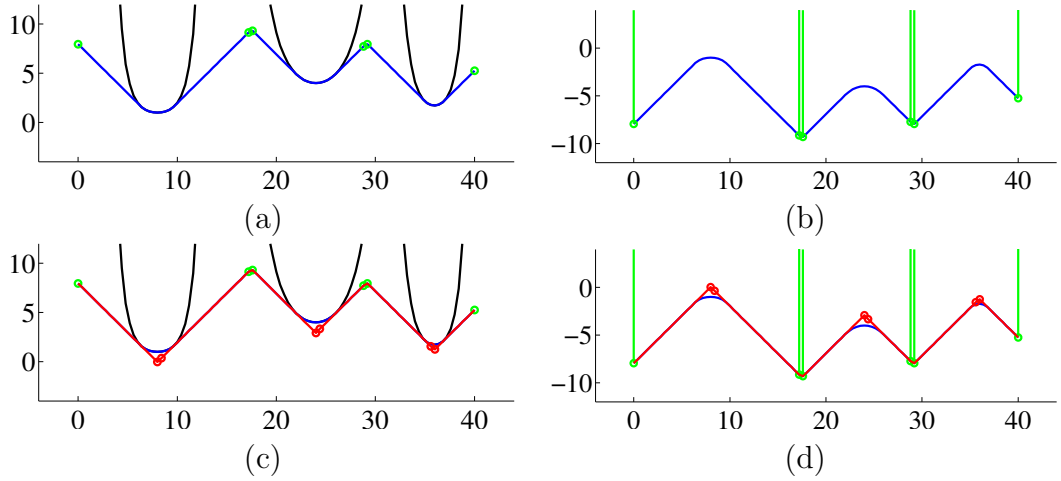


Figure 6.2: Illustrating duality of proposition 6.2.1 in one dimension. Functions in (b),(d) are negated versions of (a), (c); horizontal axis is X. (a) Black: f, blue: $\mathcal{D}(f)$, green dots: $\mathcal{A}(f)$. f is low at image boundaries, high inside regions. (b) Blue: red dots: $\mathcal{A}(-g)$. (c) Red dots: $\mathcal{S}(f') = -\mathcal{A}(-g)$. This is where fronts meet during partitioning.

extends its domain to X with value $-\infty$ wherever f is not defined. The *extended medial operator* is defined as $\mathcal{M}$ by $\mathcal{M}(f) = \mathcal{A}(f)|^X$ for $f \in \mathbb{F}$. Since $\mathcal{M}(f)$ is defined on domain X, distance or medial axis operators can be applied sequentially:

Proposition 6.2.1 *Given function f, let $g = \mathcal{M}(f)$ in a Euclidean space, define $f' = -\mathcal{M}(-g)$, $g' = \mathcal{M}(f')$. Then source function $\mathcal{S}$ and medial axis function $\mathcal{A}$ are* dual*:*

(a) $-\mathcal{S}(-g) = \mathcal{A}(f)$
(b) $\mathcal{S}(f') = -\mathcal{A}(-g) \subseteq S(f)$
(c) $g' = g$.

This result is quite condensed, but an one-dimensional example in Fig. 6.2 illustrates the idea. Proposition 6.2.1 suggests that the *extended boundary operator* $\mathcal{B}$ can be defined as $\mathcal{B}(f) = -\mathcal{M}(-f)$ for $f \in \mathbb{F}$. Then, similarly to morphological *erosion* and *dilation*, the two operators are *dual*. Also, similarly to *opening* (*closing*), composition $\mathcal{B} \circ \mathcal{M}$ ($\mathcal{M} \circ \mathcal{B}$) is idempotent and has fixed point f iff $f = \mathcal{B}(g)$ ($f = \mathcal{M}(g)$) for some g.

In practice, given the distance map $h = \mathcal{D}(f)$ and medial axis $A(f)$, EGM is invoked with input function g:

$$g(x) = \begin{cases} -h(x), & x \in A(f) \\ +\infty, & \text{otherwise} \end{cases} \tag{6.2}$$

Label map κ from MAD is used to construct *component* or equivalently *region* labels $\kappa(x)$ for all $x \in X$. *Initial Partition* $\mathcal{P}_0$ is produced by a label propagation

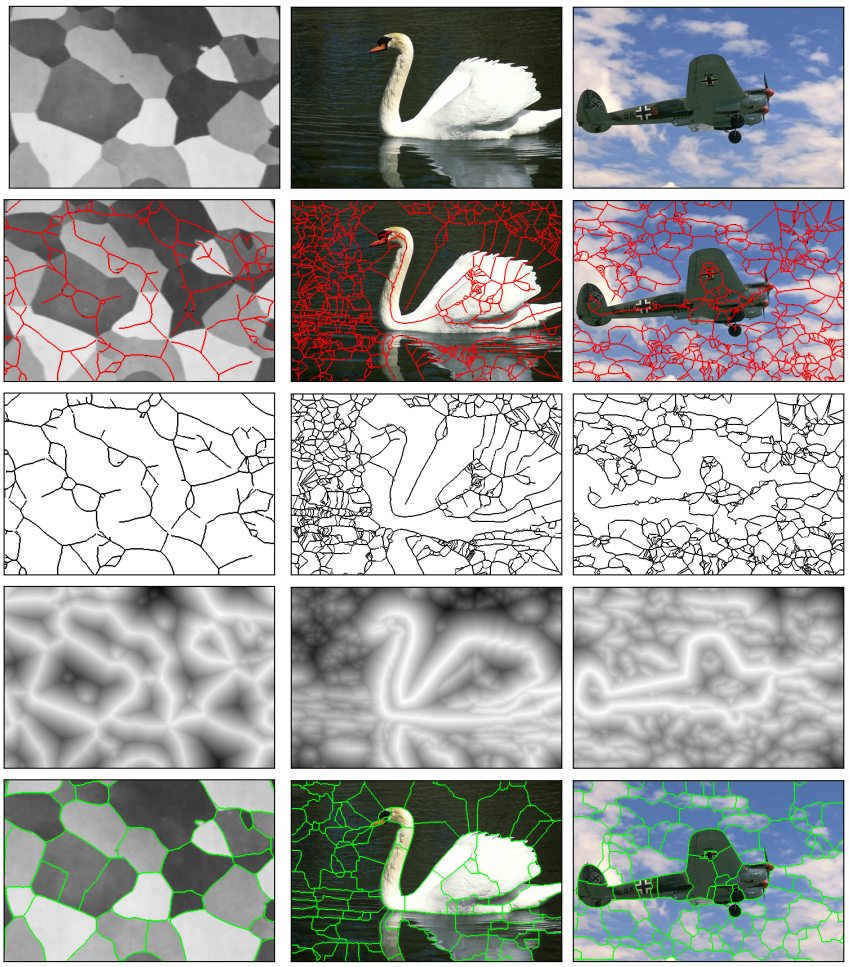


Figure 6.3: **From top to bottom**: input images, medial axis (in red) and corresponding saddle points (in black), dual source set which is identified with the medial axis as can be easily observed, dual distance map, (initial) partition.

operation:

$$\kappa(x) = \kappa(\hat{s}(x)), \qquad \text{for all } x \in X \tag{6.3}$$

where $\hat{s}(x)$ denotes the dual source of $x \in X$. The label map κ that corresponds to the initial partition $\mathcal{P}_0$ will be denoted as κ_0. Image is represented as a disjoint-set forest data structure where each pixel corresponds to a node. Initially, all nodes are in disjoint sets or equivalently each node has as parent the node itself. After the label propagation according to (6.3), each node in the disjoint-set forest has as parent its source point of the dual distance transform.

In Fig. 6.2 an example of the initial partition produced by the medial axis decomposition and the dual distance map. In addition, Fig. 6.4 an illustrates the EGM algorithm scale parameter σ effect to the finesse of the initial partition.

Figure 6.4: Initial partition for scale parameter of the exact group marching algorithm for $\sigma = 0.25$, 0.5, 1.0 and 2.0 respectively.

Chapter 7

Adjacent region merging

7.1 Efficient Merging based on similarity

The first merging technique we propose is an efficient merging technique based on a similarity measure between adjacent regions. Returning to line 24 of algorithm 3, we define the weight of each edge $e \in \mathcal{E}$ for this merging technique as:

$$w(e) = h(x(e)) \tag{7.1}$$

where $x(e)$ is the saddle point where e is generated. The choice of the above weight function as a measure of similarity will be shortly justified.

First of all, recall the definition of the weighted distance transform at point x:

$$h(x) = d(x, s(x)) + f(s(x)) \tag{7.2}$$

where $s(x)$ is the source point of x. Large value of a saddle point height potentially means that there is a large boundary discontinuity or equivalently a large gap in the binary source set. This discontinuity is associated with the term $d(x, s(x))$. In addition, it could demonstrate the existence of a weak boundary in case of a small value of $f(s(x))$. In any of the above cases, the saddle point height can be viewed as a non-negative measure of similarity between adjacent regions.

We define as similarity between two regions (components) $R, R' \subseteq X$ is defined as the *maximum* edge connecting the two regions. That is:

$$\mathrm{Sim}(R, R') = \max \{w(u, v) : u \in R, v \in R', (u, v) \in \mathcal{E}\} \tag{7.3}$$

If there is no edge connecting R and R' then let $\mathrm{Sim}(R, R') = 0$.

A region comparison predicate $Q(R, R')$ is introduced to evaluate whether a boundary exists between a pair of regions R, R'. If this predicate is true then the two regions should not be merged. The predicate checks if the similarity between

the components, $\mathrm{Sim}(R, R')$, is less or equal than a threshold function $\tau(R, R')$. The pairwise comparison predicate for the existence of a boundary is defined as:

$$Q(R, R') = \begin{cases} \text{true} , & \text{if } \mathrm{Sim}(R, R') < \tau(R, R') \\ \text{false} , & \text{otherwise} \end{cases} \tag{7.4}$$

The implemented algorithm , which we name *Efficient Similarity Merging* (ESM), has analogous basic steps as [18]. We first sort edge weights by non increasing edge weight. Then we process each weight in the latter order. Regions pairs are merged if the predicate Q is *false* for the two adjacent regions that each edge connects. A disjoint set forest representation ,as described in the previous section, is used along with union by rank and path compression techniques to improve running time. The main merging algorithm is outlined below.

