FINDING COLOR AND SHAPE PATTERNS IN IMAGES

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Scott Cohen
May 1999

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Leonidas Guibas
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Carlo Tomasi

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Jeffrey Ullman

Approved for the University Committee on Graduate Studies:

Abstract

This thesis is devoted to the Earth Mover's Distance and its use within content-based image retrieval (CBIR). The major CBIR problem discussed is the *pattern problem*: Given an image and a query pattern, determine if the image contains a region which is visually similar to the pattern; if so, find at least one such image region. The four main themes of this work are: (i) partial matching, (ii) matching under transformation sets, (iii) combining (i) and (ii), and (iv) effective pruning of unnecessary, expensive distance/matching computations.

The first pattern problem we consider is the *polyline shape search problem* (PSSP): Given *text* and *pattern* planar polylines, find all approximate occurrences of the pattern within the text, where such occurrences may be scaled and rotated versions of the pattern. For a text and a pattern with n and m edges, respectively, we present an $O(m^2n^2)$ time, $O(mn)$ space PSSP algorithm. A major strength of our algorithm is its generality, as it can be applied for any shape pattern represented as a polyline.

The main distance measure studied in this thesis is the *Earth Mover's Distance* (EMD), which is an edit distance between distributions that allows for partial matching, and which has many applications in CBIR. A discrete distribution is just a set of (point,weight) pairs. In the CBIR context, the weight associated with a particular point in a feature space is the amount of that feature present in the image. The EMD between two distributions is proportional to the minimum amount of work needed to change one distribution into the other, where one unit of work is the amount necessary to move one unit of weight by one unit of *ground distance*. We give a couple of modifications which make the EMD more amenable to partial matching: (i) the *partial* EMD in which only a given fraction of the weight in one distribution is forced to match weight in the other, and (ii) the τ-EMD which measures the amount of weight that cannot be matched when weight moves are limited to at most τ ground distance units.

An important issue addressed in this thesis is the use of efficient, effective lower bounds on the EMD to speed up retrieval times. If a system can quickly prove that the EMD is larger than some threshold, then it may be able to avoid an EMD computation and decrease

its query time. We contribute lower bounds that are applicable in the partial matching case in which distributions do not have the same total weight. The efficiency and effectiveness of our lower bounds are demonstrated in a CBIR system which measures global color similarity between images.

Another important problem in CBIR is the EMD *under transformation* ($\text{EMD}_{\mathcal{G}}$) *problem*: find a transformation of one distribution which minimizes its EMD to another, where the set of allowable transformations $\mathcal{G}$ is given. The problem of estimating the size/scale at which a pattern occurs in an image is phrased and efficiently solved as an $\text{EMD}_{\mathcal{G}}$ problem in which transformations scale the weights of a distribution by a constant factor.

For $\text{EMD}_{\mathcal{G}}$ problems with transformations that modify the points of a distribution but not its weights, we present a monotonically convergent iteration called the *FT iteration*. This iteration may, however, converge to only a locally optimal EMD value and transformation. The FT iteration is very general, as it can be applied for many different (ground distance, transformation set) pairs, and it can be modified to work with the partial EMD, as well as in some cases in which transformations change both distribution points and weights. We apply the FT iteration to the problems of (i) illumination-invariant object recognition, and (ii) point feature matching in stereo image pairs. We also present algorithms that are guaranteed to find a globally optimal transformation when matching equal-weight distributions under translation (i) on the real line with the absolute value as the ground distance, and (ii) in any finite-dimensional space with the Euclidean distance squared as the ground distance.

Our pattern problem solution is the SEDL (Scale Estimation for Directed Location) content-based image retrieval system. Three important contributions of this system are (1) a general framework for finding both color and shape patterns, (2) the previously mentioned novel scale estimation algorithm using the EMD, and (3) a directed (as opposed to exhaustive) search strategy. We show that SEDL achieves excellent results for the color pattern problem on a database of product advertisements, and the shape pattern problem on a database of Chinese characters. A few promising pattern locations are efficiently computed at query time without having to examine image areas that obviously do not contain the pattern. SEDL uses the τ-EMD to help eliminate false positives resulting from difficulties in trading off, for example, color and position distances to measure visual similarity.

Acknowledgements

I am deeply indebted to my adviser Professor Leonidas Guibas for guiding me through the Ph.D. program. Amazingly to me, he always seemed to know just the right reference for every discussion. Whenever I was stuck in some detail, he would help me see the bigger picture and offer useful comments and suggestions (although it would usually take me a while to truly understand and appreciate his words). I am also very grateful to Professor Carlo Tomasi, whom I consider my secondary adviser, for all his time and help throughout the years. He made me feel like a part of his group, and he taught me to appeal to the basics and intuition before mechanical manipulations. I also thank Professor Jeffrey Ullman and Professor Leon Simon for serving on my orals committee. I learned how to be a better teacher from these superb educators. Some of my fondest undergraduate class memories are from Professor Simon's real analysis and differential equations classes. I thank Professor Gene Golub for playing the roles of my sponsor and adviser in an undergraduate research program many years ago, for recommending me to the Stanford computer science Ph.D. program, and for serving on my orals committee.

I have always been very fortunate to have excellent teachers who instilled a love of learning within me. I recognize the contributions of all my teachers in helping me reach my current position. There are too many to give all by name, but I especially thank my high school mentor Lawrence Zimmerman for his problem solving lessons that have stayed with me all these years.

I have also been very fortunate to have the support of many close friends during my time here at Stanford. Again, there are too many to give all by name, but I would like to mention Krista Lentine, Scott Putterman, Mark Ruzon, João Comba, David Hoffman, Alok Shah, John Stoffel, Jeff Chen, Ann Chan, Joseph Silvagnoli, and Nikhyl Singhal. I also thank Tong Zheng, a very special person in my life who was always there for me. Her unconditional love and support helped get me through the most difficult times. Finally, I would be remiss if I did not say how much I enjoyed being a part of Professor Tomasi's computer vision group, including all the vislunches, vispics, and friendships with its members.

I owe special thanks to a couple of vision group members for very direct contributions to my work. I thank Yossi Rubner for his Earth Mover's Distance code, and much more importantly, for a framework upon which to build, explore, and extend. I also thank Stan Birchfield for his feature extraction code. Other members of the academic community outside of Stanford helped as well. I thank David Slater for his database of objects imaged under varying illumination, Madirakshi Das for her database of color advertisements, Oliver Lorenz for his database of Chinese characters, and Robert Ogniewicz for his medial axis code. For my first three years in the Ph.D. program, I was supported by an NSF Fellowship (grant CCR-9215219); for my last three years, I was supported by the DARPA grant for Stanford's image retrieval project (contract DAAH04-94-G-0284).

Finally, I thank my uncle Lenny, from whom I undoubtedly get my academic inclination, for his constant encouragement, support, and good advice from the very beginning of my education.

Dedicated in loving memory to my grandmother Ann

Contents

Abstract **v**

Acknowledgements **vii**

1 Introduction **1**

1.1 The Pattern Problem 4

1.2 Thesis Overview 6

2 Background **9**

2.1 The 1D Shape Pattern Problem 9

2.2 Focused Color Searching 12

2.3 From Histogram Intersection to the Earth Mover's Distance 16

2.4 The Earth Mover's Distance 18

2.5 Matching under Transformation Groups 22

2.5.1 The Hausdorff Distance 22

2.5.2 The ICP Iteration 26

2.6 The FOCUS Image Retrieval System 29

3 The Polyline Shape Search Problem **33**

3.1 Problem Setup 35

3.2 The Best Rotation 38

3.3 The 2D Search Problem 39

3.3.1 Faces 43

3.3.2 Edges and Vertices 46

3.4 The Algorithm 47

3.5 Results 50

3.6 Summary and Suggestions for Future Work 53

4 The Earth Mover's Distance (EMD) 59
4.1 Basic Definitions and Notation 62
4.2 Connection to the Transportation Problem 67
4.3 Special Cases 69
4.3.1 Point Set Matching using the EMD 69
4.3.2 The EMD in One Dimension 71
4.4 Modifications 80
4.4.1 The Partial Earth Mover's Distance 80
4.4.2 The Restricted Earth Mover's Distance 83
4.5 Use in Scale Estimation 85
4.5.1 Experiments with the Color Pattern Problem 91

5 Lower Bounds on the EMD 97
5.1 Centroid-based Lower Bounds 98
5.1.1 Equal-Weight Distributions 98
5.1.2 Unequal-Weight Distributions 100
5.1.2.1 The Centroid Lower Bound 101
5.1.2.2 The Centroid Bounding Box Lower Bound 103
5.2 Projection-based Lower Bounds 104
5.3 Experiments in Color-based Retrieval 110

6 The EMD under Transformation Sets 123
6.1 Definitions and Notation 125
6.2 A Direct Algorithm 128
6.3 The FT Iteration 128
6.3.1 Similar Work 131
6.3.2 Convergence Properties 133
6.4 The Optimal Transformation Problem 135
6.4.1 Translation 135
6.4.1.1 Minimizing a Weighted Sum of Squared L_2 Distances . . . 137
6.4.1.2 Minimizing a Weighted Sum of L_1 Distances 137
6.4.1.3 Minimizing a Weighted Sum of L_2 Distances 141
6.4.1.4 Minimizing a Weighted Sum of Cyclic L_1 Distances 142
6.4.2 Euclidean and Similarity Transformations 147
6.4.3 Linear and Affine Transformations 147
6.5 Allowing Weight-Altering Transformations 148

6.6 Some Specific Cases 151
6.6.1 The Equal-Weight EMD under Translation with $d = L_2^2$ 152
6.6.2 The Equal-Weight EMD under Translation on the Real Line 152
6.6.3 The Equal-Weight EMD under $\mathcal{G}$ with $m = n = 2$ 154
6.7 Global Convergence in $\mathcal{F} \times \mathcal{G}$? 156
6.7.1 Partial Matching 157
6.7.2 One Optimal Flow or Transformation 160
6.7.3 A Perfect Match under Translation 161
6.7.4 Equal-Weight Comparisons with Local Minima 163
6.8 Odds and Ends 169
6.8.1 L_2^2 versus L_2 171
6.8.2 Sensitivity and Growth Rate 172
6.9 Some Applications 175
6.9.1 Lighting-Invariant Object Recognition 175
6.9.2 Feature Matching in Stereo Images 177

7 The SEDL Image Retrieval System **181**
7.1 SEDL's Image Signatures and Distance Function 185
7.2 The Scale Estimation Phase 188
7.3 The Initial Placement Phase 189
7.3.1 Experiments with the Color Pattern Problem 194
7.4 The Verification and Refinement Phase 195
7.5 Results 205
7.5.1 The Product Advertisement Color Database 205
7.5.1.1 Creating Signatures 205
7.5.1.2 Query Results 212
7.5.1.3 SEDL versus FOCUS 217
7.5.2 The Chinese Character Shape Database 217
7.5.2.1 Creating Signatures 230
7.5.2.2 Query Results 230
7.5.2.3 Possible Modifications 231

8 Conclusion **241**
8.1 Thesis Summary and Discussion 241
8.2 Future Work 244
8.3 Final Thoughts 247

Bibliography **249**

List of Figures

1.1 The Importance of Partial Matching . . . 2
1.2 The Importance of a Small Partial Match . . . 3
1.3 The Importance of Scale . . . 5

3.1 PSSP Turning Angle Summaries . . . 35
3.2 PSSP Matching in Arclength Versus Turning Angle Space . . . 36
3.3 The Interval Overlap X_{ij} . . . 41
3.4 All Possible Interval Overlaps X_{ij} . . . 42
3.5 The Integral of Text Angle as a Function of Pattern Placement . . . 45
3.6 Elementary Step Notation . . . 48
3.7 Trading Off Match Length Against Match Error . . . 50
3.8 PSSP Exact Matching Examples . . . 51
3.9 PSSP Results . . . 52
3.10 More PSSP Results . . . 53
3.11 Image Summary by Straight Segments . . . 54
3.12 Another Image Summary by Straight Segments . . . 55
3.13 PSSP No Matches Example . . . 55

4.1 Example Distributions in 2D . . . 60
4.2 The EMD Morphing Process . . . 61
4.3 A Non-Optimal and an Optimal Flow between Equal-Weight Distributions 64
4.4 A Non-Optimal and an Optimal Flow between Unequal-Weight Distributions 65
4.5 The EMD between Equal-Weight Distributions on the Real Line . . . 72
4.6 Feasibility of the CDF flow . . . 73
4.7 Breakpoint Notation Used in Lemma 2 . . . 74
4.8 Flow Feasibility for Equal-Weight Distributions on the Real Line . . . 76
4.9 The Inverse CDFs . . . 79
4.10 Partial EMD Example . . . 82

4.11 The Partial EMD as a Balanced Transportation Problem 82
4.12 τ-EMD Example 84
4.13 Scale Estimation – Main Idea 86
4.14 Scale Estimation Algorithm 89
4.15 Scale Estimation – Clorox Example 92
4.16 Scale Estimation – Pattern Not in the Image 93
4.17 Scale Estimation Results – Example Set 1 94
4.18 Scale Estimation Results – Example Set 2 95
4.19 Scale Estimation Results – Example Set 3 96

5.1 The Projection Lower Bound 105
5.2 Flow Feasibility for Unequal-Weight Distributions on the Real Line 108
5.3 Query C.1.1 – 20% Blue 113
5.4 Query C.1.2 – 40% Green 114
5.5 Query C.1.3 – 60% Red 114
5.6 Query C.2.1 – 13.5% Green, 3.4% Red, 17.8% Yellow 115
5.7 Query C.2.2 – 26.0% Blue, 19.7% Violet 116
5.8 Query C.2.3 – 16.8% Blue, 22.2% Green, 1.8% Yellow 116
5.9 Query C.2.4 – 22.8% Red, 24.2% Green, 17.3% Blue 117
5.10 Query C.2.5 – 13.2% Yellow, 15.3% Violet, 15.3% Green 117
5.11 Distribution Centroids for Corel Database Images and Example Queries . . 118
5.12 Query C.3.1 – Sunset Image 120
5.13 Query C.3.2 – Image with Trees, Grass, Water, and Sky 121

6.1 FT Iteration Example 129
6.2 The Minisum Problem on the Line with Unequal Weights 139
6.3 The Minisum Problem on the Line with Equal Weights 140
6.4 The Equal-Weight EMD under Translation in 1D with $d = L_1$ 153
6.5 A Local Minimum in a 1D Partial Matching Case 158
6.6 A Local Minimum in a 2D Partial Matching Case 159
6.7 A Local Minimum in a 2D Equal-Weight Case 164
6.8 Graphs of EMD v. t Showing a Locally Optimal Translation for $d = L_2$ . . 166
6.9 Graphs of EMD v. t Showing a Locally Optimal Translation for $d = L_1$ . . 167
6.10 A Closer Look at a Locally Optimal Translation 168
6.11 Graphs of EMD v. θ Showing a Locally Optimal Rotation for $d = L_2^2$ 170
6.12 Matching Pairs of Points 172
6.13 Optimal Point Set Matchings under L_2 and L_2^2 173

6.14 Lighting-Invariant Object Recognition – A Small Object Database 176
6.15 Lighting-Invariant Object Recognition – Query Results 177
6.16 Point Feature Matching in Stereo Images – Results 179

7.1 Query Patterns and Related Database Images 182
7.2 The Three Phases in SEDL 184
7.3 Signature in Color $\times$ Position Space 186
7.4 An Example of Color Confidences 191
7.5 Initial Placement Results – Example Set 1 196
7.6 Initial Placement Results – Example Set 2 197
7.7 Initial Placement Results – Example Set 3 198
7.8 Initial Placement Results – Example Set 4 199
7.9 Initial Placement Results – Example Set 5 200
7.10 The Verification and Refinement Phase 200
7.11 Queries for the Color Product Advertisement Database 206
7.12 Region Merging Results 210
7.13 More Region Merging Results 211
7.14 Tough Advertisements to Retrieve 213
7.15 Query Results for the Color Advertisement Database 214
7.16 Recall, Precision, and Timing Results for the Color Advertisement Database 216
7.17 Advertisement Query Result – Clorox 218
7.18 Advertisement Query Result – Breathe Right 219
7.19 Advertisement Query Result – Comet 220
7.20 Advertisement Query Result – Fresh Step 221
7.21 Advertisement Query Result – Jello 222
7.22 Advertisement Query Result – Apple 223
7.23 Advertisement Query Result – Reynolds Oven Bags 224
7.24 Advertisement Query Result – Taco Bell 225
7.25 Verification and Refinement Results – Example Set 1 226
7.26 Verification and Refinement Results – Example Set 2 227
7.27 Verification and Refinement Results – Example Set 3 228
7.28 Sample Images from the Chinese Character Database 229
7.29 Medial Axis Shape Summaries 229
7.30 Chinese Characters Query Result – Example 1 232
7.31 Chinese Characters Query Result – Example 2 232
7.32 Chinese Characters Query Result – Example 3 233

7.33 Chinese Characters Query Result – Example 4 233
7.34 Chinese Characters Query Result – Example 5 234
7.35 Chinese Characters Query Result – Example 6 234
7.36 Chinese Characters Query Result – Example 7 235
7.37 Chinese Characters Query Result – Example 8 235
7.38 Chinese Characters Query Result – Example 9 236
7.39 Chinese Characters Query Result – Example 10 236
7.40 Chinese Characters Query Result – Example 11 237
7.41 Chinese Characters Query Result – Example 12 237
7.42 Chinese Characters Query Result – Example 13 238
7.43 Chinese Characters Query Result – Example 14 238
7.44 Chinese Characters Query Result – Example 15 239
7.45 Query Times for the Chinese Character Database 240

Chapter 1

Introduction

The invention of the World Wide Web has brought the need for automated image indexing and content-based image retrieval (CBIR) to the forefront of image processing and computer vision research. Although there has been significant progress in CBIR in the past five years, the general problem is far from solved. Semantic image understanding is beyond the scope of current state-of-the-art CBIR systems. A user who hopes to retrieve all database images of dogs by presenting a CBIR system with a query dog image is likely to be disappointed. Today's systems with automated indexing mechanisms record color, texture, and shape indices in the hope that similarity of these low level features and their locations in the database and query images will imply a high level semantic relationship. Thus, CBIR users today must make do with visual similarity instead of semantic similarity.

Measuring visual similarity with an eye toward semantic similarity between images is still a very difficult problem. In general, database images of interest are unconstrained input to a CBIR system. Images may be taken from any distance, at any time of day, under any weather conditions, under any illuminant, from any angle or viewpoint. Two images of the same object imaged under different illuminants and from different viewpoints will still look similar even though corresponding image pixels may be quite different in color. This visual similarity will persist even when the object is partially occluded from one of the viewpoints.

Comparing images of different scenes, on the other hand, is difficult even without the complication of lighting and viewpoint changes. If we ever want to obtain semantic similarity from measures of visual similarity, then our notion of visual similarity must allow for partial matching of images. A database image with regions that are similar to regions in a query image is likely to be related to the query in some way that the user cares about, and is therefore a good candidate for retrieval. It is common for a semantic relationship to exist even when only part of the information in the database image matches information in the


Figure 1.1: The Importance of Partial Matching. The left and right images are semantically related because both contain zebras, but there are no sky and clouds in the right image, and there are no trees in the left image. Partial matching is crucial in any CBIR system that aims to capture semantic similarity.

query, and only part of the information in the query matches information in the database image. The images in Figure 1.1, for example, are semantically related because they both contain zebras, but there are no sky and clouds in the right image, and there are no trees in the left image. It is also possible for a semantic relationship to exist when only a very small fraction of the one image can be matched to the other image. Consider, for example, the Apple logo image and the Apple advertisement shown in Figure 1.2. The Apple logo is less than one half of one percent of the advertisement. The ability to find even very small partial matches is important in CBIR.

The *image matching problem* is to identify all pairs of visually similar subregions from two images. An efficient solution to this problem is a holy grail in CBIR. Such region similarities provide crucial information to a CBIR system that attempts to make reasonable guesses as to the similarity of the semantic content of images. These guesses may be based on the relative positions of the similar regions, as well as which regions have no similar region in the other image. The set of similar regions can also be displayed to the user to show why a particular database image was retrieved for a given query.

At the heart of virtually any CBIR system is its image distance measure. Such distance measures usually do not operate directly on the images themselves, but rather on image summaries or *signatures* that record information in a form more suitable for efficient comparison. The main distance measure discussed in this thesis is the *Earth Mover's Distance*

Figure 1.2: The Importance of a Small Partial Match. The Apple logo image on the left is semantically related to the Apple advertisement on the right even though the logo covers less than one half of one percent of the total area of the advertisement.

(EMD). The use of the EMD in image retrieval was pioneered by Rubner, Tomasi, and Guibas ([69, 67, 65, 68]). This distance measure compares image signatures which are distributions of mass or weight in some underlying feature space. The weight associated with a particular point in the feature space is the amount of that feature present in the image, and a distribution is a set of (point,weight) pairs. The EMD between two distributions is proportional to the minimum work required to change one distribution into the other. The morphing process involves moving around mass within the feature space (hence the name of the distance measure). The notion of work is borrowed from physics. One unit of work is the amount of work necessary to move one unit of weight by one unit of distance in the feature space. The EMD framework has been successfully applied in color-based ([69, 67, 65, 68]) and texture-based ([69, 68, 66]) retrieval systems. The differences in these two cases are simply the feature space and the distance measure in the feature space.

1.1 The Pattern Problem

This thesis is concerned with a slightly simplified version of the image matching problem which we call the *pattern problem.*

The Pattern Problem. Given an image and a query pattern, determine if the image contains a region which is visually similar to the pattern. If so, find at least one such image region.

In contrast to the image matching problem, in the pattern problem we search for the entire query as a single subregion of the image. A solution to the pattern problem can be used to build a solution to the image matching problem if a query image can be decomposed into atomic regions of interest. This is certainly a reasonable assumption in the CBIR context since the user can manually outline a few relevant regions in the query before submitting it to the system. A routine that solves the pattern problem can be called with each of the query regions as the pattern.

The pattern problem is very difficult because of the combination of partial matching and scaling. The pattern may occur at any location in the image, and at any size. It is not known a priori where to look in the image, or how much of the image area around a given location to examine. Without a good estimate of the scale, it is very difficult for an algorithm to conclude that a pattern does not exist at a particular image location. If the assumed scale of the pattern is too large, then the image location is unfairly penalized because too much information is being examined, much of which may have no matching information in the query pattern. If the assumed scale of the pattern is too small, then much of the pattern information may not be matched because not enough area around the hypothesized pattern location is being examined. These points are illustrated in Figure 1.3, where we compare the color signatures for a query pattern and various size rectangular regions around and within the occurrence of the pattern in a database image.

Another difficult issue in the pattern problem is efficiency, even when the scale of the pattern is known. If the pattern scale is very small, then there are many nonoverlapping (and therefore independent) image locations to check for the pattern. This leads to an efficiency problem in the CBIR context in which a pattern problem must be solved for many (query, database image) pairs. To compound the problem even further, it is difficult to prune a search for a pattern in one image because of a negative search result in another image. Such pruning is possible in CBIR systems which have a true metric as an image distance measure. If query Q is far from database image P, and database image P is close

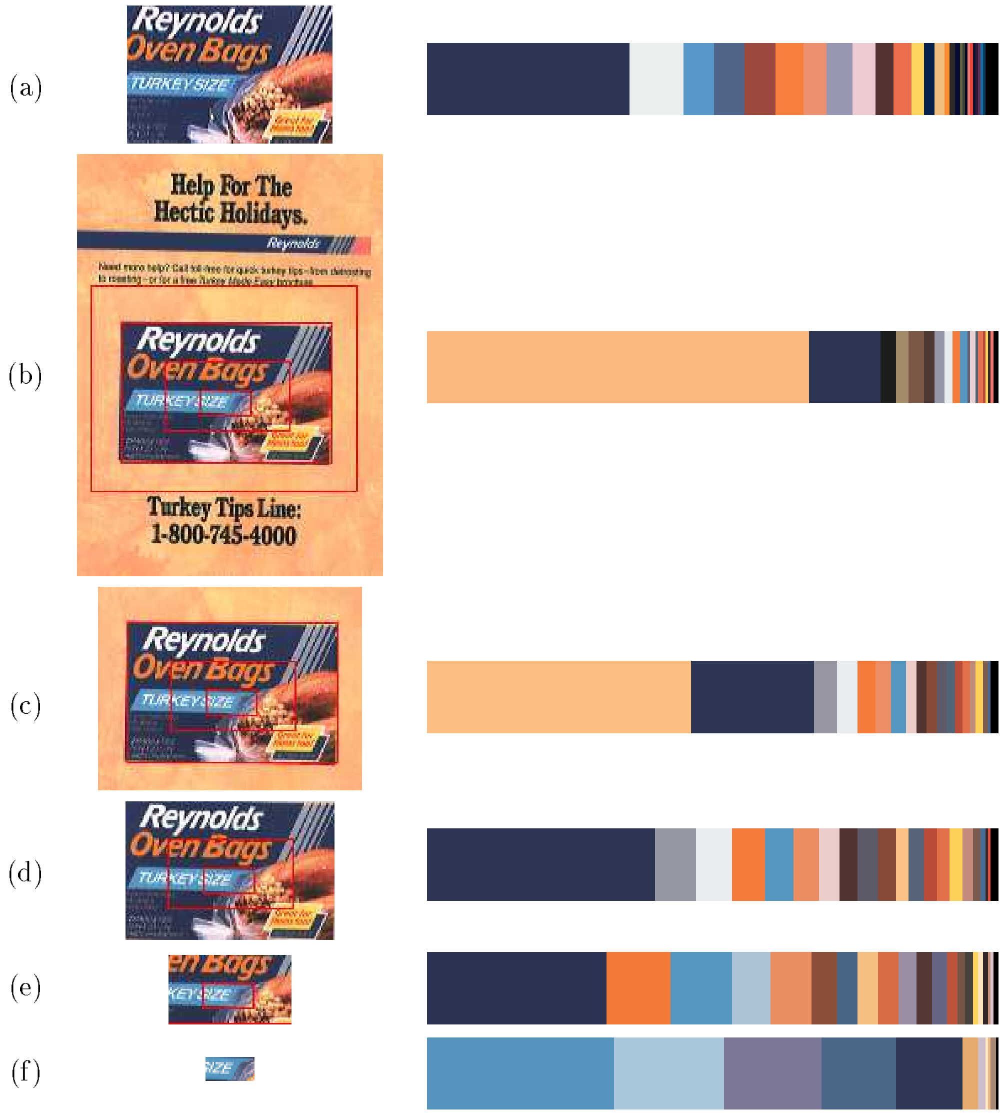


Figure 1.3: The Importance of Scale. The estimated scale at which a pattern appears is important because it determines the amount of information in the database image to compare to the information in the pattern. In row (a), we show the pattern and its color signature. In row (b), we show the database image and its color signature. In rows (c)-(f), we show various subregions of the image in (b) along with their color signatures. The EMD between the pattern signature and each of the image signatures (in CIE-Lab color space units) is (b) 27.7, (c) 19.8, (d) 5.5, (e) 9.4, (f) 20.8. Note that the subregion shown in (d) is almost exactly the occurrence of the pattern in the database image.

to database image R, then the triangle inequality implies that Q is also far from R. Any distance measure which allows for partial matching, however, will not be a metric because the triangle inequality can be violated. A query pattern Q may not occur in image P ($d(Q, P)$ is large), and image P may have many regions in common with image R ($d(P, R)$ is small), but the pattern may still occur in image R ($d(Q, R)$ is small) in a region of R which is not similar to a region in P.

In addition to the sheer number of region comparisons that a brute force approach with or without accurate scale information would perform, there is also the difficulty that each such comparison is not straightforward. Even if the system knows the scale at which the pattern may occur in an image, the pattern may occur rotated in the image with respect to the example presented to the system. Under general imaging assumptions, the comparison between regions must allow for a projective transformation between the image region and the query. In addition to this geometric transformation in image position space, in the color pattern case we might need to account for a photometric transformation in judging visual region similarity. The same object imaged under different light sources may have very different pixel colors, but will still appear very similar to a human observer. In terms of the underlying imaging system, unknown pattern scale, pose, and color appearance are the result of unknown camera-to-object distance, unknown camera viewpoint, and unknown lighting conditions. Also, perceptual color similarity is a complex and ill-understood notion which depends on context and many other factors; these issues are beyond the scope of this work.

Once geometric and photometric factors have been accounted for, the match between images of the same object from different viewpoints and under different illumination conditions will be nearly exact. However, we need to measure similarity and allow for inexact pattern matches. This raises the difficult problem of how to combine color and position information in judging the visual similarity of two color patterns.

1.2 Thesis Overview

This thesis is devoted to the pattern problem in the context of content-based image retrieval. Four main themes are present:

(i) partial matching,

(ii) matching under transformation sets,

(iii) combining (i) and (ii), and

(iv) effective pruning of unnecessary, expensive distance/matching computations.

In **chapter 2**, we give some background information to help put this thesis in the context of previous research. This includes brief descriptions of works with similar goals and that discuss similar problems to those in our work, as well as a discussion of high level differences in motivation, approach, and technique. In **chapter 3**, we solve a 1D shape pattern problem which seeks all (possibly scaled and rotated) approximate occurrences of a pattern polyline shape within another polyline.

The thesis shift gears a bit in **chapter 4** where we discuss the Earth Mover's Distance and a couple of modifications which make it more amenable for use in partial matching settings. One of these modifications is the *partial* EMD in which only a given fraction of the total weight in a distribution is forced to match weight in the other distribution. The other modification is the τ-EMD which measures the amount of weight that cannot be matched when weight moves are limited to at most τ units. Also included in this chapter is an algorithm that uses the EMD to estimate the scale at which a pattern may occur in an image. The issue of efficiency is the central theme of **chapter 5** in which we present efficient lower bounds on the EMD that often allow a system to avoid many more expensive, exact EMD calculations. These lower bounds were developed and are illustrated within the context of the color-based retrieval system described in [65].

In **chapter 6**, we extend the Earth Mover's Distance to allow for unpenalized distribution transformations. We consider the problem of computing a transformation of one distribution which minimizes its EMD to another, where the set of allowable transformations is given. The previously mentioned scale estimation problem is phrased and efficiently solved as an EMD under transformation ($\mathrm{EMD}_{\mathcal{G}}$) problem in which transformations change the weights of a distribution but leave its points fixed. For $\mathrm{EMD}_{\mathcal{G}}$ problems with transformations that modify the points of a distribution but not its weights, we present a monotonically convergent iteration called the *FT iteration*. This iteration may, however, converge to only a locally (cf. globally) optimal EMD value and transformation. The FT iteration is very general, and is modified to work with the partial EMD mentioned above, as well as in some cases when transformations modify both distribution points and weights. We also discuss cases of the $\mathrm{EMD}_{\mathcal{G}}$ problem which can be solved directly, without our iteration.

In **chapter 7**, we describe the SEDL (Scale Estimation for Directed Location) content-based image retrieval system for the pattern problem. The SEDL framework is general enough to be applied to both the color and shape pattern problems. In the shape case, images are sets of curves such as might be produced by edgel detection and linking, or by any standard drawing program. Excellent results for a color database of product advertisements and a shape database of Chinese characters are shown.

A key component in SEDL is the previously mentioned scale estimation module. The output of this module is either an estimate of the pattern scale in the database image or an assertion that the pattern does not appear in the image. In the initial placement phase that follows scale estimation, SEDL efficiently determines a handful of places in the image where the pattern might occur at the previously estimated scale. This small set of promising locations mark the starting points for the final verification and refinement phase. For each initial placement of the query at the estimated scale, SEDL checks for positional consistency of the underlying attributes (for example, colors), modifying the attribute locations by some transformation if this will help improve the match. In recognition of the difficulty of combining attribute and position information, a final check using the τ-EMD helps eliminate false positives.

Finally, we conclude in **chapter 8** with a thesis summary, main insights, and suggestions for future work.

Chapter 2

Background

In this chapter, we give a history of previous work which is related to work contained in this thesis. Along the way, we alert the reader to the high level differences between the previous work and our work.

2.1 The 1D Shape Pattern Problem

The core of this thesis begins in chapter 3 with an algorithm to find a polyline shape pattern within another polyline. Using this algorithm, one can summarize/simplify the description of a polyline by finding pieces of the polyline with more compact descriptions such as "line segment", "corner", or "circular arc".

There are many methods for finding geometric primitives in polylines. The classic pattern recognition book [56] by Pavlidis is an early computer vision reference for approximating, within some error bound, polylines with many vertices by polylines with fewer vertices. Some such approximation algorithms, which essentially find line segments within a polyline, are given in [56] in the chapter titled "Analytical Description of Region Boundaries and Curves". Three excellent, contemporary works from the computer vision community are the Lowe segmentation algorithm ([45]) to divide an edgel chain into straight segments, the "strider" algorithm of Etemadi ([23]) to find straight segments and circular arcs, and the Rosin and West algorithm ([62]) to identify line segments, elliptical arcs, and other high-order curves.

The crucial issue in all such segmentation algorithms is where to stop one description and to begin another. The three works [45], [23], and [62] are all similar in that their breakpoint selection does not use more usual, curvature-based criteria such as curvature zero crossings and extrema. Curvature is sensitive to noise, and the mentioned breakpoint conditions may divide a curve that does not have constant curvature. For example, breaking descriptors at curvature extrema may result in the division of an elliptical arc into two or more pieces. Regardless of the breakpoint strategy, a single pass through the polyline data may not yield very good results. Each of the previously mentioned algorithms also has a second phase that considers replacing two descriptions of adjacent curve pieces with a single description over their union.

Etemadi's "strider" segmentation algorithm ([23]) uses a symmetry condition to determine an initial set of breakpoints. A chain of pixels is labelled as symmetric or asymmetric using the midpoint R of the pixel chain and the line segment PQ connecting its endpoints. The line through R and perpendicular to PQ splits PQ into two segments PS and SQ. The chain from P to Q is in a symmetric state iff the difference in lengths of PS and SQ is less than $1/\sqrt{1+L^2}$, where L is the length of the chain. Note that circular arcs and straight segments are symmetric chains. The strider algorithm adds pixels to a chain until the chain is in an asymmetric state for three consecutive pixel additions. The process then begins again from the first pixel which caused the chain to become asymmetric.

The segmentation algorithms of Lowe ([45]) and Rosin and West ([62]) both use a maximum deviation criterion to compute breakpoints and a scale invariant "significance" formula to decide whether to split a chain further. The Rosin and West algorithm is a generalization of the Lowe algorithm to handle geometric primitives other than line segments. Lowe's algorithm recursively computes a segmentation tree for a pixel chain. The entire chain is approximated by a single segment and then split into two subchains at the pixel which deviates most from the approximation.[1] The *significance* of the approximation is the ratio of the approximation length to the maximum deviation. The splitting algorithm then recurses on the two subchains. The final segmentation into straight lines is computed by traversing the tree up from the leaves and retaining a segment if it is more significant than its children.

The above idea can be applied using any geometric primitive curve for which there is a method for fitting such a curve to pixel chain data. The maximum deviation between the fitted representation and the data can be obtained by computing the maximum of the minimum distances from each chain pixel to the approximation curve. The significance measure remains the same ratio of approximation length to maximum deviation. The first step in the Rosin and West strategy ([62]) is to compute a line segment representation using Lowe's algorithm. The second step applies Lowe's algorithm again to divide the line segment representation into higher order curves such as ellipses or superellipses. The step

[1] An earlier use of the idea to split at the point of maximum deviation is the Douglas-Peucker (poly)line simplification method ([16]). This method approximates a polyline with one of fewer vertices that is within a given error bound ε. If the error in approximating a polyline by a single line segment is greater than ε, then the polyline is split into two at the maximum deviation vertex, and the approximation procedure is recursively applied to the two smaller polylines.

one line segmentation is kept at the leaves of the step two segmentation tree, so not all line segments are necessarily replaced by a higher order curve. Unlike in Lowe's algorithm, the upward tree traversal considers any two adjacent approximations for combination into a single approximation (versus considering only approximations with a common parent). See [62] for algorithm details.

In the computational geometry community, the process of approximating a polyline with a smaller number of segments is called the *min-#* problem. Given a polyline and an error bound, the goal is to compute an approximation polyline with the fewest number of vertices whose distance to the original polyline is within the error bound. If the vertices of the approximation are required to be a subset of the n vertices of the original, then the problem can be solved in $O(n^2)$ time using a graph formulation. The main idea is to construct a graph G with nodes equal to the vertices of the given polyline and where there is an edge from v_i to v_j iff the polyline from v_i to v_j can be approximated within the error bound by a segment from v_i to v_j. The number of vertices in the smallest approximation is equal to the number of edges in a shortest path between the nodes for the first and last vertices. Chan and Chin [6] show how to compute G in $O(n^2)$ time (the brute force method requires $O(n^3)$ time). Since a shortest path in G can be computed in $O(n^2)$ time, the overall $O(n^2)$ bound follows.

Our work on the 1D shape pattern problem does not look for patterns from a particular family of curves. Instead the input pattern can be any polyline shape. If the given "image" polyline has n vertices and the pattern polyline has m vertices, then our algorithm requires $O(m^2n^2)$ time. This reduces to $O(n^2)$ time if the pattern is of constant size (e.g. $m = 2$ for a line segment and $m = 3$ for a corner). Our algorithm was inspired by the Arkin et al. polygon shape metric work [3]. This work compares two polygons by comparing their arclength versus turning angle graphs once the total arclength of both polygons has been normalized to one unit. To make the metric invariant to rotation, the authors allow for an up-down shift of the turning angle graph; to handle the arbitrariness of the first vertex in a polygon description, the authors allow for a cyclic left-right shift of the turning angle graph. There is some experimental evidence ([72]) that the turning angle graph is one of the best shape descriptors for judging perceptual similarity.

We adopt the approach of matching turning angle graphs in our search for a pattern. For the pattern problem, however, we need to allow for partial matching and changes in scale (in addition to changes in orientation). Partial matching means that we match the pattern graph to only a piece of the image graph. To handle a scale change, we allow the arclength axis to be stretched or contracted before matching the graphs.

2.2 Focused Color Searching

In the literature, *focused color searching* refers to the problem of finding a color pattern or model within a color image. The goal is to *focus* on a region of the image which contains the pattern or ascertain that no such region exists. We refer to this problem as the *color pattern problem.*

An early paper in this field is by Swain and Ballard ([77]) who introduced the *Histogram Backprojection* algorithm. If we denote the image and model color histograms as $I = (I_j)_{j=1}^n$ and $M = (M_j)_{j=1}^n$, respectively, then the first step in Histogram Backprojection is to compute the quotient histogram $R = (R_j)_{j=1}^n$, where $R_j = M_j/I_j$. The value R_j represents the probability that an image pixel with color j belongs to an occurrence of the model in the image, assuming that the model appears in the image. If the model appears in the image, then $I_j = M_j + C_j$, where C_j is the number of pixels in the image with color j that are not part of the model. This analysis assumes that the model described by histogram M is the same size in pixels as its occurrence within the image described by histogram I (which is true if the model is cut out from the image). The more "clutter" pixels C_j of color j, the less likely it is that a random image pixel with color j is part of the model.