Algorithm 4 ESM

```
1: procedure SEGMENTATION(graph 𝒢, initial label map κ_0)
2:     Sort ℰ = (e_1, ..., e_m) by non increasing edge weight.
3:     for q = 1, ..., m do
4:         Let e_q = (u, v), u ∈ R and v ∈ R'
5:         κ_u ← FIND(u)
6:         κ_v ← FIND(v)
7:         if ¬Q(R, R') then UNION(κ_u, κ_v)
8:     end for
9:     return label map κ
10: end procedure
11:
```

What remains undefined so far is the threshold function. The threshold function of two regions R_i and R_j is defined as the minimum between the threshold function of each region separately:

$$\tau(R, R') = \min\left(\tau(R), \tau(R')\right) \tag{7.5}$$

For the threshold function of one region, two choices have been used and tested. The first one is a simple constant, that is:

$$\tau(R) = \tau \tag{7.6}$$

where τ is a constant. On the other hand, the second choice of the threshold function is proportional to the region area, denoted as $|R|$. That is for a small region, a smaller evidence for similarity is required.

$$\tau(R) = |R| / k \tag{7.7}$$

where k is a constant and plays the role of a scale factor.

Lemma 7.1.1 *If an edge e_q and the two corresponding distinct region are considered and not merged then at least one of the two components will be in the final segmentation.*

Lemma 7.1.2 *The segmentation S produced by 5 is not too fine according to 2.2.1, using the predicate Q.*

Lemma 7.1.3 *The segmentation S produced by 5 is not too coarse according to 2.2.1, using the predicate Q.*

Detailed proofs of the above lemmas are presented in **Appendix** A.

7.2 Merging controlled by region fragmentation

In this section we present an alternative merging technique based on the fact that resulting regions from any segmentation procedure are *closed*. In addition, image representation we adopted in this framework is able to measure a region's boundary closure or inversely a region boundary fragmentation. Thus, we see image segmentation as a search for most closed region boundaries. We introduce a new logical predicate for adjacent region merging based on region boundaries fragmentation.

The source set may frequently become disconnected or fragmented. Gaps appear either due to variation of f along edges, or to region shape. Medial axis decomposition helps overcome fragmentation because for every gap there is associated a local minimum of the distance map along the medial axis, that is, a saddle point. The surrounding saddle points give rise to edges of graph $\mathcal{G}$. Returning ,again, to line 24 of algorithm 3 we define a different edge weight function which we consider as more appropriate for this method. We now define the weight $w(e)$ for each edge $e \in \mathcal{E}$ as the width of the associated gap of the source set. If $x = x(e)$ denotes the saddle point where e is generated and $y \in P(x)$ one point of the medial pair set of x, then the edge weight $w(e)$ or equivalently the *width of the associated gap* can be written as:

$$w(e) = d(x, s(x)) + d(y, s(y)) \Leftrightarrow \tag{7.8}$$

$$w(e) = \underbrace{h(x) - h(s(x))}_{d(x,s(x))} + \underbrace{h(y) - h(s(y))}_{d(y,s(y))} \tag{7.9}$$

Medial pair set may contain more than one points and four at most. In case there are more than one medial pairs of x, its choice does not significantly affect the associated width measurement. Thus, in practice we randomly choose one out of these medial pairs to measure the gap width.

Given a region R with area $a(R)$ and corresponding edge set $E(R)$, its *shape fragmentation factor* is defined as [7]:

$$\phi(R) = \frac{1}{a(R)} \sum_{e \in E(R)} w^2(e) \tag{7.10}$$

whereas $\phi(R) = 0$ if $E(R) = \emptyset$. This factor is a dimensionless, scale invariant quantity. It is an increasing function of both the width of the gaps and their cardinality (for constant sum of widths), and is identically zero for closed shapes.

With a single iteration through each edge $e \in \mathcal{E}$, we compute the sum of squared gap widths appearing in (7.10) for each component, prior to region merging.

Now, we define the predicate for merging two adjacent regions R, and R' :

$$Q(R, R') = \max\left(\phi(R'), \phi(R)\right) > \tau \tag{7.11}$$

where threshold τ is a constant. At this point it should be underlined that when a merging between a pair of adjacent regions (R, R') occurs, then the new region's $R \cup R'$ fragmentation factor is computed as following:

$$\phi(R \cup R') = \frac{\sum_{e \in E(R)} w^2(e) + \sum_{e \in E(R')} w^2(e) - 2 \cdot \sum_{e \in E(R) \cap E(R')} w^2(e)}{a(R) + a(R')} \tag{7.12}$$

The merging process is similar to the one described in the previous section. Again, we first sort edge weights by non increasing edge weight and then we process each weight in the latter order. Regions pairs are merged if the predicate Q is *true* for the two adjacent regions that each edge connects. The predicate is true if at least one of them has fragmented boundaries, i.e. it has a fragmentation factor greater than a constant τ. Our *Shape Fragmentation based Merging* (SFM) algorithm is presented below.

Algorithm 5 SFM

procedure MERGING(graph $\mathcal{G}$, initial label map κ_0)
  Sort $\mathcal{E} = (e_1, \ldots, e_m)$ by non increasing edge weight.
  for $q = 1, \ldots, m$ **do**
    Let $e_q = (u, v)$, $u \in R$ and $v \in R'$
    $\kappa_u \leftarrow$ FIND(u)
    $\kappa_v \leftarrow$ FIND(v)
    if $Q(R, R')$ **then**
      UNION(κ_u, κ_v)
      UPDATE$(\phi(R \cup R'))$ according to (7.12)
    end if
  end for
  return label map κ
end procedure

7.3 Hierarchical segmentation and ultrametric contour maps

Arbelaez [2] defines the notion of ultrametric contour maps that are used to produce hierarchical image partitions. The main idea is to represent the whole process of partition from the finest level (the initial oversegmentation) to the coarser level (entire image is one segment) in a single grayscale image. From this grayscale image named ultrametric contour map one can obtain segmentation at any desired scale by a simple thresholding operation. The definitions and description that follows can be found at [2] and are included in this section to improve readers comprehension.

Let $\mathcal{P}_0$ denote an initial partition of image domain X and $\lambda \in \mathbb{R}$ a scale parameter. A *Hierarchical Segmentation Operator* (HSO) is a mapping between a partition $\mathcal{P}_\lambda$ to $(\mathcal{P}_0, \lambda)$ in such a way that the three following properties are satisfied:

$$\mathcal{P}_\lambda = \mathcal{P}_0, \quad \forall \lambda \leq 0 \tag{7.13}$$

$$\exists \lambda_1 \in \mathbb{R}^+ : \mathcal{P}_\lambda = \{X\}, \quad \forall \lambda \geq \lambda_1 \tag{7.14}$$

$$\lambda \leq \lambda^{'} \Rightarrow \mathcal{P}_\lambda \sqsubseteq \mathcal{P}_{\lambda'} \tag{7.15}$$

where symbol $\sqsubseteq$ denotes the partial order of partitions, such that $\mathcal{P} \sqsubseteq \mathcal{P}'$ iff:

$$\forall R_i \in \mathcal{P}, \exists R_i^{'} \in \mathcal{P}' : R_i \subseteq R_i^{'} \tag{7.16}$$

According to (7.16), sets of partitions at different scales are nested and impose a hierarchical structure to the family:

$$\mathcal{H} = \{R \subseteq X \mid \exists \lambda : R \in \mathcal{P}_\lambda\} \tag{7.17}$$

The *stratification index* is defined as the scale λ at which a region appears in $\mathcal{H}$:

$$I(R) = \inf \{\lambda \in [0, \lambda_1] : R \in \mathcal{P}_\lambda\} \tag{7.18}$$

Pair $(\mathcal{H}, I)$ is called an indexed hierarchy of subsets of X.This indexed hierarchy can be represented by a dendogram, where the height of each region is given by the stratification index $I(R)$. The construction of the above hierarchy is equivalent to the definition of a distance or a metric between two elements $x, y \in X$:

$$\mathcal{Y}(x, y) = \inf\{I(R) \mid x \in R \wedge y \in R \wedge R \in \mathcal{H}\} \tag{7.19}$$

Metric $\mathcal{Y}$ belongs to a special type of distances, called *ultrametrics*, which in addition to the triangle inequality they satisfy the following property:

$$\mathcal{Y}(x, y) \leq \max\{\mathcal{Y}(x, z), \mathcal{Y}(y, z)\}, \quad x, y, z \in X \tag{7.20}$$

Alternatively, segmentation can be expressed in terms of contours. A segmentation K of domain X can be defined as a finite set of Jordan curves, the *contours*

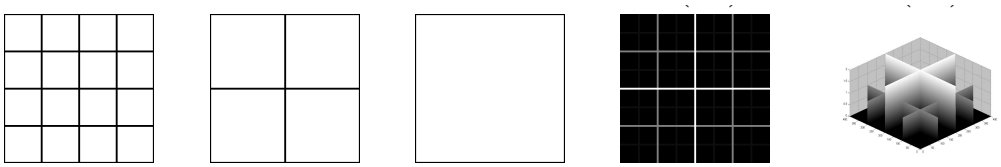


Figure 7.1: **From left to right**: Family of segmentations defined by a HSO, UCM and 3D view of UCM.

of K. The regions R_i of K are the connected components of $X \setminus K$. The contour separating adjacent regions R_i and R_j is denoted as ∂_{ij}.

Hierarchical segmentation operation can be dually expressed in terms of *contours*. Thus, equations (7.13) to (7.14) can be rewritten as following:

$$\mathcal{K}_\lambda = \mathcal{K}_0, \quad \forall \lambda \leq 0 \tag{7.21}$$

$$\exists \lambda_1 \in \mathbb{R}^+ : \mathcal{K}_\lambda = \partial X, \quad \forall \lambda \geq \lambda_1 \tag{7.22}$$

$$\lambda \leq \lambda^{'} \Rightarrow \mathcal{K}_\lambda \supseteq \mathcal{K}_{\lambda'} \tag{7.23}$$

Property (7.21) determines the set of initial contours K_0 which are related to the initial partition P_0. Property (7.22) demonstrates that all inner contours of domain X vanish at a scale λ_1. According to the third property (7.23), localization of contour is preserved through different scales.

Let $\mathcal{Y}$ be the ultrametric distance defined by a Hierarchical Segmentation Operator (HSO). The *ultrametric contour map* (UCM) associated to $\mathcal{Y}$ is defined as:

$$\mathcal{C}(\mathcal{Y})(\partial) = \inf\{\lambda \in [0, \lambda_1] \mid \partial \nsubseteq K_\lambda\}, \quad \forall \partial \in K_0 \tag{7.24}$$

The number $\mathcal{C}(\mathcal{Y})(\partial)$ is called the *saliency* of contour ∂. Note the duality with the regions, the saliency of ∂ being its scale of disappearance from the hierarchy of contours. The ultrametric contour map is a representation of a HSO in a single real valued image. Figure 7.3 presents a simple example of UCM. By definition, thresholding this soft boundary image at scale λ provides a set of closed curves, the segmentation K.

Starting from a family of nested partitions constructed by a region merging process, one can always define an ultrametric distance by considering as stratification index an increasing function of the merging order. However, in order to define a meaningful notion of scale, the distance between any two points in adjacent regions should coincide with the inter-region dissimilarity δ:

$$\mathcal{Y}(x, y) = \delta(R_i, R_j), \forall x \in R_i, \forall y \in R_j \tag{7.25}$$

This property is satisfied by setting the value of the dissimilarity at the creation of a region as its stratification index. However, for an arbitrary dissimilarity, this choice can lead to the existence of two regions $\left(R, R^{'}\right) \in \mathcal{H}^2$ such that $R \subset R^{'}$

but $f(R) > f(R^{'})$. In terms of contours, this case implies the violation of property (7.23))

Hence, we call δ an ultrametric dissimilarity if the pair $(\mathcal{H}, I)$ is an indexed hierarchy, where I is defined by:

$$I(R_i \cup R_j) = \delta(R_i, R_j) \tag{7.26}$$

for all pairs of connected regions $(R_i, R_j) \in \mathcal{H}^2$.

One can then prove that a dissimilarity δ is ultrametric if and only if:

$$\delta(R_i, R_j) \leq \delta\left(R_i \cup R_j, R_k\right) \tag{7.27}$$

where (R_i, R_j) is the region pair minimizing δ and R_k is a region connected to $R_i \cup R_j$ and appearing in the partition obtained after the merging of (R_i, R_j)).

Working with ultrametric dissimilarities, as those defined in the next section, is important for our application because it guarantees that the saliency of each contour in the UCM is *exactly* the value of the dissimilarity between the two regions it separates.

The algorithm we implemented for region merging using ultrametric contour maps proceeds as following. The hierarchy of regions is constructed by a greedy graph-based region merging algorithm. A graph $G = (\mathcal{P}, \mathcal{K})$ is defined where the nodes are the regions $\mathcal{P}$, the edges correspond to contours $\mathcal{K}$ separating adjacent regions and the weights W are a measure of *ultrametric* dissimilarity between regions. The above graph is initialized by the initial partition produced by MAD algorithm. In addition, it produces as output the corresponding Ultrametric Contour Map (UCM) which is initially set to zero for all elements in image domain. The algorithm proceeds by sorting the links by similarity and iteratively merging the most similar regions. The merging process can be summarized in the following main steps:

1) Select minimum weight contour: C^*
2) Let $R_i, R_j \in \mathcal{P}$ be the regions separated by C^*
2) $UCM(x) = W(C^*), \quad \forall x \in \partial_{ij}$
4) Set $R = R_i \cup R_j$ and update: $\mathcal{P} \leftarrow \mathcal{P} \backslash \{R_i, R_j\} \cup R$ and $\mathcal{K} \leftarrow \mathcal{K} \backslash \{C^*\}$
5) Stop if $\mathcal{K}$ is empty. Otherwise, *update weights* W and repeat.

As we have computed the ultrametric Contour Map, one can obtain the final segmentation by simply thresholding UCM at a desired scale.

7.3.1 Inter-region fragmentation ultrametric dissimilarity

In this section, we exploit the rich image representation provided by the medial axis decomposition method we are based on. Specifically, we define the *inter-region fragmentation factor* between two adjacent regions R_i and R_j as the sum of the

gap widths associated with their common boundary divided by the length of their common boundary. That is:

$$\Phi(R_i, R_j) = \frac{\sum_{e \in E(R_i) \cap E(R_j)} w(e)}{L(\partial_{ij})} \tag{7.28}$$

A large value of the inter-region fragmentation demonstrates the absence of a strong and continuous boundary between two regions. Thus, it expresses similarity between corresponding regions. Now, we define the ultrametric dissimilarity δ_Φ which is based on the inter-region fragmentation:

$$\delta_\Phi(R_i, R_j) = \exp\left(-\Phi(R_i, R_j)\right) \tag{7.29}$$

Lemma 7.3.1 *The dissimilarity δ_Φ is ultrametric.*

Lemma 7.3.1 is analytically proved in the Appendix A.

7.3.2 Mean Boundary Gradient Ultrametric Dissimilarity

In addition to the previous ultrametric dissimilarity, we experimented on and implemented an ultrametric dissimilarity based on the mean boundary strength between adjacent regions. Using mean boundary strength is a common approach [2, 4] to express adjacent region dissimilarity. The boundary strength ultrametric dissimilarity is defined as:

$$\delta_g(R_i, R_j) = \frac{\sum_g(\partial_{ij})}{L(\partial_{ij})} \tag{7.30}$$

where $\sum_g(\partial_{ij}) = \int_{\partial_{ij}} g(x(s))ds$, that is the sum of image gradient or the sum of a grayscale contour output g along the common boundary.

Lemma 7.3.2 *The dissimilarity δ_g as defined in equation (7.30) is ultrametric.*

Lemma 7.3.2 is analytically proved in the Appendix A.

Chapter 8

Evaluation

8.1 Introduction

When developing a scientific method in any field, it is more than essential to test how well the particular method works according to some objective criteria. This point is eloquently underlined by Lord Kelvin words: "When you can measure what you are speaking about and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of the meager and unsatisfactory kind". Thus, in this chapter we evaluate our method, compare it to other methods and discuss the pros and the cons of each merging technique.

8.2 Dataset

To evaluate our method we used the Berkeley Segmentation Dataset which is the most popular and wide known dataset for the purpose of image segmentation. The original Berkeley Segmentation Dataset (BSDS300) consists of 300 natural images, manually segmented by a number of different subjects. The ground-truth data for this large collection shows the diversity, yet high consistency, of human segmentation. The images are divided into a training set of 200 images, and a test set of 100 images. A new dataset (BSDS500) is an extension of the BSDS300, where the original 300 images are used for training / validation and 200 fresh images, together with human annotations, are added for testing. Each image was segmented by five different subjects on average.

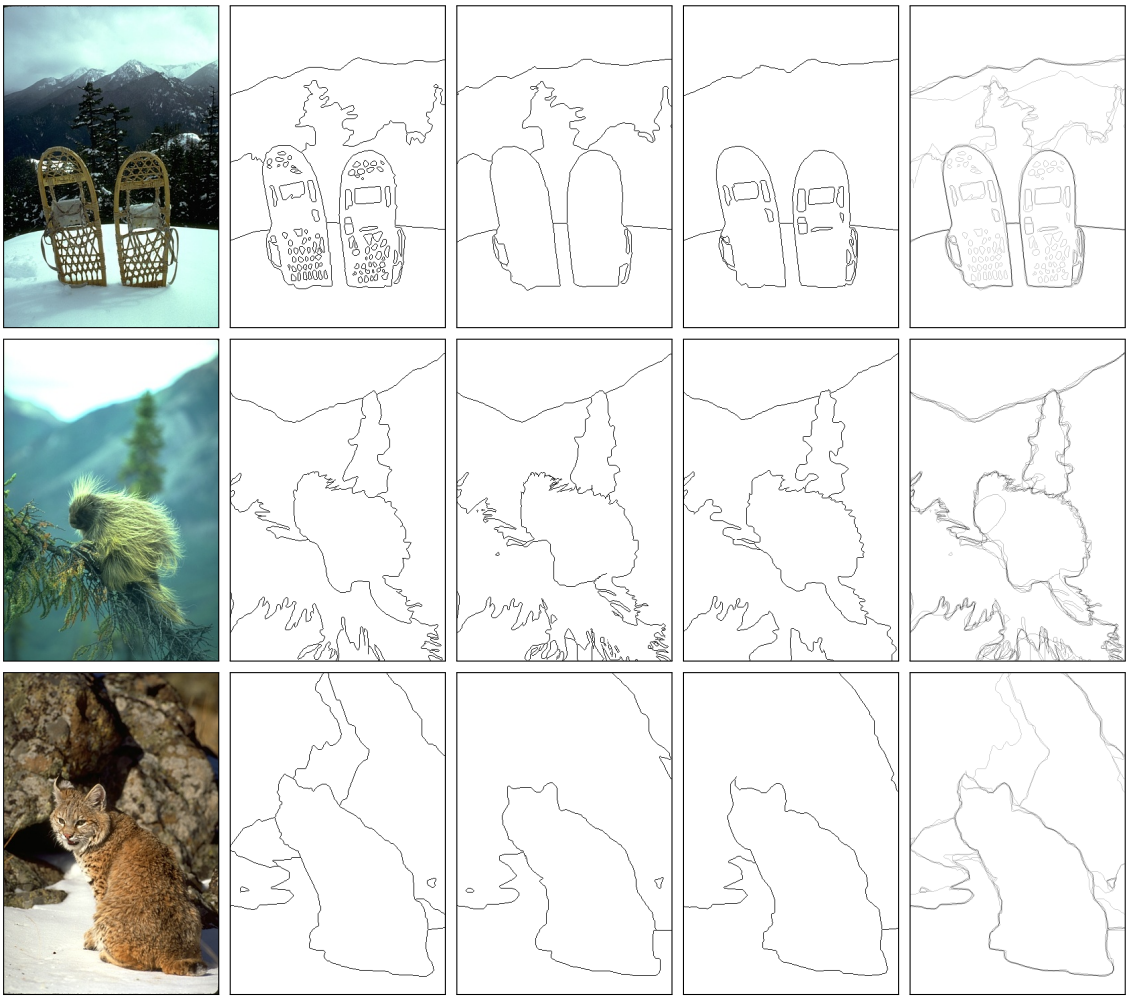


Figure 8.1: Human ground-truth of Berkeley segmentation dataset

8.3 Evaluation metrics

In the following sections, several metrics for evaluating both boundaries and regions against human ground-truth are presented as described in the documentation of the BSDS500 [4].