The second step in Histogram Backprojection is to replace the image color at every pixel with its probability of being part of the model, thus forming the *backprojection image.* Pixels with color j are replaced by the confidence value $\min(R_j, 1)$. White (confidence close to one) regions in the backprojection image are places where the model is likely to occur, and black (confidence close to zero) regions are places where the model is unlikely to occur. The final step in the Histogram Backprojection algorithm is to find the location of the maximum value in the backprojected image after it has been convolved with a mask of the same area as an expected occurrence of the model. This convolution sums confidence values over local areas, and the location of the maximum in the result is the place where the model is most likely to occur.

In the same paper [77], Swain and Ballard also introduce a measure of histogram distance called *Histogram Intersection.* The Histrogram Intersection between image histogram I and model histogram M is defined as

$$H(I, M) = \frac{\sum_{j=1}^n \min(I_j, M_j)}{\sum_{j=1}^n M_j}. \tag{2.1}$$

The numerator of (2.1) is the number of pixels from the model that have corresponding pixels of the same color in the image. The normalization in the denominator of (2.1) guarantees $0 \le H(I, M) \le 1$. This measure was used to determine the identity of an unknown model at a known position in the image. A weighted version of this measure, however, is also at work behind the scenes of the Histogram Backprojection algorithm to find the unknown location of a known model in an image. Let L^S denote the local histogram taken over a $w \times h$ rectangular subregion S of the image. Now suppose that the final Backprojection step uses a $w \times h$ rectangular mask of all ones. If we let $S_{w\times h}$ denote the collection of all $w \times h$ image subwindows, then the place where the model is most likely to occur is the solution to the optimization problem

$$\begin{aligned} \arg \max_{S \in S_{w\times h}} \sum_{(k,l)\in S} 1 \cdot \min(M_{j(k,l)}/I_{j(k,l)}, 1) &= \arg \max_{S \in S_{w\times h}} \sum_{j=1}^{n} L_j^S \min(M_j/I_j, 1) \qquad (2.2) \\ &= \arg \max_{S \in S_{w\times h}} \sum_{j=1}^{n} \left(\frac{L_j^S}{I_j} \right) \min(M_j, I_j), \end{aligned}$$

where $j(k,l)$ is the histogram bin index for the image color at pixel (k,l), and we have used the fact that $\min(M_j/I_j, 1) = \min(M_j, I_j)/I_j$. The weight quantity L_j^S/I_j is the fraction of image pixels of color j that appear in window S.

One major problem with the original Swain and Ballard approach is that its final convolution step essentially assumes the size at which the model occurs in the image. The idea of Ennesser and Medioni in [22] is to compare histograms of local image areas of different sizes and locations to the model histogram using *weighted Histogram Intersection* to measure histogram similarity. The authors suggest using the weighted Histogram Intersection formula

$$\widehat{H}_W(L^S, M) = \sum_{j=1}^{n} W_j \min(L_j^S, M_j),$$

with weights $W_j = 1/I_j$ or $W_j = R_j = M_j/I_j$. Here it is assumed that the model histogram M is normalized to match the total amount of information in the image subwindow S: $\sum_{j=1}^{n} M_j = \sum_{j=1}^{n} L_j^S$. The quantity $\min(L_j^S, M_j)$ is the number of pixels from the scaled model that have corresponding pixels of the same color in the image subwindow S. The weight W_j is an attempt to give more importance to colors j that are more distinctive in the matching process. If the local histogram region S contains the model, then $\min(L_j^S, M_j) = L_j^S$ (remember that the model histogram is normalized to the same total bin count as the local image histogram) and $\widehat{H}_R(L^S, M) = \sum_{j=1}^{n} L_j^S (M_j/I_j)$, which is the same value used in the Backprojection algorithm. This last observation follows from (2.2) and that fact that $M_j \le I_j$ (due to the normalization of the size of M).

Ennesser and Medioni's *Local Histogramming* algorithm looks for model matches as it slides a local window across the image. For each center location of the local window, the algorithm increases the size of the window until the weighted Histogram Intersection measure given above starts decreasing. There is some amount of manipulation to ensure that this scale estimation process avoids local maxima as the window is grown, and that a model is not found in pieces by overlapping local windows. See [22] for some details. The bottom line, however, is that the Local Histrogramming algorithm tries to find the model by an exhaustive search over scale-position space, where a (scale,position) pair is evaluated by a weighted Histogram Intersection between a local image histogram and the scaled model histogram. Swain and Ballard's Backprojection algorithm is an exhaustive search only over position since they assume the scale parameter. The similarity measure used to check a position is a weighted Histogram Intersection between the global image histogram and the model histogram, where the weights are dependent on the position (and assumed scale).

The same exhaustive search over scale-position space using local image histograms is performed by Vinod et al. in [83, 82], but with an upper bound on the Histogram Intersection measure that can be used to prune quickly unattractive (scale,position) pairs. The Histogram Intersection between the model histogram and two local histograms for image regions of similar position and scale will be similar since the two regions have many pixels in common. If the Histogram Intersection between the model and one such local histogram is small, then the Histogram Intersection between the model and the other local histograms will also be small. For any two *focus regions* S and T, the authors show that the Histogram Intersection measures α_S and α_T with the model histogram M are related by the inequality

$$\alpha_T \leq \frac{\min(|S \cap T|, \alpha_S |S|) + |T - S|}{|T|}.$$

The goal of the authors' *Active Search* algorithm is to output image regions that have a Histogram Intersection with the model of at least some threshold θ. The Histogram Intersection for an image region T need not be computed if the above inequality proves that it must be less than θ using the result of a previous Histogram Intersection with a region S. See [82] for the details of Active Search. In [83], the authors report that only on the order of 0.5% of possible focus regions require a Histogram Intersection computation. Although this is an excellent pruning rate, it can still leave thousands of (scale,position) pairs to be checked in the three-dimensional scale-position search space, especially if very small scale model appearances must be found.

None of the works discussed in this section attempt to verify that the positional distribution of colors in an image window are similar to the positional distribution of the model or pattern colors. The algorithms only report image regions that have a similar color histogram to that of the model. In this thesis, we are interested in verifying positional similarity as well as color similarity. This positional verification is part of the last stage of our matching algorithm which also tries to adjust the pattern scale and location to make the positions match as well as possible.

The first phase of our matching algorithm is to estimate, using only color information, the scale at which the pattern might occur in the image. Although our scale estimation algorithm does not work with a histogram color respresention, its instantiation on histograms amounts to scaling the model histogram until the point at which further increases in scale start to decrease similarity to the global image histogram. The Backprojection algorithm assumes the scale is given, while Local Histogramming and Active Search exhaustively search for the best scale at each position.

Armed with our scale estimate, the second stage of our search stategy seeks to find quickly a very small set of promising positions in the image to check for a positionally consistent match. By "very small", we mean fewer than five image locations. If our representation were histogram-based, then these would be the locations of relative maxima in the backprojected image after being convolved with a mask whose size matches our scale estimate. Although our algorithm uses constant color image regions instead of pixels as its basic units of information, the idea of backprojection is still used to identify image regions which are likely to be part of the pattern if it appears in the image. Our notion of a promising image region is also one in which there is a small distance between the color histogram of the region and the color histogram of the pattern. However, we only examine areas around image regions that have high probability of being part of a pattern occurrence in our search for promising locations (this requires preprocessing before query time). In other words, we only use the centers of image regions with a high confidence color as the centers of regions to be checked as promising, instead of using every pixel in the image as in the previously described works.

An important improvement in our application of Swain and Ballard's backprojection idea is the use of a pattern scale estimate in computing the confidences/probabilities that an image pixel of a particular color is part of a pattern occurrence. The true probability for a color c is the number of image pixels with color c that are part of a pattern occurrence within the image, divided by the total number of image pixels with color c. In [77], Swain and Ballard use the number of model pixels of color c as the numerator of this ratio. This gives the correct probabilities if the model is cut out from the image, but it can give very inaccurate probabilities for more general model inputs.

The most computationally expensive part of our search algorithm is the final stage in which we try to adjust the scale and position at which we believe the pattern occurs within the image. This stage makes use of both the colors and their spatial distribution in the image and in the pattern. The adjustment process is started from each of the promising locations found in the previous step. The Local Histogramming and Active Search algorithms also adjust scale in an attempt to get a better Histogram Intersection value, but they do so by fixed step region growing which does not consider the underlying data (color or position) to help determine how much to grow.

2.3 From Histogram Intersection to the Earth Mover's Distance

The major problem with histograms and the common bin-to-bin distance measures defined between histograms is the use of arbitrary bin boundaries. Histogram distance or similarity measures usually only match pixels in corresponding bins. This is certainly true of the Histogram Intersection similarity measure described above, as well as a simple L_1 distance between histograms which sums up the absolute difference in bin amounts over all the bins. Two pixels with very similar colors might be placed in adjacent bins if an arbitrary bin boundary is drawn between the locations in color space. Once this is done, bin-to-bin histogram comparison measures will never be able to match these two very similar pixels. The fault here lies both with the representation and the distance measure. Usually histogram bins are defined by a uniform subdivision of the underlying attribute (e.g. color) space axes. With this uniform subdivision, the bins in every histogram for every image cover the exact same region in the attribute space. This makes it easy to define distance measures between two histograms for any two images by simply comparing corresponding bins, but such bin-to-bin distance measures do not account for the arbitrariness of the histogram representation.

The notion that is missing in bin-to-bin histogram comparison measures is the distance between bins. In the color case, a distance between bins specifies how different the colors in one bin are from the colors in another bin. This dissimilarity will be relatively low for bins which cover adjacent regions in the color space. The problem of using regularly spaced bin boundaries can be overcome to a great extent by allowing matching across bins, and penalizing according to the distance between bins. This is exactly what is done in the histogram distance measure

$$D(I,M) = \min_{F=(f_{ij})} \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} d_{\text{bin}}(\text{bin } i \text{ in I}, \text{bin } j \text{ in M}), \tag{2.3}$$

where f_{ij} represents the amount of bin i from the image histogram which is matched to bin j of the model histogram. The distance asks for the minimum cost matching of the mass in the two histograms. Here we have left out some constraints on $F = (f_{ij})$, including the constraints $\sum_{j=1}^{n} f_{ij} \leq I_i$ and $\sum_{i=1}^{n} f_{ij} \leq M_j$ which state that the amount of mass from one histogram that can be matched to a bin of the other histogram cannot exceed the amount of mass in that bin.

Given the freedom to match across different regions of attribute space, there is no need to define these regions by a regular subdivision. Such a representation does not taylor itself to the data in an image and can be quite inefficient in space. If an image contains no red, then the bins of the histogram that correspond to shades of red will all have zero color mass. From this image's point of view there is no need to consider these attribute space regions in the distance function (2.3). We could compute the histogram of an image using regular subdivisions of the attribute space, throw away the zero bins, and re-number the nonzero bins in sequential order. This makes the image summary more efficient in space, but can be improved further by defining regions in attribute space according to the image data. Suppose, for example, we want regions in attribute space which are summarized by a single attribute value to have diameter less than some threshold. Such a cluster of attribute mass might cross arbitrary bin boundaries even when the bin spacing matches the threshold, thus representing this cluster by more than one entry. Each image can have a different number of clusters which cover different regions in attribute space as efficiently as possible. Thus we can change the bin-based distance measure (2.3) to the cluster-based measure

$$D(I, M) = \min_{F=(f_{ij})} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{\text{cluster}}(\text{cluster } i \text{ in } I, \text{cluster } j \text{ in } M), \tag{2.4}$$

Here the image has m clusters, the model has n clusters, and we must restrict $\sum_{j=1}^{n} f_{ij} \leq I_i$ and $\sum_{i=1}^{m} f_{ij} \leq M_j$. The distance measure (2.4) (once the constraints on F have been fully specified, and the proper normalization has been applied) represents the Earth Mover's Distance (EMD) between distributions of mass in an attribute space. A more precise description of the EMD and the distributions on which it operates will be given in chapter 4. See the works of Rubner et al. [69, 68] and Gong et al. [26] which effectively argue for the use of clustering over histogramming in color-based image retrieval. Experiments in [68] show the superiority of the EMD for color-based image retrieval over many bin-to-bin histogram dissimilarity measures, including Histogram Intersection ([77]), and cross-bin measures, including a common quadratic-form distance ([51]).

2.4 The Earth Mover's Distance

The history of the Earth Mover's Distance begins in 1781 when Gaspard Monge posed the following optimization problem.

> When one must transport soil from one location to another, the custom is to give the name *clearing* to the volume of soil that one must transport and the name *filling* ("remblai") to the space that it must occupy after transfer.
>
> Since the cost of the transportation of one molecule is, all other things being equal, proportional to its weight and the interval that it must travel, and consequently the total cost of transportation being proportional to the sum of the products of the molecules each multiplied by the interval traversed; given the shape and position, the clearing and filling, it is not the same for one molecule of the clearing to be moved to one or another spot of the filling. Rather, there is a certain distribution to be made of the molecules from clearing to filling, by which the sum of the products of molecules by intervals travelled will be the least possible, and the cost of the total transportation will be a *minimum.* (Monge in [50], p. 666, as quoted in [60], p. viii)

If $c(x, y)$ denotes the per unit mass cost of transporting material from $x \in A$ to $y \in B$ for equal-volume sets A and B, then Monge's mathematical formulation is to compute

$$\inf_t \int_{x \in A} c(x, t(x))\, dx$$

over all volume preserving maps $t : A \to B$. In this original formulation, $c(x, y)$ is the Euclidean distance between x and y, the objective function is nonlinear in t, and the set of admissible transportations t is nonconvex.

A major development in the *mass transfer problem* (MTP) came in 1942 when Kantorovich reformulated the problem as a linear optimization over a convex set. If A and B are distributions of mass with density functions $w(x)$ and $u(y)$, respectively, then Kantorovich asked to compute

$$\inf_f \int_x \int_y c(x, y) f(x, y)\, dx dy$$

over all probability density functions $f(x, y)$ with fixed marginals $\int_x f(x, y)\, dx = u(y)$ and $\int_y f(x, y)\, dy = w(x)$. Here it is assumed that A and B have equal total mass $\int_x w(x)\, dx = \int_y u(y)\, dy$. In Kantorovich's formulation, the objective function is linear in f, and the set of admissible f's is convex.

When A and B are (finite) discrete distributions of equal total mass, where A has mass w_i at location x_i and B has mass u_j at location y_j, then Kantorovich's formulation becomes the transportation problem ([32]) from mathematical programming: compute

$$\min_{F=(f_{ij})} \sum_i \sum_j f_{ij} c_{ij}$$

such that

$$f_{ij} \geq 0, \ \sum_i f_{ij} = u_j, \ \sum_j f_{ij} = w_i,$$

where $c_{ij} = c(x_i, y_j)$. Here F is a density function on $\{ x_i \} \times \{ y_j \}$ with fixed marginals $w = (w_i)$ and $u = (u_j)$.[2] If we think of transforming A into B, the f_{ij} is the amount of mass at x_i which *flows* to y_j.

In this thesis, we are mainly concerned with the discrete MTP. Readers interested in the continuous MTP, its history, theory, connection with the discrete MTP, modifications, and applications should see the recent two volume work [60],[61] of Rachev and Rüschendorf, as well as Rachev's survey paper [59].

Often a result for the finite, discrete MTP has a corresponding result for the continuous MTP. For example, in section 5.1.1 we prove that if $c(x, y) = ||x-y||$ for a vector norm $||\cdot||$ or $c(x, y) = ||x - y||_2^2$, then the cost $c(\bar{x}, \bar{y})$ between the centroids $\bar{x} = \sum_i w_i x_i$ and $\bar{y} = \sum_j u_j y_j$ is a lower bound on the EMD between the finite, discrete distributions $\{ (x_i, w_i) \}$ and $\{ (y_j, u_j) \}$ (here we assume $\sum_i w_i = \sum_j u_j = 1$). Virtually the same proof with summations replaced by integrals shows that the distance between the means $\bar{x} = \int_x x w(x) \, dx$ and $\bar{y} = \int_y y u(y) \, dy$ is a lower bound on the EMD between the probablity distributions $w(x)$ and $u(y)$. This should not be surprising because an integral is just the limit of a Riemann sum. The proof of a continuous MTP result can follow the summation manipulations done in the proof of the discrete MTP result, but care must be exercised to ensure that limit signs can be interchanged or can "pass though" functions when necessary.

For another example of a property that holds for both the continuous and discrete MTPs, consider the problem of matching a finite, discrete distribution $\{ (y_1, u_1), \ldots, (y_n, u_n) \}$ to a translate $\{ (y_1 + t, u_1), \ldots, (y_n + t, u_n) \}$. In section 6.7.3, we prove that the matching $f_{ij} = \delta_{ij} u_j$ which matches all the mass at y_i to the mass at $y_i + t$ is optimal. The same proof, which uses the previously discussed centroid lower bound, shows that $f(x, y) = \delta(x - (y - t)) u(y)$ is an optimal matching between $u(y)$ and $u(y - t)$. A consequence, for example, is that the EMD between two uniform normal distributions with means μ_1,μ_2 and equal variances is $||\mu_1 - \mu_2||_2^2$ when $c(x, y) = ||x - y||_2^2$.

There are not many explicit results in the literature for the EMD between two continuous

[2] The discrete formulation follows from the continuous one by putting $w(x) = \sum_i w_i \delta(x - x_i)$ and $u(y) = \sum_j u_j \delta(y - y_j)$, where $\delta(x)$ is the Dirac delta distribution.

distributions, but there is a nice result for matching two normal distributions. Let $N(\mu, \Sigma)$ denote a normal distribution with mean μ and covariance matrix Σ. If $c(x,y) = ||x-y||_2^2$, then (p. 119 in [60],[59],[53],[17])

$$\mathrm{EMD}(N(\mu_1, \Sigma_1), N(\mu_2, \Sigma_2)) = ||\mu_1 - \mu_2||_2^2 + \mathrm{tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_2^{1/2}\Sigma_1\Sigma_2^{1/2})^{1/2}). \tag{2.5}$$

This result is symmetric in Σ_1 and Σ_2 since it can be shown that $\mathrm{tr}((\Sigma_2^{1/2}\Sigma_1\Sigma_2^{1/2})^{1/2}) = \mathrm{tr}((\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2})^{1/2})$. For uniform Gaussians in $\mathbf{R}^K$ with $\Sigma_1 = \sigma_1^2 I_K$ and $\Sigma_2 = \sigma_2^2 I_K$, formula (2.5) reduces to

$$\mathrm{EMD}(N(\mu_1, \sigma_1^2 I_K), N(\mu_2, \sigma_2^2 I_K)) = ||\mu_1 - \mu_2||_2^2 + K(\sigma_1 - \sigma_2)^2,$$

which agrees with our result for the equal-variance case $\sigma_1 = \sigma_2$. The result (2.5) is actually a consequence of a more general theorem which also yields the EMD between uniform distributions over equal-volume ellipsoids (p. 119 in [60],[17]).

Let us now return to the case of discrete distributions with a discussion of the *match distance* between histograms defined by Werman, Peleg, and Rosenfeld in 1985 ([86]). The main contribution of this work is the use of a distance between bins as given in (2.3). The match distance is defined between two histograms $H^0 = (h_i^0)_{i=1}^n$ and $H^1 = (h_i^1)_{i=1}^n$ with equal total bin counts $\sum_i h_i^0 = \sum_i h_i^1$. The *unfolding* of histogram $H = (h_i)_{i=1}^n$ is defined to be the multiset $\mathrm{UF}(H)$ with h_i copies of bin i. For example, the unfolding of the 2D histogram

$$H = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 3 & 0 \end{bmatrix} \qquad \text{is} \qquad \mathrm{UF}(H) = \{\ (1,1), (1,1), (2,3), (3,2), (3,2), (3,2)\ \},$$

where bins are labelled by (row,column) pairs. The cost of a 1-1 match between $UF(H^0)$ and $UF(H^1)$ is the sum of the distances between matching bins. The match distance between H^0 and H^1 is the cost of the minimum cost 1-1 matching between the multisets $UF(H^0)$ and $UF(H^1)$.

The match distance between H^0 and H^1 is, in fact, the Earth Mover's Distance between the distributions $\{\ (\text{bin } i, h_i^0)\ \}$ and $\{\ (\text{bin } i, h_i^1)\ \}$:

$$\min_{F=(f_{ij})} \sum_{i=1}^n \sum_{j=1}^n f_{ij} d_{\mathrm{bin}}(\text{bin } i, \text{bin } j)$$

such that

$$f_{ij} \geq 0, \; \sum_{i=1}^{n} f_{ij} = h_j^0, \; \sum_{j=1}^{n} f_{ij} = h_i^1.$$

It turns out that a transportation problem with all integer masses has an optimal flow $F = (f_{ij})$ with only integer values ([32]). Such an optimal flow is therefore also an optimal 1-1 matching between $UF(H^0)$ and $UF(H^1)$.

The match distance is used by Peleg, Werman, and Rom [57] as a uniform framework for changing the spatial and gray-level resolution of an image. In this work, images are 2D histograms of photons where the value of bin/pixel (i, j) is its gray level $I(i, j)$. The authors pose the problem of reducing the number of gray levels in an image from $2G + 1$ (labelled $0..2G$) to $G+1$ as an optimization problem that seeks an image I' with gray levels $0..G$ such that the match distance between I and $2I'$ is as small as possible, where the bin distance is the Euclidean distance between pixel locations. This problem is reduced to computing a minimal sum-of-distances pairing of pixels in I with odd gray level. See [57] and [85] for details. Since the minimal pairing computation takes time $O(m^3)$, where m is the number of pixels in I with odd gray level, the authors suggest using a linear time algorithm to find an approximately optimal matching.

Peleg, Werman, and Rom also formulate a change in spatial resolution from N pixels $\{ x_i \}$ in I to N' pixels $\{ y_j \}$ in I' as a match distance problem. The final gray level at pixel y_j is taken to be a linear combination of the pixels in I: $I'(y_j) = \sum_i \alpha_{ij} I(x_i)$. The match distance is used to compute the weights α_{ij} between $N'I$ and NI' (which both have total mass NN'). In other words, one multiset has N' copies of every pixel in I and the other multiset has N copies of every pixel in I'. If the optimal matching is r_{ij}, then $\alpha_{ij} = r_{ij}/N$ so that the gray level range of I' is the same as that of I. The authors point out that their formulation is not restricted to rectangular grids and can be adapted to weight some pixels (for example, those close to an image edge) more heavily than others.

The previously discussed works concentrate on the case when the total mass of the two distributions is equal. In this thesis, we pay close attention to the partial matching case in which one distribution is "heavier" than the other. For example, in section 5.1.2 we use the centroid lower bound between equal-mass distributions to derive a lower bound on the EMD between unequal-mass distributions. In the unequal-mass case, some of the mass in the heavier distribution is not matched to mass in the lighter distribution. In this dissertation, we also focus on the problem of finding an optimal transformation (from a predefined set of transformations) of the mass locations in one distribution so that its EMD to another distribution is minimized. Finally, we note that our focus is on the EMD between distributions of masses located at points. This does not expose the full generality of the

EMD, for the EMD can be used to compare two weighted collections of any objects for which there is a notion of distance of between objects to provide the costs in the transportation problem. We could use the EMD, for example, to compare two distributions of distributions, where the cost between the lower level distributions is itself an Earth Mover's Distance.

The seminal work which developed the EMD for use in content-based image retrieval is Rubner's thesis ([64]). We shall refer to his work throughout this dissertation.

2.5 Matching under Transformation Groups

In measuring visual similarity, it is often important to allow some transformation of locations in feature space before measuring the distance between two collections of attributes. In the color case, for example, changing the lighting of a scene causes some transformation of the underlying pixel colors. The images of the same scene under different illuminations may still look quite similar even though the pixel values may be quite different. Directly comparing histograms with Histogram Intersection or distributions of color mass with the EMD will not capture such visual similarity. Another example of the importance of allowing transformations is when the underlying attribute is (or includes) the image plane position of ink on a page. Two images of the same object from different distances and viewpoints will be visually similar despite differences in scale, orientation, and image location. Directly comparing the position of the ink will not capture this visual similarity.

Although not its intended purpose, the EMD defines a distance between point sets if one considers the mass at each point in a set to be one unit. The EMD between point sets is the minimum of the average distance between corresponding points taken over all one-to-one correspondences between the sets. In the next two subsections, we consider two other widely used distances between point sets along with the methods used to compute these distances under transformation groups.

2.5.1 The Hausdorff Distance

The Hausdorff distance between finite point sets $A = \{ a_1, \ldots, a_m \}$ and $B = \{ b_1, \ldots, b_n \}$ is defined as

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

where

$$h(A, B) = \max_{a \in A} \min_{b \in B} \rho(a, b) \tag{2.6}$$

is the directed Hausdorff distance from A to B, and $\rho(a, b)$ is the distance between points a and b. The directed Hausdorff distance $h(A, B)$ from A to B is small whenever each point of A is close to some point in B. The directed distance is appropriate when trying to find a model within an image, since we want all the points in the model set to be close to some point in the image set, but not necessarily vice-versa. In this case, we use the directed distance from the model point set to the image point set. The symmetric Hausdorff distance $H(A, B)$ is small when each point in A is close to some point in B, and each point in B is close to some point in A. The Hausdorff distance defines a metric between finite point sets if the underlying point distance ρ is a metric. The Hausdorff distance can be computed trivially in $O(mn)$ time, but with some cleverness ([1]) this time can be improved to $O((m+n)\log(m+n))$.

In contrast to the EMD applied to point sets, the Hausdorff distance does not use one-to-one correspondences between points. In the directed distance computation (2.6), the same point $b \in B$ can be the nearest neighbor to many different points $a \in A$. This many-to-one matching can be quite advantageous for problems involving very large point sets which do not require one-to-one correspondences. To be fair to the EMD, it is more general than the Hausdorff distance in the sense that it can be used to match distributions, of which point sets are a special case.

For a transformation group $\mathcal{G}$, the Hausdorff distance under $\mathcal{G}$ is defined as

$$M_{\mathcal{G}}(A, B) = \min_{g \in \mathcal{G}} H(A, g(B)).$$

In words, we find the transformation $g \in \mathcal{G}$ which matches A and $g(B)$ as closely as possible. The problem of computing the Hausdorff distance under a transformation group has been considered for various transformation groups $\mathcal{G}$ (e.g. translation, Euclidean), with different norms ρ (e.g. L_2 or L_∞), and in different dimensions d (e.g. 1, 2, ≥ 3). An $O(n \log n)$ time algorithm for the translation case in one dimension is given in [63]. In [8], Chew et al. give an algorithm for the translation case with the L_∞ point distance that runs in time $O(n^3 \log^2 n)$ in dimension $d = 2$ in time $O(n^{(4d-2)/2} \log^2 n)$ in dimension $d \geq 3$. For point sets in the plane with the L_2 distance, the Hausdorff distance under translation can be computed in $O(n^3 \log n)$ time ([35]) and the Hausdorff distance under Euclidean transformations (translation plus rotation) can be computed in $O(n^5 \log n)$ time ([9]). In presenting the time bounds, we have assumed that $m = O(n)$.

The Hausdorff matchers of Rucklidge et al. ([36],[70]) are aimed at the practical problem of finding a binary model within a binary image. Here the point set defined by a binary image is the nonnegative integer point set which includes exactly the locations of the pixels which are "on" in the image. In [36], a rotated, translated version of the model can be located, while in [70] an affine transformation of the model is allowed. The underlying distance measure is the directed Hausdorff distance under Euclidean transformations in [36], and the directed Hausdorff distance under affine transformations in [70]. Actually, these works use a more robust version of the Hausdorff distance which prevents outliers from affecting the distance computation. The partial directed Hausdorff distance from the model point set B to the image point set A is defined as

$$h_K(B, A) = K^{th}_{b \in B} \min_{a \in A} \rho(a, b),$$

where $K^{th}_{b \in B}$ denotes the K^{th} ranked distance of the distances $\min_{a \in A} \rho(a, b)\ \forall b \in B$, and $0 < K \leq n = |B|$. The user specifies the fraction f, $0 < f \leq 1$ which determines $K = \lfloor fn \rfloor$. In this way, the user does not need to know the number of points in the model. The fraction f allows for a fraction $1 - f$ of the points in the model to be outliers (not near any image points).

The Hausdorff matchers described in [36] and [70] both search in a discretized transformation space. The search space is limited to a finite (but usually very large) number of transformations by the user. The transformation space is discretized into a rectangular grid so that moving by one grid unit in transformation space produces only a small change in the transformed model. The authors leverage off the fact that if the Hausdorff distance is large for a particular grid transformation, then it will also be large at grid transformations in a neighborhood of that transformation. This allows some transformations to be eliminated from consideration without explicitly computing the Hausdorff distance. In [36], the search algorithm loops over all grid transformations and skips transformations that can be ruled out by Hausdorff distances evaluated at previously visited grid cells. Here "ruled out" means that it can be proven that a transformation yields a Hausdorff distance which is (1) larger than a user specified threshold if all matches within a given threshold are desired or (2) larger than the best Hausdorff distance seen so far if the user only wants the best match to be reported.

In [70], the affine search space is six-dimensional, so developing efficient search techniques is crucial. A multi-level cell decomposition strategy is used instead of a loop over all grid transformations. The initial space of transformations to search is tiled with rectilinear cells of equal size (i.e. an equal number of grid transformations inside). If a cell can be proven *not* to contain a transformation which needs to be reported to the user, then it is not searched further. Otherwise, the cell is marked as "interesting". After considering all the cells at the current level, each of the interesting ones are subdivided and the process is repeated. This search process is a breadth first search in the tree representing the recursive cell decomposition. In the case when a single match is required (either any match or the minimum distance match), a best-first search in which we investigate the most promising interesting cells first will help reduce the total search time. See [70] for the measure used to rank interesting cells.

The same idea of a hierarchical search in a discretized transformation space is used again by Rucklidge ([71]) to find a gray level model in a gray level image. In this work, the search accounts for an affine transformation of the geometry of the gray level pattern. The pruning strategies given are applicable to a number of block distance functions such as SSD (sum of squared differences) and MAD (mean absolute deviation), and other functions which use the gray level differences between corresponding pixels of the image and the transformed model. Distance functions which compare the gray level of a transformed model pixel to a neighborhood of the corresponding image pixel can also be accomodated to account for noise and small shape changes which cannot be captured by an affine transformation. The following results were reported: (1) correct identification of a translated version of a 28×70 model within a 640×480 image examining only 1303 cells versus the approximately 250000 possible translations, (2) correct registration of a figure under rigid motion examining about 7.2×10^7 cells compared to the approximately 5×10^9 rigid grid motions, and (3) correct patch-by-patch matching of two affinely-related images examining only 6.8×10^8 cells compared to over 2×10^{13} possible transformations. No running times were given.

The main strength of the works [36], [70], and [71] is that these methods do not miss any good matches and are guaranteed to report the globally optimal match under whatever distance measure (SSD, MAD, etc.) is used. In contrast, our search strategy will move from any point in transformation space to a locally optimal transformation, with no guarantee that this transformation is globally optimal. The key to the correctness and efficiency of our strategy lies in the efficient selection of a few good initial transformations from which to begin our iteration so that it will converge to an optimal or nearly optimal transformation. The key to the correctness of Rucklidge's works is that all transformations are potentially considered, while the key to their efficiency is the ability to prune large areas of the transformation space without explicitly considering individual transformations.

The pruning rates reported in [36], [70], and [71] are quite impressive, but there are still many transformations that must be explicitely considered – probably too many for the application of content-based image retrieval (CBIR) in which hundreds or thousands of pattern problems must be solved within a minute or so. Given our CBIR motivation, it is more important for us to solve *almost all* pattern problems *correctly and very quickly*, rather than to solve *all* pattern problems *correctly but more slowly*. Another main difference in our approaches is the behavior of our approaches in an area around the best match. Even in the case when only the best match must be reported (as opposed to all matches within a certain threshold), the Rucklidge algorithms must consider many transformations around the best match because around any good match are other good matches which although not globally optimal are near optimal enough to be quite difficult to eliminate without a closer look. In our approach of following a downhill path in transformation space (i.e. a sequence of transformations which always improves the EMD value), getting close to the best match speeds up the convergence of the iteration. If "close" is close enough, the downhill iteration will move from suboptimal but good matches near the optimal match to the optimal one, and it will do so without visiting every good match in the neighborhood of the best one. Finally, we note that we have also extended the EMD for robustness reasons so that only some mass in the lighter distribution needs to be matched to mass in the heavier distribution.

2.5.2 The ICP Iteration

The *iterative closest point* (ICP) iteration ([5]) was developed to register two 3D shapes. Given a 3D "model" shape and a 3D "data" shape, the goal is to find the rotation and translation of the data shape which registers it with part of the model. Here the word *model* is used in the exact opposite of the way it is used in the previous discussion of Hausdorff matchers – the model shape is searched for the data shape pattern. The model shape can be represented as the union of any collection of geometric primitives such as points, line segments, curves, triangles, etc., as long as there is a routine available to compute the point on a primitive which is closest to a given point. The data shape must be represented as (or decomposed into) a point set. The distance from the data shape point set $B = \{ b_1, \ldots, b_n \}$ to the model shape primitive set $A = \{ a_1, \ldots, a_m \}$ is given by

$$D^{\mathrm{ICP}}(B, A) = \sum_{j=1}^{n} \min_{a \in A} d^2(b_j, a) = \sum_{j=1}^{n} \min_{a \in A} \min_{p \in a} ||p - b_j||_2^2,$$

where $d(b, a)$ is the L_2 distance from point b to the closest point on the geometric primitive a. If we assume that the model is represented as a 3D point set, then the ICP distance function becomes

$$D^{\mathrm{ICP}}(B, A) = \sum_{j=1}^{n} \min_{a \in A} ||a - b_j||_2^2.$$

The ICP distance from B to A is very similar to the Hausdorff distance from B to A, except that D^{ICP} sums up the distances to the nearest neighbors instead of taking the maximum.

The ICP iteration is designed to solve the optimization problem

$$D_{\mathcal{G}}^{\mathrm{ICP}}(B,A) = \min_{g\in\mathcal{G}} D^{\mathrm{ICP}}(g(B),A) = \min_{g\in\mathcal{G}} \sum_{j=1}^{n} \min_{a\in A} ||a - g(b_j)||_2^2,$$

where $\mathcal{G} = \mathcal{E}$, the group of Euclidean transformations. The idea here is to alternately compute (1) the distance minimizing transformation for a fixed set of nearest neighbor correspondences, and then (2) the nearest neighbor correspondences for a fixed data shape transformation. More precisely, the two steps are

$$\psi^{(k)}(j) = \arg\min_i ||a_i - g^{(k)}(b_j)||_2^2 \quad \forall j = 1,\ldots,n \quad \text{and} \tag{2.7}$$

$$g^{(k+1)} = \arg\min_{g\in\mathcal{G}} \sum_{j=1}^{n} ||a_{\psi^{(k)}(j)} - g(b_j)||_2^2. \tag{2.8}$$

The correspondence $\psi^{(k)}$ maps the index of a point in $g^{(k)}(B)$ to the index of the closest point in A. The use of the Euclidean distance squared facilitates the transformation step because the least squares optimization problem (2.8) has a known, closed-form solution for $\mathcal{G} = \mathcal{E}$ (and for many other transformation sets $\mathcal{G}$). It is not difficult to show that the sequence $< D^{\mathrm{ICP}}(g^{(k)}(B),A) >_k$ is a monotonically descreasing, convergent sequence starting with any initial transformation $g^{(0)}$. The least squares registration (2.8) reduces the average distance between corresponding points during each iteration, while the the nearest neighbor correspondences reduce the individual distance from the points in the transformed data shape to corresponding points in the model.

The idea behind the ICP iteration is a very general one. To see this, we shall rewrite the correspondence step as

$$\psi^{(k)} = \arg\min_{\psi\in\Psi} \sum_{j=1}^{n} ||a_{\psi(j)} - g^{(k)}(b_j)||_2^2, \tag{2.9}$$

where Ψ is the set of all functions from $[1..n]$ to $[1..m]$. Since the functions in Ψ are not restricted in any way, the solution to (2.9) makes $\psi^{(k)}(j) \in [1..m]$ equal to the index of the point in A which is closest to $g^{(k)}(b_j)$ (as in (2.7)). Computing $D_{\mathcal{G}}^{\mathrm{ICP}}(B,A)$ is an optimization problem over $\Psi \times \mathcal{G}$. The ICP iteration alternates between finding the best $\psi \in \Psi$ for a fixed $g \in \mathcal{G}$, and the best $g \in \mathcal{G}$ for a fixed $\psi \in \Psi$. In this way, a path in $\Psi \times \mathcal{G}$ is traced out for which the ICP objective function always decreases or remains the same. The ICP iteration is an example of the general *alternation strategy* in which the path followed in search space always proceeds downhill along subpaths over which some subset of the search space variables are constant. Mathematically, the alternation idea applies to any optimization problem $\text{opt}_{V=V_1\cup V_2\cup\cdots\cup V_N} f(V_1, V_2, \ldots, V_N)$ for which one can solve $V_i^*(V_1, \ldots, V_{i-1}, V_{i+1}, \ldots, V_N) = \arg\text{opt}_{V_i} f(V_1, V_2, \ldots, V_N)$.

When the objective function to be minimized is bounded below, the *go downhill* strategy employed by the alternation strategy is guaranteed to converge, albeit possibly to only a local (as opposed to global) minimum (an analogous statement holds for the alternation strategy applied to maximization problems). This is the case in the ICP iteration since the distance function to be optimized is nonnegative. Of course, the main drawback of the alternation strategy is that there is no guarantee of convergence to the global optimum.

Our method for computing the EMD under a transformation set applies the alternation strategy to obtain a decreasing, convergent sequence of EMD values. As in the ICP problem of registering shapes, there is a step to determine the best correspondences for a fixed transformation. The correspondence step in matching distributions with the EMD is more complicated than (2.9). In the EMD case, there is a real-valued variable f_{ij} that indicates how much mass at location i in one distribution is matched to location j in the the other distribution. The correspondences $F = (f_{ij})$ must be constrained so that each location does not match more mass than it possesses. In addition to allowing transformations of mass locations, we can also handle some sets of transformations which alter both mass locations and amounts.