8.3.1 Precision-recall framework

In the field of information retrieval, precision is the fraction of retrieved items that are relevant to the search:

$$Precision = \frac{|\{\text{Relevant items}\} \cap \{\text{Retrieved items}\}|}{|\{\text{Retrieved items}\}|} \tag{8.1}$$

Recall in information retrieval is the fraction of relevant items that are successfully retrieved:

$$Recall = \frac{|\{\text{Relevant items}\} \cap \{\text{Retrieved items}\}|}{|\{\text{Relevant items}\}|} \tag{8.2}$$

The *precision-recall* framework has been used to evaluate the output of a contour detector or a segmentation algorithm as both outputs are image regions boundaries. This framework considers two aspects of boundary detection performance. *Precision* (P) measures the fraction of true positives in the boundaries produced by a detector or a segmentation method. *Recall* (R) measures the fraction of ground-truth boundaries detected. Thus, *precision* quantifies the amount of noise in the output of a contour detection or a segmentation method, while *recall* quantifies the amount of ground-truth detected. Computation of precision and recall is performed as following: the machine boundary map is separately corresponded with each human map in turn. Only those machine boundary pixels that match no human boundary are counted as false positives. The hit rate is simply averaged over the different humans, so that to achieve perfect recall the machine boundary map must explain all of the human data. For detectors that provide real-valued outputs, one obtains a curve parametrized by detection threshold, quantifying performance across operating regimes. The global F-measure, defined as the harmonic mean of precision and recall, provides a useful summary score for the algorithm. The F-measure is defined as:

$$F = \frac{2PR}{P + R} \tag{8.3}$$

8.3.2 Variation of information

The *Variation of Information* metric was introduced for the purpose of clustering comparison. It measures the distance between two clusterings $\mathcal{C}$ and $\mathcal{C}'$ in terms of

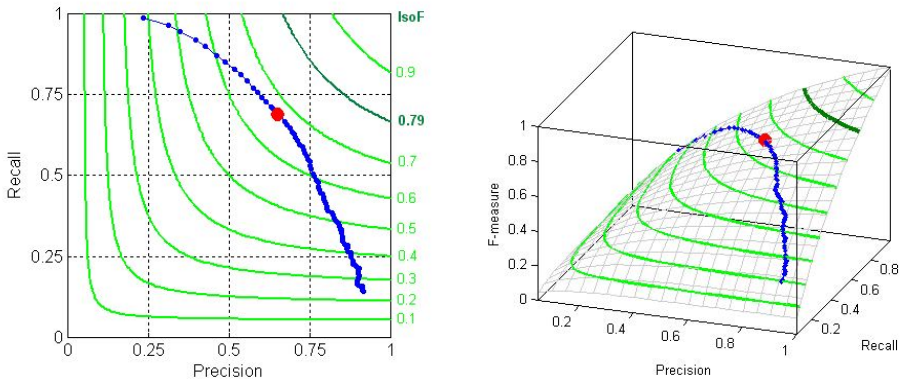


Figure 8.2: The Precision-recall curve is presented in blue. The light green curves represent the isolevel lines of the function $F(P, R)$. They can be used to situate the Precision-Recall curves, since the quality of a detector is judged by considering the image of its curve by the function F. Thus, the point with *maximal* F-measure, in red, can be interpreted as the highest point of the curve on the surface $z = F(P, R)$, the objective being the point $F(1, 1) = 1$. The isolevel line $F(P, R) = 0.79$, in dark green, corresponds to the human consistency on the test data set of the BSDS300, obtained by comparing the human segmentations among them. This line represents the reference with respect to which the performance of machines for the task of boundary detection is measured. (From [1]).

their average conditional entropy given by:

$$VI(\mathcal{C}, \mathcal{C}') = H(\mathcal{C}) + H(\mathcal{C}') - 2I(\mathcal{C}, \mathcal{C}') \tag{8.4}$$

where H and I represent respectively the entropies and mutual information between two clusterings of data $\mathcal{C}'$ and $\mathcal{C}'$. In the case of image segmentation, these clusterings are test and ground truth segmentations. Although VI possesses some interesting theoretical properties, its perceptual meaning and applicability in the presence of several ground-truth segmentations remains unclear.

8.3.3 Rand index

Originally, the *Rand Index* was introduced for general clustering evaluation. It operates by comparing the compatibility of assignments between pairs of elements in the clusters. The *Rand Index* between test and ground truth segmentations S and G is given by the sum of the number of pairs of pixels that have the same label in S and G and those that have different labels in both segmentations, divided by the total number of pairs of pixels. Variants of the Rand Index have been proposed for dealing with the case of multiple ground-truth segmentations. Given a set of ground-truth segmentations $\{G_k\}$, the *Probabilistic Rand Index* is defined as:

$$PRI(S, \{G_k\}) = \frac{1}{T} \sum_{i<j} [c_{ij} p_{ij} + (1 - c_{ij})(1 - p_{ij})] \tag{8.5}$$

where c_{ij} is the event that pixels i and j have the same label and p_{ij} its probability. T is the total number of pixel pairs. Using the sample mean to estimate p_{ij}, amounts to averaging the Rand Index among different ground-truth segmentations. The PRI has been reported to suffer from a small dynamic range and its values across images and algorithms are often similar. In, this drawback is addressed by normalization with an empirical estimation of its expected value.

8.3.4 Segmentation covering

The *overlap* between two regions R_1 and R_2 is defined as:

$$\mathcal{O}(R_1, R_2) = \frac{|R_1 \cap R_2|}{|R_1 \cup R_2|} \tag{8.6}$$

The covering of a segmentation S_1 by a segmentation S_2 is defined as:

$$\mathcal{C}(S_1 \to S_2) = \frac{1}{N} \sum_{R \in S} |R| \cdot \max_{R_2 \in S_2} \mathcal{O}(R_1, R_2) \tag{8.7}$$

where N denotes the total number of pixels in the image.

Similarly, the covering of a machine segmentation S by a family of ground-truth segmentations is defined by first covering S separately with each human segmentation $\{G_k\}$, and then averaging over the different humans. To achieve perfect covering the machine segmentation must explain all of the human data.

8.4 Results

In the following figures we show some representative results of the segmentation techniques we implemented. Results that follow are produced using the gPb contour detector as input to our framework. All implemented merging techniques depend on a single threshold value. The threshold value that maximizes the F-measure for the entire test set is called the *optimal dataset scale* (ODS) whereas the threshold value that maximizes the F-measure for each image of the test set is called the *optimal image scale* (OIS). Thus, we expect segmentations produced at OIS to be closest to human ground-truth segmentations.

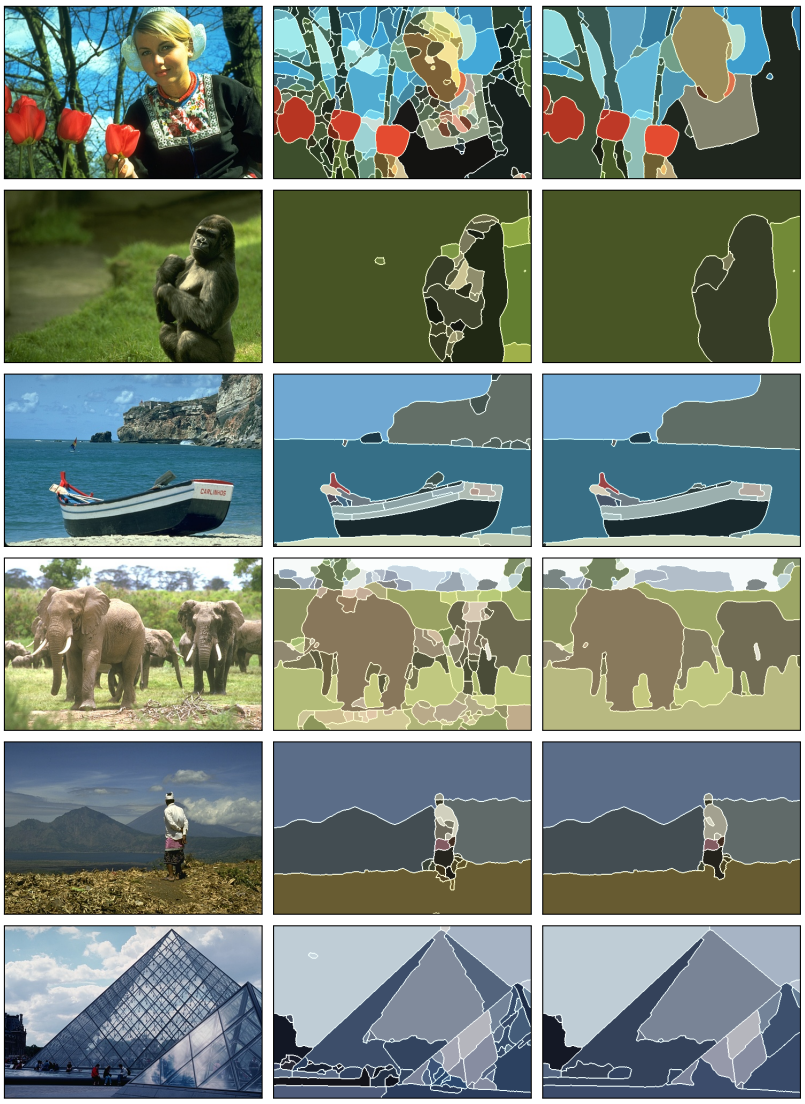


Figure 8.3: **Segmentation results on the BSDS500 produced by the gPb-mad-esm algorithm using a constant value threshold. From left to right**: input image, and segmentations obtained by thresholding at the optimal dataset scale (ODS) and optimal image scale (OIS). All images are from the test set.

Figure 8.4: **Additional segmentation results on the BSDS500 produced by the gPb-mad-esm algorithm using a constant value threshold. From top to bottom**: input image, and segmentations corresponding to the optimal dataset scale (ODS) and optimal image scale (OIS). All images are from the test set.

Figure 8.5: **Segmentation results on the BSDS500 produced by the gPb-mad-esm algorithm using a threshold based on region size. From left to right**: input image, segmentations corresponding to the optimal dataset scale (ODS) and optimal image scale (OIS). All images are from the test set.

Figure 8.6: **Segmentation results on the BSDS500 produced by the gPb-mad-sfm algorithm using a threshold based on region size. From top to bottom**: input image, and segmentations segmentations corresponding to the optimal dataset scale (ODS) and optimal image scale (OIS). All images are from the test set.

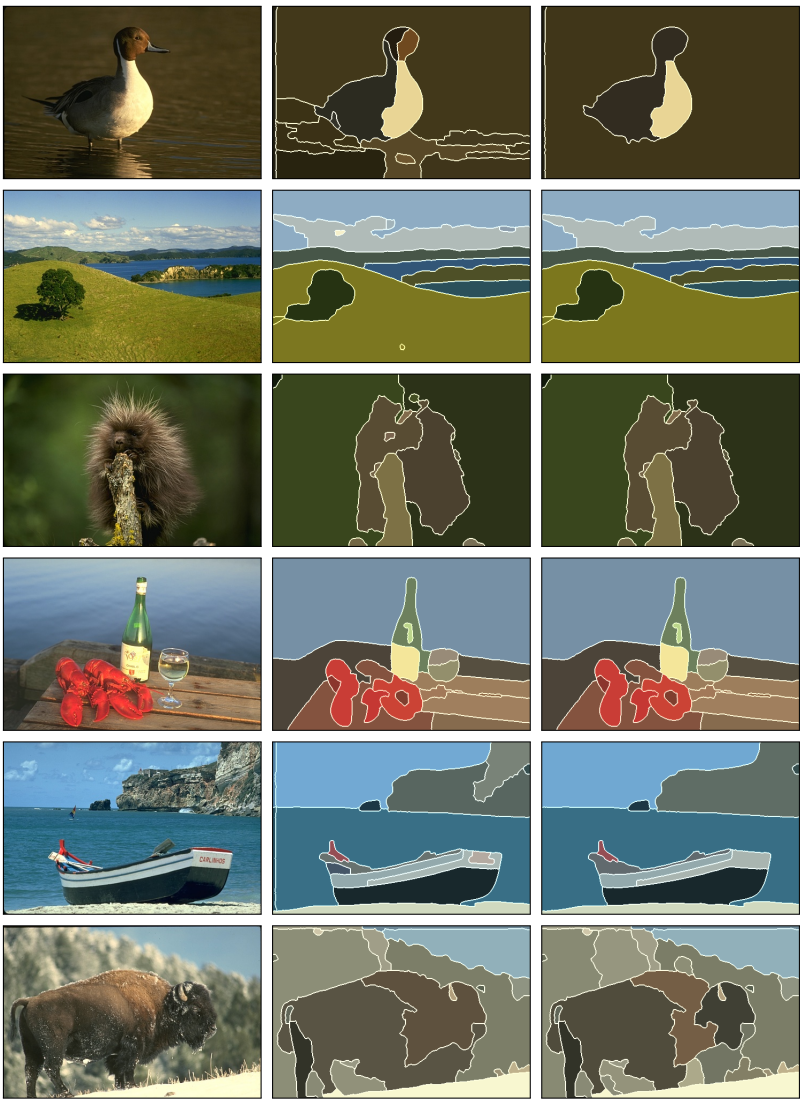


Figure 8.7: **Hierarchical segmentation results on the BSDS500 by using the inter-region fragmentation ultrametric dissimilarity. From left to right**: input image, segmentations obtained by thresholding at the optimal dataset scale (ODS) and optimal image scale (OIS).

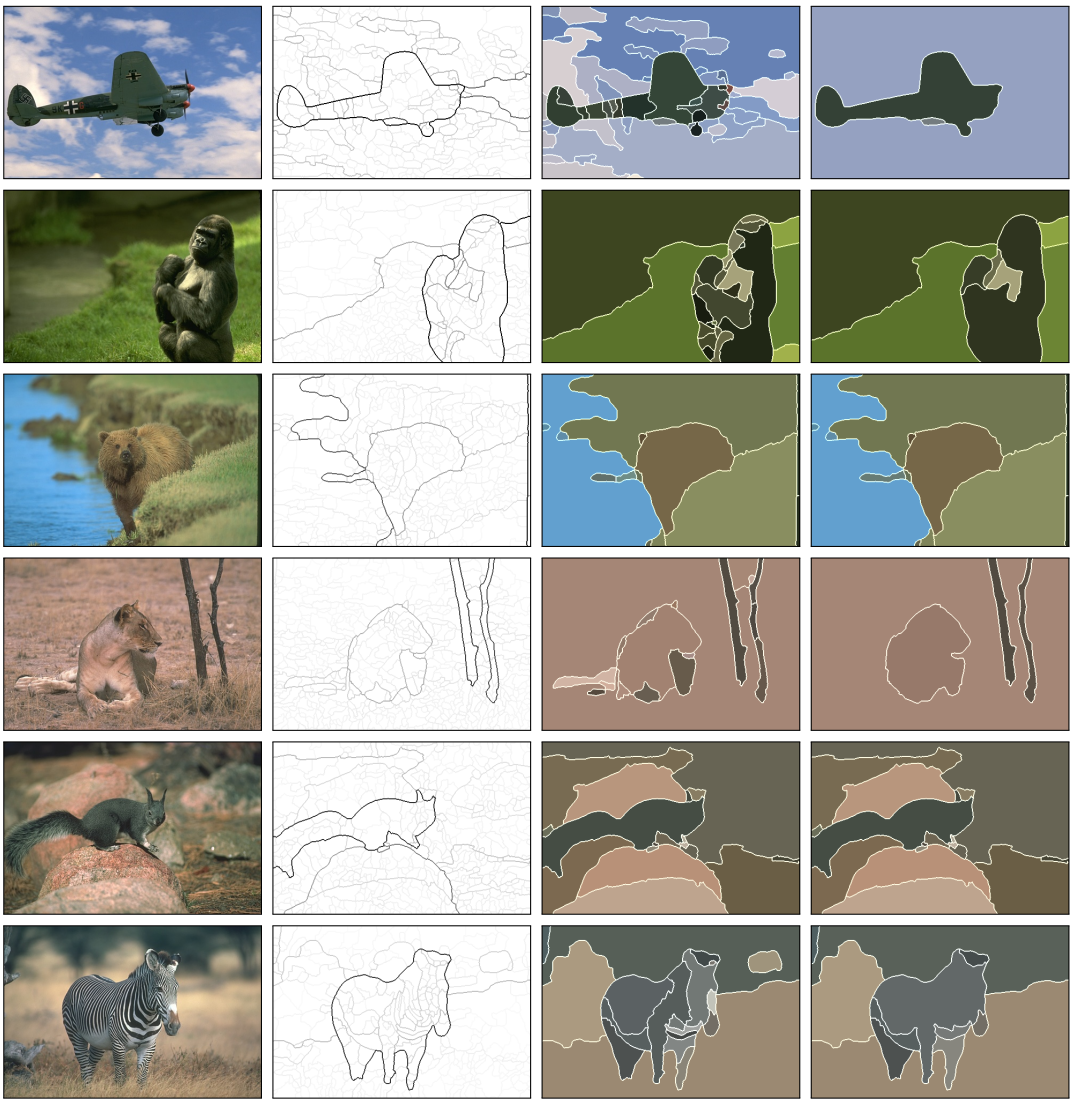


Figure 8.8: **Hierarchical segmentation results on the BSDS500 by using the boundary strength ultrametric dissimilarity . From left to right**: input image, ultrametric contour map and segmentations obtained by thresholding at the optimal dataset scale (ODS) and optimal image scale (OIS).