The key to the effective use of a correspondence-transformation alternation to solve CBIR matching problems is the selection of a small number of promising transformations from which to start the iteration. The number must be small for efficiency reasons, and one of the initial transformations must be close to the globally optimal one so that one of the sequences converges to an optimal or to a nearly optimal match. In the case when the whole model shape matches the data shape, the ICP authors suggest a method for computing a good initial rotation from eigenshape analyses of the model and data shapes. A promising initial translation in this case lines up the centroids of the two shapes. This is a good initial transformation strategy when all the data in the model shape is matched, but comparing global descriptors such as those mentioned above will not work in the partial matching case when the data shape matches only some (possibly very small) amount of the model. In this case, the authors suggest a dense sampling of the transformation space to produce a set of initial transformations.

Even in the case of partial matching, it may be possible to quickly determine a few very promising places in an image to look for a pattern. Consider the color pattern problem. If, for example, there is a yellow blob in the pattern and yellow appears only in a few places within the image, then one of those few places is likely to contain the pattern if it appears at all within the image. Even if yellow appears in many places within the image, it may be possible to quickly eliminate most of these places based on the other colors surrounding the yellow. The pattern will not match an image region which has very different colors or very different ratios of color amounts even if the same colors are present. These two simple checks do not use the positions of the colors within in an image region. The number of image regions checked for initial pattern placement can be kept quite low by using only the most distinctive pattern colors with respect to the image to select these regions. By examining the amounts of the most distinctive pattern colors in the pattern and in the image, one can often obtain a very good estimate of the pattern scale. This scale estimate is crucial in the preceeding discussion since the scale determines how much of the image to compare to the pattern.

The basic ideas discussed above are also applicable to the shape pattern problem in which only differences in scale and location, not in orientation, are allowed. The orientation of ink on the page in the shape pattern problem plays the same role as color in the color pattern problem: the orientation does not change under the allowable transformations. Distinctive pattern orientations with respect to the image (i.e. orientations which occur in the image mainly in an image region which contains the pattern) give a lot of leverage in computing a scale estimate and possible locations of the pattern within the image. Even if the rotated versions of the pattern are to be located in the image, distinctive relative orientations may give the needed leverage to determine efficiently a small set of initial similarity transformations. The ideas discussed above are used to select initial transformations in SEDL, which is our CBIR pattern problem system.

2.6 The FOCUS Image Retrieval System

The FOCUS (Fast Object Color-based qUery System) image retrieval system designed by Das et al. ([14]) addresses the color pattern problem in the CBIR context. An important non-technical contribution of the FOCUS work is that it highlights the importance of partial matching in CBIR systems without assumptions about scale, orientation, position, or background. The system operates in two phases. During the first phase, database images which do not contain all the colors present in the query pattern are eliminated from further consideration. The second phase uses spatial color adjacency relationships to filter the phase one results.

The preprocessing step for phase one identifies the peaks in local color (HSV) histograms measured over a uniform grid of rectangular image subregions. These peaks are recorded as the colors present in the image. Local histograms are used instead of a global histogram to help prevent color peaks in the occurrence of a query pattern from being masked or shifted by peaks from the background colors. Ideally, one of the local histogram subregions should be the image area which is exactly the query occurrence since this would guarantee an image color peak which exactly matches a query color peak. Of course, this cannot be done because the problem under investigation is to find the query. The color peaks for all images in the database are collected into an indexing structure which supports range queries. Each peak is tagged with the image and the cell within the image in which it occurs. The phase one index also includes a frequency table which specifies the number of images that will be returned by a range query (over a fixed size range) for every point in a discretization of the HSV color space.

At phase one query time, the peaks in a global color histogram over the query image are computed. For each query peak, a range query centered at that peak is performed to identify all database images with a similar color. Using the frequency table, the query peaks are processed in order of increasing number of returned images. The image lists for each query peak are joined to form a list of images that have peaks matching every query peak. The result of phase one is this list of images along with a set of peak correspondences for each image on the list.

The preprocessing step for phase two matching builds a *spatial proximity graph* (SPG) that captures the spatial adjacency relationships between color regions in an image. The nodes in an image SPG correspond roughly to (color peak,cell) pairs. The construction process starts with a list of color peaks for each cell. Two color peaks are connected if it is possible that their underlying regions are adjacent in the image; there is an edge between two color peaks if the peaks are in the same cell or if two peaks of the same color are four-neighbors on the cell grid. There is some collapsing of connected nodes of the same color to obtain scale invariance. See [14] for details. The edges in an SPG show all possible pixel level adjacencies that could occur (although the SPG construction does not perform any pixel level processing), but some false adjacencies may be included.

During phase two, the SPG of each image returned in the phase one result is searched for an occurrence of the query SPG as a subgraph. If an image SPG does not contain the query SPG, then it is assumed that the image does not contain the query. The subgraph search is done after each image SPG node label is replaced by the query color label of the matching peak (obtained from the phase one result). Image nodes whose color does not match any query color are eliminated before the search. The reduced image SPG can be checked for the query subgraph in time $O(n^m)$, where n is the size of the query adjacency matrix and m is the maximum number of occurrences of a color label in the reduced image SPG (usually ≤ 3 according to [14]). The distance between a query and database image is given by the sum of HSV distances between query and corresponding image peaks. The phase two matching removes images from the phase one result list ordered by this distance function.

The main high-level difference between SEDL and FOCUS is that we use absolute position information, while Das et al. use relative position information. One positive consequence of using relative information is that adjacencies of color regions remain the same regardless of the scale, orientation, position, or continuous deformation of a pattern occurence within an image. In the FOCUS work, there is no transformation to estimate in order to determine if the pattern is present. The search for an optimal transformation is replaced by the search for a subgraph in the image SPG. Underlying both searches is the question of which regions can match one another.

In FOCUS, a query region can only match an image region whose color is most similar to that of the query region (and is within some threshold). Deciding which color in the query matches which color in the image makes the subgraph isomorphism problem computationally feasible. Position information is used to determine if a pair of image regions can match a pair of query regions. A pair of adjacent regions cannot match a pair of non-adjacent regions. In contrast, we decide which regions can match by using a continuous distance measure that is a weighted combination of color and position distance between regions. Our notion of a good match is that for each query region there is a nearby image region of a similar color. This translates into a small combined color-position distance. Of course, our choice to use such a distance function forces us to make the very difficult decision of how much to weigh the individual color and position distances.

The corresponding difficulty in FOCUS is the lack of robustness that may result from using the `possibly-adjacent-to` relation – the absence of a single edge in the image SPG can cause the algorithm to conclude that the pattern does not exist within the image. The FOCUS answer to this problem is to err on the side of introducing false adjacencies in favor of missing true adjacencies. Recall that their SPG (spatial proximity graph) captures all *possible* spatial adjacencies between image colors. False adjacencies may be introduced because the graph construction works at the level of cells instead of pixels.

The size of the cell is crucial here. At one extreme, using a single cell equal to the whole image will give many false adjacencies, but results in a very small SPG to match. In this case, the nodes of the SPG are all the colors present in the image and there is an edge between every pair of colors. Also, the likelihood that query color peaks will be masked by background colors increases as the cell size increases. At the other extreme, using cells which are pixels results in no false adjacencies but gives very large SPGs that cannot be matched in times which are acceptable for an image database application. Suppose, for example, there is an orange and blue checkerboard that fills one cell. Then this cell contributes a blue node connected to an orange node. If we change the cell size to the size of a single checkerboard square, then each checkboard square contributes one blue or orange node and all four-neighbors of opposite colors are connected. The challenge faced by the FOCUS work is to choose the cell size so that there are not too many false adjacencies but so that the sizes of the SPGs are small enough to match SPGs quickly.

In [14], Das et al. report a recall rate of 90% and a precision of 75% after phase two. These results are excellent, although the results obtained directly from their web demo at `http://vis-www.cs.umass.edu/~mdas/color_proj.html` were not as good (08/17/98). In section 7.5.1, we show that SEDL achieves better recall and competitive precision results. FOCUS, however, has a sizeable advantage over SEDL in speed, requiring less than a second on average for a query in a database of 1200 images of product advertisements and nature images. SEDL requires an average of about 40 seconds per query on a size 361 subset of the FOCUS database.

Finally, we note that FOCUS never really verifies the occurrence of the pattern within the image. It simply eliminates images that cannot possibly contain the pattern. The hope is that the pattern is distinctive enough to eliminate almost all images except those that contain the pattern. In this elimination process, FOCUS does *not* use the sizes of uniform color regions or the amounts of colors present. In contrast, SEDL uses this information to estimate the scale and location of the pattern within the image. Another difference is that SEDL can show the user where it believes the pattern is located, while FOCUS cannot do so. Such feedback may be useful if users can change system parameters in an attempt to improve query results. A further discussion of differences between SEDL and FOCUS is given in section 7.5.1.3.

Chapter 3

The Polyline Shape Search Problem

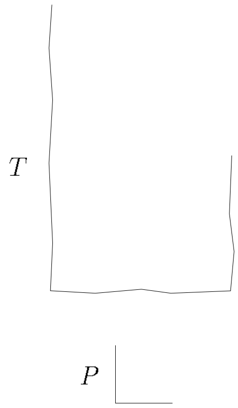
Shape comparison is a fundamental problem in computer vision because of the importance of shape information in performing recognition tasks. For example, many model-based object recognition algorithms work by matching boundary contours of imaged objects ([39, 73, 43, 44, 10, 55]). In addition to this traditional application, shape information is also one of the major components in some content-based image retrieval systems (see e.g. [51] and [46]). The goal in such a system is to find database images that look similar to a given query image or drawing. Images and queries are usually summarized by their color, shape, and texture content. For example, in the illustration retrieval system described in [11], we suggest a shape index which records *what* basic shapes (such as line segments, corners, and circular arcs) fit *where* in the drawing. The method in this chapter can be used to index shape information in images once contours have been extracted.

In this chapter, a shape is a polyline in the plane. We consider the following problem: Given two planar polylines, referred to as the *text* and the *pattern*, find all approximate occurrences of the pattern within the text, where such occurrences may be scaled and rotated versions of the pattern. We call this problem of locating a polyline pattern shape within a polyline text shape the *polyline shape search problem*, or PSSP for short. The PSSP is difficult because we allow both scaling and partial matching. Allowing scaling of the input pattern requires us to make precise certain inherent tradeoffs in PSSP matching. Consider for example, trying to match a line segment pattern into a text polyline. The pattern will match a segment of the text polyline exactly at a continuum of different scales. In this case, we only wish to report the maximum length match. In addition, suppose that the angle between two consecutive segments of the text is very close to 180°. In this case, we may prefer a single longer match with a very small error that spans the two segments instead of two shorter, zero error, matches for each segment.

Two closely related problems to the PSSP are the *segment matching problem* ([40, 30, 31]) and the *polyline simplification problem* ([6, 24, 37, 49, 79]). The segment matching problem is to find approximate matches between a short polygonal arc and pieces of a longer polygonal arc. In searching for these matches, the short arc is allowed to rotate, but its length/scale remains constant. The PSSP generalizes the segment matching problem by allowing scaling. In the polyline simplification problem, a polyline and error bound are given, and we seek an approximation polyline with the fewest number of segments whose distance to the given polyline is within the error bound. (This problem is also known in the literature as the *min-#* problem.) For a polyline with n vertices, the planar polyline simplification problem can be solved in $O(n^2)$ time if the vertices of the approximation are required to be a subsequence of the vertices of the given polyline ([6]), and in $O(n)$ time if the vertices of the approximation can be arbitrary points in the plane but the given polyline is the graph of a piecewise linear function ([37]). One can think of the polyline simplification problem as an attempt to find long segments within the input polyline, i.e. the PSSP with a line segment pattern. The PSSP generalizes the polyline simplification problem because the PSSP allows for any polyline pattern, not just a line segment. PSSP matches are *not* required to start and end at vertices of the input text (and hence PSSP matches may overlap). For a text polyline with n vertices, our algorithm for the PSSP with a line segment pattern requires $O(n^2)$ time.

This chapter is organized as follows. In section 3.1, we describe the framework for our solution to the PSSP, including the match score for a given scale, orientation, and position of the pattern within the text. In section 3.2, we derive the orientation of the pattern which gives the best match for a fixed pattern scale and position. This leaves us with a 2D search problem in scale-position space (the position is the text arclength at which to begin the comparsion of the pattern with the text). In section 3.3, we show how a certain set of lines divides up the scale-position plane into regions in which we have analytic formulae for our scoring function. In section 3.4, we fully describe our line sweep algorithm for the PSSP. Section 3.5 shows some results of our algorithm. Finally, we summarize our work and suggest areas for future work in section 3.6.

The material in this chapter has been published in [13].



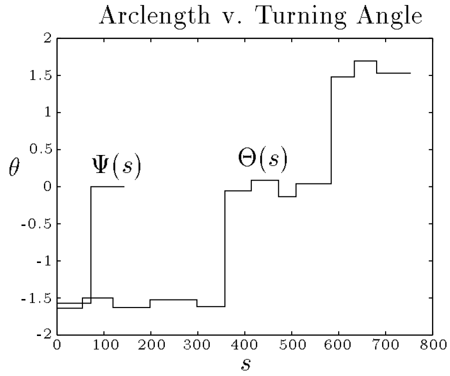


Figure 3.1: PSSP Turning Angle Summaries. (a) Text T above the corner pattern P. (b) Arclength versus cumulative turning angle functions $\Theta(s)$ and $\Psi(s)$ for T and P, respectively (s in points, θ in radians).

3.1 Problem Setup

Let T and P denote the text and pattern polylines, respectively. We will use the familiar arclength versus cumulative turning angle graph in judging the quality of a match. We denote these summary graphs for the text and pattern as $\Theta(s)$ and $\Psi(s)$, respectively. Figure 3.1 shows an example. If T consists of n segments and P consists of m segments, then $\Theta(s)$ and $\Psi(s)$ are piecewise constant functions with n and m pieces, respectively. We denote the text arclength breakpoints as $0 = c_0 < c_1 < \cdots < c_n = L$, where L is the length of the text. The value of $\Theta(s)$ over the interval (c_i, c_{i+1}) is denoted by θ_i, $i = 0, \ldots, n-1$. Similarly, we denote the pattern arclength breakpoints as $0 = a_0 < a_1 < \cdots < a_m = l$, where l is the length of the pattern, and the value of $\Psi(s)$ over the interval (a_j, a_{j+1}) is ψ_j, $j = 0, \ldots, m-1$.

Rotating the pattern by angle γ simply shifts its summary graph by γ along the turning angle axis, while scaling the pattern by a factor α stretches its summary graph by a factor of α along the arclength axis. Hereafter, a scaled, rotated version of the input pattern will be referred to as the *transformed pattern*. The comparison between the transformed pattern and the text will be done in the summary coordinate system. The text arclength at which to begin the comparison will be denoted by β. Since the length of the transformed pattern is αl, the transformed pattern summary graph is compared to the text summary graph from β to $\beta + \alpha l$. Finding the pattern within the text means finding a stretching, right shift, and up shift of the pattern summary graph $\Psi(s)$ that makes it closely resemble the corresponding

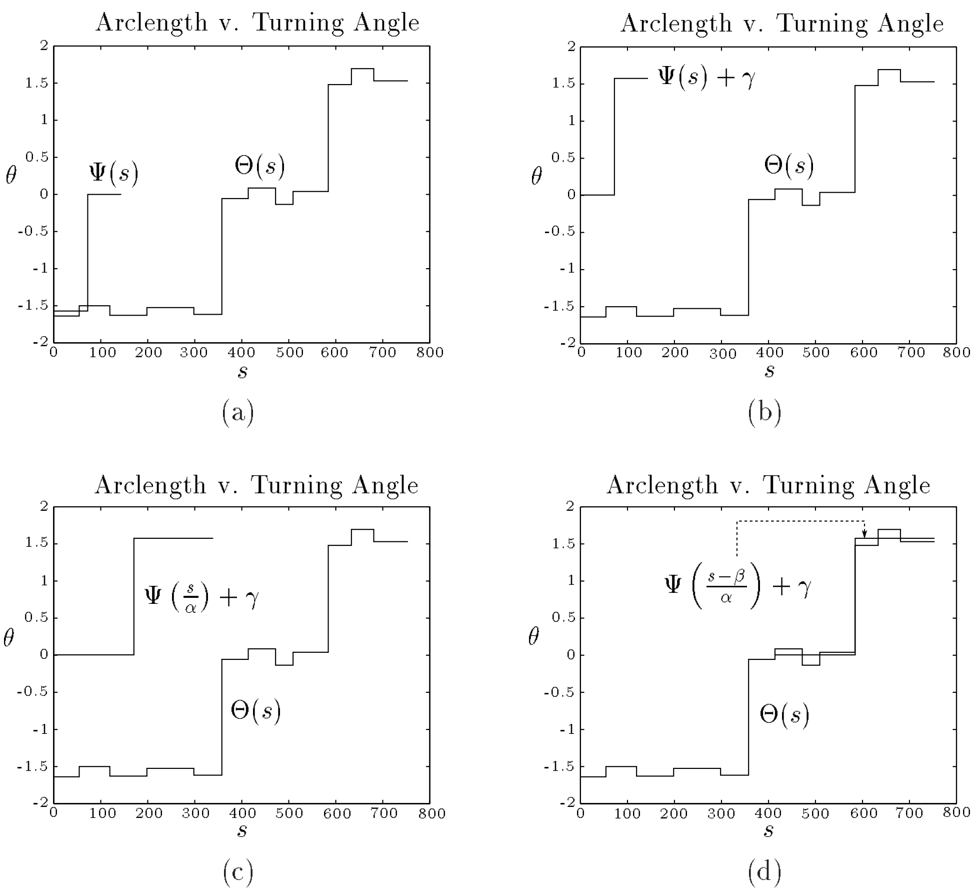


Figure 3.2: PSSP Matching in Arclength Versus Turning Angle Space. (a) $\Theta(s)$ and $\Psi(s)$ from Figure 1(b). (b) Rotating the pattern by γ shifts its summary graph up by γ. (c) Scaling the rotated pattern by α stretches the summary graph by a factor of α. (d) Finally, we slide the transformed pattern summary graph over by an amount β to obtain a good match.

piece of the text summary graph. Figure 3.2 illustrates this intuition. The stretching (α), up shifting (γ), and right shifting (β) of the pattern summary graph $\Psi(s)$ correspond to scaling, rotating, and sliding the pattern along the text, respectively. The problem of finding the pattern within the text is thus a search problem in the scale-position-rotation space (α, β, γ).

The preceeding discussion tacitly assumes that the pattern and text are both *open* polylines. If, for example, the text is closed (i.e. the text is a polygon), then the above strategy will miss a pattern match that crosses the arbitrary start and finish text arclengths $s = 0$ and $s = L$ which actually correspond to the same point on the text. Such a match will not be missed if we match the pattern summary curve to a text summary curve which runs from $s = 0$ to $s = 2L$, where $\Theta(s + L) = \Theta(s) + 2\pi$ for $s \in [0, L)$. Since the underlying text summary is repeated twice, we need to be careful not to report two identical matches. In what follows, however, we ignore the issue of open/closed polylines, and simply assume that our pattern and text are open polylines.

In judging the quality of a match at a given scale, orientation, and position, we need to consider both the error of the match and the length of the match. Is a long match with a large error better than a short match with a small error? Using the mean squared error, for example, indicates an indifference to the length of the match. If the pattern were a single segment, then it would fit with zero mean squared error at a continuum of different scales and locations along each edge of the text polyline. The mean squared error metric cannot distinguish among these matches. In order to do so, we will use a match scoring function which rewards longer matches. When the mean squared error (length) of two matches is equal, the longer (lower mean squared error) match will have a higher score than the shorter (higher mean squared error) match. Of course, there is still the issue of how to compare a match to a shorter (longer), lower mean squared error (higher mean squared error) match. There is no one correct answer to this balancing question – the answer depends, for example, on the underlying input noise model. Here we opt for a simple balancing of match length versus match error which, as we shall see, yields very good results and is amenable to analysis via standard calculus optimization techniques. Obviously, other match score functions are possible.

In moving toward our scoring metric, we define the mean squared error $e(\alpha, \beta, \gamma)$ as

$$e(\alpha, \beta, \gamma) = \frac{\int_{s=\beta}^{\beta+\alpha l} \left(\Theta(s) - \left(\Psi\left(\frac{s-\beta}{\alpha}\right) + \gamma\right)\right)^2 \, ds}{\alpha l}.$$

Note that $\Psi((s - \beta)/\alpha) + \gamma$, $s \in [\beta, \beta + \alpha l]$, is the summary graph of the transformed pattern, starting at text arclength β. The score $S(\alpha, \beta, \gamma)$ of a match is defined in terms of the mean squared error as

$$S(\alpha, \beta, \gamma) = \frac{\alpha l}{L(1 + e(\alpha, \beta, \gamma))}.$$

The product αl is the length of the match (α, β, γ). Our goal is to find local maxima of the score function S over a suitable domain D. This domain is defined by restricting the values of α and β so that the domain of definition of the stretched, shifted pattern summary graph is completely contained in $[0, L]$ (which is the domain of definition of the text summary graph):

$$D = \{ \ (\alpha, \beta) \mid \alpha > 0, \ \beta \geq 0, \ \text{and} \ \alpha l + \beta \leq L \ \}.$$

Although the range of the mean squared error e is $[0, \infty)$, the range of the score S over the domain D is $[0, 1]$. A match of length L with zero mean squared error receives the highest score of one. Instead of trying to locate local maxima of S in D, we will try to find local minima of its reciprocal

$$R(\alpha, \beta, \gamma) \equiv \frac{1}{S(\alpha, \beta, \gamma)} = \frac{L}{\alpha l}(1 + e(\alpha, \beta, \gamma)).$$

Note that the rotation γ affects only the mean squared error portion of the score.

At a local maximum location $(\alpha_*, \beta_*, \gamma_*)$ of S, a small change in pattern scale, orientation, or position within the text decreases the match score. We do not, however, want to report all local maxima because two very similar matches may be reported. We want to report a complete set of independent matches. By independent matches, we mean that any two reported matches should be significantly different in at least one of the defining components: scale, orientation, and position. If a pattern fits very well into a piece of text at a particular scale, orientation, and position within the text, then the pattern at the same scale and orientation but with a slightly different position will also fit very well. Both matches should not be reported. We report only those matches at which S is a local maximum in a topological neighborhood of the match, where the topological elements are the vertices, edges, and regions of an arrangement of a certain set of lines in scale-position space. For example, we output a vertex (α, β) of the arrangement only if it is a better match than its neighboring vertices and all local maximum locations on adjacent edges. (The complete algorithm is given in section 3.4.) Using a topological neighborhood instead of a geometric one is only a heuristic used to achieve our goal of reporting independent matches.

3.2 The Best Rotation

In this section we fix (α, β) and derive the rotation angle $\gamma = \gamma_*(\alpha, \beta)$ which minimizes the mean squared error $e(\alpha, \beta, \gamma)$ and, hence, the reciprocal match score $R(\alpha, \beta, \gamma)$. This is straightforward because e is differentiable with respect to γ:

$$\frac{\partial e}{\partial \gamma}(\alpha, \beta, \gamma) = 2\left(\gamma - \frac{\int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) - \Psi\left(\frac{s-\beta}{\alpha}\right)\right)\, ds}{\alpha l}\right).$$

The derivative $\partial e / \partial \gamma$ is equal to zero exactly when

$$\gamma = \gamma_*(\alpha, \beta) = \frac{\int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) - \Psi\left(\frac{s-\beta}{\alpha}\right)\right)\, ds}{\alpha l},$$

the mean value of the difference $\Theta - \Psi$ (more precisely, $\Theta(s) - \Psi((s-\beta)/\alpha)$) over the arclength interval of the match. Since $\partial^2 e/\partial\gamma^2(\alpha,\beta,\gamma) \equiv 2 > 0$, we conclude that for fixed α and β, the rotation angle that minimizes the mean squared error is $\gamma = \gamma_*(\alpha,\beta)$ given above. If we define $e_*(\alpha,\beta) \equiv e(\alpha,\beta,\gamma_*(\alpha,\beta))$, then

$$e_*(\alpha,\beta) = \frac{\int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) - \Psi\left(\frac{s-\beta}{\alpha}\right)\right)^2\, ds}{\alpha l} - \left(\frac{\int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) - \Psi\left(\frac{s-\beta}{\alpha}\right)\right)\, ds}{\alpha l}\right)^2.$$

The function $e_*(\alpha,\beta)$ is the variance of $\Theta - \Psi$ over the arclength interval of the match.

3.3 The 2D Search Problem

The result of the previous section allows us to eliminate the rotation parameter from consideration in our score and reciprocal score functions. We define $R_*(\alpha,\beta) = R(\alpha,\beta,\gamma_*(\alpha,\beta))$. Our goal now is to find local minima in the domain D of

$$R_*(\alpha,\beta) = \frac{L}{\alpha l}\left(1 + \frac{I_2(\alpha,\beta)}{\alpha l} - \left(\frac{I_1(\alpha,\beta)}{\alpha l}\right)^2\right), \tag{3.1}$$

where

$$I_1(\alpha,\beta) = \int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) - \Psi\left(\frac{s-\beta}{\alpha}\right)\right)\, ds \quad \text{and} \tag{3.2}$$

$$I_2(\alpha,\beta) = \int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) - \Psi\left(\frac{s-\beta}{\alpha}\right)\right)^2\, ds. \tag{3.3}$$

Consider the evaluation of the integral $I_1(\alpha,\beta)$ for a fixed pair (α,β). Since Θ and Ψ are piecewise constant functions, this integral can be reduced to a finite summation of terms such as the product of $(\theta_i - \psi_j)$ with the length of the overlap of the ith arclength interval (c_i, c_{i+1}) of $\Theta(s)$ and the jth arclength interval $(a_j\alpha+\beta, a_{j+1}\alpha+\beta)$ of $\Psi((s-\beta)/\alpha)$. In precise mathematical terms, we have

$$I_1(\alpha,\beta) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1}(\theta_i - \psi_j) \times X_{ij},$$

where $X_{ij} = |(c_i, c_{i+1}) \cap (a_j\alpha+\beta, a_{j+1}\alpha+\beta)|$ and $|J|$ is the length of interval J.

Let l_{ij} denote the line $a_j\alpha + \beta = c_i$, $i = 0,\ldots,n$, $j = 0,\ldots,m$, in the $\alpha\beta$-plane. The four lines

$$l_{ij} \quad : \quad a_j\alpha + \beta = c_i,$$

$$\begin{aligned} l_{i+1,j} &: a_j\alpha + \beta = c_{i+1}, \\ l_{i,j+1} &: a_{j+1}\alpha + \beta = c_i, \quad \text{and} \\ l_{i+1,j+1} &: a_{j+1}\alpha + \beta = c_{i+1} \end{aligned}$$

divide the $\alpha\beta$-plane into regions in which we may write down explicit analytic formulae for $X_{ij} = X_{ij}(\alpha, \beta) = |(c_i, c_{i+1}) \cap (a_j\alpha + \beta, a_{j+1}\alpha + \beta)|$. In each region, the formula for the size of the intersection is (at most) a degree one polynomial in α and β. For example, when $c_i < a_j\alpha + \beta < c_{i+1} < a_{j+1}\alpha + \beta$, it is easy to check that $X_{ij} = c_{i+1} - (a_j\alpha + \beta)$. This situation is depicted in Figure 3.3. Note that the given formula for X_{ij} also holds if we replace $<$ with $\leq$ in the comparisons between text and transformed pattern breakpoints. All possible formulae for $X_{ij}(\alpha, \beta)$ are illustrated in Figure 3.4. Now let $\mathcal{L}$ denote the set of $(n+1)(m+1)$ lines l_{ij} and let $\mathcal{A} = \mathcal{A}(\mathcal{L})$ denote the arrangement[1] in $\alpha\beta$-space of the lines in $\mathcal{L}$. In each face f of $\mathcal{A}$, we have a degree one polynomial formula for X_{ij}, $X_{ij}^f = u_{ij}^f\alpha + v_{ij}^f\beta + w_{ij}^f$. As explained above, the formula for $X_{ij} = X_{ij}(\alpha, \beta)$ is determined by the above–below relationship between (α, β) and each of the four lines $l_{ij}, l_{i+1,j}, l_{i,j+1}$, and $l_{i+1,j+1}$. From this fact, it is easy to see that the above–below relationship of (α, β) and the line l_{ij} affects only the four intersection formulae $X_{ij}, X_{i-1,j}, X_{i,j-1}$, and $X_{i-1,j-1}$.

Note that $\mathcal{A}$ is an arrangement of non-vertical lines. The slope of l_{ij} is $-a_j \leq 0$, so all lines have negative or zero slope. Note also that $\mathcal{A}$ is degenerate because there are many pairs of parallel lines ($l_{i_1,j} \parallel l_{i_2,j}$). An arrangement vertex v_{ijpq} is the intersection of l_{ij} and l_{pq}. The scaling and sliding $(\alpha, \beta) = v_{ijpq}$ of the pattern lines up exactly two pairs of breakpoints: $a_j\alpha + \beta$ coincides with c_i and $a_q\alpha + \beta$ coincides with c_p. An arrangement edge e is an open segment along some line l_{ij}. A scaling and sliding $(\alpha, \beta) \in e$ lines up exactly one pair of breakpoints: $a_j\alpha + \beta$ coincides with c_i. For an open arrangement face f, any scaling and sliding $(\alpha, \beta) \in f$ lines up no pairs of breakpoints.

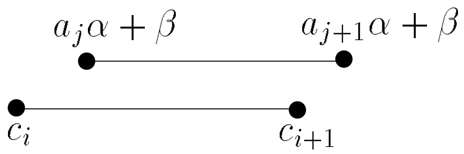
Now fix a face f of the arrangement $\mathcal{A}$ and let $Y_{ij} = \theta_i - \psi_j$. Then for (α, β) in the closure $\bar{f}$, the integrals (3.2), (3.3) in the formula for R_* can be written as

$$I_1^f(\alpha, \beta) = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} X_{ij}^f Y_{ij}, \quad \text{and} \tag{3.4}$$

$$I_2^f(\alpha, \beta) = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} X_{ij}^f (Y_{ij})^2. \tag{3.5}$$

[1] For a survey of arrangements, see [19].

(a)



$$X_{ij} = c_{i+1} - (a_j\alpha + \beta)$$

$$\begin{array}{rcll} a_j\alpha+\beta & > & c_i & (\text{above } l_{ij}) \\ a_j\alpha+\beta & < & c_{i+1} & (\text{below } l_{i+1,j}) \\ a_{j+1}\alpha+\beta & > & c_i & (\text{above } l_{i,j+1}) \\ a_{j+1}\alpha+\beta & > & c_{i+1} & (\text{above } l_{i+1,j+1}) \end{array}$$

(b)

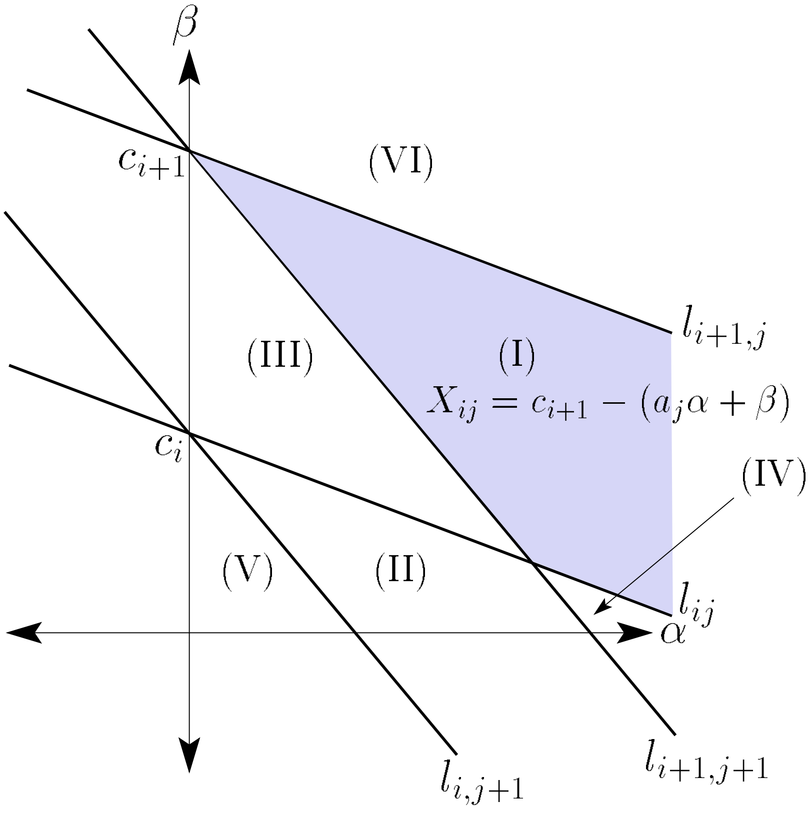


Figure 3.3: The Interval Overlap X_{ij}. (a) Intervals for which $c_i < a_j\alpha+\beta < c_{i+1} < a_{j+1}\alpha+\beta$. In this case, $X_{ij} = c_{i+1} - (a_j\alpha+\beta)$. (b) The corresponding region in scale-position space is shown in gray. The formulae for all six regions (I)–(VI) bounded by $\alpha \geq 0$, $\beta \geq 0$, l_{ij}, $l_{i+1,j}$, $l_{i,j+1}$, and $l_{i+1,j+1}$ are given in Figure 3.4.

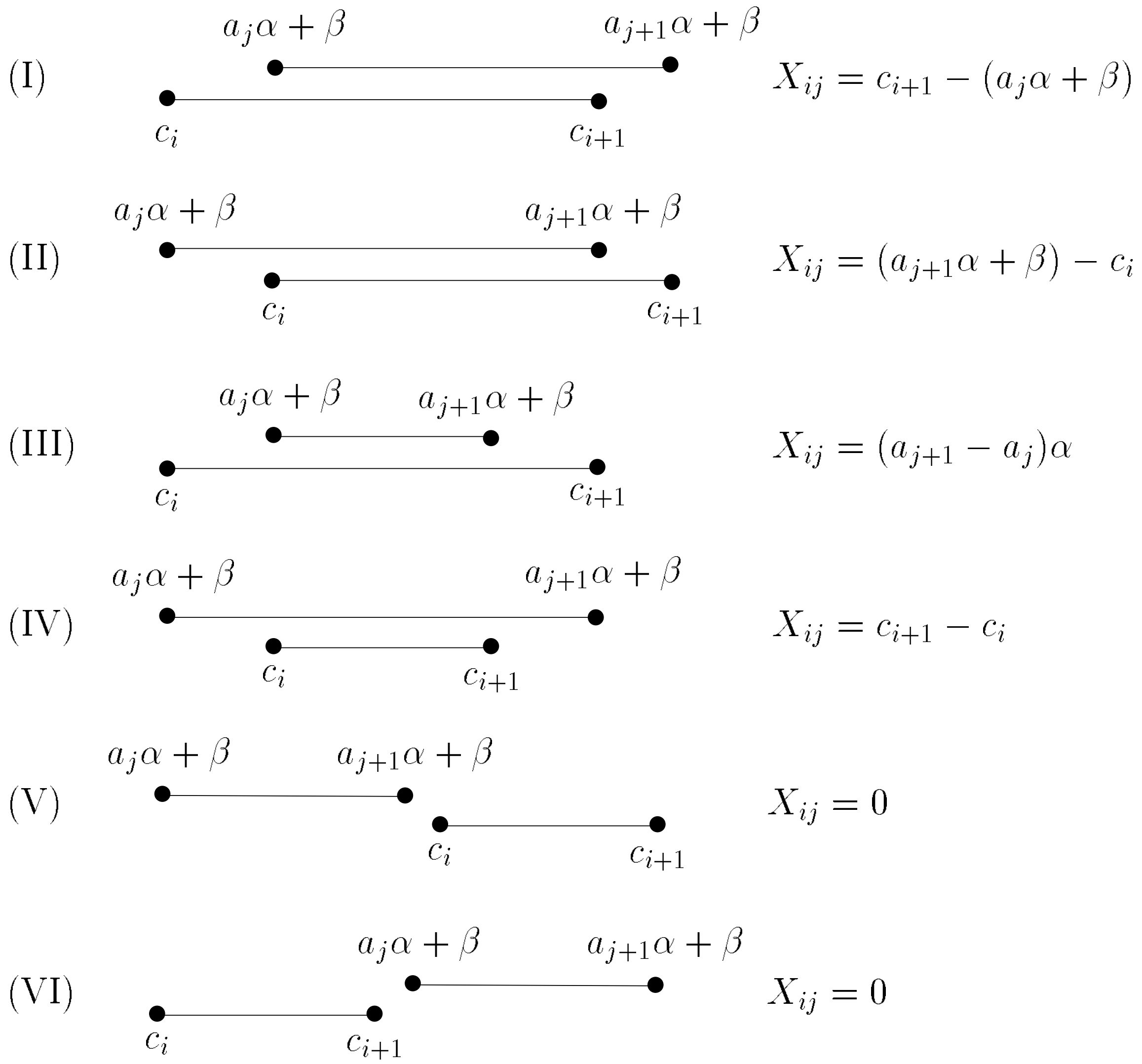


Figure 3.4: All Possible Interval Overlaps X_{ij}. In all cases, $X_{ij} = X_{ij}(\alpha, \beta)$ is (at most) a degree one polynomial in α and β. (I) $c_i \leq a_j\alpha + \beta \leq c_{i+1} \leq a_{j+1}\alpha + \beta$. (II) $a_j\alpha + \beta \leq c_i \leq a_{j+1}\alpha + \beta \leq c_{i+1}$. (III) $c_i \leq a_j\alpha + \beta \leq a_{j+1}\alpha + \beta \leq c_{i+1}$. (IV) $a_j\alpha + \beta \leq c_i \leq c_{i+1} \leq a_{j+1}\alpha + \beta$. (V) $a_j\alpha + \beta \leq a_{j+1}\alpha + \beta \leq c_i \leq c_{i+1}$. (VI) $c_i \leq c_{i+1} \leq a_j\alpha + \beta \leq a_{j+1}\alpha + \beta$.