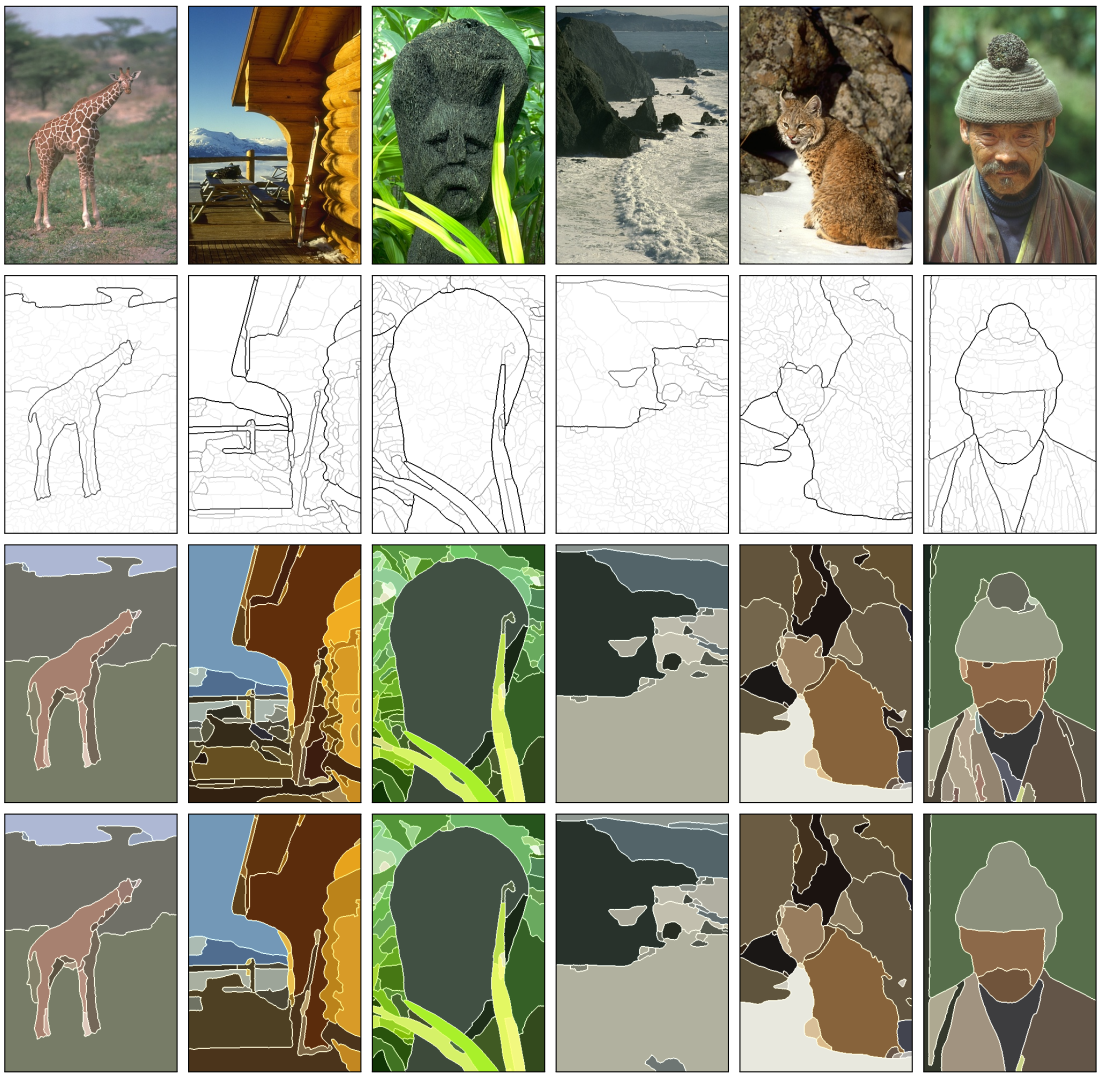


Figure 8.9: **Hierarchical segmentation results on the BSDS500 produced by using the boundary strength ultrametric dissimilarity. From left to right**: input image, ultrametric contour map and segmentations obtained by thresholding at the optimal dataset scale (ODS) and optimal image scale (OIS).

8.5 Evaluation

To provide a basis of comparison for the the merging techniques of our segmentation framework, we make use of the state-of-art gPb-owt-ucm [3] algorithm along with its baseline, the Canny-owt-ucm algorithm. In addition, we provide results as presented in [3] for the region merging by Felzenszwalb and Huttenlocher [18] (Felz-Hutt), Mean Shift [15], Multiscale Normalized Cuts [16] and for a fixed hierarchy of regions such as the Quad-Tree with 8 levels.

For our implemented techniques, let ucm_g denote our merging technique based on boundary strength ultrametric dissimilarity, ucm_ϕ denote our merging technique based on inter-region fragmentation ultrametric dissimilarity, esm_c denote the ESM algorithm with a constant threshold value and esm_a denote the ESM algorithm with using a threshold based on region size. In addition, we denote our grayscale canny contour detector as gCanny.

We evaluate each method using the boundary based precision-recall framework as well as the Variation of Information, Probabilistic Rand Index, and Covering criteria described in the previous sections. The BSDS serves as ground-truth for both the boundary and region quality measures, since the human-drawn boundaries are closed and hence are also segmentations.

8.5.1 Boundary benchmarks

The overall results for boundary evaluation criteria are presented in Table 8.5.1. Results for several different segmentation methods (upper table) and contour detectors (lower table) are given. Shown are the F-measures when choosing an optimal scale for the entire dataset (ODS) or per image (OIS). The boundary benchmark is considered to have the largest discriminative power among the evaluation criteria, clearly separating the Quad-Tree from all the data-driven methods.

8.5.2 Region benchmarks

The overall results for region evaluation criteria are presented in Table 8.5.1. For each segmentation method, the leftmost three columns report the score of the covering of ground-truth segments according to optimal dataset scale (ODS), optimal image scale (OIS), or Best covering criteria. The rightmost four columns compare the segmentation methods against ground-truth using the Probabilistic Rand Index (PRI) and Variation of Information (VI) benchmarks, respectively. Among the region benchmarks, the covering criterion has the largest dynamic range, followed by PRI and VI.

8.6 Discussion

From all merging techniques we implemented, the gPb-mad-ucm_g has the best performance in both boundary and region benchmarks. It is almost as good as the state-of-art gPb-owt-ucm algorithm in the boundary benchmark and in Covering metric, exactly as good in Probabilistic Rand Index metric and significantly *outperforms* the gPb-owt-ucm algorithm in the Variation of Information metric. The gPb-mad-ucm_ϕ technique that merges according to the inter-region ultrametric dissimilarity performs quite well in the boundary benchmark but it has the worst

	BSDS500	
	ODS	OIS
Human	0.80	0.80
gPb-owt-ucm [3]	**0.73**	**0.76**
gPb-mad-ucm$_g$	0.72	0.75
gPb-mad-esm$_c$	0.69	0.72
gPb-mad-ucm$_\phi$	0.69	0.71
gPb-mad-sfm	0.67	0.70
gPb-mad-esm$_a$	0.64	0.68
Mean Shift [15]	0.64	0.68
NCuts [16]	0.64	0.68
gCanny-mad-ucm$_g$	0.61	0.65
Felz-Hutt [18]	0.61	0.64
Canny-owt-ucm [3]	0.60	0.64
gCanny-mad-esm$_c$	0.58	0.61
gCanny-mad-esm$_a$	0.58	0.61
gCanny-mad-sfm	0.56	0.59
gCanny-mad-ucm$_\phi$	0.55	0.58
Quad-Tree	0.38	0.39
gPb [3]	0.71	0.74
gCanny	0.61	0.64
Canny [3]	0.60	0.63

Table 8.1: Boundary benchmarks on the BSDS500.

performance in the region benchmark. This is quite expected as the inter-region fragmentation dissimilarity is actually a *binary approximation* of the boundary strength between two regions. Consider the following case: two regions are separated by a relatively small boundary segment which has a medium contour strength. If the are stronger gradients in the neigborhood of this segment, then probably there will not be formed any sources on the this boundary segment. As a result, this segment will be considered as one of very weak dissimilarity and the two corresponding regions will be merged despite the fact that there is underlying boundary strength.

The efficient similarity merging technique using a constant threshold seems to be the second best technique out the five we proposed and tested. It has the advantage that it is efficient, easy to implement and produces good results. Its main drawback is that it does not take into account the region size. Thus, it will not merge two small adjacent regions even if there is relatively strong evidence for similarit. In addition, using a threshold function based on region area, instead of a constant threshold, does not actually produce the desired results. It creates a preference for regions size rather than a proper normalization of regions similarity. Moreover, the merging technique controlled by shape fragmentation factor has the advantage that it takes into account more global information than the other methods. One the one hand, it can merge small regions if their boundaries are fragmented enough. On the

	BSDS500						
	Covering			PRI		VI	
	ODS	OIS	Best	ODS	OIS	ODS	OIS
Human	0.72	0.72	–	0.88	0.88	1.17	1.17
gPb-owt-ucm[3]	**0.59**	**0.65**	**0.74**	**0.83**	**0.86**	1.69	1.48
gPb-mad-ucm$_g$	0.58	0.64	**0.74**	**0.83**	**0.86**	**1.62**	**1.39**
gPb-mad-esm$_c$	0.55	0.62	0.71	0.82	**0.86**	1.83	1.51
Mean Shift [15]	0.54	0.58	0.66	0.79	0.81	1.85	1.64
Felz-Hutt [18]	0.52	0.57	0.69	0.80	0.82	2.21	1.87
gPb-mad-sfm	0.52	0.56	0.62	0.79	0.82	1.83	1.70
gPb-mad-esm$_a$	0.51	0.54	0.60	0.79	0.80	1.86	1.82
Canny-owt-ucm [3]	0.49	0.55	0.66	0.79	0.83	2.19	1.89
myCanny-mad-ucm$_g$	0.48	0.55	0.65	0.79	0.83	2.10	1.77
gPb-mad-ucm$_\phi$	0.46	0.54	0.63	0.77	0.80	2.07	1.81
NCuts [16]	0.45	0.53	0.67	0.78	0.80	2.23	1.89
gCanny-mad-esm$_c$	0.45	0.53	0.63	0.78	0.83	2.31	1.91
gCanny-mad-sfm	0.42	0.49	0.58	0.77	0.80	2.18	1.95
gCanny-mad-esm$_a$	0.40	0.47	0.53	0.76	0.77	2.41	2.27
gCanny-mad-ucm$_\phi$	0.35	0.42	0.50	0.74	0.77	2.43	2.29
Quad-Tree	0.32	0.37	0.46	0.73	0.74	2.46	2.32

Table 8.2: Region benchmarks on the BSDS500.

other hand, it can break up uniform regions.

Finally, we see that the use of a contour detector that does not take into account texture information such as the implemented Canny detector, yields in quite poor result not only for the techniques of our framework but for algorithms such the owt-ucm. So, the choice of a contour detector is of high importance for the problem at hand. Natural images are very complicated and usually textured and thus, a sophisticated contour detector as the gPb is necessary. If the problem was to segment another type of images, e.g. textureless biomedical images, then probably our baseline contour detector would be sufficient to produce good results and preferable than the computationally costly gPb detector.

Chapter 9

Conclusions

9.1 Conclusions

In the framework of this thesis, we presented new image segmentation techniques based on a recently developed medial axis decomposition procedure. We investigated various merging techniques and explored several possible applications of a medial decomposition procedure for the purposes of segmenting natural images. We evaluated the implemented merging techniques using the Berkeley Segmentation Dataset and compared with some state of the art algorithms. The performance of developed techniques, with a proper contour detection as input, proved to be comparable with the state of the art. Although medial axis is considered, in general, unstable, we demonstrated that medial axis can be applied for image segmentation purposes with success, producing meaningful segmentations which are close to human visual perception.

9.2 Future work

Medial axis decomposition technique for the purpose of image segmentation has not been fully investigated in the narrow limits of a diploma thesis. We plan to further investigate several different directions in the future. Some of them are:

- A modified distance transform including contour orientation information.
- Integration of local characteristics and features provided by the medial axis decomposition method into a more globalized procedure.
- Usage of the segmentation along with a medial axis shape description for the purpose of object recognition.
- A faster algorithm to compute the weighted distance transform as it consumes significant part of the method total running time.

- A more robust ultrametric dissimilarity based on boundary strength. Again, orientation can be exploited to take more accurate measurements of the boundary strength between adjacent regions.
- A combination of some merging techniques we have implemented.

Appendix A

Proofs

Proof of Lemma 4.3.1 (From [6])
(a) $\rightarrow$ (b). Using definitions (4.17), (4.16) and (4.15), it follows that if y is a source,

$$d(x,y) + f(y) \leq d(x,z) + f(z) \quad \forall z \in X \tag{A.1}$$

for some $x \in X$, or

$$f(y) \leq d(x,z) - d(x,y) + f(z) \quad \forall z \in X. \tag{A.2}$$

Now, using the triangle inequality and the fact that $d(y,y) = 0$, we derive that

$$d(y,y) + f(y) \leq d(y,z) + f(z) \quad \forall z \in X, \tag{A.3}$$

which, similarly to (A.1), implies that $y \succcurlyeq y$.

(b) $\rightarrow$ (c). If $y \succcurlyeq y$, then by definition (4.16), $\mathcal{D}(f)(y) = f(y)$.

(c) $\rightarrow$ (d). If $\mathcal{D}(f)(y) = f(y)$, then by definition (4.16) $y \in \hat{S}(y)$, or $y \succcurlyeq y$. Suppose there is some other point $z \in S(y)$, then $z \in \hat{S}(y)$ or $z \succcurlyeq y \succcurlyeq y$. By definition (4.17) $z \notin S(y)$, a contradiction. Therefore $\hat{S}(y) = \{y\}$ implying that $S(y) = \{y\}$.

(d) $\rightarrow$ (e) and (d) $\rightarrow$ (a) are straightforward. □

Proof of Lemma 4.3.2 (From [6])
Let

$$g(x) = \begin{cases} f(x), & x \in S(f) \\ +\infty, & \text{otherwise.} \end{cases} \tag{A.4}$$

By definition (4.15), for all $x \in X$,

$$\mathcal{D}_d(g)(x) = \bigwedge_{y \in X} d(x,y) + g(y) \tag{A.5}$$

$$= \bigwedge_{y \in S(f)} d(x,y) + f(y). \tag{A.6}$$

On the other hand, assuming source existence, it follows from definition (4.17) that

$$\mathcal{D}_d(f)(x) = d(x, s(x)) + f(s(x)), \quad x \in X. \tag{A.7}$$

But since $s(x) \in S(f)$, definition (4.15) for f gives

$$\mathcal{D}_d(f)(x) = \bigwedge_{y \in S(f)} d(x, y) + f(y), \quad x \in X. \tag{A.8}$$

The above imply that $\mathcal{D}_d(f) = \mathcal{D}_d(g)$, where (by construction) g is uniquely determined by $f|_{S(f)}$, as claimed.

Proof of Lemma 5.2.1 (From [6])
Let $y \in S(f)$. By lemma 4.3.1, $S(y) = \{y\}$ hence $|S(y)| = 1$. Then y cannot be a medial point: $y \notin A(f)$. □

Proof of Lemma 5.2.4 (From [6])
(a) Clearly,

$$\ell(u', v') = \ell(u', u) + \ell(u, v) + \ell(v, v'), \tag{A.9}$$

with all four lengths being non-zero. By (strict) triangle inequality,

$$\delta(u', v') < \ell(u', u) + \delta(u, v) + \ell(v, v'). \tag{A.10}$$

Hence, substituting $\ell(u', u) + \ell(v, v')$ by $\ell(u', v') - \ell(u, v)$ from (A.9),

$$\delta(u', v') < \delta(u, v) + \ell(u', v') - \ell(u, v), \tag{A.11}$$

and, by definition (**??**),

$$\text{res}(x, y) < \text{res}(x', y') \tag{A.12}$$

as claimed.

Proof of Lemma 7.1.1 (An analogous proof can be found at [18])
Let $e_q = (v_i, v_j)$, $w(e_q)$ the corresponding edge weight. Let C_i^{q-1} and C_j^{q-1} the components of vertices v_i and v_j at step q. If C_i^{q-1} and C_j^{q-1} are not merged, then there are two cases. Either $w(e_q) < \tau(C_i^{q-1})$ or $w(e_q) < \tau(C_j^{q-1})$. Consider the first case. Since edges are considered in non-increasing weight order then $w(e_k) \leq w(e_q) < \tau(C_i^{q-1})$, for all $k \geq q+1$. Thus, no additional merging will happen to this component and it will then appear to the final segmentation, i.e. $C_i = C_i^{q-1}$. The second case is exactly analogous.

Proof of Lemma 7.1.2 (An analogous proof can be found at [18])
By definition, in order for S to be too fine there is some pair of components for

which Q does not hold. Thus, we have:

$$Q(C_i^{q-1}, C_j^{q-1}) = false \Rightarrow$$
$$Sim(C_i^{q-1}, C_j^{q-1}) > \tau(C_i^{q-1}, C_j^{q-1}) \Rightarrow$$
$$\max_{v_i \in C_i^{q-1}, v_j \in C_j^{q-1}, (v_i, v_j) \in E} w((v_i, v_j)) > \tau(C_i^{q-1}, C_j^{q-1}) \Rightarrow$$
$$\exists \quad e = (v_i, v_j) : w(e) > \tau(C_i^{q-1}, C_j^{q-1})$$

So there exists an edge $e = (v_i, v_j)$ between components C_i^{q-1}, C_j^{q-1} that will not cause their merging. By lemma 7.1.1, one of them at least will be in the final segmentation. Without loss of generality, suppose that $C_i = C_i^{q-1}$. Then:

$$w(e) > \tau(C_i^{q-1}, C_j^{q-1}) \Rightarrow$$
$$w(e) > \tau(C_i, C_j^{q-1}) \geq \tau(C_i, C_j) \Rightarrow$$
$$Q(C_i, C_j) = true$$

So the predicate $Q(C_i, C_j)$ is true, which is a contradiction. Thus, there is no pair of components for which Q does not hold and subsequently the segmentation produced in not too fine.

Proof of Lemma 7.1.2 (An analogous proof can be found at [18])
In order for S to be too coarse there must be some proper refinement, T, that is not too fine. Consider the maximum weight edge e that is internal to a component $C \in S$ but connects distinct components $A, B \in T$. Note that by the definition of refinement $A \subset C$ and $B \subset C$. Since T is not too fine, either $w(e) < \tau(A)$ or $w(e) < \tau(B)$. Without loss of generality, say the former is true. By construction any edge connecting A to another sub-component of C has weight smaller or equal to $w(e)$. So the algorithm must have formed A before forming C, and in forming C it must have merged A with some other sub-component of C. The weight of the edge that caused this merge must be small or equal to $w(e)$. However, the algorithm would not have merged A in this case because $w(e) < \tau(A)$, which is a contradiction.

Proof of Lemma 7.3.1 Proving that dissimilarity δ_Φ defined in equation 7.29 is ultrametric, is equivalent to proving that δ_Φ satisfies property (7.27). Let (R_i, R_j) denote the pair of regions that minimize δ_Φ and R_k denote any region connected to $R_i \cup R_j$ and belonging to the partition obtained after the merging of (R_i, R_j)). In addition, let $\mathcal{A}_{ij} = \sum_{e \in E(R_i) \cap E(R_j)} w(e)$.

$$\delta_\Phi(R_i, R_j) \leq \delta_\Phi(R_i, R_k) \Rightarrow$$
$$\exp(-\Phi(R_i, R_j)) \leq \exp(-\Phi(R_i, R_k)) \Rightarrow$$
$$\Phi(R_i, R_j) \geq \Phi(R_i, R_k) \Rightarrow$$
$$\frac{\mathcal{A}_{ij}}{L(\partial_{ij})} \geq \frac{\mathcal{A}_{ik}}{L(\partial_{ik})} \Rightarrow$$
$$L(\partial_{ik}) \cdot \mathcal{A}_{ij} \geq L(\partial_{ij}) \cdot \mathcal{A}_{ik} \tag{A.13}$$

Similarly,

$$L(\partial_{jk}) \cdot \mathcal{A}_{ij} \geq L(\partial_{ij}) \cdot \mathcal{A}_{jk} \tag{A.14}$$

Combining equations (A.13) and (A.14) we have:

$$\begin{gathered}
L(\partial_{jk}) \cdot \mathcal{A}_{ij} + L(\partial_{ik}) \cdot \mathcal{A}_{ij} \geq L(\partial_{ij}) \cdot \mathcal{A}_{jk} + L(\partial_{ij}) \cdot \mathcal{A}_{ik} \Rightarrow \\
(L(\partial_{jk}) + L(\partial_{ik})) \cdot \mathcal{A}_{ij} \geq L(\partial_{ij}) \cdot (\mathcal{A}_{jk} + \mathcal{A}_{ik}) \Rightarrow \\
\frac{\mathcal{A}_{ij}}{L(\partial_{ij})} \geq \frac{\mathcal{A}_{jk} + \mathcal{A}_{ik}}{L(\partial_{jk}) + L(\partial_{ik})} \Rightarrow \\
\frac{\sum_{e \in E(R_i) \cap E(R_j)} w(e)}{L(\partial_{ij})} \geq \frac{\sum_{e \in E(R_j) \cap E(R_k)} w(e) + \sum_{e \in E(R_i) \cap E(R_k)} w(e)}{L(\partial_{jk}) + L(\partial_{ik})} \Rightarrow \\
\frac{\sum_{e \in E(R_i) \cap E(R_j)} w(e)}{L(\partial_{ij})} \geq \frac{\sum_{e \in E(R_i \cup R_j) \cap E(R_k)} w(e)}{L(\partial_{i \cup j,k})} \Rightarrow \\
\Phi(R_i, R_j) \geq \Phi(R_i \cup R_j, R_k) \Rightarrow \\
\exp(-\Phi(R_i, R_j)) \leq \exp(-\Phi(R_i \cup R_j, R_k)) \Rightarrow \\
\delta_\Phi(R_i, R_j) \leq \delta_\Phi(R_i \cup R_j, R_k)
\end{gathered} \tag{A.15}$$