Substituting $X_{ij}^f = u_{ij}^f\alpha + v_{ij}^f\beta + w_{ij}^f$ into these formulae and gathering like terms gives

$$\begin{aligned} I_1^f(\alpha,\beta) &= \widetilde{u}^f\alpha + \widetilde{v}^f\beta + \widetilde{w}^f \qquad \text{and} & (3.6)\\ I_2^f(\alpha,\beta) &= \widehat{u}^f\alpha + \widehat{v}^f\beta + \widehat{w}^f, & (3.7)\end{aligned}$$

where $\widetilde{u}^f = \sum_{ij} u_{ij}^f Y_{ij}$, $\widehat{u}^f = \sum_{ij} u_{ij}^f (Y_{ij})^2$, and similarly for $\widetilde{v}^f$, $\widehat{v}^f$, $\widetilde{w}^f$, and $\widehat{w}^f$. Combining (3.1), (3.6), and (3.7), we can write R_* in the closed region $\bar{f}$ as

$$R_*^f(\alpha,\beta) = \frac{L}{\alpha^3 l^3}(A^f\alpha^2 + B^f\alpha\beta + C^f\beta^2 + D^f\alpha + E^f\beta + F^f) \tag{3.8}$$

for constants

$$\begin{aligned} A^f &= l^2 + l\widehat{u}^f - (\widetilde{u}^f)^2, & \\ B^f &= l\widehat{v}^f - 2\widetilde{u}^f\widetilde{v}^f, & (3.9)\\ C^f &= -(\widetilde{v}^f)^2, & (3.10)\\ D^f &= l\widehat{w}^f - 2\widetilde{u}^f\widetilde{w}^f, & \\ E^f &= -2\widetilde{v}^f\widetilde{w}^f, \qquad \text{and} & (3.11)\\ F^f &= -(\widehat{w}^f)^2. & \end{aligned}$$

Our 2D search problem is to find pairs $(\alpha,\beta) \in D$ at which $R_*(\alpha,\beta)$ is a local minimum.

3.3.1 Faces

Can a local minimum of R_* occur in the interior of a face?[2] We have not been able to rule out this possibility, although we will argue that local minima inside faces are somewhat rare. We will also show that every open ball around a face local minimum location (α_*,β_*) contains other points (α,β) such that $R_*(\alpha,\beta) = R_*(\alpha_*,\beta_*)$. In this sense, there are no *strict* local minima of R_* inside an arrangement face. Furthermore, the same value $R_*(\alpha_*,\beta_*)$ can always be found on an arrangement edge or vertex which is adjacent to the face containing (α_*,β_*). Given the above considerations, we ignore the possibility of face local minima in our algorithm. In the rest of this section, we argue the previously made claims.

In order to determine if a local minimum of R_* can occur inside a face f, we must examine the function $R_*^f(\alpha,\beta)$ defined on $\bar{f}$ (see (3.8)).

[2] In [13], we concluded that there are no local minima of R_* inside an arrangement face. Our proof (given in the appendix of [13]), however, contained an error in the handling of the case $\widetilde{v}^f = 0$.

Theorem 1 *The function $R_*^f(\alpha,\beta)$ has no local minima in f if $\widetilde{v}^f \neq 0$ or $\widehat{v}^f \neq 0$. Note from equations (3.6) and (3.7) that $\widetilde{v}^f$ and $\widehat{v}^f$ are the coefficients of β in $I_1^f(\alpha,\beta)$ and $I_2^f(\alpha,\beta)$, respectively. Thus, the theorem can be rephrased to say that $R_*^f(\alpha,\beta)$ has no local minima if one of the integrals $I_1^f(\alpha,\beta)$ or $I_2^f(\alpha,\beta)$ depends on β.*

Proof. The first and second partial derivatives of R_*^f with respect to β are

$$\begin{aligned}\frac{\partial R_*^f}{\partial \beta} &= \frac{L}{\alpha^3 l^3}(B^f\alpha + 2C^f\beta + E^f) \\ \frac{\partial^2 R_*^f}{\partial \beta^2} &= \frac{2L}{\alpha^3 l^3}C^f.\end{aligned} \tag{3.12}$$

If $\widetilde{v}^f \neq 0$, then $C^f = -(\widetilde{v}^f)^2 < 0$, and, consequently, $\partial^2 R_*^f/\partial\beta^2 < 0$ (since $\alpha > 0$). The concavity of R_*^f with respect to β implies that $R_*^f(\alpha,\beta)$ cannot have a local minimum in the β direction for any α. Therefore, $R_*^f(\alpha,\beta)$ cannot have a local minimum when $\widetilde{v}^f \neq 0$.

So now suppose that $\widetilde{v}^f = 0$. From the formulae (3.9), (3.10), and (3.11), we see that $B^f = l\widehat{v}^f$, $C^f = 0$, and $E^f = 0$ in this case. The first derivative (3.12) then reduces to $\partial R_*^f/\partial\beta = (L/(\alpha^3 l^2))\widehat{v}^f$. Thus $\partial R_*^f/\partial\beta = 0$ iff $\widehat{v}^f = 0$. If follows that $R_*^f(\alpha,\beta)$ cannot have a local minimum when $\widetilde{v}^f = 0$ and $\widehat{v}^f \neq 0$.

We have shown that $R_*^f(\alpha,\beta)$ cannot have a local minimum when $\widetilde{v}^f \neq 0$ or ($\widetilde{v}^f = 0$ and $\widehat{v}^f \neq 0$). These conditions are, of course, logically equivalent to $\widetilde{v}^f \neq 0$ or $\widehat{v}^f \neq 0$. ■

Next, we seek an explicit condition on the text polyline which guarantees that $\widetilde{v}^f \neq 0$.

Define $i_0(\alpha,\beta)$ and $i_1(\alpha,\beta)$ to be the text indices where the transformed pattern graph domain $[\beta, \beta+\alpha l]$ begins and ends:

$$i_0(\alpha,\beta) = i_0 \text{ if } \beta \in [c_{i_0}, c_{i_0+1}) \qquad \text{and} \qquad i_1(\alpha,\beta) = i_1 \text{ if } \beta+\alpha l \in [c_{i_1}, c_{i_1+1}).$$

See Figure 3.5. For any $(\alpha,\beta) \in f$, no two breakpoints of the text and the transformed pattern graph line up. This means that $i_0(\alpha,\beta) \equiv i_0^f$ and $i_1(\alpha,\beta) \equiv i_1^f$ are constant for $(\alpha,\beta) \in f$. It also means that $I_1^f(\alpha,\beta)$ is differentiable with respect to β at points in f. By (3.2), $\partial I_1^f/\partial\beta = \partial/\partial\beta(\int_{s=\beta}^{\beta+\alpha l}\Theta(s)\,ds)$ since $\int_{s=\beta}^{\beta+\alpha l}\Psi((s-\beta)/\alpha)\,ds = \alpha\int_{s=0}^{l}\Psi(s)\,ds$ is independent of β. It is then easy to see from Figure 3.5 that $\partial I_1^f/\partial\beta = \theta_{i_1^f} - \theta_{i_0^f}$. But from (3.6), we also have that $\partial I_1^f/\partial\beta = \widetilde{v}^f$. Therefore $\widetilde{v}^f = \theta_{i_1^f} - \theta_{i_0^f}$ and $\widetilde{v}^f \neq 0$ iff $\theta_{i_1^f} \neq \theta_{i_0^f}$. Combining this fact and Theorem 1, we get

Corollary 1 *If $\theta_{i_0^f} \neq \theta_{i_1^f}$, then R_*^f has no local minima in f.*

The previous discussion does not eliminate the possibility of a local minimum of R_*^f if $i_0^f = i_1^f$. Fortunately, we have the following lemma.

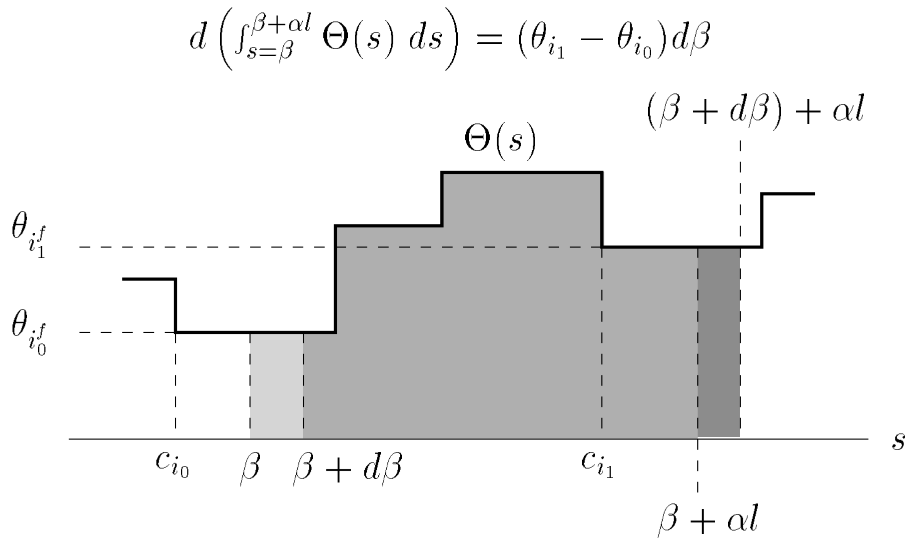


Figure 3.5: The Integral of Text Angle as a Function of Pattern Placement. Consider the integral $I(\beta) = \int_{s=\beta}^{\beta+\alpha l} \Theta(s)\, ds$, where α is fixed. Clearly, $I(\beta+d\beta) = I(\beta) - \int_{s=\beta}^{\beta+d\beta} \Theta(s)\, ds + \int_{s=\beta+\alpha l}^{\beta+d\beta+\alpha l} \Theta(s)\, ds$. If (α, β) is in the open face f, then we can choose $d\beta > 0$ small enough so that $(\alpha, \beta+d\beta)$ is still in f, $\Theta(\beta) = \Theta(\beta+d\beta) = \theta_{i_0^f}$, and $\Theta(\beta+\alpha l) = \Theta(\beta+d\beta+\alpha l) = \theta_{i_1^f}$. Here, we have $I(\beta + d\beta) - I(\beta) = (\theta_{i_1^f} - \theta_{i_0^f})d\beta$.

Lemma 1 *If $i_0^f = i_1^f$, then R_*^f has no local minima in f unless $R_*^f \equiv 0$.*

Proof. If $i_0^f = i_1^f$, then the text turning angle graph $\Theta(s) \equiv \theta_{i_0}$ over $[\beta, \beta + \alpha l]$ for every pair $(\alpha, \beta) \in f$. From equations (3.2) and (3.3), we can explicitly compute $I_1^f(\alpha, \beta)$ and $I_2^f(\alpha, \beta)$. The results are $I_1^f(\alpha, \beta) = \alpha l(\theta_{i_0^f} - \overline{\psi})$ and $I_2^f(\alpha, \beta) = \alpha l(\theta_{i_0^f}^2 - 2\theta_{i_0^f}\overline{\psi} + \overline{\psi^2})$, where $\overline{\psi} = (\int_{s=0}^{l} \Psi(s)\, ds)/l$ and $\overline{\psi^2} = (\int_{s=0}^{l} \Psi^2(s)\, ds)/l$. Substituting these results into equation (3.1) for R_*^f, we get $R_*^f(\alpha, \beta) = K/\alpha$, where K is constant with respect to α. If $K = 0$, then $R_*^f \equiv 0$. If $K \neq 0$, then $\partial R_*^f/\partial\alpha = -K/\alpha^2 \neq 0$, and hence again R_*^f has no local minima. ■

The following theorem combines Corollary 1 and Lemma 1.

Theorem 2 *If all text cumulative angles θ_i are distinct, then R_*^f has no local minima in f unless $R_*^f \equiv 0$.*

Proof. Consider the face f. If $i_0^f = i_1^f$, then the conclusion follows from Lemma 1. If $i_0^f \neq i_1^f$, then $\theta_{i_0^f} \neq \theta_{i_1^f}$ by assumption, and the conclusion follows from Corollary 1. ■

An almost immediate consequence of Theorem 2 is

Corollary 2 *If the text contains only right turns or only left turns (e.g. if the text is a piece of a convex polygon), then R_*^f has no local minima in f unless $R_*^f \equiv 0$.*

Proof. If the text only has right turns, then $\theta_0 < \theta_1 < \cdots < \theta_{m-1}$; if the text only has left turns, then $\theta_0 > \theta_1 > \cdots > \theta_{m-1}$. In either case, the cumulative turning angles θ_i are distinct, and the result follows from Theorem 2. ∎

By Theorem 1, if there is a local minimum in f, we must have $\widetilde{v}^f = \widehat{v}^f = 0$. In this case, R_*^f reduces to $R_*^f(\alpha,\beta) = (L/(\alpha^3 L^3))(A^f\alpha^2 + D^f\alpha + F^f)$, which is independent of β. In order to determine if there is a local minimum in f, we minimize this function with respect to α. Here $\partial R_*^f/\partial\alpha = (-L/(\alpha^4 l^3))(A^f\alpha^2 + 2D^f\alpha + 3F^f)$, so there are at most two values of α at which a local minimum can occur. Obviously, these values can be determined in constant time. Even if R_*^f has a local minimum in the right halfspace $H = \{\,(\alpha,\beta)\ :\ \alpha > 0\,\}$, it may not occur in the open face f. Although we have not been able to eliminate the possibility of a local minimum inside a face, we have the following theorem.

Theorem 3 *There are no strict local minima of R_* in the interior of an arrangement face.*

By a *strict* local minimum at $(\alpha_*,\beta_*) \in f$, we mean that $R_*^f(\alpha_*,\beta_*) < R_*^f(\alpha,\beta)$ for all $(\alpha,\beta) \neq (\alpha_*,\beta_*)$ in a small enough open ball centered at (α_*,β_*) and contained within f. Theorem 3 follows from the fact that a local minimum in f implies that R_*^f is independent of β (as argued above). If (α_*,β_*) is the location of such a local minimum, then $R_*^f(\alpha_*,\beta_*) = R_*^f(\alpha_*,\beta)\ \forall(\alpha_*,\beta) \in \bar{f}$. In particular, $R_*^f(\alpha_*,\beta_*) = R_*^f(\alpha_*,\beta)$ if $(\alpha_*,\beta) \in \partial f$. Therefore, a small value $R_*^f(\alpha_*,\beta_*)$ will also occur on the edges and/or vertices of f which intersect the vertical line $\alpha = \alpha_*$.

Let us summarize the results of this section. We have not able to eliminate the possibility of a local minimum inside an arrangement face. However, such a local minimum cannot occur unless $\widetilde{v}^f = \widehat{v}^f = 0$ (Theorem 1), a fairly restrictive condition which implies that R_*^f does not depend on the shift β. Even if there were a local minimum of R_* at $(\alpha_*,\beta_*) \in f$, $R_*(\alpha_*,\beta_*)$ will *not* be strictly smaller than R_* at other points in a neighborhood of (α_*,β_*) (Theorem 3). Furthermore, the value $R_*^f(\alpha_*,\beta_*)$ will also occur on some arrangement edge or vertex adjacent to f. Since local minima inside faces are somewhat rare, are never strict local minima, and the same value can always be found on arrangement edges or vertices, we ignore the possibility of face local minima in our algorithm.

3.3.2 Edges and Vertices

Now consider an edge e that bounds face f. We want to know whether R_* has a local minimum at some $(\alpha,\beta) \in e$. For simplicity of computation, we check instead whether R_* has a local minimum at some $(\alpha,\beta) \in e$ *along the direction of* e. This is a weaker condition than R_* having a local minimum at (α,β), and it is this weaker condition that defines an "edge minimum" in the algorithm presented in the next section. The edge e is part of a line $l_{ij} : a_j\alpha + \beta = c_i$. Combining this line equation with the equation (3.8) for R_*^f, we get a function R_*^e (R_* restricted to edge e) which is a rational cubic in α (the numerator is quadratic, but the denominator is cubic):

$$R_*^e(\alpha) = \frac{L}{\alpha^3 l^3}(\eta_{ij}^f \alpha^2 + \xi_{ij}^f \alpha + \rho_{ij}^f) \qquad \text{if } e \subseteq l_{ij},\ e \subseteq \partial f,$$

where

$$\begin{aligned} \eta_{ij}^f &= A^f - a_j B^f + a_j^2 C^f\ , \\ \xi_{ij}^f &= c_i B^f - 2a_j c_i C^f + D^f - a_j E^f\ , \text{ and} \\ \rho_{ij}^f &= c_i^2 C^f + c_i E^f + F^f\ . \end{aligned}$$

Thus we are left with a basic optimization problem. There are at most two local minima points $(\alpha_*, \beta_*) \in e$ for R_*^e since

$$\frac{dR_*^e}{d\alpha} = -\frac{L}{\alpha^4 l^3}(\eta_{ij}^f \alpha^2 + 2\xi_{ij}^f \alpha + 3\rho_{ij}^f).$$

These edge local minima locations can be determined in constant time given the formula for R_*^f.

The function R_* consists of piecewise rational cubic patches glued together at arrangement edges. Local minima of R_* may occur at arrangement vertices, so we must also examine the values of R_* at these locations. In fact, in practice we have found that values at vertices are smaller than minima on incident edges, even with the weaker notion of edge minimum given above. We shall say a bit more about this at the beginning of the results section 3.5.

3.4 The Algorithm

The user specifies a minimum match length $\text{matchlen}_{\text{min}}$ and a maximum match length $\text{matchlen}_{\text{max}}$ (default maximum is L), along with a bound on the maximum mean absolute error mae_{max} of a reported match. It is a bit easier to think in terms of the mean absolute error than in terms of the mean squared error since the former is in units of radians (or degrees), while the latter is in units of radians squared. If we let $f(s) = |\Theta(s) - (\Psi((s - \beta)/\alpha) + \gamma)|$, then the mean absolute error is $\text{mae}(\alpha, \beta, \gamma) = < f, I > /(\alpha l)$, where $< f, g > = \int_{s=\beta}^{\beta+\alpha l} f(s)g(s)\,ds$ and $I(s) \equiv 1$. The mean squared error is $\text{mse}(\alpha, \beta, \gamma) = ||f||^2/(\alpha l)$, where $||f||^2 = < f, f >$. Applying the Cauchy-Schwarz inequality gives $(< f, I >)^2 \leq ||f||^2 ||I||^2 =$

$||f||^2(\alpha l)$, from which it follows that $\mathrm{mae}^2(\alpha, \beta, \gamma) \leq \mathrm{mse}(\alpha, \beta, \gamma)$. Therefore, the bound $\mathrm{mae}_{\max}$ can be guaranteed as long as we require the reported match to have a mean squared error which is less than or equal to $\mathrm{mse}_{\max} = \mathrm{mae}^2_{\max}$.

We say that (α, β) is *admissible* if the match length $\alpha l \in [\mathrm{matchlen}_{\min}, \mathrm{matchlen}_{\max}]$ and the mean squared error $e_*(\alpha, \beta) \leq \mathrm{mse}_{\max}$. Note that the mean squared error $e_* = \frac{\alpha l}{L} R_* - 1$, and so we can quickly determine the mean squared error from the reciprocal score R_*. Of all admissible locations (α, β), we report only those which are locally the best. An admissible vertex is reported iff its reciprocal score is less than the reciprocal scores of all adjacent admissible vertices and of all admissible edge minima locations on adjacent edges. An admissible edge minimum is reported iff its reciprocal score is less than the reciprocal scores of all its admissible vertices (at most two) and the other admissible edge minimum (if one exists) on the same edge. Checking only a constant number of topologically neighboring elements does not guarantee that two very similar matches (which are geometrically close in $\alpha\beta$-space) will not be reported. Using topological closeness instead of geometric closeness is only a heuristic.

Our algorithm outputs matches during a topological sweep ([21]) over the $O(mn)$ lines l_{ij} in the degenerate arrangement $\mathcal{A}$. Our sweep implementation uses Edelsbrunner's "Simulation of Simplicity" technique ([20]) to cope with the degenerate input. During an elementary step, the topological sweep line moves from a face f_1 into a face f_2 through an elementary step vertex v. Please refer to Figure 3.6. During this step, we first compute the formula

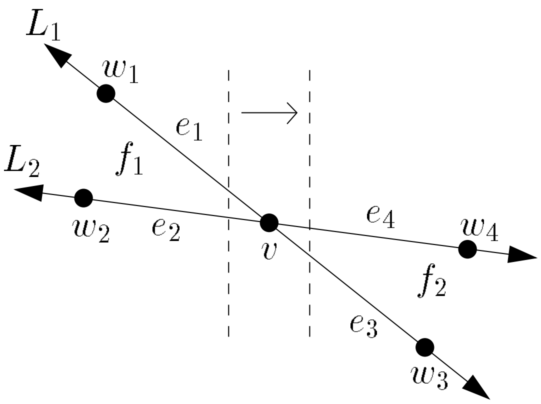


Figure 3.6: Elementary Step Notation.

for $R_*^{f_2}(\alpha, \beta)$. This requires computing the coefficients $\widetilde{u}^{f_2}$, $\widetilde{v}^{f_2}$, $\widetilde{w}^{f_2}$, $\widehat{u}^{f_2}$, $\widehat{v}^{f_2}$, and $\widehat{w}^{f_2}$ in the formulae (3.6) and (3.7) for the integrals $I_1(\alpha, \beta)$ and $I_2(\alpha, \beta)$, $(\alpha, \beta) \in \bar{f}_2$. Note that computing and summing the $O(mn)$ terms in (3.4) and (3.5) during each elementary step would require total time $O(m^3n^3)$ because there are $O(m^2n^2)$ elementary steps. Fortunately, only a constant number of terms in (3.4) and (3.5) change when we move from face f_1 to face f_2. This is because only a constant number (at most eight) of intersection formulae X_{ij}^f are affected by above–below relationships involving L_1 and L_2. Hence, the values $\widetilde{u}^{f_2}, \widetilde{v}^{f_2}, \widetilde{w}^{f_2}, \widehat{u}^{f_2}, \widehat{v}^{f_2}, \widehat{w}^{f_2}$ in (3.6) and (3.7) can be computed in constant time from the values $\widetilde{u}^{f_1}, \widetilde{v}^{f_1}, \widetilde{w}^{f_1}, \widehat{u}^{f_1}, \widehat{v}^{f_1}, \widehat{w}^{f_1}$, and the formula for $R_*^{f_2}(\alpha, \beta)$ can be computed in $O(1)$ time from the formula for $R_*^{f_1}(\alpha, \beta)$. The latter formula was computed when the sweep line first entered face f_1.

The remaining work during an elementary step over v is straightforward. In Figure 3.6, e_3 and e_4 are the arrangement edges with v as left endpoint. These edges are part of the lines L_1 and L_2, respectively, and have right endpoints w_3 and w_4, respectively. From the formula for $R_*^{f_2}(\alpha, \beta)$, we compute the formulae for $R_*^{L_1}(\alpha, \beta)$ and $R_*^{L_2}(\alpha, \beta)$, as well as the values $R_*(v), R_*(w_3), R_*(w_4)$. From the formulae for $R_*^{L_1}(\alpha, \beta)$ and $R_*^{L_2}(\alpha, \beta)$, we compute the locations on e_3 and e_4 of any local minima of R_* in the direction of these edges. The above computations take constant time given the formula for $R_*^{f_2}(\alpha, \beta)$. During the elementary step at v, we decide whether to output the vertex v and any local edge minima locations on e_3 and e_4. During previous elementary steps, it was determined if it is still possible to report v by comparing the value of R_* at v to the value of R_* at w_1, w_2, and at all edge minima locations on e_1 and e_2. The remaining values of R_* needed to decide whether or not to report v are computed during the elementary step over v, as described above. During this step, we also compute all values of R_* needed to decide whether to report any local edge minima locations on e_3 and e_4. For each location (α, β) to be reported, we compute $\gamma = \gamma_*(\alpha, \beta) = I_1(\alpha, \beta)/\alpha l$ and report the triple (α, β, γ).

The above discussion shows that the elementary step work specific to our setting may be performed in $O(1)$ time. Thus, the total time to perform the topological sweep over the $O(mn)$ lines l_{ij} is $O(m^2n^2)$. The total space required by our algorithm is the $O(mn)$ storage required by a generic topological line sweep which does not store the discovered arrangement. In the common case of a simple pattern, such as a line segment or corner, $m = O(1)$ and our algorithm requires $O(n^2)$ time and $O(n)$ space.

The α, γ components of a reported triple (α, β, γ) give the scaling and rotation components of a similarity transformation of the pattern which makes it look like a piece of the text. The β component tells us where along the text this similar piece is located. To get the translation parameters of the similarity transformation, we sample the transformed pattern and the corresponding similar piece of the text, and find the translation parameters which minimize the mean squared error between the translated pattern point set and the text point set.

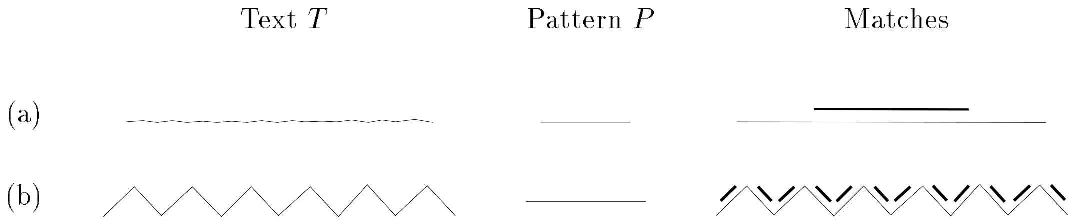


Figure 3.7: Trading Off Match Length Against Match Error. In both examples, the pattern is a line segment and the maximum absolute error input parameter $\text{mae}_{\max} = 9°$. To help make individual matches clear, we show a darker, smaller scale version of the pattern slightly offset from each match. (a) One match over the length of the entire "noisy" straight line text is found. (b) Twelve matches, one for each of the sides of the "mountain range", are found.

3.5 Results

In practice, local edge minima are very rarely reported because there is almost always a smaller admissible minimum at one of the two edge vertices. Essentially, the algorithm reports admissible vertices which have a reciprocal score which is lower than any other adjacent admissible vertex. Recall that arrangement vertices (α, β) give scalings and shifts of the pattern which cause two of its arclength breakpoints to line up with two of the text arclength breakpoints. Our experimentation has thus showed that the best matches of arclength versus turning angle graphs are usually those that line up two pairs of breakpoints (as opposed to one pair for points on arrangement edges and zero pairs for points in arrangement faces).

Figure 3.7 shows two experiments that illustrate the balancing of match error versus match length. In both cases, the pattern shape is a line segment. In the first case, the text is a "noisy" straight line. The angles between consecutive segments are all nearly 180°. With $\text{mae}_{\max} = 9°$, our algorithm finds one match over the length of the entire piece of text. In the second case, the text looks more like a mountain range. The angles between consecutive segments are far from flat angles. With $\text{mae}_{\max} = 9°$, our algorithm finds twelve matches, one for each of the sides of the mountain range.

A successful PSSP algorithm must produce a complete set of independent matches for given user tolerances. Figures 3.8 shows the result of applying our PSSP algorithm in two "exact" situations. In the first example, the pattern is simply a rotated version of the text. In the second example, the pattern is a rotated, scaled-down version of a portion of the text. For both inputs, only the one correct match is reported. Figure 3.9 shows the output of our

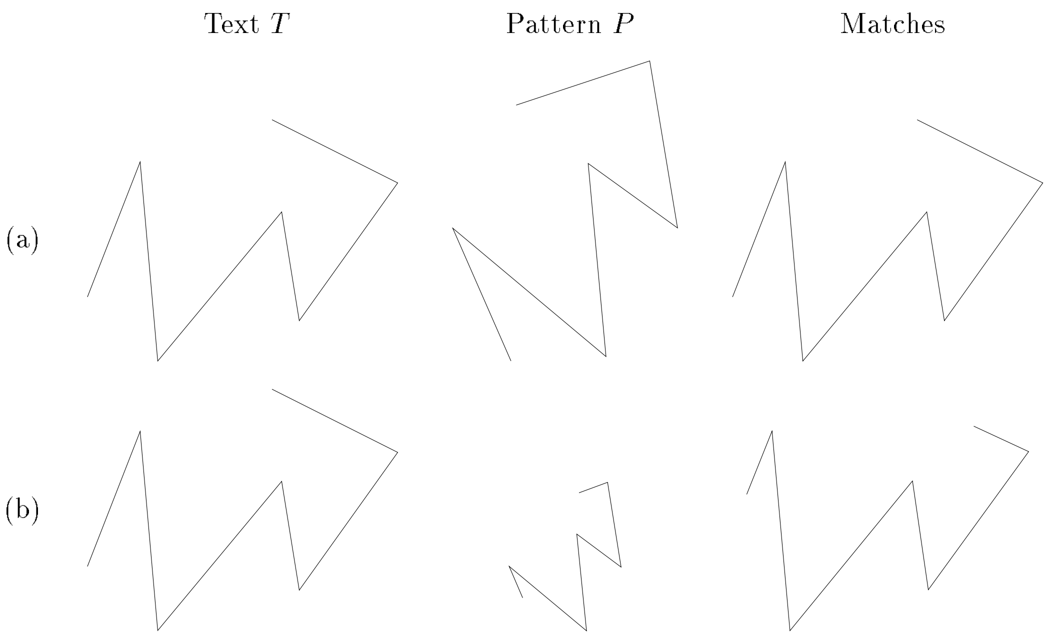


Figure 3.8: PSSP Exact Matching Examples. The columns (from left to right) show the text, pattern, and matches found. With $\mathrm{mae}_{\max} = 9^\circ$, our algorithm finds only the one exact match in both examples. (a) The pattern is a rotated version of the text. (b) The pattern is a rotated, scaled version of a piece of the text.

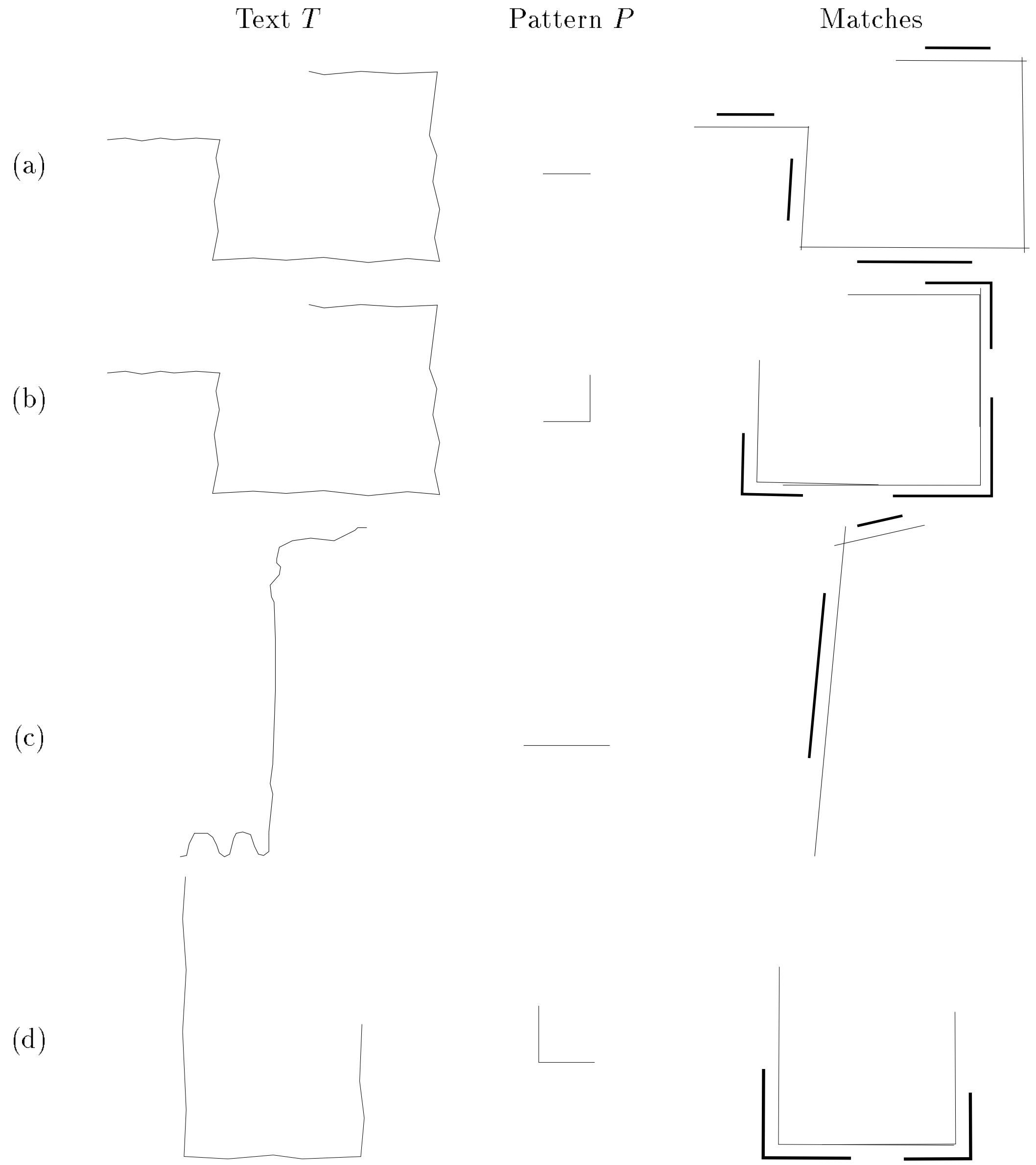


Figure 3.9: PSSP Results. As in Figure 3.7, each match is accompanied by a darker, smaller scale version of the pattern which is slightly offset from the match. (a) $\text{mae}_{\text{max}} = 15°$. Each of the five noisy lines gives rise to exactly one segment match. (b) $\text{mae}_{\text{max}} = 9°$. The three left turn text corners are found with the left turn corner pattern. (c) $\text{mae}_{\text{max}} = 20°$. Our algorithm finds the two (relatively) long, straight portions of the text. (d) $\text{mae}_{\text{max}} = 9°$. Both left turn text corners are found.

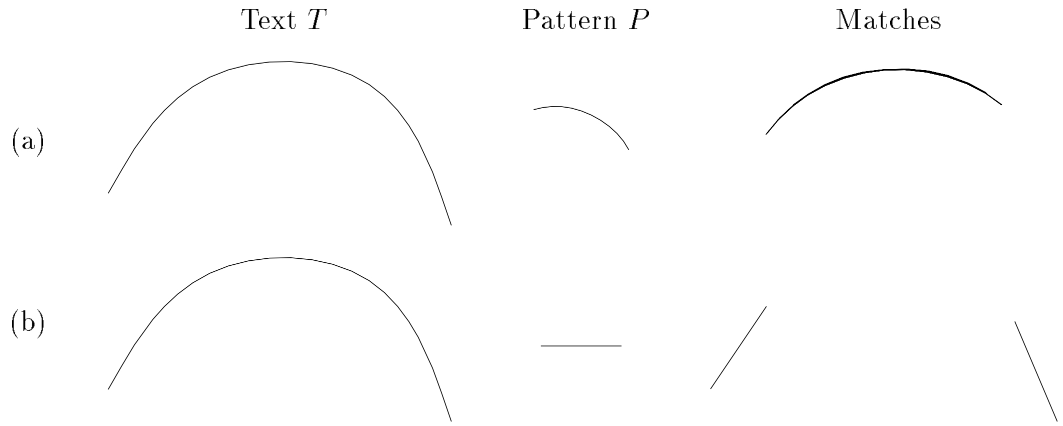


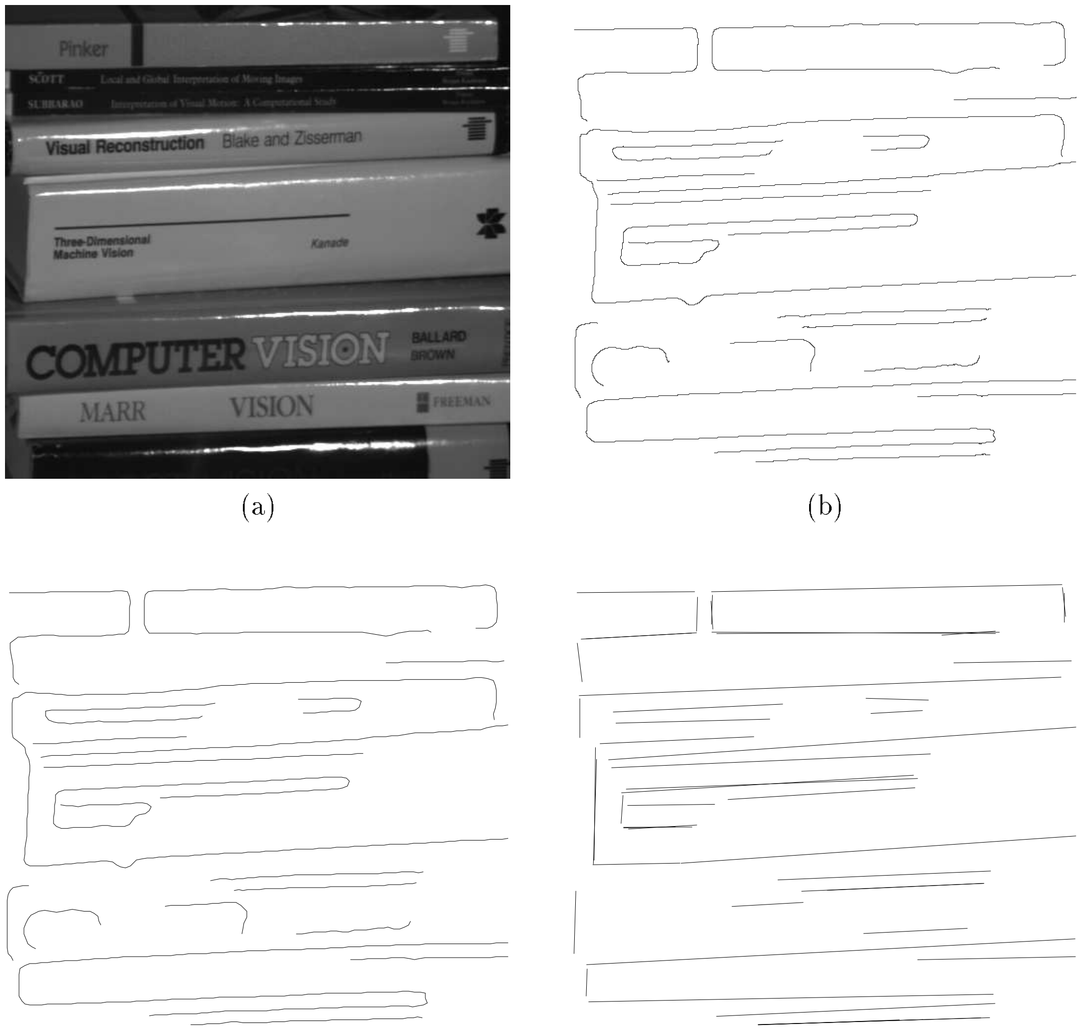
Figure 3.10: More PSSP Results. In both examples, the text is a curve with straight line and circular portions. (a) The circular arc pattern only matches the circular part of the text. (b) The line segment pattern only matches the straight parts of the text.

algorithm on four (text, pattern) pairs in which the pattern is either a straight segment or a corner. In Figure 3.9(b) we clearly see that the order of the vertices in the pattern and text makes a difference in the matches found by our algorithm — the three left turn text corners are found, but the right turn text corner is missed. To get around this dependence on the representation of the polyline, we could run our algorithm again on the pattern represented by vertices in the reverse order (so that the pattern is a right turn instead of a left turn). In Figure 3.10, we fit a circular arc and straight line pattern into a text curve with both circular and straight portions. The circular arc does not match the straight portions of the text, nor does the line segment match the circular portion of the text. In the examples shown in Figures 3.11 and 3.12, we use our method to summarize the straight line content of an image.