Proof of Lemma 7.3.2 Proving that dissimilarity δ_g is ultrametric is analogous to proof of lemma 7.3.1. Let (R_i, R_j) denote the pair of regions that minimize δ_Φ and R_k denote any region connected to $R_i \cup R_j$ and belonging to the partition obtained after the merging of (R_i, R_j)).

$$\begin{gathered}
\delta_g(R_i, R_j) \leq \delta_g(R_i, R_k) \Rightarrow \\
\frac{\sum_g(\partial_{ij})}{L(\partial_{ij})} \leq \frac{\sum_g(\partial_{ik})}{L(\partial_{ik})} \Rightarrow \\
L(\partial_{ik}) \cdot \sum_g(\partial_{ij}) = L(\partial_{ij}) \cdot \sum_g(\partial_{ik})
\end{gathered} \tag{A.16}$$

(A.17)

Similarly,

$$L(\partial_{jk}) \cdot \sum_g(\partial_{ij}) \leq L(\partial_{ij}) \cdot \sum_g(\partial_{jk}) \tag{A.18}$$

Combining equations (A.16) and (A.18) we have:

$$\begin{gathered}
(L(\partial_{ik}) + L(\partial_{jk})) \cdot \sum_g(\partial_{ij}) \leq L(\partial_{ij}) \cdot (\sum_g(\partial_{ik}) + \sum_g(\partial_{jk})) \Rightarrow \\
\frac{\sum_g(\partial_{ij})}{L(\partial_{ij})} \leq \frac{(\sum_g(\partial_{ik}) + \sum_g(\partial_{jk}))}{L(\partial_{ik}) + L(\partial_{jk}))} \Rightarrow \\
\delta_g(R_i, R_j) \leq \delta_g(R_i \cup R_j, R_k)
\end{gathered} \tag{A.19}$$

Bibliography

[1] P. Arbelaez. Notes on the evaluation methodology. http://www.cs.berkeley.edu/~arbelaez/Notes.html.

[2] P. Arbelaez. Boundary extraction in natural images using ultrametric contour maps. In *Proceedings of POCV*, 2006.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. From contours to regions: An empirical evaluation. In *Proceedings of Computer Vision and Pattern Recognition*, pages 2294–2301, 2009.

[4] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 898–916, 2011.

[5] A.B. Arehart, L. Vincent, and B.B. Kimia. Mathematical morphology: The Hamilton-Jacobi connection. In *ICCV*, pages 215–219, 1993.

[6] Y. Avrithis and K. Rapantzikos. The medial feature detector: Stable regions from image boundaries (long version). Unpublished, 2010.

[7] Y. Avrithis and K. Rapantzikos. The medial feature detector: Stable regions from image boundaries. In *Proceedings of International Conference on Computer Vision*, 2011.

[8] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using em and its application to content-based image retrieval. In *Proceedings of IEEE International Conference on Computer Vision*, pages 675–692, 1998.

[9] S. Beucher and F. Meyer. *The Morphological Approach to Segmentation : The Watershed Transformation. In Mathematical Morphology in Image Processing*, chapter 12, pages 433–481. Ed. E. R. Dougherty, 1993.

[10] H. Blum. A transformation for extracting new descriptors of shape. In *Models for the Perception of Speech and Visual Form*. MIT Press, Cambridge, 1967.

[11] H. Blum. Biological shape and visual science. *Journal of Theoretical Biology*, 38:205–287, 1973.

[12] Max Born and Emil Wolf. *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light (7th Edition)*. Cambridge University Press, 7th edition, 1999.

[13] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 679–698, 1986.

[14] Fan R K Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[15] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[16] T. Cour, F. Bénézit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Proceedings of Computer Vision and Pattern Recognition*, pages 1124–1131, 2005.

[17] H. Demirel and G. Anbarjafari. Iris recognition system using combined colour statistics. In *IEEE International Symposium on Signal Processing and Information Technology,*, pages 175 –179, 2008.

[18] P.F. Felzenszwalb and D.P Huttenlocher. Efficient graph-based image segmentation. *Internationl Journal of Computer Vision*, 59(2):167–181, 2004.

[19] P.F. Felzenszwalb and D.P Huttenlocher. Distance transforms of sampled functions. *Faculty of Computing and Information Science, Cornell Univ*, 2004.

[20] V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation by image exploration. In *Proceedings of the European Conference on Computer Vision*, 2004.

[21] C.F.F.C. Filho and M.G.F. Costa. Iris segmentation exploring color spaces. In *Image and Signal Processing 2010 3rd International Congress on*, volume 4, pages 1878 –1882, 2010.

[22] M. Fussenegger, A. Opelt, A. Pinz, and P. Auer. Object recognition using segmentation for feature detection. In *In ICPR*, pages 41–44, 2004.

[23] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., 2006.

[24] S. Kim. An $O(N)$ Level Set Method for Eikonal Equations. *SIAM journal on scientific computing*, 22(6):2178–2193, 2001.

[25] I. Kovacs and B. Julesz. A closed curve is much more than an incomplete one: Effect of closure in figure-ground segmentation. *Proceedings of the National Academy of Sciences, USA*, 90:7495–7497, 1993.

[26] V. Lamme. The neurophysiology of figure-ground segregation in primary visual cortex. *Journal of Neuroscience*, 15:1605–1615, 1995.

[27] T. S. Lee, D. Mumford, R. Romero, and V. Lamme. The role of the primary visual cortex in higher level vision. *Vision Research*, 38:2429–2454, 1998.

[28] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77:259–289, 2008.

[29] Hwei-Jen Lin, Shu-Yi Wang, Shwu-Huey Yen, and Yang-Ta Kao. Face detection based on skin color segmentation and neural network. In *Proceedings of Neural Networks and Brain, 2005.*, volume 2, pages 1144 –1149, 2005.

[30] L. Linardakis and N. Chrisochoides. A static medial axis domain decomposition for 2d geometries. *ACM Transactions on Mathematical Software*, 34(1):1–19, 2005.

[31] P. Maragos. *Image Analysis And Computer Vision.* National Technical University of Athens, 2005. Book draft version.

[32] P. Maragos and M.A. Butt. Curve evolution, differential morphology, and distance. *Fundamenta Informaticae*, 41:91–129, 2000.

[33] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* Henry Holt and Co., Inc. New York, NY, USA, 1982.

[34] D. R. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, pages 416–425, 2001.

[35] David R. Martin, Charless Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 530–549, 2004.

[36] B. M. Mehtre and B. Chatterjee. Segmentation of fingerprint images - a composite method. *Pattern Recognition*, 22(4):381–385, 1989.

[37] H. Mobahi, S. Rao, A. Yang, S. Sastry, and Y. Ma. Segmentation of natural images by texture and boundary compression. *International Journal of Computer Vision*, pages 1–13, 2011.

[38] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42:577–685, 1989.

[39] R.L. Ogniewicz and O. Kübler. Hierarchic voronoi skeletons. *Pattern Recognition*, 28(3):343–359, 1995.

[40] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9:62–66, 1979.

[41] D. L. Pham, C. Xu, and J. L. Prince. A survey of current methods in medical image segmentation. *Annual Review of Biomedical Engineering*, 2:315–338, 2000.

[42] A. Rosenfeld and J. L. Pfaltz. Distance functions on digital pictures. *Pattern Recognition*, 1:33–61, 1968.

[43] J.A. Sethian. Fast marching methods. *SIAM journal on scientific computing*, 41(2):199–235, 1999.

[44] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997.

[45] K. Siddiqi, S. Bouix, A. Tannenbaum, and S.W. Zucker. Hamilton-jacobi skeletons. *Internationl Journal of Computer Vision*, 48(3):215–231, 2002.

[46] George Stockman and Linda G. Shapiro. *Computer Vision*. Prentice Hall, 1st edition, 2001.

[47] P. Sundberg, T. Brox, M. Maire, P. Arbelaez, and J. Malik. Occlusion boundary detection and figure/ground assignment from optical flow. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.

[48] H. Tek and B. B. Kimia. Curve evolution, wave propagation, and mathematical morphology. In *Proc. 4th Int. Symposium on Mathematical Morphology*, June 1998.

[49] P.W. Verbeek and B.J.H. Verwer. Shading from shape, the eikonal equation solved by grey-weighted distance transform. *PRL*, 11(10):681–690, 1990.

[50] L. Vincent. Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms. *IEEE Transactions on Image Processing*, 2: 176–201, 1993.

[51] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 2002.

[52] Chee S. W. A block-based map segmentation for image compressions. *IEEE Transanctions on Circuits Syst. Video Techn.*, 8:592–601, 1998.

[53] L. Wang, J. Shi, G. Song, and I. Shen. Object detection combining recognition and segmentation. In *Proceedings of the Asian conference on Computer vision*, pages 189–199, 2007.

[54] M. Wertheiner. Laws of organization in perceptual forms (partial translation). In W. B. Ellis, editor, *A Sourcebook of Gestalt Psychology*, pages 71–88. Harcourt, Brace and Company, 1938.

[55] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15:1101–1113, 1993.

[56] L. Yatziv, A. Bartesaghi, and G. Sapiro. $O(N)$ Implementation of the fast marching algorithm. *Journal of Computational Physics*, 212(2):393–399, 2006.

[57] Y. Yi, D. Qu, and F. Xu. Face detection method based on skin color segmentation and eyes verification. In *Proceedings of Artificial Intelligence and Computational Intelligence*, volume 3, pages 495 –501, 2009.