Obviously, if the pattern shape is not present anywhere within the text (within user specified tolerances) then a PSSP algorithm should not report any matches. Given the text and pattern shown in Figure 3.13 our PSSP algorithm reports no matches for $\mathrm{mae}_{\mathrm{max}} = 9°$ and $\mathrm{matchlen}_{\mathrm{min}}$ set to prevent very tiny matches.

3.6 Summary and Suggestions for Future Work

In this chapter we developed an algorithm to find where a planar "pattern" polyline fits well into a planar "text" polyline. By allowing the pattern to rotate and scale, we find



(a) (b)

(c) (d)

Figure 3.11: Image Summary by Straight Segments. (a) The image to be summarized is 512×480. (b) The result of Canny edge detection ($\sigma = 6$ pixels) and edgel linking is a set of polylines with a total of 7219 vertices. (c) The result of subsampling each of the polylines by a factor of 6 leaves a total of 1212 vertices. (d) Finally, fitting a straight segment to each of the subsampled polylines using our PSSP algorithm gives a set of 50 segments. As mentioned in the text, checking only a constant number of topologically neighboring elements before reporting a match (α, β) does not guarantee that two very similar matches (which are geometrically close) will not be reported. This heuristic is responsible for the "double edges" in the image summary.

(a) (b) (c)

Figure 3.12: Another Image Summary by Straight Segments. (a) The "tools" image. (b) The result of Canny edge detection and edgel linking of the pliers' contour. (c) The result of fitting a straight segment into the pliers' contour using our PSSP algorithm.

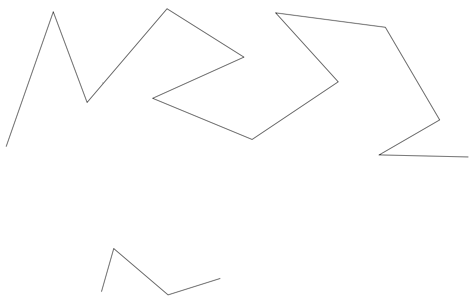


Figure 3.13: PSSP No Matches Example. The text is shown above the pattern. Our PSSP algorithm reports no matches for $mae_{max} = 9°$ and $matchlen_{min}$ set to prevent very tiny matches.

portions of the text which are similar in shape to the pattern. We dealt with the issue of match length versus match error by using a match scoring function that balances these two factors. All comparisons were performed on the arclength versus cumulative turning angle representations of the polylines. This allowed us to reduce the complexity of the problem: To compare two planar polylines, we compare their one-dimensional arclength versus turning angle graphs. In addition, the space operations of scaling, rotation, and sliding the pattern along the text cause simple changes to the pattern summary graph. Another reduction in complexity was gained by using the L_2^2 distance to compare the graphs. This allowed us to eliminate the rotation parameter from the search space, leaving a 2D scale-position space. Thus, we converted a four-dimensional search problem in the space of similarity transformations to a two-dimensional search problem in scale-position space.

Our line sweep strategy essentially examines all possible pattern scales and positions within the text. If, however, the pattern does not fit well at a certain scale and location, then it will not fit well at nearby scales and locations. Finding "certificates of dissimilarity" which would allow us to prune our (α, β) search space is a topic for future research. Another idea for speeding up our algorithm is to use the results of pattern searches in coarse versions of the text (with relatively high error tolerances) to guide searches in finer versions of the text, with the final search in the text itself. This hierarchical search strategy aims to reduce the total search time by reducing the number of expensive comparisons between the pattern and the full text at the expense of many cheaper comparisons with coarser versions of the text. The two techniques discussed above will help speed up the search for one pattern within one text. Suppose, however, that we have multiple patterns that we want to fit into multiple texts. This is the case, for example, if we are trying to summarize an image by recording which basic shapes/patterns fit where in the image. Can we do better than the brute force approach of applying our algorithm to each (text, pattern) pair?

The problem considered in this chapter can be generalized in a few different ways. For example, we may want to find affinely transformed versions of a pattern within a text. In addition to the shape transformation group, we could expand the class of shapes to include planar shapes other than polylines, such as circular arcs or cubic splines. Also, we may want to remove the restriction that the one-dimensional shapes are planar and allow, for example, polylines in three dimensions. To complete the generalization of the problem, we could allow the two-dimensional shapes such as polyhedra or spheres. Another possible topic of further research is to use the ideas in this chapter to develop a metric on polyline shapes which returns a small distance between two polylines when a scaled, rotated version of one matches well a portion of the other. See [3][34][54][2] for work on shape metrics.

One strategy for handling other one-dimensional planar pattern and text shapes is to apply our method on polyline approximations of the input shapes. An accurate polyline approximation, however, may require a very large number of segments and our algorithm uses $O(m^2n^2)$ time. In choosing the arclength versus turning angle representation of a shape, we restricted ourselves to one-dimensional planar shapes. In addition, our algorithm relies heavily on the simple changes to this representation when the underlying polyline is uniformly scaled and rotated. This no longer holds if we allow, for example, affine transformations of the pattern.

Although our algorithm does not generalize to handle more general shape search problems, it produces excellent results for planar polylines under similarity transformations. This very specific problem is far from trivial because we match possibly scaled versions of the pattern to pieces of the text, and we require only the "best" matches to be reported. The main idea of the algorithm is to divide up the search plane into regions in which it is (relatively) easy to handle a complex scoring function. This general idea is obviously applicable to other search problems.

Chapter 4

The Earth Mover's Distance (EMD)

A very general distance measure with applications in content-based image retrieval is the Earth Mover's Distance (EMD) between distributions ([68]). The EMD has been successfully used in a common framework for measuring image similarity with respect to color ([69, 67, 65, 68]) and texture ([69, 68, 66]). In this framework, the summary or *signature* of an image is a finite collection of weighted points. For example, in [69] the color signature of an image is a collection of dominant image colors in the CIE-Lab color space ([88]), where each color is weighted by the fraction of image pixels classified as that color. In [69], the texture signature of a single texture image is a collection of spatial frequencies in log-polar coordinates, where each frequency is weighted by the amount of energy present at that frequency. To complete the uniform framework, a distance measure on weight distributions is needed to measure similarity between image signatures.

The *Earth Mover's Distance* (EMD) between two distributions is proportional to the minimum amount of *work* required to change one distribution into the other. Here one unit of work is defined as the amount of work necessary to move one unit of weight by one unit of distance. The distance measure between weight locations is known as the *ground distance*. The morphing process between equal-weight distributions can be visualized as weight flowing from one distribution to the other until the distributions are identical. Figures 4.1(a)-(c) and 4.2(a)-(c) illustrate the minimum work morphing for three different pairs of equal-weight distributions.

In chapter 2, we used *mass* instead of *weight* in our EMD description because the EMD optimization problem was originally called the *mass transfer problem*. We consider the two terms interchangeable, although our notation given in the next section corresponds better

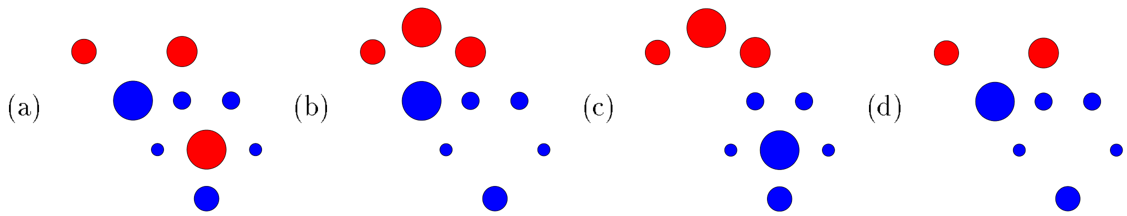


Figure 4.1: Example Distributions in 2D. Each of the examples (a), (b), (c), and (d) shows two distributions, one whose points are centered at the red discs and one whose points are centered at the blue discs. The area of a disc is equal to the weight at its center point in the distribution. The pairs (a), (b), and (c) are equal-weight pairs. In (d), the red distribution is lighter than the blue distribution. This example is the same as (b) with one of the red weights removed.

with *weight*, and in physics the units of weight $\times$ distance are the same as the units for work. On the other hand, *mass* corresponds better with the term *Earth Mover's Distance*. This name was suggested by Jorge Stolfi ([76]) who got the idea from some CAD programs for road design which have a function that computes the optimal earth displacement from roadcuts to roadfills.

An important property of the EMD is that it allows *partial matching*. When the total weights of the distributions are unequal, the EMD requires all the weight in the lighter distribution to be matched to weight in the heavier distribution. Some weight in the heavier distribution, however, will not be matched to weight in the lighter distribution. The matching process between unequal-weight distributions can be visualized as a flow in two different ways: (i) weight flows from the lighter distribution to the heavier distribution until the lighter distribution becomes a sub-distribution of the heavier one, or (ii) weight flows from the heavier distribution to the lighter distribution until all the weight in the lighter distribution has been covered. A type (i) flow visualization for the unequal-weight distributions in Figure 4.1(d) is shown in Figure 4.2(d).

The EMD matching process can also be visualized as filling holes with piles of dirt. The holes are located at the points in the lighter distribution, and the dirt piles are located at the points in the heavier distribution. The volume of a hole or dirt pile is given by the weight value of its position. In the equal-weight case, either distribution can be used to define the dirt piles or the holes, and all the dirt is needed to fill the holes. In the unequal-weight case, there is dirt leftover once all the holes are filled.

In the next section, we give the formal definition of the Earth Mover's Distance and discuss some of its properties. The work minimization problem which defines the EMD is a

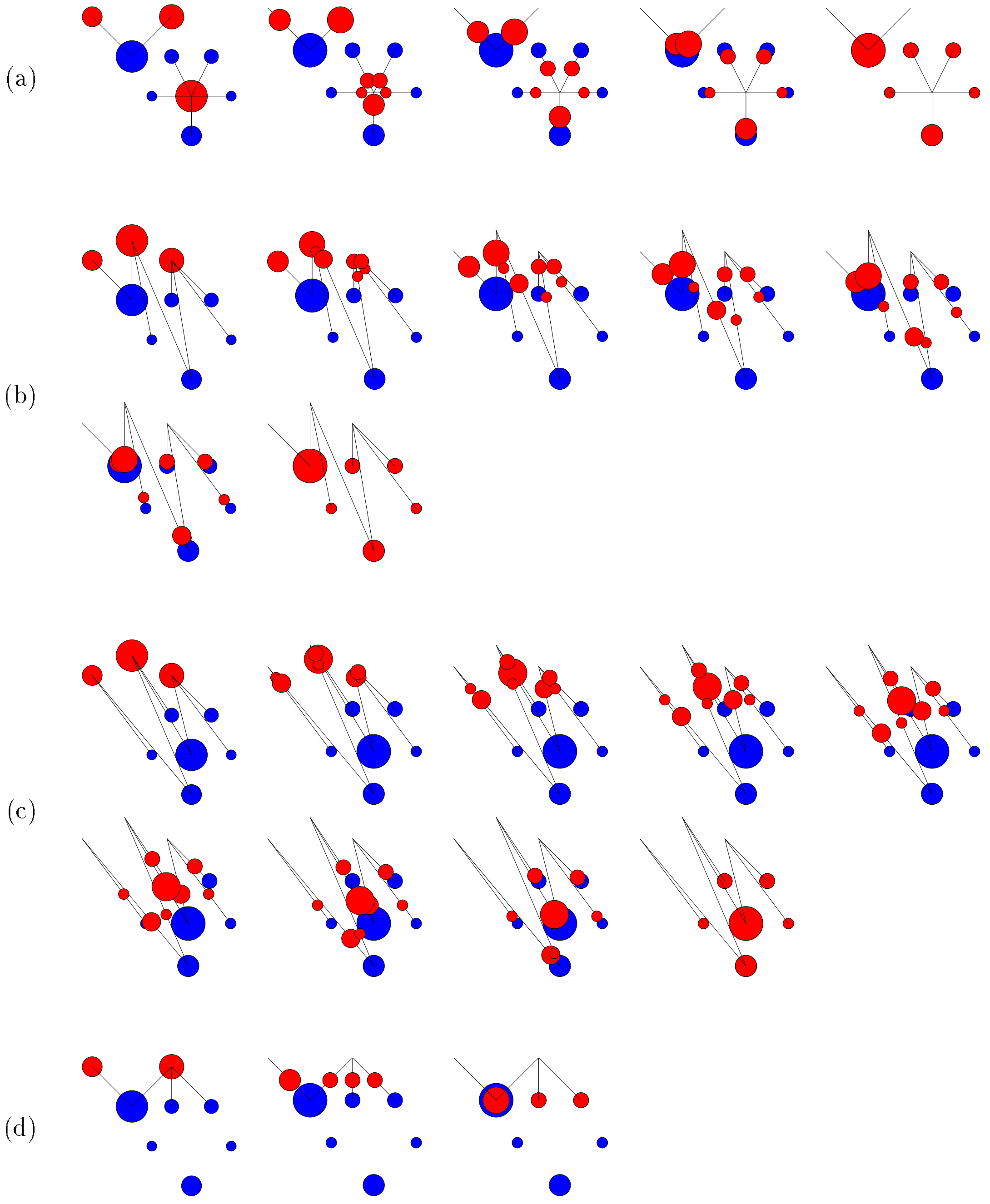


Figure 4.2: The EMD Morphing Process. Here we show the least work morphing for the equal-weight examples (a), (b), (c), and the unequal-weight example (d) in Figure 4.1. Weight flows from the red distributions to matching weight in the blue distributions. The amount of work done between frames is the same for every pair of adjacent frames shown (except possibly between the last two frames in each sequence). The EMD is smaller between pair (a) than pair (b), and smaller between pair (b) than pair (c). In (d), some of the blue weight is not matched to any red weight.

type of linear program known as the transportation problem. We discuss the transportation problem and its connection to the EMD in section 4.2. In section 4.3, we consider some special cases of matching (i) distributions which define ordinary point sets (section 4.3.1), and (ii) equal-weight distributions on the real line (section 4.3.2). In section 4.4, we give a couple of modifications to the EMD which make it more amenable to partial matching. In section 4.4.1, we present the partial EMD which forces only some fraction of the weight of the lighter distribution to be matched. In section 4.4.2, we discuss the τ-EMD which measures the amount of weight that *cannot* be matched if we only allow weight to flow over ground distances that do not exceed τ. Finally, in section 4.5 we use the EMD to estimate the size at which a color pattern may appear within an image. Please refer back to section 2.3 for a comparison of the EMD and bin-to-bin histogram distance measures.

4.1 Basic Definitions and Notation

We denote a discrete, finite *distribution* $\mathbf{x}$ as

$$\mathbf{x} = \{ \ (x_1, w_1), \ldots, (x_m, w_m) \ \} \equiv (X, w) \in \mathbf{D}^{K,m}$$

where $X = [\ x_1 \ \cdots \ x_m \ ] \in \mathbf{R}^{K \times m}$ and $w_i \geq 0$, for all $i = 1, \ldots, m$. Here K is the dimension of ambient space of the points $x_i \in \mathbf{R}^K$, and m is the number of points. The (total) *weight* of the distribution $\mathbf{x}$ is $w_\Sigma = \sum_{i=1}^m w_i$. Given two distributions $\mathbf{x} = (X, w) \in \mathbf{D}^{K,m}$ and $\mathbf{y} = (Y, u) \in \mathbf{D}^{K,n}$, a *flow* between $\mathbf{x}$ and $\mathbf{y}$ is any matrix $F = (f_{ij}) \in \mathbf{R}^{m \times n}$. Intuitively, f_{ij} represents the amount of weight at x_i which is matched to weight at y_j. The term *flow* is meant to evoke the image of weight flowing from the points in the heavier distribution to the points in the lighter distribution until all the weight in the lighter distribution has been covered. If one distribution is known to be heavier than the other, then we shall write that a flow is *from* the heavier distribution *to* the lighter distribution. The flow F is a *feasible flow* between $\mathbf{x}$ and $\mathbf{y}$ iff

$$\begin{aligned}
f_{ij} &\geq 0 && i = 1, \ldots, m, \ j = 1, \ldots, n, && (4.1)\\
\sum_{j=1}^{n} f_{ij} &\leq w_i && i = 1, \ldots, m, && (4.2)\\
\sum_{i=1}^{m} f_{ij} &\leq u_j && j = 1, \ldots, n, \quad \text{and} && (4.3)\\
\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} &= \min(w_\Sigma, u_\Sigma). && && (4.4)
\end{aligned}$$

Constraint (4.1) requires the amount of x_i matched to y_j to be nonnegative. Constraint (4.2) ensures that the weight in $\mathbf{y}$ matched to x_i does not exceed w_i. Similarly, (4.3) ensures that the weight in $\mathbf{x}$ matched to y_j does not exceed u_j. Finally, constraint (4.4) forces the total amount of weight matched to be equal to the weight of the lighter distribution.

Let $\mathcal{F}(\mathbf{x}, \mathbf{y})$ denote the set of all feasible flows between $\mathbf{x}$ and $\mathbf{y}$. The work done by a feasible flow $F \in \mathcal{F}(\mathbf{x}, \mathbf{y})$ in matching $\mathbf{x}$ and $\mathbf{y}$ is given by

$$\mathrm{WORK}(F, \mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij},$$

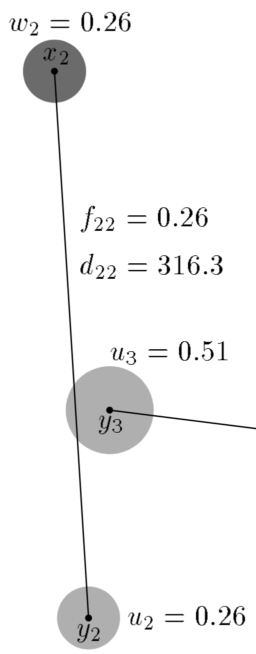
where $d_{ij} = d(x_i, y_j)$ is the distance between x_i and y_j. An example ground distance is the Euclidean distance $d(x_i, y_j) = ||x_i - y_j||_2$. The *Earth Mover's Distance* $\mathrm{EMD}(\mathbf{x}, \mathbf{y})$ between $\mathbf{x}$ and $\mathbf{y}$ is the minimum amount of work to match $\mathbf{x}$ and $\mathbf{y}$, normalized by the weight of the lighter distribution:

$$\mathrm{EMD}(\mathbf{x}, \mathbf{y}) = \frac{\min_{F=(f_{ij}) \in \mathcal{F}(\mathbf{x}, \mathbf{y})} \mathrm{WORK}(F, \mathbf{x}, \mathbf{y})}{\min(w_\Sigma, u_\Sigma)}. \tag{4.5}$$

In the next section, we connect the work minimization problem in the numerator of (4.5) to a special type of linear program called the transportation problem ([32]). The normalization by the minimum weight makes the EMD equal to the average distance travelled by weight during an optimal (i.e. work minimizing) flow, and ensures that the EMD does not change if all the weights in both distributions are scaled by the same factor. Examples of feasible non-optimal and optimal flows between equal-weight distributions are shown in Figure 4.3, and between unequal-weight distributions are shown in Figure 4.4. In the unequal-weight case, some of the weight in the heavier distribution is unmatched by a feasible flow (more precisely, $w_\Sigma - u_\Sigma$ $\mathbf{x}$-weight is unmatched if $\mathbf{x}$ is heavier than $\mathbf{y}$).

The EMD is a metric when the total weights of the distributions are equal and the ground distance between weights is a metric ([68]). The only difficult part of the proof is showing the triangle inquality $\mathrm{EMD}(\mathbf{x}, \mathbf{z}) \leq \mathrm{EMD}(\mathbf{x}, \mathbf{y}) + \mathrm{EMD}(\mathbf{y}, \mathbf{z})$. One way to morph $\mathbf{x}$ into $\mathbf{z}$ is to morph $\mathbf{x}$ into $\mathbf{y}$ and then $\mathbf{y}$ into $\mathbf{z}$. If dirt travels from x_i to y_j to z_k, then the metric assumption for the ground distance yields $d(x_i, z_k) \leq d(x_i, y_j) + d(y_j, z_k)$; i.e., it is cheaper just to transport the dirt directly from x_i to z_k. If we use an optimal matching $F^* = (f^*_{ij})$ to change $\mathbf{x}$ into $\mathbf{y}$ and an optimal matching $G^* = (g^*_{jk})$ to change $\mathbf{y}$ into $\mathbf{z}$, then the composite morphing $H = (h_{ik})$ to change $\mathbf{x}$ into $\mathbf{z}$ cannot cost more than $\mathrm{EMD}(\mathbf{x}, \mathbf{y}) + \mathrm{EMD}(\mathbf{y}, \mathbf{z})$. The EMD triangle inequality then follows from the fact that $\mathrm{EMD}(\mathbf{x}, \mathbf{z})$ is the minimum cost of any morphing from $\mathbf{x}$ to $\mathbf{z}$. The composite flow H is

(a)



(b)

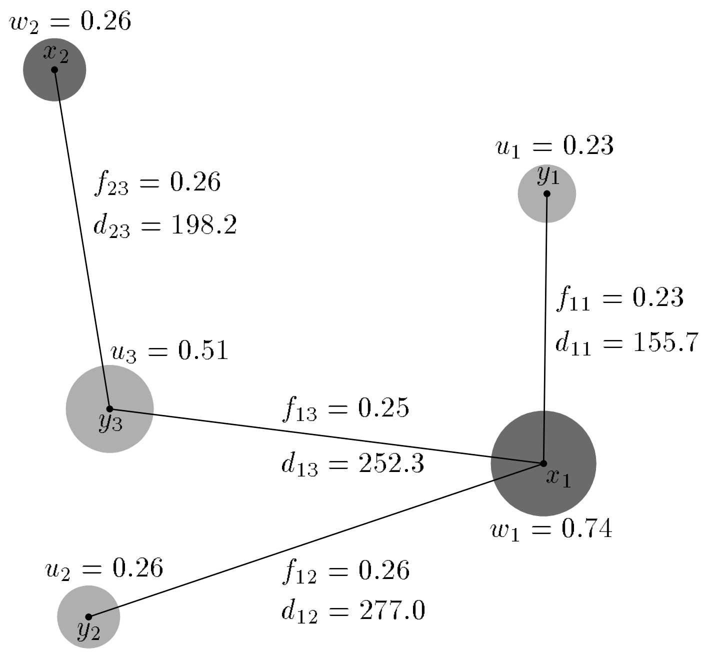


Figure 4.3: A Non-Optimal and an Optimal Flow between Equal-Weight Distributions. The area of the disc around a weight location is equal to the amount of weight at that location. (a) The amount of work done to match $\mathbf{x}$ and $\mathbf{y}$ by this feasible flow is $0.23 \times 155.7 + 0.51 \times 252.3 + 0.26 \times 316.3 = 246.7$. This flow is not optimal. (b) This flow is a work minimizing flow. The total amount of work done is $0.23\times155.7+0.26\times277.0+0.25\times252.3+0.26\times198.2 = 222.4$. Since the total weight of both $\mathbf{x}$ and $\mathbf{y}$ is one, the EMD is equal to the minimum amount of work: $\mathrm{EMD}(\mathbf{x},\mathbf{y}) = 222.4$.

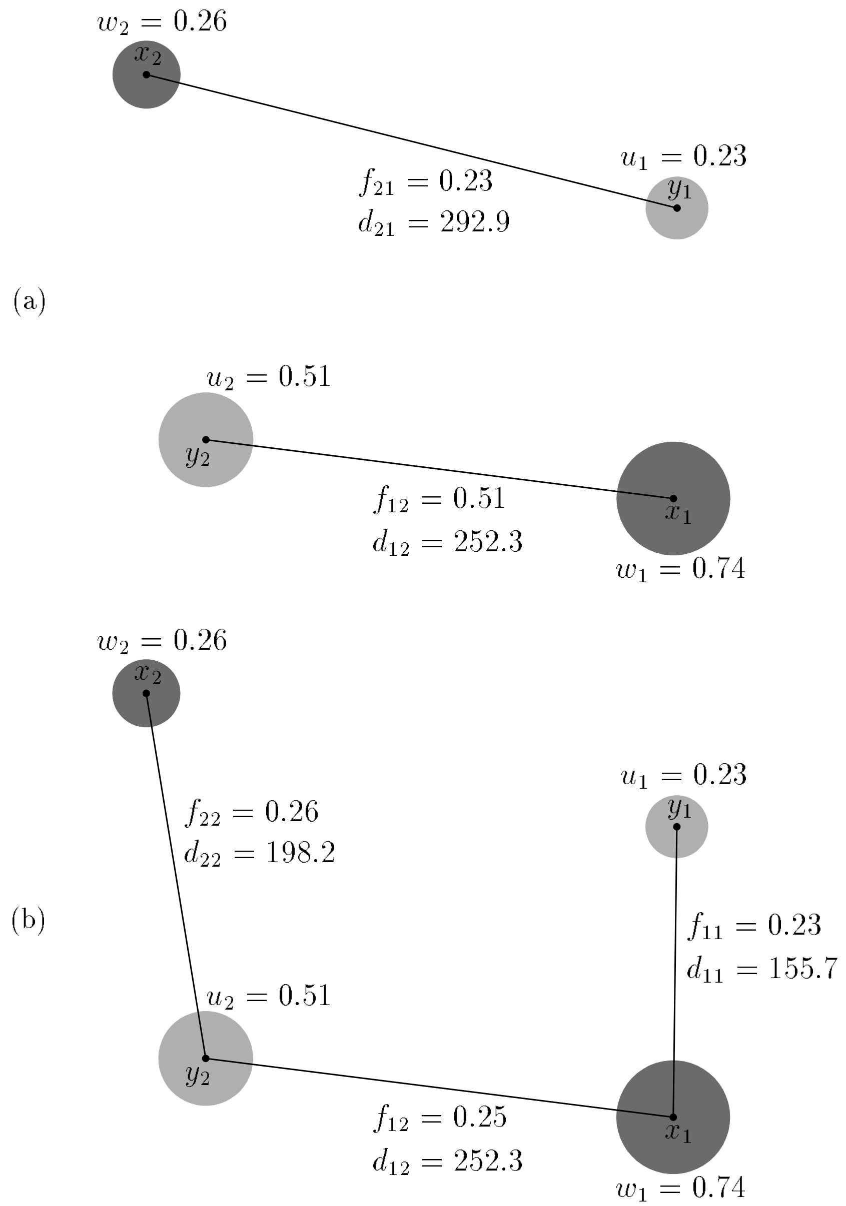


Figure 4.4: A Non-Optimal and an Optimal Flow between Unequal-Weight Distributions. Here $\mathbf{x}$ is heavier than $\mathbf{y}$. (a) The amount of work done to match $\mathbf{x}$ and $\mathbf{y}$ by this feasible flow is $0.51 \times 252.3 + 0.23 \times 292.9 = 196.0$. For this flow, 0.23 of the weight at x_1 and 0.03 of the weight at x_2 are not used in the matching. This flow is not optimal. (b) This flow is a work minimizing flow. The total amount of work for this flow to cover $\mathbf{y}$ is $0.23 \times 155.7 + 0.25 \times 252.3 + 0.26 \times 198.2 = 150.4$. For this flow, 0.26 of the weight at x_1 is not used in the matching. Since the total weight of the lighter distribution is 0.74, $\mathrm{EMD}(\mathbf{x}, \mathbf{y}) = 150.4/0.74 = 203.3$.

derived from the flows F^* and G^* as the sum of interval intersections

$$h_{ik} = \sum_{j=1}^{n} \left| \left[\sum_{\widehat{\imath}=1}^{i-1} f^*_{\widehat{\imath}j}, \sum_{\widehat{\imath}=1}^{i} f^*_{\widehat{\imath}j} \right] \bigcap \left[\sum_{\widehat{k}=1}^{k-1} g^*_{j\widehat{k}}, \sum_{\widehat{k}=1}^{k} g^*_{j\widehat{k}} \right] \right|.$$

See [68] for the intuition for this formula. Since the L_p $(p \geq 1)$ distance functions are metrics, the EMD is a metric between equal-weight distributions whenever the ground distance is an L_p distance.

Another commonly used distance function is $d = L_2^2$, the square of the ordinary L_2 distance. The L_2^2 distance function does not obey the triangle inequality, but it is a weak metric between points since

$$||p - q||_2^2 \leq 2(||p - r||_2^2 + ||q - r||_2^2) \qquad \forall p, q, r.$$

Thus, the morphing H from $\mathbf{x}$ to $\mathbf{z}$ costs no more than $2(\text{EMD}(\mathbf{x}, \mathbf{y}) + \text{EMD}(\mathbf{y}, \mathbf{z}))$ when the gorund distance is L_2^2. It follows that

$$\text{EMD}^{L_2^2}(\mathbf{x}, \mathbf{z}) \leq 2(\text{EMD}^{L_2^2}(\mathbf{x}, \mathbf{y}) + \text{EMD}^{L_2^2}(\mathbf{y}, \mathbf{z})). \tag{4.6}$$

Thus, the EMD is a weak metric between equal-weight distributions when $d = L_2^2$.

When the distributions are not necessarily equal-weight, the EMD is no longer a metric. If $\mathbf{x}$ is lighter than $\mathbf{y}$, then a feasible flow matches all the weight in $\mathbf{x}$ to part of the weight in $\mathbf{y}$. If $\mathbf{x}$ and $\mathbf{z}$ are both lighter than $\mathbf{y}$, then it can happen that $\text{EMD}(\mathbf{x}, \mathbf{y})$ and $\text{EMD}(\mathbf{y}, \mathbf{z})$ are small, but $\text{EMD}(\mathbf{x}, \mathbf{z})$ is large. This is because $\mathbf{x}$ and $\mathbf{z}$ might match well two parts of $\mathbf{y}$ that have little or no weight in common. There is no reason that two such parts of $\mathbf{y}$ must be similar under the EMD.

In the examples and discussion given thus far, the EMD measures the distance between two collections of weighted points based on a ground distance between points. This does not, however, expose the full generality of the EMD. The coordinates of distribution points are not used directly in the EMD formulation; only the ground distances between points are needed. Therefore, there is no need to work in a point feature space; the only requirement is that ground distances between features can be computed. In general, the EMD is a distance measure between two sets of weighted objects which is built upon a distance between individual objects. In this thesis, however, we focus mainly on the case of distributions of weight in some point feature space.

4.2 Connection to the Transportation Problem

The transportation problem (TP) is a special type of linear program (LP) which seeks to find the minimum cost way to transport goods from a set of sources or suppliers $i = 1, \ldots, m$ to a set of destinations or demanders $j = 1, \ldots, n$. Supplier i has a supply of s_i units, and demander j has a demand of d_j units. The cost per unit transported from supplier i to demander j is denoted by c_{ij}, and the number of units transported is denoted by x_{ij}. Assuming that the total supply $s_\Sigma = \sum_{i=1}^m s_i$ is equal to the total demand $d_\Sigma = \sum_{j=1}^n d_j$, the transportation problem is to compute

$$\min_{(x_{ij})} \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij}$$

subject to

$$\begin{array}{rcll} x_{ij} & \geq & 0 & i = 1, \ldots, m,\ j = 1, \ldots, n, \\ \displaystyle\sum_{j=1}^n x_{ij} & = & s_i & i = 1, \ldots, m, \\ \displaystyle\sum_{i=1}^m x_{ij} & = & d_j & j = 1, \ldots, n. \end{array}$$

If the total supply and demand are not equal, then it is impossible to satisfy the given constraints. The equality constraints can be written as

$$\begin{bmatrix} 1 & 1 & \cdots & 1 & & & & & & & & & \\ & & & & 1 & 1 & \cdots & 1 & & & & & \\ & & & & & & & & \ddots & & & & \\ & & & & & & & & & 1 & 1 & \cdots & 1 \\ 1 & & & & 1 & & & & & 1 & & & \\ & 1 & & & & 1 & & & \cdots & & 1 & & \\ & & \ddots & & & & \ddots & & & & & \ddots & \\ & & & 1 & & & & 1 & & & & & 1 \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{12} \\ \vdots \\ x_{1n} \\ x_{21} \\ x_{22} \\ \vdots \\ x_{2n} \\ \vdots \\ x_{m1} \\ x_{m2} \\ \vdots \\ x_{mn} \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_m \\ d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix}.$$

The structure of this coefficient matrix can be exploited to improve both the time and space required by the simplex algorithm on a transportation problem. A detailed description of the *transportation simplex method* can be found in [32]. A C-code implementation of transportation simplex algorithm is currently available at `http://robotics.stanford.edu/~rubner/research.html`.

The transportation simplex algorithm can still be applied when the total supply s_Σ is greater than the total demand d_Σ. The goal is still to find the minimum cost way to satisfy all the demand. In this case, however, there will be some supply left over after the demand has been satisfied. The LP for the unbalanced case is

$$\min_{(x_{ij})} \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\begin{aligned} x_{ij} &\geq 0 & i = 1, \ldots, m,\ j = 1, \ldots, n, \\ \sum_{j=1}^{n} x_{ij} &\leq s_i & i = 1, \ldots, m, \\ \sum_{i=1}^{m} x_{ij} &= d_j & j = 1, \ldots, n. \end{aligned}$$

In order to apply the transportation simplex method, we convert the unbalanced TP to an equivalent balanced TP. This is done by adding a dummy demander $n + 1$ with demand $d_{n+1} = s_\Sigma - d_\Sigma$, and for which $c_{i,n+1} = 0$ for $i = 1, \ldots, m$. The total demand in the modified problem is equal to the total supply, and the minimum cost is the same for the balanced and unbalanced problems. The dummy demander gives the suppliers a place to dump their leftover supply at no cost.

Let us now connect the work minimization LP to the unbalanced transportation problem. If, for example, $u_\Sigma \leq w_\Sigma$, then the work minimization LP can be rewritten as

$$\min_{(f_{ij})} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}$$

subject to

$$\begin{aligned} f_{ij} &\geq 0 & i = 1, \ldots, m,\ j = 1, \ldots, n, \\ \sum_{j=1}^{n} f_{ij} &\leq w_i & i = 1, \ldots, m, \end{aligned}$$

$$\sum_{i=1}^{m} f_{ij} = u_j \qquad j = 1, \ldots, n.$$

This LP is an unbalanced transportation problem, where the supplies are $w_1, \ldots, w_m$, the demands are $u_1, \ldots, u_n$, and the costs are $d_{ij} = d(x_i, y_j)$. Similarly, if $w_\Sigma \leq u_\Sigma$, then the suppliers are from distribution $\mathbf{y} = (Y, u)$ and the demanders are from distribution $\mathbf{x} = (X, w)$. In the case of equal-weight distributions, $w_\Sigma = u_\Sigma$, the work LP reduces to

$$\min_{(f_{ij})} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}$$

subject to

$$\begin{aligned} f_{ij} &\geq 0 \qquad i = 1, \ldots, m,\ j = 1, \ldots, n, \\ \sum_{j=1}^{n} f_{ij} &= w_i \qquad i = 1, \ldots, m, \\ \sum_{i=1}^{m} f_{ij} &= u_j \qquad j = 1, \ldots, n, \end{aligned}$$

which is a balanced transportation problem. Thus, the work minimization problem in the numerator of equation (4.5) is a transportation problem, and it can be solved efficiently by applying the transportation simplex method.

4.3 Special Cases

In this section, we examine two special cases of the EMD when the input distributions are restricted in some way. In section 4.3.1, we show that the EMD reduces to an optimal one-to-one matching of points when all point weights in the two distributions are equal to one. In section 4.3.2, we consider the case of equal-weight distributions on the real line. In this case, we give a very efficient algorithm to compute the EMD in one pass over the points.

4.3.1 Point Set Matching using the EMD

A point set is a special case of a distribution in which all weights are equal to one. In the language of the transportation problem, all the supplies and demands are equal to one unit. A slightly more general restricted input to the transportation problem is one in which all supplies and demands are equal to integers. Here we can assume that the transportation problem is balanced, for a dummy demander that absorbs any excess supply will also have an integer demand. The integer input restriction adds structure to the transportation problem in what is usually known as the *integer solutions property.* This property states that when all the supplies and demands are integers, all feasible flows located at vertices of the feasible convex polytope $\mathcal{F}$ have integer values ([32]). Hence, all optimal feasible vertex flows consist only of integer values when all supplies and demands are integers. The transportation simplex method returns an optimal vertex flow.

Now let us return to the specific case when all supplies and demands are equal to one. From constraints (4.1), (4.2), and (4.3), it follows that $0 \leq f_{ij} \leq w_i$ and $0 \leq f_{ij} \leq u_j$ in every feasible flow $F = (f_{ij})$. This means that $0 \leq f_{ij} \leq 1$ in every feasible flow between point sets. Combining this fact with the integer solutions property, there exists an optimal feasible flow $F^* = (f^*_{ij})$ at a vertex of $\mathcal{F}$ with $f^*_{ij} \in \{0, 1\}$ $\forall i, j$. As previously noted, the transportation simplex method will return such a solution. The flow values involving a dummy demander needed to create a balanced transportation problem will be integers, but need not be binary. This is irrelevant for our purposes since such flow variables are not really part of the solution. The bottom line is that for distributions $\mathbf{x} \in \mathbf{D}^{K,m}$ and $\mathbf{y} \in \mathbf{D}^{K,n}$ which are point sets in $\mathbf{R}^K$ with $m \geq n$,

$$\mathrm{EMD}(\mathbf{x}, \mathbf{y}) = \frac{\min_{F=(f_{ij}) \in \mathcal{F}(\mathbf{x},\mathbf{y})} \sum_{i=1}^m \sum_{j=1}^n f_{ij} d(x_i, y_j)}{\min(w_\Sigma, u_\Sigma)} = \frac{\min_{\phi \in \Phi} \sum_{j=1}^n d(x_{\phi(j)}, y_j)}{n},$$

where Φ is the set of one-to-one correspondences

$$\Phi = \{\, \phi : \{\, 1, \ldots, n \,\} \to \{\, 1, \ldots, m \,\} \mid \phi(j_1) = \phi(j_2) \Leftrightarrow j_1 = j_2 \,\}.$$

Thus, the EMD between point sets measures the average distance between corresponding points in an optimal one-to-one matching.

It is worthwhile to note that although the transportation simplex method will find an optimal one-to-one matching between point sets, it does not take advantage of the fact that the supplies and demands are all equal to one. A transportation problem with such supplies and demands is known as an *assignment problem*, and there are specialized algorithms to solve assignment problems ([75]). One word of caution is needed before applying an assignment problem algorithm instead of a transportation problem algorithm to match point sets. Creating a balanced assignment problem (i.e. one in which the two point sets have the same number of points) by adding dummy points to the smaller set will result in a large increase in the number of problem variables if the two sets have very different sizes. A transportation problem can be balanced with the addition of only one dummy demander which creates far fewer dummy variables than in the assignment case.

Thus for very unbalanced point set matching problems, it may be more efficient to apply the transportation simplex method than an assignment problem algorithm which operates only on balanced problems.

4.3.2 The EMD in One Dimension

Let $x = (X, w) \in \mathbf{D}^{1,m}$ and $y = (Y, u) \in \mathbf{D}^{1,n}$ be distributions on the real line. Assume the points in $\mathbf{x}$ and $\mathbf{y}$ are sorted by position:

$$x_1 < x_2 < \cdots < x_m \qquad \text{and} \qquad y_1 < y_2 < \cdots < y_n.$$

We also assume in this section that the ground distance between points is the absolute value $(d = L_1)$.

Define the *cumulative distribution function* (CDF) of $\mathbf{x}$ as

$$W(t) = \begin{cases} 0 & \text{if } t \in (-\infty, x_1) \\ \sum_{i=1}^{k} w_i & \text{if } t \in [x_k, x_{k+1}),\ \ 1 \le k \le m-1 \\ w_\Sigma = \sum_{i=1}^{m} w_i & \text{if } t \in [x_m, \infty). \end{cases}$$

Similarly, the CDF of $\mathbf{y}$ is

$$U(t) = \begin{cases} 0 & \text{if } t \in (-\infty, y_1) \\ \sum_{j=1}^{l} u_j & \text{if } t \in [y_l, y_{l+1}),\ \ 1 \le l \le n-1 \\ u_\Sigma = \sum_{j=1}^{n} u_j & \text{if } t \in [y_n, \infty). \end{cases}$$

If $\mathbf{x}$ and $\mathbf{y}$ are equal weight, then the minimum work to transform one distribution into the other is the area between the graphs of the CDFs of $\mathbf{x}$ and $\mathbf{y}$. We shall prove this fact later in this section in Theorem 5. An example is shown in Figure 4.5.

The flow naturally defined by the CDFs is called the *CDF flow*, and is denoted $F^{\text{CDF}} = (f_{ij}^{\text{CDF}})$. Once again, see Figure 4.5. If we let

$$\begin{array}{rclcll} W_k & = & W(x_k) & = & \sum_{i=1}^{k} w_i & \quad \text{and} \\ U_l & = & U(y_l) & = & \sum_{j=1}^{l} u_j, & \end{array}$$

then the CDF flow is given by

$$f_{ij}^{\text{CDF}} = |[W_{i-1}, W_i] \cap [U_{j-1}, U_j]|.$$

Theorem 4 *The flow F^{CDF} is a feasible flow between equal-weight distributions $\mathbf{x}$ and $\mathbf{y}$;*

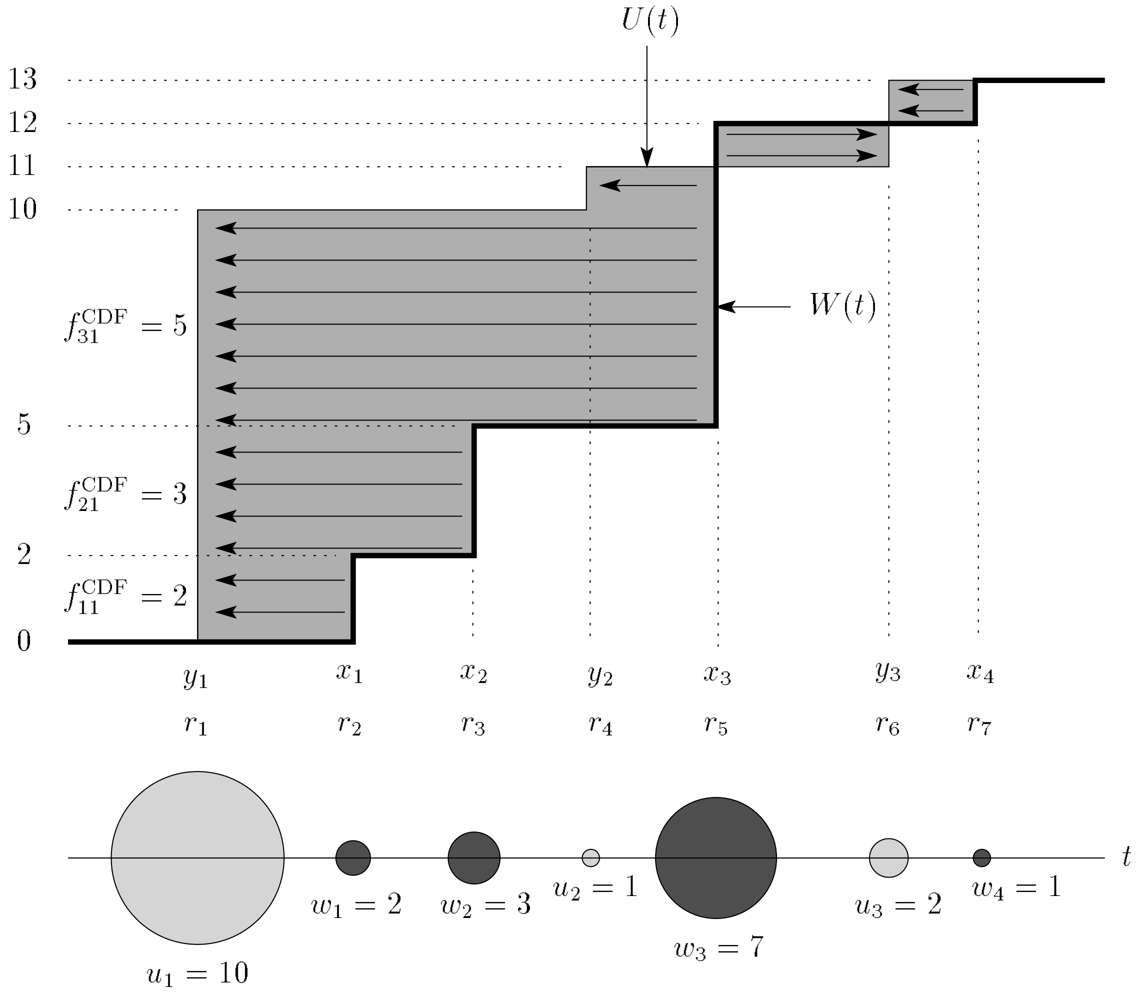


Figure 4.5: The EMD between Equal-Weight Distributions on the Real Line. The cumulative distribution functions (CDFs) for the equal-weight line distributions $\mathbf{x}$ and $\mathbf{y}$ are $W(t)$ and $U(t)$, repsectively. The minimum work to transform $\mathbf{x}$ into $\mathbf{y}$ is equal to the area between the two CDFs. An optimal transforming flow $F^{\mathrm{CDF}} = (f_{ij}^{\mathrm{CDF}})$, called the *CDF flow*, is shown with directed lines from $\mathbf{x}$-weight to matching $\mathbf{y}$-weight. The CDF flow is $f_{11}^{\mathrm{CDF}} = 2$, $f_{21}^{\mathrm{CDF}} = 3$, $f_{31}^{\mathrm{CDF}} = 5$, $f_{32}^{\mathrm{CDF}} = 1$, $f_{33}^{\mathrm{CDF}} = 1$, $f_{43}^{\mathrm{CDF}} = 1$, and $f_{ij}^{\mathrm{CDF}} = 0$ for all other pairs (i, j). The EMD between $\mathbf{x}$ and $\mathbf{y}$ is obtained by dividing the minimum work by the total weight of the distributions ($w_\Sigma = u_\Sigma = 13$ in this example).

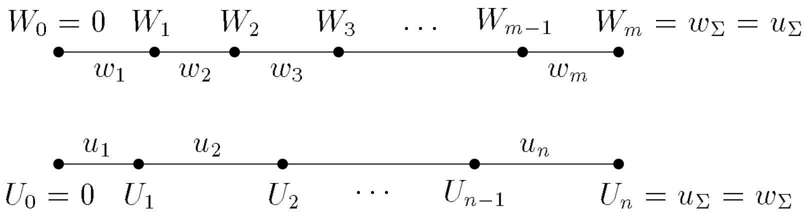


Figure 4.6: Feasibility of the CDF flow. The length of the $\mathbf{x}$-interval $[W_{i-1}, W_i]$ is the $\mathbf{x}$-weight w_i, and the length of the $\mathbf{y}$-interval $[U_{j-1}, U_j]$ is the $\mathbf{y}$-weight u_j. It should be clear from this figure that $\sum_{j=1}^n f_{ij}^{\mathrm{CDF}} = w_i$ and $\sum_{i=1}^m f_{ij}^{\mathrm{CDF}} = u_j$, where $f_{ij}^{\mathrm{CDF}} = |[W_{i-1}, W_i] \cap [U_{j-1}, U_j]|$.

i.e., $F^{\mathrm{CDF}} \in \mathcal{F}(\mathbf{x}, \mathbf{y})$.

Proof. Obviously $f_{ij}^{\mathrm{CDF}} \geq 0$. It remains to show that

$$\sum_{j=1}^{n} f_{ij}^{\mathrm{CDF}} = w_i \qquad \text{and} \qquad \sum_{i=1}^{m} f_{ij}^{\mathrm{CDF}} = u_j.$$

Note that the disjoint (except at interval endpoints) unions

$$\bigcup_{i=1}^{m} [W_{i-1}, W_i] = [0, w_\Sigma] \qquad \text{and} \qquad \bigcup_{j=1}^{n} [U_{j-1}, U_j] = [0, u_\Sigma]$$

cover exactly the same interval $[0, w_\Sigma] = [0, u_\Sigma]$. See Figure 4.6. It follows that

$$\begin{aligned}
\textstyle\sum_{j=1}^n f_{ij}^{\mathrm{CDF}} &= \textstyle\sum_{j=1}^n |[W_{i-1}, W_i] \cap [U_{j-1}, U_j]| \\
&= \left|[W_{i-1}, W_i] \cap \left(\textstyle\bigcup_{j=1}^n [U_{j-1}, U_j]\right)\right| && \text{(interior disjointness of } [U_{j-1}, U_j]) \\
&= |[W_{i-1}, W_i] \cap [0, u_\Sigma]| \\
&= |[W_{i-1}, W_i] \cap [0, w_\Sigma]| \\
&= |[W_{i-1}, W_i]| && ([W_{i-1}, W_i] \subseteq [0, w_\Sigma]) \\
\textstyle\sum_{j=1}^n f_{ij}^{\mathrm{CDF}} &= w_i.
\end{aligned}$$

Similar reasoning proves that $\sum_{i=1}^m f_{ij}^{\mathrm{CDF}} = u_j$. ■

Now denote the sorted list of breakpoints $x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n$ as

$$r_1 \leq r_2 \leq \cdots \leq r_{m+n}.$$

See Figure 4.5. In order to prove the optimality of F^{CDF}, we need the following lemma.

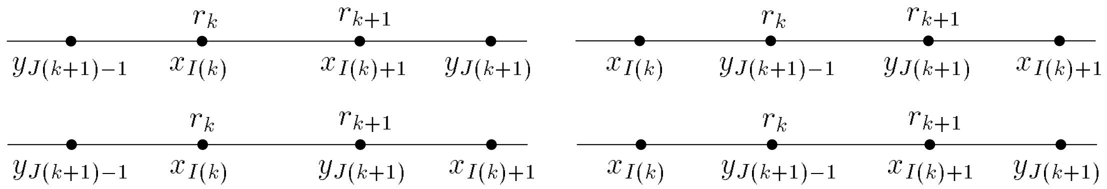


Figure 4.7: Breakpoint Notation Used in Lemma 2. $I(k)$ is the largest i such that $x_i \leq r_k$, and $J(k+1)$ is the smallest j such that $y_j \geq r_{k+1}$. The leftmost and rightmost labelled points are not necessarily r_{k-1} and r_{k+2}.

Lemma 2 *The feasible flow F^{CDF} between equal-weight distributions $\mathbf{x}$ and $\mathbf{y}$ moves exactly $|W(r_k) - U(r_k)|$ weight from $\mathbf{x}$ to $\mathbf{y}$ over the interval (r_k, r_{k+1}). More precisely, it moves $W(r_k) - U(r_k)$ $\mathbf{x}$-weight from r_k to r_{k+1} if $W(r_k) \geq U(r_k)$ and $U(r_k) - W(r_k)$ from r_{k+1} to r_k if $U(r_k) > W(r_k)$.*

Proof. Let $I(k)$ be the largest i such that $x_i \leq r_k$, and let $J(k+1)$ be the smallest j such that $y_j \geq r_{k+1}$. The four possible configurations are shown in Figure 4.7. Note also that $I(k)+1$ is the smallest i such that $x_i \geq r_{k+1}$ and $J(k+1)-1$ is the largest j such that $y_j \leq r_k$. The key observations here are that

$$W_{I(k)} = W(r_k) \qquad \text{and} \qquad U_{J(k+1)-1} = U(r_k) \tag{4.7}$$

for all four possible configurations.

The amount of $\mathbf{x}$-weight $\alpha_{k\to k+1}$ that flows from r_k to r_{k+1} during the feasible flow F^{CDF} is

$$\begin{aligned}
\alpha_{k\to k+1} &= \sum_{i=1}^{I(k)} \sum_{j=J(k+1)}^{n} f_{ij}^{\mathrm{CDF}} \\
&= \sum_{i=1}^{I(k)} \sum_{j=J(k+1)}^{n} |[W_{i-1}, W_i] \cap [U_{j-1}, U_j]| \\
&= \left| \left(\cup_{i=1}^{I(k)} [W_{i-1}, W_i] \right) \cap \left(\cup_{j=J(k+1)}^{n} [U_{j-1}, U_j] \right) \right| && (4.8) \\
&= |[0, W_{I(k)}] \cap [U_{J(k+1)-1}, U_n]| \\
&= |[0, W(r_k)] \cap [U(r_k), u_\Sigma]| && \text{(by (4.7))} \\
\alpha_{k\to k+1} &= \begin{cases} W(r_k) - U(r_k) & \text{if } W(r_k) \geq U(r_k) \\ 0 & \text{otherwise} \end{cases} . && (4.9)
\end{aligned}$$

The line (4.8) follows from the previous line by the interior disjointness of the intervals $[W_{i-1}, W_i]$ and the interior disjointness of the intervals $[U_{j-1}, U_j]$. Similarly, the amount of

$\mathbf{x}$-weight $\alpha_{k+1\to k}$ that flows from r_{k+1} to r_k is

$$\begin{aligned}
\alpha_{k+1\to k} &= \sum_{i=I(k)+1}^{m} \sum_{j=1}^{J(k+1)-1} f_{ij}^{\mathrm{CDF}} \\
&= \sum_{i=I(k)+1}^{m} \sum_{j=1}^{J(k+1)-1} |[W_{i-1}, W_i] \cap [U_{j-1}, U_j]| \\
&= \left| \left(\cup_{i=I(k)+1}^{m} [W_{i-1}, W_i]\right) \cap \left(\cup_{j=1}^{J(k+1)-1} [U_{j-1}, U_j]\right) \right| \\
&= |[W_{I(k)}, W_m] \cap [0, U_{J(k+1)-1}]| \\
&= |[W(r_k), w_\Sigma] \cap [0, U(r_k)]| \qquad \text{(by (4.7))} \\
\alpha_{k+1\to k} &= \begin{cases} U(r_k) - W(r_k) & \text{if } U(r_k) > W(r_k) \\ 0 & \text{otherwise} \end{cases} . \qquad (4.10)
\end{aligned}$$

The desired result follows from (4.9) and (4.10). ■

We are now ready to prove the main result of this section.

Theorem 5 *If* $\mathbf{x} = (X, w) \in \mathbf{D}^{1,m}$ *and* $\mathbf{y} = (Y, u) \in \mathbf{D}^{1,n}$ *have equal weight* $w_\Sigma = u_\Sigma$*, then*

$$\mathrm{EMD}(\mathbf{x}, \mathbf{y}) = \frac{\int_{-\infty}^{\infty} |W(t) - U(t)|\, dt}{w_\Sigma}.$$

Furthermore, F^{CDF} *is an optimal feasible flow between* $\mathbf{x}$ *and* $\mathbf{y}$*.*

Proof. Note that $W(t)$ and $U(t)$ are constant over the interval $t \in [r_k, r_{k+1})$ for $k = 1, \ldots, m+n-1$, $W(t) = U(t) \equiv 0$ for $t \in (-\infty, r_1)$, and $W(t) = U(t) \equiv w_\Sigma = u_\Sigma$ for $t \in [r_{m+n}, \infty)$. Therefore the integral of the absolute difference of the CDFs may be written as the finite summation

$$\int_{-\infty}^{\infty} |W(t) - U(t)|\, dt = \sum_{k=1}^{m+n-1} E_k \,, \qquad (4.11)$$

where

$$E_k = (r_{k+1} - r_k)\, |W(r_k) - U(r_k)|.$$

Consider the interval (r_k, r_{k+1}). At any position t in this interval, the absolute difference $|W(t) - U(t)|$ is equal to $|W(r_k) - U(r_k)|$. Suppose that $W(r_k) > U(r_k)$. Then in any feasible flow from $\mathbf{x}$ to $\mathbf{y}$, the net flow from r_k to r_{k+1} must be exactly $W(r_k) - U(r_k)$. If the net flow is less than this amount, then there will be less $\mathbf{x}$-weight than $\mathbf{y}$-weight in $[r_{k+1}, \infty)$ after the flow is complete. If the net flow is more than this amount, then there will be more $\mathbf{x}$-weight than $\mathbf{y}$-weight in $[r_{k+1}, \infty)$ after the flow is complete. See Figure 4.8(a).

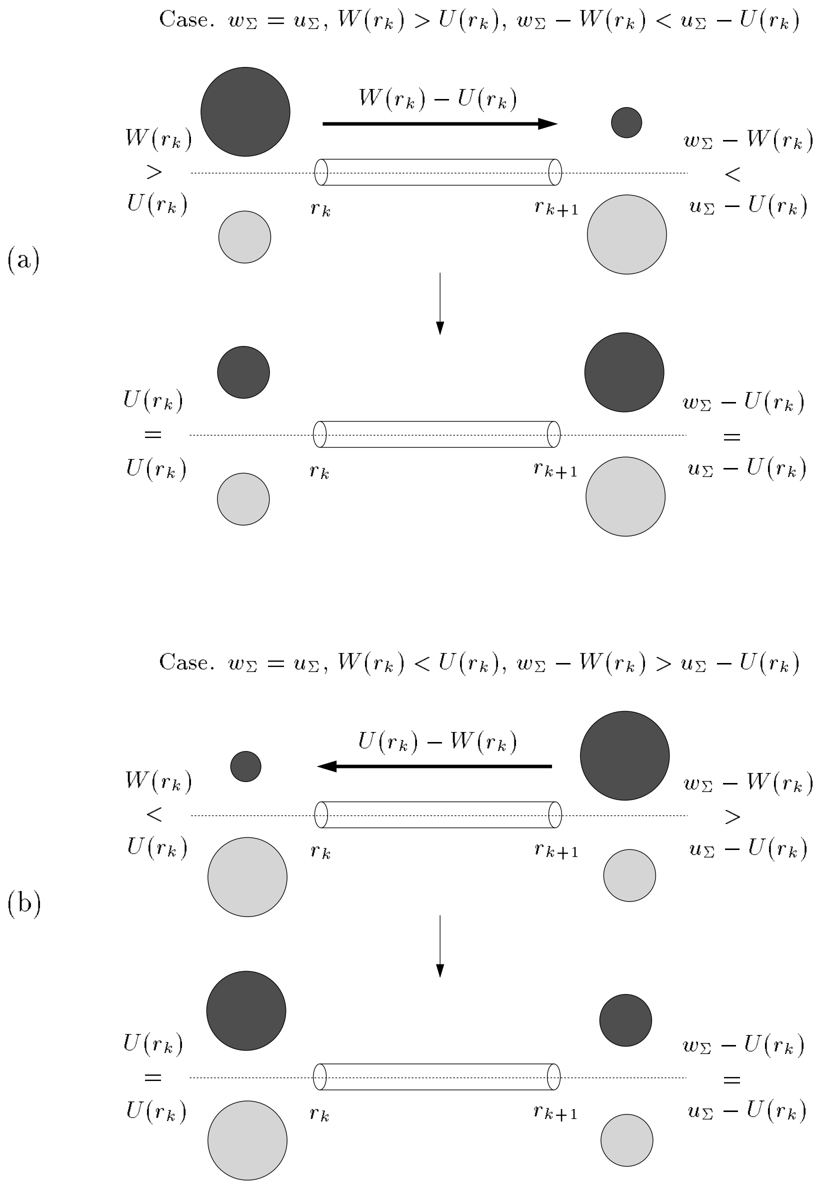


Figure 4.8: Flow Feasibility for Equal-Weight Distributions on the Real Line. $\mathbf{x} = (X, w)$ and $\mathbf{y} = (Y, u)$ are distributions in 1D. Here $r_1 \leq \cdots \leq r_{m+n}$ is the position-sorted list of points in $\mathbf{x}$ and $\mathbf{y}$, and $W(t)$ and $U(t)$ are the CDFs for $\mathbf{x}$ and $\mathbf{y}$, respectively. (a) $W(r_k) > U(r_k)$, $w_\Sigma - W(r_k) < u_\Sigma - U(r_k)$. In this case, a flow from $\mathbf{x}$ to $\mathbf{y}$ is feasible only if the net flow of $\mathbf{x}$-weight from r_k to r_{k+1} is exactly $W(r_k) - U(r_k)$. (b) $W(r_k) < U(r_k)$, $w_\Sigma - W(r_k) > u_\Sigma - U(r_k)$. In this case, a flow from $\mathbf{x}$ to $\mathbf{y}$ is feasible only if the net flow of $\mathbf{x}$-weight from r_{k+1} to r_k is exactly $U(r_k) - W(r_k)$.

Similar logic shows that if $U(r_k) > W(r_k)$, then the net flow of $\mathbf{x}$-weight from r_{k+1} to r_k must be exactly $U(r_k) - W(r_k)$. This case is illustrated in Figure 4.8(b). In either case, the amount of work E_k done in moving weight from $\mathbf{x}$ over the interval (r_k, r_{k+1}) is at least E_k, and

$$\min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y})} \mathrm{WORK}(F,\mathbf{x},\mathbf{y}) \geq \sum_{k=1}^{m+n-1} E_k. \tag{4.12}$$

To complete the proof, note that Lemma 2 says that F^{CDF} is a feasible flow[1] which requires work $\sum_{k=1}^{m+n-1} E_k$ to match $\mathbf{x}$ and $\mathbf{y}$. It follows that

$$\min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y})} \mathrm{WORK}(F,\mathbf{x},\mathbf{y}) \leq \mathrm{WORK}(F^{\mathrm{CDF}},\mathbf{x},\mathbf{y}) = \sum_{k=1}^{m+n-1} E_k. \tag{4.13}$$

Combining (4.12), (4.13), and (4.11) gives the desired result after normalizing by $w_\Sigma = u_\Sigma$. ■

Pseudocode to compute the EMD between equal-weight distributions in one dimension (with ground distance equal to the L_1 distance) is given below. This code is a direct translation of (4.11) and computes the integral by a sweep over the position axis, summing areas of rectangles with bases $r_{k+1} - r_k$ and heights $|W(r_k) - U(r_k)|$. Again, see Figure 4.5.

```
function emd = EMD1(x, y)
/* assumes K = 1, wΣ = uΣ, ground distance is L1 */
/* assumes x1 ≤ x2 ≤ ··· ≤ xm, y1 ≤ y2 ≤ ··· ≤ yn */
   work = wsum = usum = r = 0
   /* first increment of work will be 0, regardless of r */
   i = j = 1
   xnext = x1
   ynext = y1
   while ((i ≤ m) or (j ≤ n))
      if (xnext ≤ ynext)
         work += |wsum-usum|*(xnext-r)
         wsum += wi
         r = xnext
         i += 1
         xnext = (i ≤ m) ? xi : ∞
      else
         work += |wsum-usum|*(ynext-r)
         usum += uj
         r = ynext
         j += 1
```

[1] In [11], it is incorrectly stated that there is a unique feasible flow between equal-weight distributions in 1D. In fact, there may even be more than one optimal feasible flow. For example, suppose $X = [0\ 1]$, $w = [1\ 1]$, $Y = [1\ 2]$, and $u = [1\ 1]$. Then F^{CDF} is given by $f_{11}^{\mathrm{CDF}} = f_{22}^{\mathrm{CDF}} = 1$, $f_{12}^{\mathrm{CDF}} = f_{21}^{\mathrm{CDF}} = 0$. The feasible flow F^* given by $f_{11}^* = f_{22}^* = 0$, $f_{12}^* = f_{21}^* = 1$ is also an optimal feasible flow between $\mathbf{x} = (X, w)$ and $\mathbf{y} = (Y, u)$.

```
            ynext = (j ≤ n) ? y_j : ∞
        end if
    end while
    return (work / usum)
end function
```

Assuming that the points in $x \in \mathbf{D}^{1,m}$ and $y \in \mathbf{D}^{1,n}$ are in sorted order, the routine EMD_1 runs in linear time $\Theta(m+n)$. The combined sorted list $r_1, \ldots, r_{m+n}$ of points in $\mathbf{x}$ and $\mathbf{y}$ is discovered by walking along the two sorted lists of points. At any time during the algorithm, there is a pointer to the next $\mathbf{x}$ and next $\mathbf{y}$ value to be considered. The value r_{k+1} then follows in constant time from the value of r_k.

The function EMD_1 does not compute the optimal CDF flow $F^{\text{CDF}} = (f_{ij}^{\text{CDF}})$. We can rewrite the EMD_1 routine as shown below so that it also returns the optimal CDF flow with the single EMD value. This code is a direct translation of the fact that $\int_{-\infty}^{\infty} |W(t) - U(t)|\ dt = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}^{\text{CDF}} |x_i - y_j|$ (which follows from Theorem 5) and computes the integral by a sweep over the cumulative-weight axis, summing areas of rectangles with bases f_{ij}^{CDF} and heights $|x_i - y_j|$. See Figure 4.9.

```
function [emd,CDFflow] = EMD_1(x, y)
/* assumes K = 1, w_Σ = u_Σ, ground distance is L_1 */
/* assumes x_1 ≤ x_2 ≤ ··· ≤ x_m, y_1 ≤ y_2 ≤ ··· ≤ y_n */
    work = prev = CDFflow.nFlow = 0
    i = j = 1
    wsum = w_1 /* holds W_i */
    usum = u_1 /* holds U_j */
    while ((i ≤ m) and (j ≤ n))
        CDFflow[CDFflow.nFlow].from = i
        CDFflow[CDFflow.nFlow].to = j
        if (usum ≤ wsum) /* check (U_j ≤ W_i) */
            f_ij^CDF = usum-prev
            work += f_ij^CDF × |x_i − y_j|
            prev = usum
            usum += u_j
            j += 1
        else
            f_ij^CDF = wsum-prev
            work += f_ij^CDF × |x_i − y_j|
            prev = wsum
            wsum += w_i
            i += 1
        end if
```

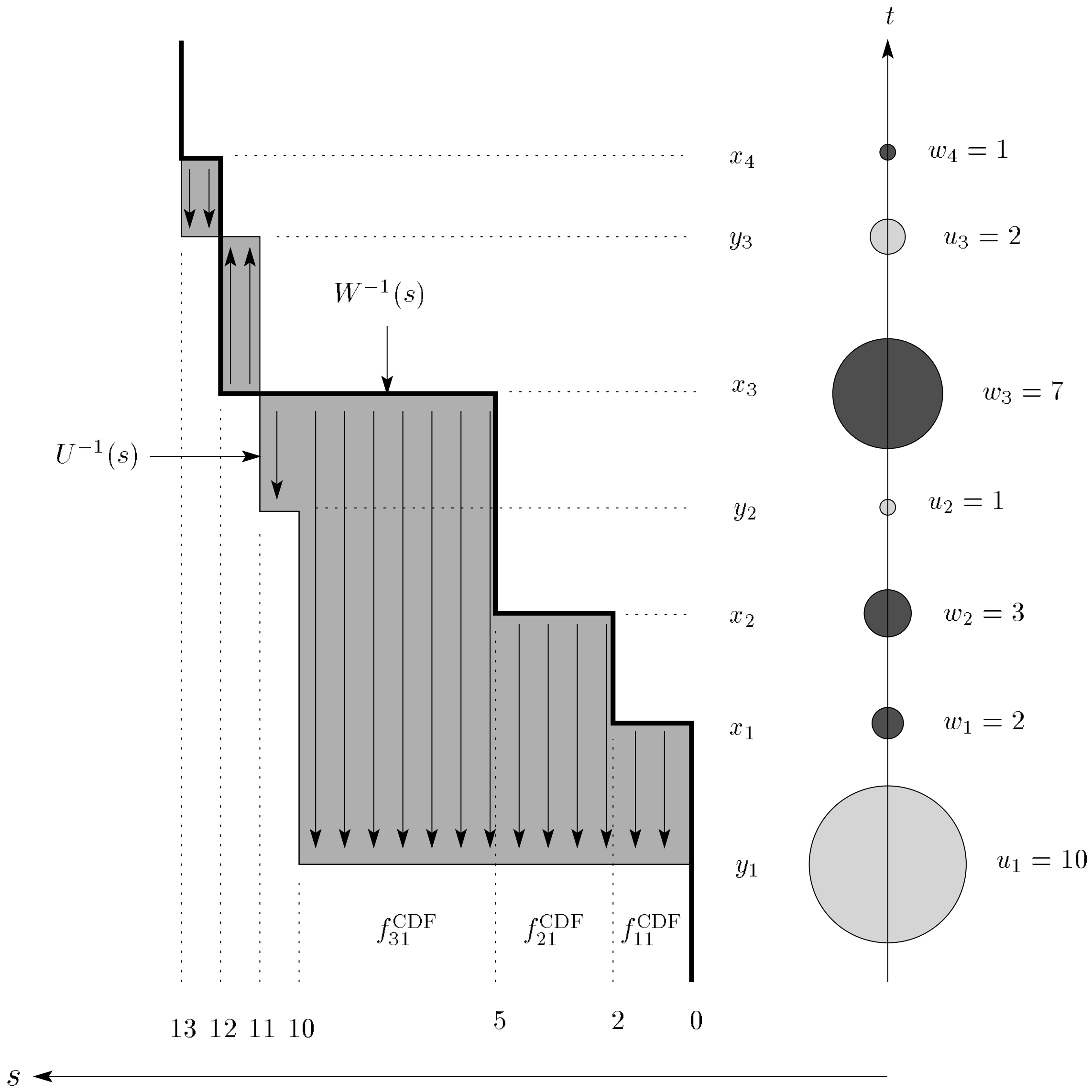


Figure 4.9: The Inverse CDFs. The area between the inverse CDFs $W^{-1}(s)$ and $U^{-1}(s)$ over $s \in [0, w_\Sigma] = [0, u_\Sigma]$ is clearly the same as the area between the CDFs $W(t)$ and $U(t)$ (see Figure 4.5) over $t \in (-\infty, \infty)$.

CDFflow[CDFflow.nFlow].amount = f_{ij}^{CDF}
CDFflow.nFlow += 1
end while
emd = work / usum
return (emd,CDFflow)
end function

This version of EMD_1 also runs in $\Theta(m+n)$ time since there is a constant amount of computation done at each of the $m+n$ breakpoints $W_1,\ldots,W_m,U_1,\ldots,U_n$. All flow variables f_{ij}^{CDF} not explicitly contained in the variable CDFflow are equal to zero.

4.4 Modifications

We now discuss some useful modifications to the EMD. As initially stated, the EMD computation forces all the weight in the lighter distribution to match weight in the heavier distribution. In section 4.4.1, we extend the EMD to take another parameter $0 < \gamma \leq 1$ which specifies the fraction of the lighter distribution to be matched. The *partial Earth Mover's Distance* computation automatically selects the best weight from the lighter distribution to match.[2] The ability to compute the best partial match is important for robustness in the presence of outliers and/or missing data.[3] The γ parameter is an attempt to avoid penalizing the non-matching parts of two distributions which have a lot in common. Remember that the goal is to measure visual similarity by matching summary distributions, and visual similarity may follow from only a partial match. An alternative similarity measure which accounts for this fact asks "How much weight can be matched when flow distances are limited to at most some given ground distance τ?". This *restricted Earth Mover's Distance* is the subject of section 4.4.2.

4.4.1 The Partial Earth Mover's Distance

The *partial Earth Mover's Distance* EMD^γ matches only a given fraction $0 < \gamma \leq 1$ of the weight of the lighter distribution or some absolute amount of weight $0 < \gamma \leq \min(w_\Sigma, u_\Sigma)$. The former case in which γ is a relative quantity is called the *relative* partial EMD, and the latter case in which γ is an absolute quantity is called the *absolute* partial EMD. In a

[2]This name may be slightly misleading since the EMD already does partial matching. When one distribution is heavier than the other, all the weight in the lighter distribution is matched, but some of the weight in the heavier distribution is unmatched. With the partial EMD, some of the weight of the lighter distribution is unmatched.

[3]The EMD is robust to a small amount of outlier mass since the large ground distances needed to match the outlier mass are weighted by small fractions of mass moved.

relative partial EMD problem, the amount of weight matched is $M(\gamma) = \gamma \min(w_\Sigma, u_\Sigma)$; in an absolute partial EMD problem, the amount of weight matched is $M(\gamma) = \gamma$. In either case, the conditions for a feasible flow are

$$\begin{aligned} f_{ij} &\geq 0 && i = 1,\dots,m,\ j = 1,\dots,n, \\ \sum_{j=1}^{n} f_{ij} &\leq w_i && i = 1,\dots,m, \\ \sum_{i=1}^{m} f_{ij} &\leq u_j && j = 1,\dots,n, \quad \text{and} \\ \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} &= M(\gamma). && \end{aligned} \tag{4.14}$$

The only difference in the feasibility conditions for the partial EMD and the ordinary EMD are in the final conditions (4.14) and (4.4) which indicate the total amount of weight to match. If we denote the set of feasible flows between $\mathbf{x}$ and $\mathbf{y}$ for partial match parameter γ as $\mathcal{F}^\gamma(\mathbf{x},\mathbf{y})$, then we define the partial EMD as

$$\mathrm{EMD}^\gamma(\mathbf{x},\mathbf{y}) = \frac{\min_{F=(f_{ij})\in\mathcal{F}^\gamma(\mathbf{x},\mathbf{y})} \mathrm{WORK}(F,\mathbf{x},\mathbf{y})}{M(\gamma)}. \tag{4.15}$$

Since γ is given, the denominator of (4.15) is fixed for an EMD^γ computation. An example partial EMD computation is shown in Figure 4.10. In section 4.2, we showed that the work minimization problem for the ordinary EMD computation can be solved as a transportation problem. We now show that the same is true for the (relative or absolute) partial EMD computation. Therefore, the same transportation problem code that is used to compute the EMD can also be used to compute the partial EMD.

Suppose that $\mathbf{x}$ is at least as heavy as $\mathbf{y}$; i.e., $w_\Sigma \geq u_\Sigma$. Then the work minimization problem in the numerator of (4.15) is the balanced transportation problem

$$\begin{aligned} s_i &= w_i \quad i = 1,\dots,m \\ s_{m+1} &= u_\Sigma - M(\gamma) \\ d_j &= u_j, \quad j = 1,\dots,n \\ d_{n+1} &= w_\Sigma - M(\gamma) \\ c_{ij} &= d(x_i, y_j) \quad i = 1,\dots,m,\ j = 1,\dots,n \\ c_{m+1,j} &= 0 \quad j = 1,\dots,n \\ c_{i,n+1} &= 0 \quad i = 1,\dots,m \\ c_{m+1,n+1} &= \infty. \end{aligned}$$

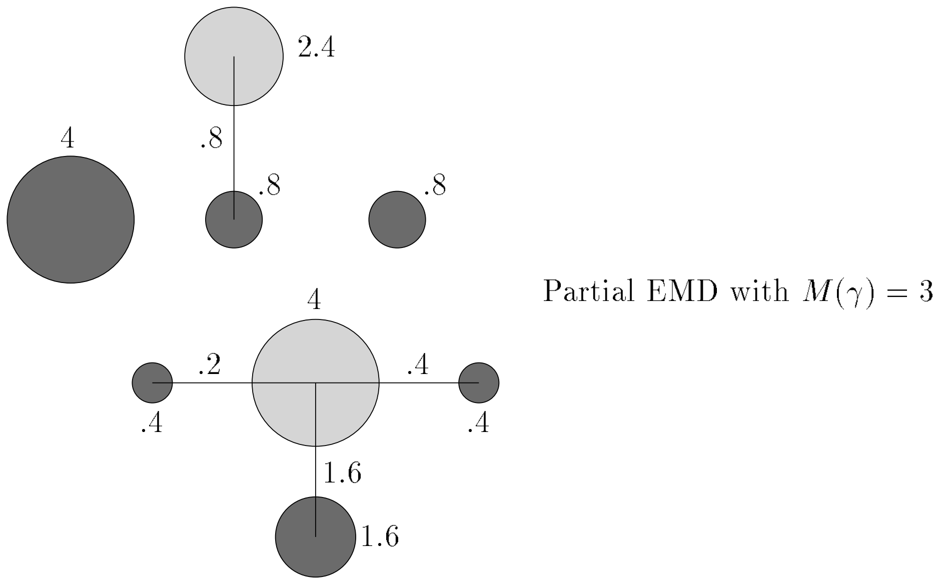


Figure 4.10: Partial EMD Example. The dark gray distribution has total weight 8, while the light gray distribution has total weight 6.4. An optimal flow for the partial EMD when $M(\gamma) = 3$ units of weight must be matched is shown by the labelled edges. All ground distances used in this flow are equal, and less than all ground distances not used.

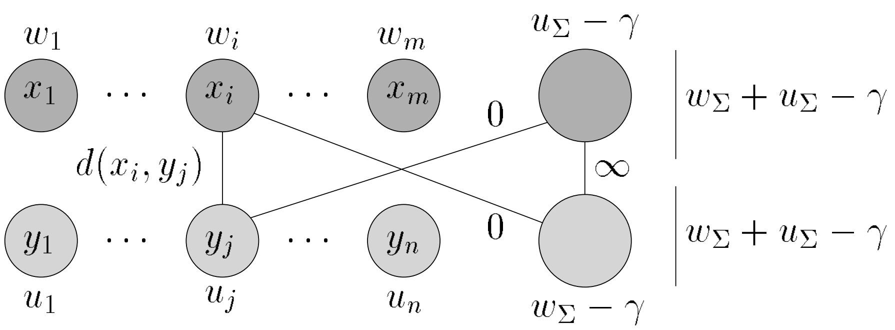


Figure 4.11: The Partial EMD as a Balanced Transportation Problem. See the text for an explanation.

A graphical representation is shown in Figure 4.11. The total supply s_Σ and total demand d_Σ are both equal to $w_\Sigma + u_\Sigma - M(\gamma)$. The dummy supplier $m+1$ is given a supply which equal to the unmatched weight of $\mathbf{y}$, while the dummy demander $n+1$ is given a demand which is equal to the unmatched weight of $\mathbf{x}$. The weight of the dummy supplier is prevented from matching the weight of the dummy demander with the requirement $c_{m+1,n+1} = \infty$, so all of the weight of the dummy supplier will be matched at no cost to demanders $j = 1, \ldots, n$. Of the remaining supply w_Σ possessed by suppliers $i = 1, \ldots, m$, $w_\Sigma - M(\gamma)$ will be matched at no cost to the dummy demander. Therefore only $M(\gamma)$ weight will be matched at possibly nonzero cost. An algorithm to solve the transportation problem will find the optimal way to transport this weight from suppliers $i = 1, \ldots, m$ to demanders $j = 1, \ldots, n$. If $\mathbf{x}$ is lighter than $\mathbf{y}$, then the above formulation with the roles of $\mathbf{x}$ and $\mathbf{y}$ interchanged allows the partial EMD work minimization problem to be formulated as a balanced transportation problem.

In section 4.3.1, we discussed the special case in which the two distributions compared by the EMD are point sets. The EMD yields an optimal one-to-one matching in which each point in the smaller set is matched to a point in the larger set. Using the partial EMD instead of the EMD, we can find an optimal matching which matches only some subset of the points in the smaller set. The total number of points to be matched using EMD^γ is $M(\gamma)$. As long as γ is selected so that $M(\gamma)$ is an integer, all the supplies and demands in the corresponding transportation problem will be integers (see the above formulations), with all the non-dummy supplies and demands equal to one. Applying the transportation simplex algorithm will yield an optimal one-to-one matching between size $M(\gamma)$ subsets of the smaller and larger sets. If the distributions $\mathbf{x} \in \mathbf{D}^{K,m}$ and $\mathbf{y} \in \mathbf{D}^{K,n}$ are point sets in $\mathbf{R}^K$ with $m \geq n$, then

$$\begin{aligned} \text{EMD}^\gamma(\mathbf{x}, \mathbf{y}) &= \frac{\min_{F=(f_{ij}) \in \mathcal{F}^\gamma(\mathbf{x},\mathbf{y})} \sum_{i=1}^m \sum_{j=1}^n f_{ij} d(x_i, y_j)}{M(\gamma)} \\ &= \frac{\min_{\phi \in \Phi^\gamma} \sum_{j \in \text{domain}(\phi)} d(x_{\phi(j)}, y_j)}{M(\gamma)}, \end{aligned}$$

where Φ^γ is the set of one-to-one partial correspondences

$$\Phi^\gamma = \{\ \phi : S \to [1..m] \ \mid \ S \subseteq [1..n],\ |S| = M(\gamma),\ \phi(j_1) = \phi(j_2) \Leftrightarrow j_1 = j_2 \ \ \forall j_1, j_2 \in S\ \}.$$

It is important to note that only the number of points to be matched is given; the partial EMD figures out the best subsets to match.

4.4.2 The Restricted Earth Mover's Distance

The *restricted Earth Mover's Distance* τ-EMD is a measure of how much weight can be matched when ground distances for transportation are limited to a threshold τ. When comparing two distributions $\mathbf{x} = (X, w)$ and $\mathbf{y} = (Y, u)$, the maximum amount of weight that can be matched if transportation distances $d_{ij} = d(x_i, y_j)$ are unrestricted is $M = \min(w_\Sigma, u_\Sigma)$. Let M_τ denote the maximum amount of weight that can be matched using

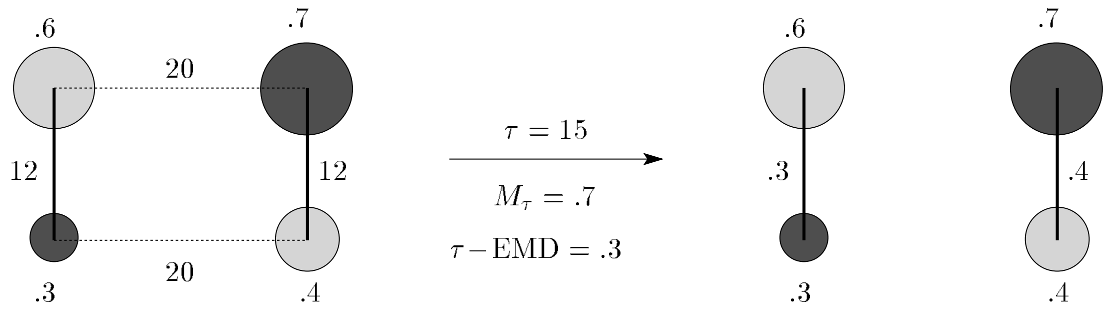


Figure 4.12: τ-EMD Example. (left) Equal-weight distributions with distances between weight locations. (right) If $\tau = 15$, then the maximum amount of weight that can be matched is $M_\tau = 0.7$. The maximal matching is indicated by the labelled edges. Weight cannot be matched between the locations which are $20 > \tau$ units apart. Since the total weight of both distributions is one, the fraction of weight that cannot be matched is τ-EMD $= 0.3$.

only distances $d_{ij} \leq \tau$. Then we define the restricted EMD as

$$\tau\text{-EMD}(\mathbf{x}, \mathbf{y}) = 1 - \frac{M_\tau(\mathbf{x}, \mathbf{y})}{\min(w_\Sigma, u_\Sigma)}. \tag{4.16}$$

Note that the τ-EMD actually equals the fraction of weight that *cannot* be matched, with $0 \leq \tau\text{-EMD}(\mathbf{x}, \mathbf{y}) \leq 1$, so that τ-EMD is a dissimilarity measure rather than a similarity measure. The extreme values are zero when the maximum amount of weight can be matched, and one when none of the weight can be matched. An example is shown in Figure 4.12. We now show that the computation of $M_\tau(\mathbf{x}, \mathbf{y})$ is a transportation problem.

The $M_\tau(\mathbf{x}, \mathbf{y})$ computation is the linear program

$$M_\tau(\mathbf{x}, \mathbf{y}) = \max_{F \in \mathcal{F}(\mathbf{x}, \mathbf{y})} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} [d(x_i, y_j) \leq \tau],$$

where

$$d_\tau(x_i, y_j) = [d(x_i, y_j) \leq \tau] = \begin{cases} 1 & \text{if } d(x_i, y_j) \leq \tau \\ 0 & \text{otherwise} \end{cases}.$$

The flow constraints are the same as for the original EMD computation, but now we sum up the matched weight f_{ij} whenever $d(x_i, y_j) \leq \tau$. The transportation problem here is

$$-M_\tau(\mathbf{x}, \mathbf{y}) = \min_{F \in \mathcal{F}(\mathbf{x}, \mathbf{y})} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} (-d_\tau(x_i, y_j))$$

In fact, this is an original EMD computation with ground distances $(-d_\tau(x_i, y_j))$. There is no restriction in the transportation problem that costs be nonnegative, and the transportation simplex method makes no such assumption.

4.5 Use in Scale Estimation

In this section, we show how to use the EMD to estimate the scale at which a pattern appears within an image. Figure 4.13(a) shows an example of the color pattern problem. Scale estimation is a critical step in solving the pattern problem accurately and efficiently. A good scale estimate is important for accuracy because the scale determines how much information in the image is compared to the pattern; it is important for efficiency because trying many scales will be inefficient, especially if one is interested in finding very small occurrences of the pattern. Along with a scale estimate, our method also returns a measure indicating the distance between the pattern at the predicted scale and the image. If this distance is large, then the pattern probably does not occur within the image.

We regard an image as a distribution of color mass or curve orientation mass (for the shape pattern problem) in position space. Our scale estimation method uses the EMD to compare image summary distributions after marginalizing away position. Ignoring position information does throw away useful information, but it reduces the complexity of the summary distributions and, therefore, allows fast scale estimation. We will show that it is possible to get very good scale estimates without position information if the pattern has a single distinctive feature with respect to the image. In the color pattern problem, the marginalized distribution is a distribution in color space. In order to keep the distribution size small, the colors of an image are clustered into a small number of groups (approximately twenty). The weight of a color cluster in CIE-Lab space ([88]) is the fraction of the total image area classified as that color. Thus the total weight of a summary distribution is one.

Suppose that a pattern occurs in an image as a fraction $c^* \in (0,1]$ of the total image area. An example is shown in Figure 4.13(a). Let $\mathbf{x}$ and $\mathbf{y} = (Y,u)$ denote unit-weight color signatures of the image and pattern, respectively. See Figure 4.13(b),(d). Since (Y, c^*u) is lighter than $\mathbf{x}$, the EMD finds the optimal matching between c^* of the image color weight and the color weight in (Y, c^*u). Consider the ideal case of an exact pattern occurrence, with the same color clusters used in $\mathbf{x}$ and $\mathbf{y}$ for the pattern colors. Then the c^* of $\mathbf{x}$'s color weight contributed by the pattern occurrence will match exactly the color weight in (Y, c^*u), and $\text{EMD}(\mathbf{x}, (Y, c^*u)) = 0$. Furthermore, $\text{EMD}(\mathbf{x}, (Y, cu)) = 0$ for $c \in (0, c^*]$ since there is still enough image weight of each pattern color to match all the weight in (Y, cu). In general, we will prove that $\text{EMD}(\mathbf{x}, (Y, cu))$ decreases as c decreases and eventually becomes constant for $c \in (0, c^0]$, as shown in Figure 4.13(e). If the graph levels off at a small EMD,

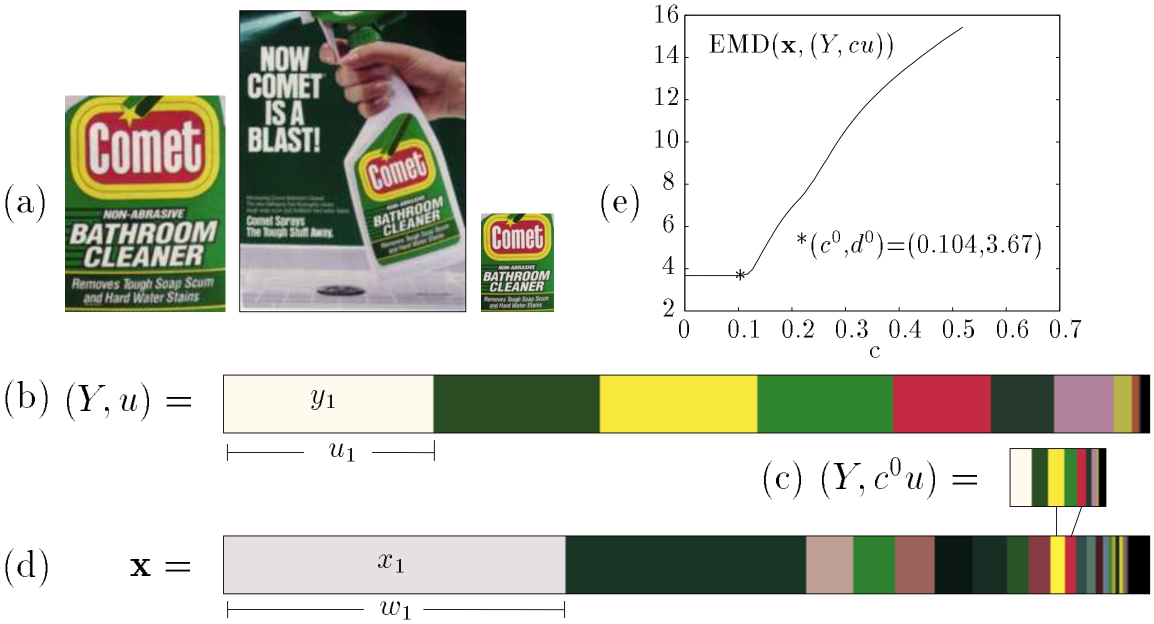


Figure 4.13: Scale Estimation – Main Idea. (a) pattern, image, and pattern scaled according to the scale estimate. (b) pattern signature. (c) pattern signature with weights scaled by the estimate. (d) image signature. (e) EMD($\mathbf{x}$, (Y, cu)) versus c.

then the pattern may occur in the image, and we take c^0 to be the scale estimate.

The main property of this scale estimation method is that in the ideal case it overestimates the scale by the *minimum* amount of background clutter over all pattern colors, where the amount of background clutter for a color is the amount of that color present in the image but not part of the pattern occurrence. Just one pattern color with a small amount of background clutter is enough to obtain an accurate scale estimate. Consider the example in Figure 4.13. The scale estimate c^0 is such that the amounts of red and yellow in the scaled pattern signature (Y, c^0u) are roughly equal to the amounts of red and yellow in the image, as shown in Figure 4.13(c). At scale c^0, there is still plenty of image weight to match the other pattern colors in (Y, c^0u). If there were a bit more red and yellow in the image, then the scale estimate c^0 would be a bit too high. In this example, red and yellow have zero background clutter since the only place that they occur in the image is within the pattern occurrence. Note that an accurate scale estimate is computed even in the presence of the dark green in the Comet label for which there is a lot of background clutter.

The preceeding discussion tacitly assumes that the pattern occurs only once in the image. Since our method does not use the positions of colors, it cannot tell the difference between two pattern occurrences at scales c_1 and c_2, and one larger occurrence at scale

$c_1 + c_2$. In this two pattern occurrence example, the computed scale estimate will be at least $c_1 + c_2$ if the same color clusters are used in $\mathbf{x}$ and $\mathbf{y}$ for the pattern colors.

We now study of the function $E(c) = \text{EMD}((X,w),(Y,cu))$, where $\mathbf{x} = (X,w)$ and $\mathbf{y} = (Y,u)$ are equal-weight distributions with total weight one, and $0 < c \leq 1$. The distribution (Y,cu) has total weight $c \leq 1$. The function $E(c)$ is thus given by

$$E(c) = \frac{\min_{(f_{ij})\in\mathcal{F}(\mathbf{x},(Y,cu))} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\, d(x_i,y_j)}{c},$$

where $(f_{ij}) \in \mathcal{F}(\mathbf{x},(Y,cu))$ iff

$$\begin{aligned} f_{ij} &\geq 0 & & i=1,\dots,m,\ j=1,\dots,n, \\ \sum_{i=1}^{m} f_{ij} &= cu_j & & j=1,\dots,n, \quad \text{and} \\ \sum_{j=1}^{n} f_{ij} &\leq w_i & & i=1,\dots,m. \end{aligned}$$

Now set $h_{ij} = f_{ij}/c$. Then

$$E(c) = \min_{(h_{ij})\in\mathcal{F}((X,\frac{1}{c}w),\mathbf{y})} \sum_{i=1}^{m}\sum_{j=1}^{n} h_{ij}\, d(x_i,y_j),$$

where $(h_{ij}) \in \mathcal{F}((X,\frac{1}{c}w),\mathbf{y})$ iff

$$h_{ij} \geq 0 \qquad i=1,\dots,m,\ j=1,\dots,n, \tag{4.17}$$

$$\sum_{i=1}^{m} h_{ij} = u_j \qquad j=1,\dots,n, \quad \text{and} \tag{4.18}$$

$$\sum_{j=1}^{n} h_{ij} \leq \frac{1}{c} w_i \qquad i=1,\dots,m. \tag{4.19}$$

Note that

$$\mathcal{F}((X,w/c_1),\mathbf{y}) \subseteq \mathcal{F}((X,w/c_2),\mathbf{y}) \iff \frac{1}{c_1} \leq \frac{1}{c_2} \iff c_2 \leq c_1. \tag{4.20}$$

Algebraically, the fact that the feasible region $\mathcal{F}((X,\frac{1}{c}w),\mathbf{y})$ increases as c decreases (and vice-versa) is because the final m constraints (4.19) involving the w_i's get weaker (stronger) as c decreases (increases). Logically, this fact make sense because the less mass that the EMD is asked to matched, the more ways there are to perform the matching.

Since $E(c)$ is a minimum over $\mathcal{F}((X,\frac{1}{c}w),\mathbf{y})$, it follows from (4.20) that $E(c)$ is a non-decreasing function of c:

$$E(c_1) \geq E(c_2) \quad \text{iff} \quad c_1 \geq c_2. \tag{4.21}$$

In fact, however, we can say something stronger than (4.21). Consider the convex polytope $Q \subseteq \mathbf{R}^{mn}$ defined by (4.17) and (4.18), and the convex polytope $P(c) \subseteq \mathbf{R}^{mn}$ defined by (4.19), so that

$$\mathcal{F}((X, w/c), \mathbf{y}) = Q \cap P(c). \tag{4.22}$$

Q is bounded since its constraints imply that $0 \le h_{ij} \le u_j$ for $i = 1, \ldots, m$, $j = 1, \ldots, n$. The polytope $P(c)$ converges to $\mathbf{R}^{mn}$ as c decreases to zero since $\frac{1}{c}$ increases to ∞. Since Q is bounded, there is some c^0 for which $Q \subseteq P(c)\ \forall c \le c^0$. From this fact and (4.22), it follows that

$$\mathcal{F}((X, w/c), \mathbf{y}) = Q \quad \forall c \le c^0,$$

and, hence,

$$E(c) = E(c^0) \quad \forall c \le c^0. \tag{4.23}$$

Thus $E(c)$ decreases as c decreases, until some point c^0 at which the curve flattens out. Examples are shown in figures 4.13(e), 4.15, and 4.16.

To help with the intuition for (4.23), consider a simple color pattern problem example. Suppose the pattern contains 30% red, 40% white, and 30% blue, and the pattern is 20% of the image. In the ideal case of perfect color matches, the image has at least 6% red, 8% white, and 6% blue due to the presence of the pattern. If the pattern is scaled by less than $c = 0.20 = 20\%$, then its distribution will contain less than 6% red, 8% white, and 6% blue, and all of this color mass can be matched perfectly to the color masses of the image. The EMD will be zero for all values $0 < c \le 0.20$.

We take as our scale estimate the largest c for which there is no real improvement in the EMD when c is decreased. What constitutes "no real improvement" in the value of the EMD is given as a parameter ε_d. There is also a parameter ε_c to specify the accuracy that is required for the scale estimate. Finally, the parameter $c_{\min}$ gives the smallest scale to be examined. The largest c for which there is no improvement in the EMD can be found via a binary search along the c-axis. See Figure 4.14. The pseudocode given below returns the scale estimate c^0, the EMD value $d^0 = \text{EMD}(\mathbf{x}, (Y, c^0 u))$, and an optimal flow flow^0 at the scale estimate.

function $[c^0, d^0, \text{flow}^0] = \text{ScaleEstimate}(\mathbf{x}, \mathbf{y}, c_{\min}, \varepsilon_c, \varepsilon_d)$
/* $\mathbf{x} = (X, w)$, $\mathbf{y} = (Y, u)$ */
/* *assumes* $w_\Sigma = u_\Sigma = 1$ */
  $c_{\max} = 1$
  $[d_{\min}, \text{flow}_{\min}] = \text{EMD}((X, w), (Y, c_{\min} u))$
  /* *loop invariant:* $c_{\min} \le c^0 \le c_{\max}$ */

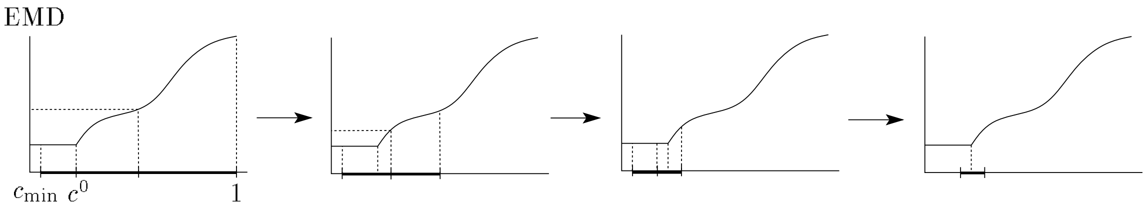


Figure 4.14: Scale Estimation Algorithm. Binary search narrows down the interval in which c^0 must occur.

while $(c_{\max} - c_{\min} > \varepsilon_c)$
  $c_{\text{mid}} = (c_{\min} + c_{\max})/2$
  $[d_{\text{mid}}, \text{flow}_{\text{mid}}] = \text{EMD}((X, w), (Y, c_{\text{mid}} u))$
  if $(|d_{\text{mid}} - d_{\min}| \leq \varepsilon_d)$
    $d_{\min} = d_{\text{mid}}$
    $c_{\min} = c_{\text{mid}}$
    $\text{flow}_{\min} = \text{flow}_{\text{mid}}$
  else
    $c_{\max} = c_{\text{mid}}$
  end if
end while
return $(c_{\min}, d_{\min}, \text{flow}_{\min})$
end function

Here $c_{\min}$ is the smallest scale pattern that the user wants to find. If the returned distance d^0 is greater than a user supplied threshold τ, then we report that the pattern does not occur within the image. Otherwise, we take c^0 as an estimate of the pattern scale within the image.

The ScaleEstimate routine requires at most $\lceil \log_2(1/\varepsilon_c) \rceil + 1$ EMD computations since the length of the initial interval $[c_{\min}, c_{\max}] = [c_{\min}, 1]$ is at most one, and this interval is cut in half after each EMD call within the while loop (the "+1" is from the initial EMD call at $c = c_{\min}$ outside the loop). If, for example, $\varepsilon_c = 0.001$, then at most 11 EMD calls are made before $|[c_{\min}, c_{\max}]| \leq \varepsilon_c$. Note that the point sets of the distributions remain constant (X and Y) throughout the execution of ScaleEstimate. Thus the cost matrix (c_{ij}), $c_{ij} = d_{ij} = d(x_i, y_j)$, for the EMD transportation problems can be computed once at the beginning of ScaleEstimate and used for all subsequent EMD computations.

If we pass the threshold τ to ScaleEstimate, then the routine can exit after the first call $\text{EMD}((X, w), (Y, c_{\min} u))$ if this quantity is greater than τ. This can happen, for example, if a large part of the pattern is red, but there is no color similar to red in the image. No matter how small the value of c, the distances that image color mass must flow in color space to cover the red pattern mass will be large in this case. Recall that the EMD is equal to the average ground distance that mass travels during an optimal flow. Since a large fraction of the total mass moved must travel large distances, the average distance moved, and hence the EMD, will be large. If $\mathrm{EMD}((X, w), (Y, c_{\min}u)) > \tau$, then ScaleEstimate performs only one EMD computation before concluding that the pattern does not occur in the image.

ScaleEstimate may also benefit in efficiency from the use of efficient, effective lower bounds on the EMD. Only the result of comparing $\mathrm{EMD}((X, w), (Y, c_{\mathrm{mid}}u))$ with the current $d_{\min}$ is needed to determine in which half of $[c_{\min}, c_{\max}]$ the scale estimate c^0 occurs. The actual value of the EMD is not needed if it can be proven by other means that $\mathrm{EMD}((X, w), (Y, c_{\mathrm{mid}}u)) > d_{\min}$ (as in the first and second frames in Figure 4.14). If so, ScaleEstimate can update $c_{\max} = c_{\mathrm{mid}}$ without performing an EMD computation. Also, if a lower bound on $\mathrm{EMD}((X, w), (Y, c_{\min}u))$ is greater than τ, then ScaleEstimate can exit without performing a single EMD computation. Whether or not ScaleEstimate runs faster using lower bounds depends on how long the lower bounds take to compute and how often they succeed in pruning an EMD computation. Lower bounds on the EMD are discussed in Chapter 5. We now shift from our discussion of efficiency issues to a more general discussion of the ScaleEstimate algorithm.

Consider the ideal case of perfectly matching features in the pattern and image. As previously mentioned, our scale estimation method overestimates the scale by the *minimum* amount of background clutter over all pattern colors in the ideal case. Suppose, for example, the pattern distribution is 50% red, 20% white, and 30% blue, and that the pattern represents 20% of the total image area. Then the image has at least 10% red, 4% white, and 6% blue from the pattern. Suppose the exact distribution of the image is 40% red, 5% white, 25% blue, 15% green, and 15% yellow. The white mass from the image will not be covered completely until the pattern distribution is scaled by $c = 0.25 = 25\%$. After this point, there will be no gain in EMD with further decreases in scale. If the image had 4% white instead of 5% white, our scale estimate would have been exactly correct at $c^0 = 0.20 = 20\%$.

In general, the reasoning is not so clear cut because corresponding parts of the pattern and image will not have exactly the same color (and even if the color matches were perfect in the original images, clustering in color space to produce the small distributions will likely destroy that perfection), and optimal matching strategies can match color mass from one color in the pattern to several colors in the image. In practice, we have observed scale estimates which are a little smaller than predicted by an ideal case analysis. This is true in the Comet example shown in Figure 4.13, where there is zero background clutter for yellow and red but the scale is slightly underestimated.

4.5.1 Experiments with the Color Pattern Problem

Figures 4.15–4.19 illustrate the performance of our scale estimation algorithm for the color pattern problem, where the patterns are product logos and the images are product advertisements.

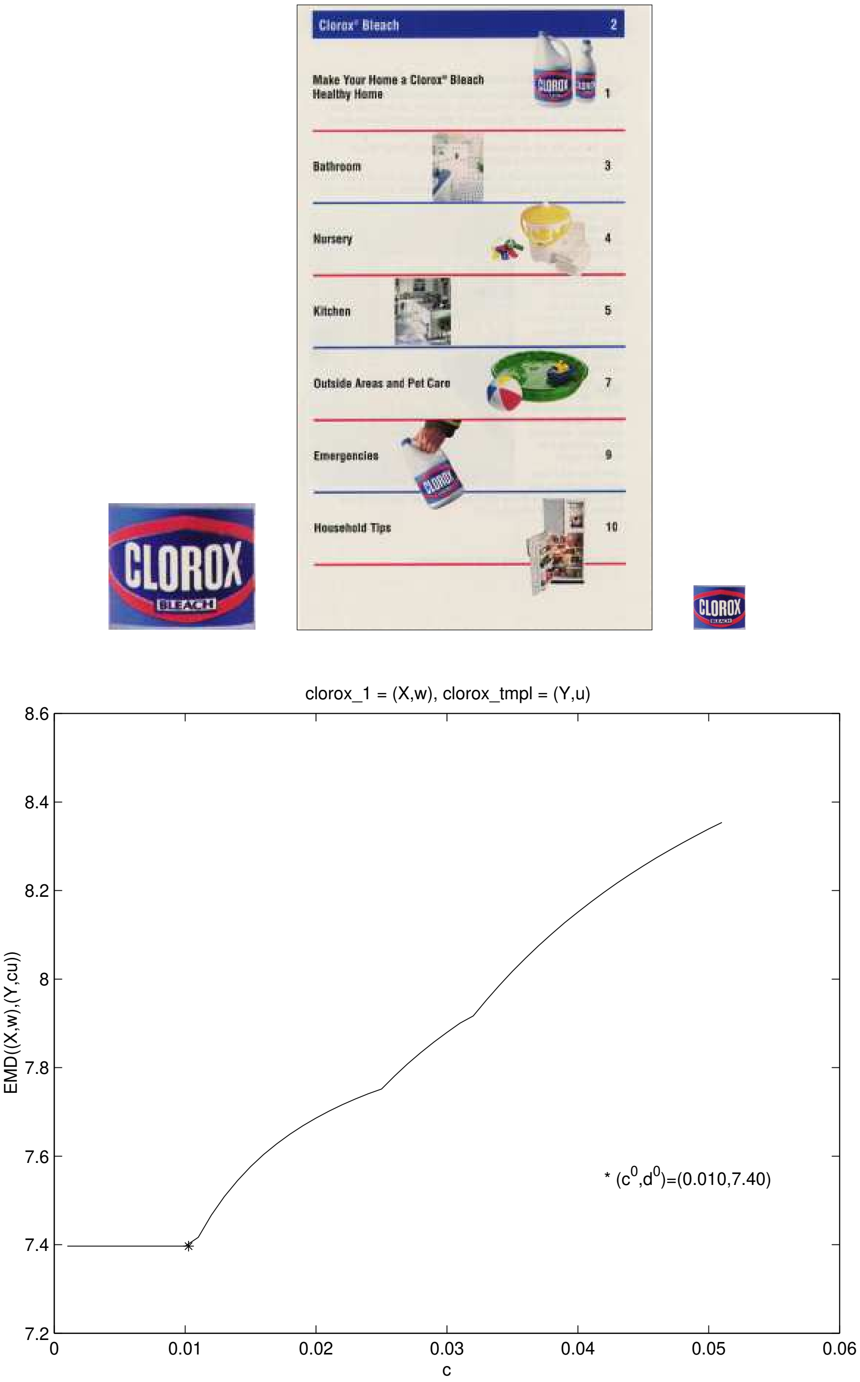


Figure 4.15: Scale Estimation – Clorox Example. From left to right in the first row, we see the Clorox logo (pattern), a Clorox advertisement, and the Clorox logo scaled according to the scale estimate given by the graph in the second row. The graph predicts that the pattern occurs at scale $c^0 = 1.0\%$ of the image. The top left, top right, and bottom Clorox logos occupy approximately 0.5%, 0.2%, and 0.5%, respectively, of the Clorox advertisement area.

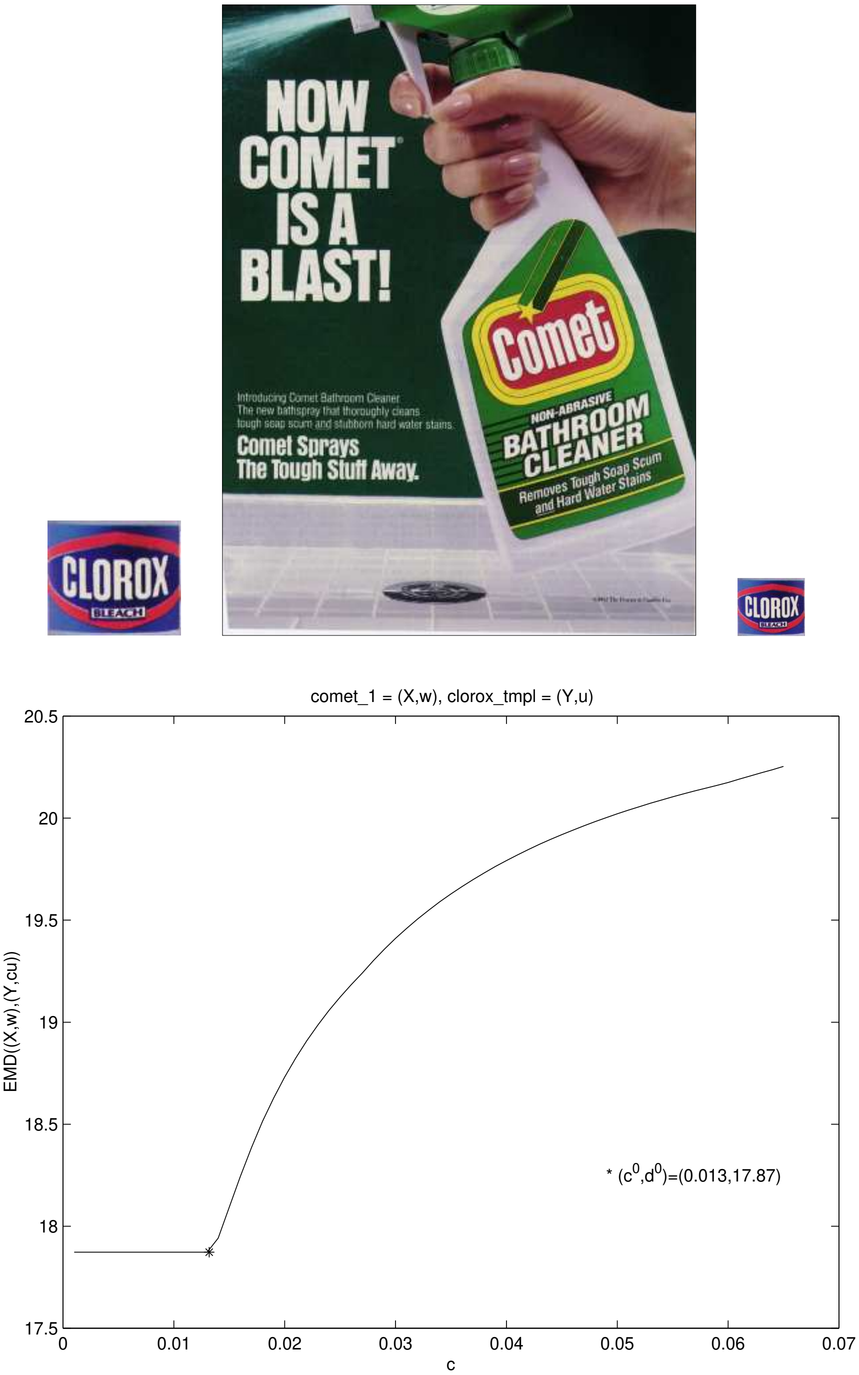


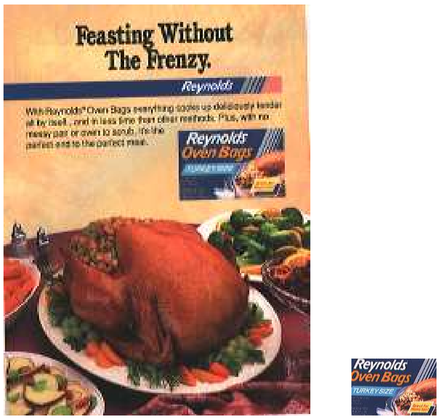
Figure 4.16: Scale Estimation – Pattern Not in the Image. From left to right in the first row, we see the Clorox logo (pattern), a Comet advertisement, and the Clorox logo scaled according to the scale estimate given by the graph in the second row. The graph predicts that the pattern occurs as $c^0 = 1.3\%$ of the image. However, the EMD at scale c^0 is $d^0 = 17.87$ units in CIE-Lab space. This large EMD value indicates that the pattern probably does *not* occur within the image.

Figure 4.17: Scale Estimation Results – Example Set 1. In each of the examples, the advertisement is shown on the left and the scaled (according to our prediction) logo is shown on the right. Let c^0 denote the predicted scale and c denote the (approximate) measured scale (in terms of fraction of advertisement area). (a) $c^0 = 2.2\%$, $c = 3.8\%$. (b) $c^0 = 4.2\%$, $c = 6.3\%$. (c) $c^0 = 2.6\%$, $c = 4.0\%$. (d) $c^0 = 6.3\%$, $c = 7.9\%$.

(a)

(b)

(c)



(d)

Figure 4.18: Scale Estimation Results – Example Set 2. In each of the examples, the advertisement is shown on the left and the scaled (according to our prediction) logo is shown on the right. Let c^0 denote the predicted scale and c denote the (approximate) measured scale (in terms of fraction of advertisement area). (a) $c^0 = 5.1\%$, $c = 3.9\%$. (b) $c^0 = 5.0\%$, $c = 6.0\%$. (c) $c^0 = 3.6\%$, $c = 4.7\%$. (d) $c^0 = 3.8\%$, $c = 5.7\%$.

Figure 4.19: Scale Estimation Results – Example Set 3. In each of the examples, the advertisement is shown on the left and the scaled (according to our prediction) logo is shown on the right. Let c^0 denote the predicted scale and c denote the (approximate) measured scale (in terms of fraction of advertisement area). (a) $c^0 = 3.8\%$, $c = 4.3\%$. (b) One of the cigarette boxes is $c = 4.0\%$ of the image. Our scale estimate $c^0 = 8.0\%$ is too large because the pattern occurs twice in the image. (c) $c^0 = 3.5\%$, $c = 2.4\%$. (d) The box of Tide occupies $c = 1.4\%$. Our scale estimate $c^0 = 3.5\%$ is too large because the pattern occurs twice in the image.

Chapter 5

Lower Bounds on the EMD

In the content-based retrieval systems described in [65] and [69], the distance between two images is taken as the EMD between the two corresponding signatures. The query time is dominated by the time to perform the EMD computations. In a nearest neighbor query, the system returns the R database images which are closest to the given query. During query processing, an exact EMD computation need not be performed if there is a lower bound on the EMD which is greater than the Rth smallest distance seen so far. The goal is to perform queries in time that grows in a sublinear fashion with the number of database images. The motivation is system scalability to very large databases.

It is known ([68]) that the distance between the centroids of two equal-weight distributions is a lower bound on the EMD between the distributions if the ground distance is induced by a vector norm. There are, however, common situations in which distributions will have unequal total weights. For example, consider once again the color-based retrieval work described in [65]. Assuming all the pixels in an image are classified, the weight of every database signature is one. EMD comparisons between unequal-weight distributions arise whenever the system is presented with a *partial* query such as: "give me all images with at least 20% sky blue and 30% green". The query signature consists of two points in CIE-Lab space with weights equal to 0.20 and 0.30, and therefore has total weight equal to 0.50. Since one cannot assume that all database images and queries will contain the same amount of information, lower bounds on the EMD between unequal-weight distributions may be quite useful in retrieval systems.

This chapter is organized as follows. In section 5.1, we extend the centroid-distance lower bound to the case of unequal-weight distributions. In section 5.2, we present lower bounds which use projections of distribution points onto random lines through the origin and along the directions of the axes. In section 5.3, we show some experiments that use our lower bounds in the previously mentioned color-based image retrieval system.

A preliminary version of most of the material in this chapter is contained in the technical report [12].

5.1 Centroid-based Lower Bounds

The centroid $\overline{\mathbf{x}}$ of the distribution $\mathbf{x} = (X, w) \in \mathbf{D}^{K,m}$ is defined as

$$\overline{\mathbf{x}} = \frac{\sum_{i=1}^{m} w_i x_i}{w_\Sigma}.$$

In section 5.1.1 we shall prove that the distance between the centroids of equal-weight distributions is a lower bound on the EMD between the distributions if the ground distance is induced by a vector norm or if $d = L_2^2$. There is also, however, a centroid-based lower bound if the distributions are not equal weight. If $\mathbf{x} = (X, w)$ is heavier than $\mathbf{y} = (Y, u)$, then all of the weight in $\mathbf{y}$ is matched to part of the weight in $\mathbf{x}$. The weight in $\mathbf{x}$ which is matched to $\mathbf{y}$ by an optimal flow is a sub-distribution $\mathbf{x}'$ of $\mathbf{x}$. Formally, a *sub-distribution* $\mathbf{x}' = (X', w')$ of $\mathbf{x} = (X, w) \in \mathbf{D}^{K,m}$, denoted $\mathbf{x}' \subset \mathbf{x}$, is a distribution with $X' = X$ and $0 \leq w' \leq w$:

$$\mathbf{x}' = \{\ (x_1, w_1'), \ldots, (x_m, w_m')\ \} = (X, w') \in \mathbf{D}^{K,m}, \qquad 0 \leq w_j' \leq w_j \text{ for } i = 1, \ldots, m.$$

In words, the points of a sub-distribution $\mathbf{x}'$ are the same as the points of $\mathbf{x}$ and the weights of $\mathbf{x}'$ are bounded by the weights of $\mathbf{x}$. One can visualize a sub-distribution $\mathbf{x}' \subset \mathbf{x}$ as the result of removing some of the dirt in the piles of dirt in $\mathbf{x}$. The minimum distance between the centroid of $\mathbf{y}$ and the locus of the centroid of sub-distributions of $\mathbf{x}$ of total weight u_Σ is a lower bound on $\mathrm{EMD}(\mathbf{x}, \mathbf{y})$. Details are given in section 5.1.2.

5.1.1 Equal-Weight Distributions

Let us first consider the case when the ground distance between points is induced by a vector norm. This is true, for example, if the ground distance is one of the L_p distances $(p \geq 1)$.

Theorem 6 *Suppose $\mathbf{x} = (X, w) \in \mathbf{D}^{K,m}$ and $\mathbf{y} = (Y, u) \in \mathbf{D}^{K,n}$ are distributions of equal total weight $w_\Sigma = u_\Sigma$. Then*

$$\mathrm{EMD}^{||\cdot||}(\mathbf{x}, \mathbf{y}) \geq ||\overline{\mathbf{x}} - \overline{\mathbf{y}}||$$

if the ground distance $d(x, y) = ||x - y||$ and $||\cdot||$ is a norm.

Proof. The equal-weight requirement implies that for any feasible flow $F = (f_{ij})$,

$$\sum_{i=1}^{m} f_{ij} = u_j \qquad \text{and} \tag{5.1}$$

$$\sum_{j=1}^{n} f_{ij} = w_i. \tag{5.2}$$

Then

$$\begin{aligned}
\left\| \sum_{i=1}^{m} w_i x_i - \sum_{j=1}^{n} u_j y_j \right\| &= \left\| \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} x_i - \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} y_j \right\| && ((5.1), (5.2)) \\
&= \left\| \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}(x_i - y_j) \right\| \\
&\leq \sum_{i=1}^{m} \sum_{j=1}^{n} ||f_{ij}(x_i - y_j)|| && (\Delta\text{-inequality}) \\
&= \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} ||x_i - y_j|| && (f_{ij} \geq 0) \\
\left\| \sum_{i=1}^{m} w_i x_i - \sum_{j=1}^{n} u_j y_j \right\| &\leq \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} ||x_i - y_j||.
\end{aligned}$$

Dividing both sides of the last inequality by $w_\Sigma = u_\Sigma$ yields

$$||\overline{\mathbf{x}} - \overline{\mathbf{y}}|| \leq \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} ||x_i - y_j||}{w_\Sigma} \tag{5.3}$$

for any feasible flow F. Replacing F by a work minimizing flow gives the desired result. ■

The centroid lower bound for the equal-weight case also holds when the ground distance is L_2^2, despite the fact that the square of the Euclidean norm is *not* itself a norm.

Theorem 7 *Suppose* $\mathbf{x} = (X, w) \in \mathbf{D}^{K,m}$ *and* $\mathbf{y} = (Y, u) \in \mathbf{D}^{K,n}$ *are distributions of equal total weight* $w_\Sigma = u_\Sigma$*. Then*

$$\mathrm{EMD}^{||\cdot||_2^2}(\mathbf{x}, \mathbf{y}) \geq ||\overline{\mathbf{x}} - \overline{\mathbf{y}}||_2^2,$$

where the ground distance $d(x, y) = ||x - y||_2^2$ *and* $||\cdot||_2$ *is the Euclidean norm.*

Proof. Applying the Cauchy-Schwarz inequality $(\sum_k a_k^2)(\sum_k b_k^2) \geq (\sum_k a_k b_k)^2$ with $a_k = \sqrt{f_{ij}}$ and $b_k = \sqrt{f_{ij}}||x_i - y_j||_2$ gives

$$\left(\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\right)\left(\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_2^2\right) \geq \left(\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_2\right)^2 \tag{5.4}$$

for every feasible flow F. The first factor on the left-hand side of (5.4) is equal to the total weight $u_\Sigma = w_\Sigma$. From (5.3) in the proof of Theorem 6, the right-hand side of the inequality is greater than or equal to $||\overline{\mathbf{x}} - \overline{\mathbf{y}}||_2^2 u_\Sigma^2$. Combining these facts with (5.4) shows that

$$\frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_2^2}{u_\Sigma} \geq ||\overline{\mathbf{x}} - \overline{\mathbf{y}}||_2^2$$

for every feasible flow F. Replacing F by an optimal feasible flow yields the desired result. ■

5.1.2 Unequal-Weight Distributions

Let $\mathbf{x} = (X, w) \in \mathbf{D}^{K,m}$ and $\mathbf{y} = (Y, u) \in \mathbf{D}^{K,n}$ be distributions with $w_\Sigma \geq u_\Sigma$. In any feasible flow $F = (f_{ij})$ from $\mathbf{x}$ to $\mathbf{y}$, all of the weight u_j must be matched to weight in $\mathbf{x}$ so that $\sum_{i=1}^{m} f_{ij} = u_j$, and the total amount of matched weight is $\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} = u_\Sigma$. Let

$$\mathbf{x}^F = \{\ (x_1, \sum_{j=1}^{n} f_{1j}), (x_2, \sum_{j=1}^{n} f_{2j}), \ldots, (x_m, \sum_{j=1}^{n} f_{mj})\ \} = (X, w^F).$$

Clearly, $w_\Sigma^F = u_\Sigma$. By Theorem 6 in the previous section, we know that

$$\mathrm{EMD}(\mathbf{x}^F, \mathbf{y}) \geq \left|\left|\overline{\mathbf{x}^F} - \overline{\mathbf{y}}\right|\right| \tag{5.5}$$

when the ground distance is induced by a vector norm $||\cdot||$. Note that Theorem 7 implies that the lower bound (5.5) also holds when $d = L_2^2$ if we replace $||\cdot||$ by $||\cdot||_2^2$.

From (5.5), it follows that

$$\mathrm{EMD}(\mathbf{x}^F, \mathbf{y}) \geq \min_{F' \in \mathcal{F}(\mathbf{x},\mathbf{y})} \left|\left|\overline{\mathbf{x}^{F'}} - \overline{\mathbf{y}}\right|\right|, \tag{5.6}$$

where the minimum is taken over all feasible flows F' from $\mathbf{x}$ to $\mathbf{y}$. Since (5.6) holds for every feasible flow F from $\mathbf{x}$ to $\mathbf{y}$, we can replace F by a work minimizing flow F^* and obtain

$$\mathrm{EMD}(\mathbf{x}, \mathbf{y}) = \mathrm{EMD}(\mathbf{x}^{F^*}, \mathbf{y}) \geq \min_{F' \in \mathcal{F}(\mathbf{x},\mathbf{y})} \left|\left|\overline{\mathbf{x}^{F'}} - \overline{\mathbf{y}}\right|\right|. \tag{5.7}$$

The minimum on the right-hand side of the inequality (5.7) can be re-stated as the minimum distance of the centroid of $\mathbf{y}$ to the centroid of any sub-distribution of $\mathbf{x}$ of total weight u_Σ:

$$\min_{F' \in \mathcal{F}(\mathbf{x},\mathbf{y})} \left|\left|\overline{\mathbf{x}^{F'}} - \overline{\mathbf{y}}\right|\right| = \min_{\substack{\mathbf{x}' = (X, w') \subset \mathbf{x} \\ w'_\Sigma = u_\Sigma}} \left|\left|\overline{\mathbf{x}'} - \overline{\mathbf{y}}\right|\right|. \tag{5.8}$$

We now argue that (5.8) holds. Clearly, $\mathbf{x}^{F'}$ is a sub-distribution of $\mathbf{x}$ with total weight u_Σ for every $F' \in \mathcal{F}(\mathbf{x},\mathbf{y})$. It remains to argue that any sub-distribution $\mathbf{x}' \subset \mathbf{x}$ with total weight u_Σ is $\mathbf{x}^{F'}$ for some $F' \in \mathcal{F}(\mathbf{x},\mathbf{y})$. Let F' be any feasible flow between the two equal-weight distributions $\mathbf{x}'$ and $\mathbf{y}$ (the set of such feasible flows is nonempty). The feasible flows $\mathcal{F}(\mathbf{x},\mathbf{y})$ between $\mathbf{x}$ and $\mathbf{y}$ are exactly those flows which match all u_Σ of $\mathbf{y}$-weight to $u_\Sigma \leq w_\Sigma$ of $\mathbf{x}$-weight. Therefore, $\mathcal{F}(\mathbf{x}',\mathbf{y}) \subset \mathcal{F}(\mathbf{x},\mathbf{y})$, and $F' \in \mathcal{F}(\mathbf{x}',\mathbf{y}) \Rightarrow F' \in \mathcal{F}(\mathbf{x},\mathbf{y})$.

Combining (5.7) and (5.8) gives

$$\mathrm{EMD}(\mathbf{x},\mathbf{y}) \geq \min_{\substack{\mathbf{x}' = (X, w') \subset \mathbf{x} \\ w'_\Sigma = u_\Sigma}} \left|\left|\overline{\mathbf{x}'} - \overline{\mathbf{y}}\right|\right|. \tag{5.9}$$

In section 5.1.2.1 we show how this minimization problem can be formulated as the minimization of a quadratic function (if $d = L_2$) subject to linear constraints. However, solving this quadratic programming problem is likely to take more time than computing the EMD itself. In section 5.1.2.2 we show how to compute a bounding box for the locus of the centroid of any sub-distribution of $\mathbf{x}$ of total weight u_Σ. The minimum ground distance from the centroid of $\mathbf{y}$ to the bounding box is a lower bound of the EMD, although it is obviously not as tight as the lower bound in (5.9).

5.1.2.1 The Centroid Lower Bound

Given a distribution $\mathbf{x} = (X, w) \in \mathbf{D}^{K,m}$, the locus of the centroid of sub-distributions of $\mathbf{x}$ of weight αw_Σ, $0 < \alpha \leq 1$, is

$$C^\alpha(\mathbf{x}) = \left\{ \frac{\sum_{i=1}^m \widetilde{w}_i x_i}{\widetilde{w}_\Sigma} \; : \; 0 \leq \widetilde{w}_i \leq w_i, \; 0 < \widetilde{w}_\Sigma = \alpha w_\Sigma \right\}.$$

If we let $v_i = \widetilde{w}_i / \widetilde{w}_\Sigma$ and $\widehat{w}_i = w_i / (\alpha w_\Sigma)$, then

$$C^\alpha(\mathbf{x}) = \left\{ \sum_{i=1}^m v_i x_i \; : \; 0 \leq v \leq \widehat{w} = \frac{1}{\alpha} \frac{w}{w_\Sigma}, \; v_\Sigma = 1 \right\}$$

or, in terms of matrix multiplication,

$$C^{\alpha}(\mathbf{x}) = \{\, Xv \;:\; 0 \le v \le \widehat{w} = \frac{1}{\alpha}\frac{w}{w_\Sigma},\; 1^T v = 1 \,\}. \tag{5.10}$$

The symbol "1" is overloaded in the constraint $1^T v = 1$; on the left-hand side it is a vector of m ones, while on the right-hand side it is simply the integer one. It is easy to see from (5.10) that

$$C^{\alpha_1}(\mathbf{x}) \supseteq C^{\alpha_2}(\mathbf{x}) \quad \text{if } \alpha_1 \le \alpha_2.$$

The locus $C^{\alpha}(\mathbf{x})$ is a convex polytope. The intersection of the $2m$ halfspaces $v \ge 0$ and $v \le \widehat{w}$ is a box P_1. The intersection of P_1 with the hyperplane $1^T v = 1$ is another convex polytope P_2 of one dimension less. Finally, applying the linear map X to P_2 gives the convex polytope $C^{\alpha}(\mathbf{x})$. In [4], Bern et al. characterize and provide algorithms to compute the locus $C_{L,H}(S)$ of the centroid of a set S of points with approximate weights, where weight w_i lies in a given interval $[l_i, h_i]$ and the total weight W is bounded as $L \le W \le H$. The locus $C^{\alpha}(\mathbf{x}) = C_{1,1}(X)$ if $[l_i, h_i] = [0, \widehat{w}_i]$.

Now suppose that $\mathbf{y} = (Y, u) \in \mathbf{D}^{K,n}$ is a lighter distribution than $\mathbf{x}$. In the previous section we argued that the EMD is bounded below by the minimum ground distance from $\overline{\mathbf{y}}$ to a point in $C^{u_\Sigma/w_\Sigma}(\mathbf{x})$. We denote this minimum distance as $\mathrm{CLOC}(\mathbf{x},\mathbf{y})$ because it uses the locus of the centroid of sub-distributions of $\mathbf{x}$ of weight u_Σ. Thus $\mathrm{EMD}(\mathbf{x},\mathbf{y}) \ge \mathrm{CLOC}(\mathbf{x},\mathbf{y})$. If $d = L_2$, then this lower bound can be computed by minimizing a quadratic objective function subject to linear constraints:

$$(\mathrm{CLOC}(\mathbf{x},\mathbf{y}))^2 = \min_{v}\; ||Xv - \overline{\mathbf{y}}||_2^2$$

subject to

$$\begin{array}{rcl} v & \ge & 0 \\ v & \le & \widehat{w} = \dfrac{1}{u_\Sigma} w \\ 1^T v & = & 1. \end{array}$$

The above minimization problem consists of m variables and $2m+1$ linear constraints which are taken directly from (5.10). It can be written more compactly as

$$(\mathrm{CLOC}(\mathbf{x},\mathbf{y}))^2 = \min_{p \in C^{u_\Sigma/w_\Sigma}(\mathbf{x})} ||p - \overline{\mathbf{y}}||_2^2, \tag{5.11}$$

where it is assumed that $u_\Sigma \le w_\Sigma$.

5.1.2.2 The Centroid Bounding Box Lower Bound

As previously mentioned, the computation of the CLOC lower bound as described in the previous section is likely to require more time than an exact EMD computation. Yet the centroid locus $C^\alpha(\mathbf{x})$ can still be very useful in finding a fast to compute lower bound on the EMD. The idea is to precompute a bounding box $B^\alpha(\mathbf{x})$ for $C^\alpha(\mathbf{x})$ for a sample of α values, say $\alpha = 0.05k$ for $k = 1, \ldots, 20$. When given a lighter query distribution $\mathbf{y}$ at query time, the minimum distance from $\overline{\mathbf{y}}$ to the bounding box $B^{\alpha_\mathbf{y}}(\mathbf{x})$ is a lower bound on $\mathrm{EMD}(\mathbf{x}, \mathbf{y})$, where $\alpha_\mathbf{y}$ is the largest sample α value which does not exceed the total weight ratio u_Σ / w_Σ (the correctness of $\alpha_\mathbf{y}$ follows from the containment property (5.14)). We call this lower bound the CBOX lower bound (the C stands for *centroid* and the *BOX* comes from *bounding box*), and it is formally defined as

$$\mathrm{CBOX}(\mathbf{x}, \mathbf{y}) = \min_{p \in B^{u_\Sigma / w_\Sigma}(\mathbf{x})} ||p - \overline{\mathbf{y}}||, \tag{5.12}$$

where, once again, it is assumed that $u_\Sigma \leq w_\Sigma$. This lower bound computation will be very fast because the bounding boxes are precomputed and the query time computation of the minimum distance of the point $\overline{\mathbf{y}}$ to the box $B^{\alpha_\mathbf{y}}(\mathbf{x})$ is a constant time operation (it is linear in the dimension K, but does not depend on the number of points in $\mathbf{x}$ or $\mathbf{y}$). When $d = L_2^2$, we replace the norm $||\cdot||$ in (5.12) with $||\cdot||_2^2$.

If we write the matrix X in terms of its rows as

$$X = \begin{bmatrix} r_1^T \\ \vdots \\ r_K^T \end{bmatrix} \in \mathbf{R}^{K \times m}, \qquad \text{then} \qquad Xv = \begin{bmatrix} r_1^T v \\ \vdots \\ r_K^T v \end{bmatrix} \in \mathbf{R}^K.$$

The computation of an axis-aligned bounding box for the centroid locus $C^\alpha(x)$ can be accomplished by solving the $2K$ linear programs

$$a_k = \min_v \; r_k^T v, \quad b_k = \max_v \; r_k^T v \qquad k = 1, \ldots, K$$

subject to

$$\begin{aligned} v &\geq 0 \\ v &\leq \widehat{w} = \frac{1}{\alpha w_\Sigma} w \\ 1^T v &= 1. \end{aligned} \tag{5.13}$$

Each of these linear programs has m variables and $2m+1$ constraints. The axis-aligned bounding box for the centroid locus $C^\alpha(\mathbf{x})$ is

$$B^\alpha(\mathbf{x}) = \prod_{k=1}^{K} [a_k, b_k].$$

As with the true centroid loci $C^\alpha(\mathbf{x})$, we have a containment property for the bounding boxes $B^\alpha(\mathbf{x})$:

$$B^{\alpha_1}(\mathbf{x}) \supseteq B^{\alpha_2}(\mathbf{x}) \quad \text{if } \alpha_1 \leq \alpha_2. \tag{5.14}$$

This fact can be verified by observing that the constraints over which the minima a_k and maxima b_k are computed get weaker as α decreases (the only constraint involving α is (5.13)). Note also that the box $B^\alpha(\mathbf{x})$ includes its "interior" so that the lower bound $\text{CBOX}(\mathbf{x},\mathbf{y})$ is zero if $\overline{\mathbf{y}}$ lies inside $B^{\alpha_\mathbf{y}}(\mathbf{x})$. Using the CBOX lower bound instead of the CLOC lower bound trades off computation speed for pruning power since the former is much faster to compute, but[1]

$$\text{EMD}(\mathbf{x},\mathbf{y}) \geq \text{CLOC}(\mathbf{x},\mathbf{y}) \geq \text{CBOX}(\mathbf{x},\mathbf{y}).$$

Nevertheless, the pruning power of the CBOX lower bound will be high when the query distribution is well-separated from many of the database distributions (which implies that the centroids will also be well-separated).

5.2 Projection-based Lower Bounds

For v on the unit sphere S^{K-1} in $\mathbf{R}^K$, the projection $\text{proj}_v(\mathbf{x})$ of the distribution $\mathbf{x} = (X, w) \in \mathbf{R}^{K,m}$ along the direction v is defined as

$$\text{proj}_v(\mathbf{x}) = \{\ (v^T x_1, w_1), (v^T x_2, w_2), \ldots, (v^T x_m, w_m)\ \} = (v^T X, w) \in \mathbf{D}^{1,m}.$$

In words, the projection along v is obtained by using the lengths of the projections of the distribution points along v and leaving the corresponding weights unchanged. The following lemma shows that the EMD between projections is a lower bound on the EMD between the original distributions. See Figure 5.1.

Lemma 3 *Let* $v \in S^{K-1}$. *Then* $\text{EMD}^{L_2}(\mathbf{x},\mathbf{y}) \geq \text{EMD}^{L_1}(\text{proj}_v(\mathbf{x}), \text{proj}_v(\mathbf{y}))$.

[1]The inequality $\text{CLOC}(\mathbf{x},\mathbf{y}) \geq \text{CBOX}(\mathbf{x},\mathbf{y})$ follows from the fact that $B^\alpha(\mathbf{x}) \supseteq C^\alpha(\mathbf{x})$ (since $B^\alpha(\mathbf{x})$ is a bounding box for $C^\alpha(\mathbf{x})$) and the definitions (5.11) and (5.12).

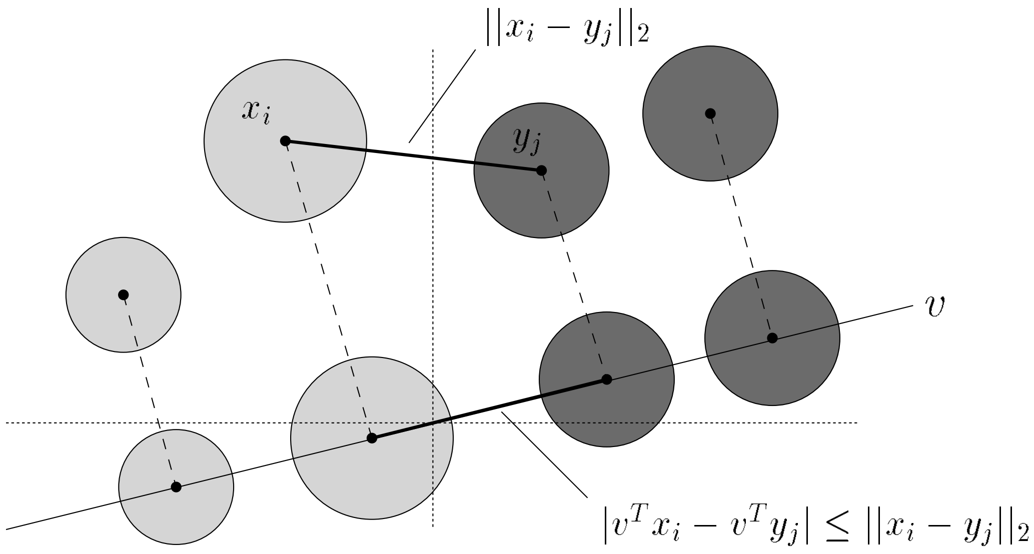


Figure 5.1: The Projection Lower Bound. The EMD with $d = L_2$ between two distributions is greater than or equal to the EMD with $d = L_1$ between the projections of the distributions onto a line through the origin. This is because all ground distances decrease or remain the same after projection. See Lemma 3 and its proof.

Proof. This theorem follows easily from the definition of the EMD and the fact that

$$\begin{aligned}
|v^T x_i - v^T y_j| &= |v^T (x_i - y_j)| \\
&= ||v||_2 \, ||x_i - y_j||_2 \, |\cos\theta_{v,(x_i - y_j)}| \\
&= ||x_i - y_j||_2 \, |\cos\theta_{v,(x_i - y_j)}| \\
|v^T x_i - v^T y_j| &\leq ||x_i - y_j||_2.
\end{aligned}$$

∎

The following theorem is an immediate consequence of Lemma 3.

Theorem 8 *Let* $V = \{v_1, \ldots, v_L\} \subset S^{K-1}$ *and*

$$\mathrm{PMAX}(V, \mathbf{x}, \mathbf{y}) = \max_{v \in V} \ \mathrm{EMD}(\mathrm{proj}_v(\mathbf{x}), \mathrm{proj}_v(\mathbf{y}))$$

Then $\mathrm{EMD}(\mathbf{x}, \mathbf{y}) \geq \mathrm{PMAX}(V, \mathbf{x}, \mathbf{y})$.

For this lower bound to be of practical use, we must be able to compute it efficiently. In section 4.3.2, we presented a straightforward, $\Theta(m + n)$ time algorithm to compute the EMD between equal-weight distributions on the real line. In combination with Theorem 8, this algorithm provides the means to compute quickly a lower bound on the EMD between two equal-weight distributions.

One pruning strategy is to pick a set of random directions V along which to perform projections, and apply Theorem 8 to obtain a lower bound. The hope is that the differences between two distributions will be captured by looking along one of the directions in V. Another pruning strategy is to use the set of orthogonal axis directions for the set V. The following corollary is an immediate consequence of Theorem 8.

Corollary 3 *Let* $E = \{e_1, \ldots, e_K\} \subset S^{K-1}$ *be the set of axis directions, and let*

$$\mathrm{PAMAX}(\mathbf{x}, \mathbf{y}) = \mathrm{PMAX}(E, \mathbf{x}, \mathbf{y}).$$

Then $\mathrm{EMD}(\mathbf{x}, \mathbf{y}) \geq \mathrm{PAMAX}(\mathbf{x}, \mathbf{y})$.

Looking along the space axes is intuitively appealing when each axis measures a specific property. For example, suppose that distribution points are points in the CIE-Lab color space ([88]). If two images are very different in terms of the luminance values of pixels, then comparing the signature projections along the L-axis will reveal this difference and allow the system to avoid an exact EMD computation.

When the projection directions are the coordinate axes, we can prove a lower bound which involves the sum of the EMDs along axis directions.

Theorem 9 *If*

$$\mathrm{PASUM}(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \mathrm{EMD}(\mathrm{proj}_{e_k}(\mathbf{x}), \mathrm{proj}_{e_k}(\mathbf{y})),$$

then $\mathrm{EMD}(\mathbf{x}, \mathbf{y}) \geq \mathrm{PASUM}(\mathbf{x}, \mathbf{y})$.

Proof. The proof uses the fact that $||a||_2 \geq (1/\sqrt{K})||a||_1$ for any vector $a \in \mathbf{R}^K$ ([25]). It follows that

$$\begin{aligned}
\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_2 &\geq \frac{1}{\sqrt{K}} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_1 \\
&= \frac{1}{\sqrt{K}} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \sum_{k=1}^{K} \left|x_i^{(k)} - y_j^{(k)}\right| \\
&= \frac{1}{\sqrt{K}} \sum_{k=1}^{K}\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \left|x_i^{(k)} - y_j^{(k)}\right| \\
\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_2 &\geq \frac{1}{\sqrt{K}} \sum_{k=1}^{K}\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \left|x_i^{(k)} - y_j^{(k)}\right|,
\end{aligned}$$

where the superscript (k) denotes the kth component of a vector. Therefore,

$$\begin{aligned}
\min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y})} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_2 &\geq \min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y})} \frac{1}{\sqrt{K}} \sum_{k=1}^{K}\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\left|x_i^{(k)} - y_j^{(k)}\right| \\
&\geq \frac{1}{\sqrt{K}}\sum_{k=1}^{K} \min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y})} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\left|x_i^{(k)} - y_j^{(k)}\right| \\
&= \frac{1}{\sqrt{K}}\sum_{k=1}^{K} (\min(w_\Sigma, u_\Sigma) \times \mathrm{EMD}(\mathrm{proj}_{e_k}(\mathbf{x}), \mathrm{proj}_{e_k}(\mathbf{y}))) \\
&= \frac{1}{\sqrt{K}} \min(w_\Sigma, u_\Sigma) \sum_{k=1}^{K} \mathrm{EMD}(\mathrm{proj}_{e_k}(\mathbf{x}), \mathrm{proj}_{e_k}(\mathbf{y})) \\
\min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y})} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}||x_i - y_j||_2 &\geq \frac{1}{\sqrt{K}} \min(w_\Sigma, u_\Sigma) \sum_{k=1}^{K} \mathrm{EMD}(\mathrm{proj}_{e_k}(\mathbf{x}), \mathrm{proj}_{e_k}(\mathbf{y})).
\end{aligned}$$

Dividing both sides of the last inequality by $\min(w_\Sigma, u_\Sigma)$ gives the desired result. ■

Note that $\mathrm{PASUM}(\mathbf{x},\mathbf{y})$ may be rewritten as

$$\mathrm{PASUM}(\mathbf{x},\mathbf{y}) = \sqrt{K}\left(\frac{\sum_{k=1}^{K} \mathrm{EMD}(\mathrm{proj}_{e_k}(\mathbf{x}), \mathrm{proj}_{e_k}(\mathbf{y}))}{K}\right).$$

This alternate expression makes it clear that $\mathrm{PASUM}(\mathbf{x},\mathbf{y})$ is a better lower bound than $\mathrm{PAMAX}(\mathbf{x},\mathbf{y})$ iff the square root of the dimension times the average axis projection distance is greater than the maximum axis projection distance.

Our projection bounds require EMD computations between distributions on the real line. In section 4.3.2, we gave a very efficient algorithm to compute the EMD between equal-weight distributions (with the L_1-distance as the ground distance). If the distributions have different total weight, we must fall back on the transportation simplex method to compute the 1D EMD. Using arguments similar to those used in section 4.3.2, we can, however, compute a lower bound on the EMD between unequal-weight distributions on the line. The idea to determine intervals over which certain amounts of mass must flow in any feasible flow.

Once again consider the interval (r_k, r_{k+1}), and WLOG assume $w_\Sigma > u_\Sigma$ and that $\mathbf{x}$-weight is moved to match all the $\mathbf{y}$-weight. When there is more $\mathbf{x}$-weight than $\mathbf{y}$-weight in both $(-\infty, r_k]$ and $[r_{k+1}, \infty)$, then there will be feasible flows in which no $\mathbf{x}$-weight travels through (r_k, r_{k+1}). If there is more $\mathbf{x}$-weight than $\mathbf{y}$-weight in $(-\infty, r_k]$, but less $\mathbf{x}$-weight than $\mathbf{y}$-weight in $[r_{k+1}, \infty)$, then $(u_\Sigma - U(r_k)) - (w_\Sigma - W(r_k))$ of the $\mathbf{x}$-weight must be moved from r_k to r_{k+1} in order to cover the $\mathbf{y}$-weight in $[r_{k+1}, \infty)$. See Figure 5.2(a). If there is less $\mathbf{x}$-weight than $\mathbf{y}$-weight in $(-\infty, r_k]$, but more $\mathbf{x}$-weight than $\mathbf{y}$-weight in $[r_{k+1}, \infty)$,

(a)

(b)

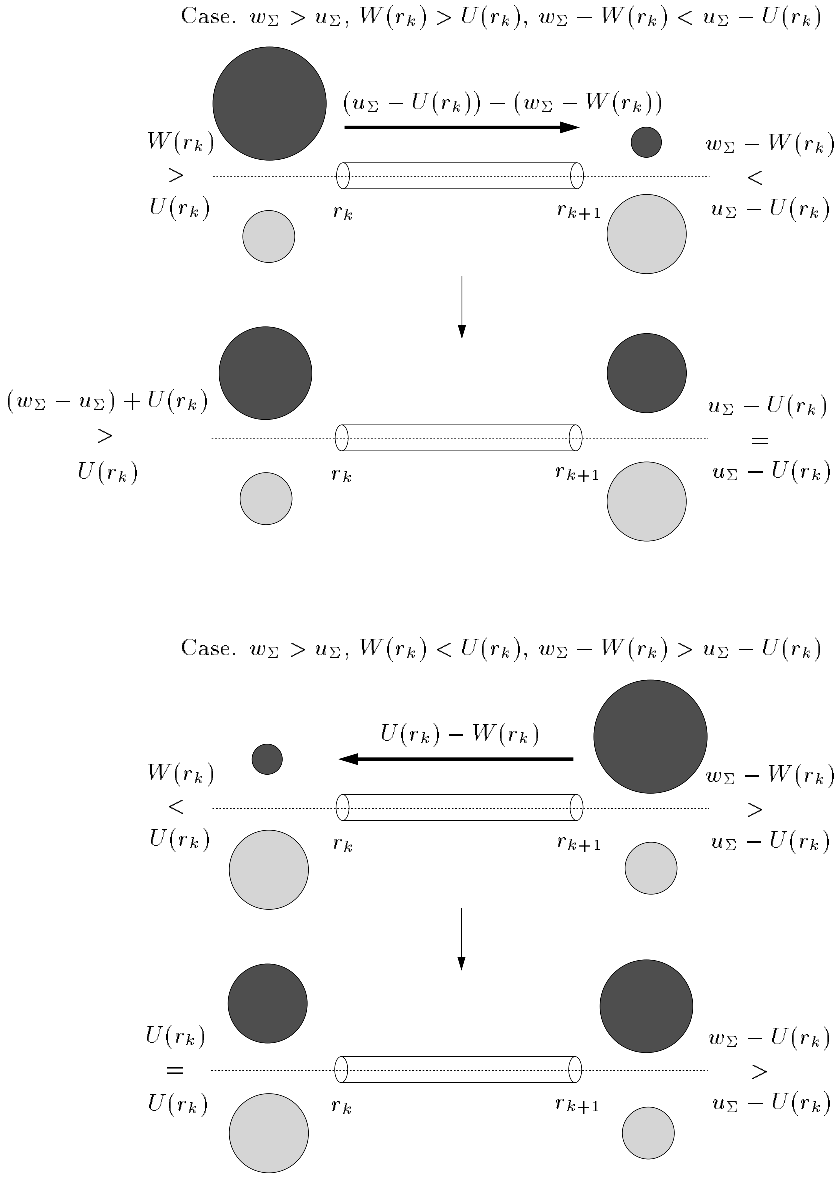


Figure 5.2: Flow Feasibility for Unequal-Weight Distributions on the Real Line. $\mathbf{x} = (X, w)$ and $\mathbf{y} = (Y, u)$ are distributions in 1D with $w_\Sigma > u_\Sigma$. All $\mathbf{y}$-weight must be covered by $\mathbf{x}$-weight. (a) $W(r_k) > U(r_k)$, $w_\Sigma - W(r_k) < u_\Sigma - U(r_k)$. In any feasible flow from $\mathbf{x}$ to $\mathbf{y}$, at least $(w_\Sigma - W(r_k)) - (u_\Sigma - U(r_k))$ of $\mathbf{x}$-weight must travel from r_k to r_{k+1} during the flow. (b) $W(r_k) < U(r_k)$, $w_\Sigma - W(r_k) > u_\Sigma - U(r_k)$. In any feasible flow from $\mathbf{x}$ to $\mathbf{y}$, at least $U(r_k) - W(r_k)$ of $\mathbf{x}$-weight must travel from r_{k+1} to r_k during the flow.

then $U(r_k) - W(r_k)$ of the $\mathbf{x}$-weight must be moved from r_{k+1} to r_k in order to cover the $\mathbf{y}$-weight in $(-\infty, r_k]$. This case is illustrated in Figure 5.2(b). Under the assumption that $w_\Sigma > u_\Sigma$, it *cannot* be the case that there is less $\mathbf{x}$-weight than $\mathbf{y}$-weight in both $(-\infty, r_k]$ and $[r_{k+1}, \infty)$.

Pseudocode for the lower bound described in the previous paragraph is given below. The routine is named FSBL because the lower bound follows simply from flow feasibility (FeaSiBiLity) conditions.

```
function emdlb = FSBL(x, y)
/* assumes K = 1, w_Σ ≥ u_Σ, ground distance is L_1 */
/* assumes x_1 ≤ x_2 ≤ ··· ≤ x_m, y_1 ≤ y_2 ≤ ··· ≤ y_n */
   work = wcumsum = ucumsum = r = 0
   /* first increment of work will be 0, regardless of r */
   wsum = Σ_{i=1}^{m} w_i
   usum = Σ_{j=1}^{n} u_j
   i = j = 1
   xnext = x_1
   ynext = y_1
   while ((i ≤ m) or (j ≤ n))
      next = min(xnext,ynext)
      if (usum-ucumsum > wsum-wcumsum)
         work += ((usum-ucumsum)-(wsum-wcumsum))*(next-r)
      elseif (ucumsum > wcumsum)
         work += (ucumsum-wcumsum)*(next-r)
      end if
      if (xnext ≤ ynext)
         wcumsum += w_i
         i += 1
         xnext = (i ≤ m) ? x_i : ∞
      else
         ucumsum += u_j
         j += 1
         ynext = (j ≤ n) ? y_j : ∞
      end if
      r = next
   end while
   return (work / usum)
end function
```

We have argued that

Theorem 10 *If* $\mathbf{x}$ *and* $\mathbf{y}$ *are distributions on the real line, then* $\mathrm{EMD}(\mathbf{x}, \mathbf{y}) \geq \mathrm{FSBL}(\mathbf{x}, \mathbf{y})$.

If $w_\Sigma = u_\Sigma$, then $(u_\Sigma - U(r_k) > w_\Sigma - W(r_k)) \equiv (W(r_k) > U(r_k))$, $(u_\Sigma - U(r_k)) - (w_\Sigma - W(r_k)) = W(r_k) - U(r_k)$, and the routine computes the exact value $\mathrm{EMD}(\mathbf{x}, \mathbf{y})$.

Theorem 11 *If* $\mathbf{x}$ *and* $\mathbf{y}$ *are equal-weight distributions on the real line, then* $\mathrm{EMD}(\mathbf{x}, \mathbf{y}) = \mathrm{FSBL}(\mathbf{x}, \mathbf{y})$.

Assuming that the points in $\mathbf{x} \in \mathbf{D}^{1,m}$ and $\mathbf{y} \in \mathbf{D}^{1,n}$ are in sorted order, the routine FSBL runs in linear time $\Theta(m+n)$. The combined sorted list $r_1, \ldots, r_{m+n}$ of points in $\mathbf{x}$ and $\mathbf{y}$ is discovered by walking along the two sorted lists of points. At any time during the algorithm, there is a pointer to the next $\mathbf{x}$ and next $\mathbf{y}$ value to be considered. The value r_{k+1} then follows in constant time from the value of r_k.

The FSBL lower bound may be substituted for the EMD function in the PMAX, PAMAX, and PASUM lower bounds to obtain efficient to compute, projection-based lower bounds

$$\begin{aligned} \mathrm{PMAX}_{\mathrm{FSBL}}(V, \mathbf{x}, \mathbf{y}) &= \max_{v \in V} \mathrm{FSBL}(\mathrm{proj}_v(\mathbf{x}), \mathrm{proj}_v(\mathbf{y})) \\ &= \mathrm{PMAX}(V, \mathbf{x}, \mathbf{y}) \qquad \text{when } w_\Sigma = u_\Sigma \end{aligned}$$

$$\begin{aligned} \mathrm{PAMAX}_{\mathrm{FSBL}}(\mathbf{x}, \mathbf{y}) &= \max_{k=1,\ldots,K} \mathrm{FSBL}(\mathrm{proj}_{e_k}(\mathbf{x}), \mathrm{proj}_{e_k}(\mathbf{y})) \\ &= \mathrm{PAMAX}(\mathbf{x}, \mathbf{y}) \qquad \text{when } w_\Sigma = u_\Sigma \end{aligned}$$

$$\begin{aligned} \mathrm{PASUM}_{\mathrm{FSBL}}(\mathbf{x}, \mathbf{y}) &= \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \mathrm{FSBL}(\mathrm{proj}_{e_k}(\mathbf{x}), \mathrm{proj}_{e_k}(\mathbf{y})) \\ &= \mathrm{PASUM}(\mathbf{x}, \mathbf{y}) \qquad \text{when } w_\Sigma = u_\Sigma \end{aligned}$$

in which $\mathbf{x}$ and $\mathbf{y}$ are not necessarily equal weight. The second equality in each of the three pairs of equalities follows directly from Theorem 11 and the definitions of $\mathrm{PMAX}(V, \mathbf{x}, \mathbf{y})$, $\mathrm{PAMAX}(\mathbf{x}, \mathbf{y})$, and $\mathrm{PASUM}(\mathbf{x}, \mathbf{y})$.

5.3 Experiments in Color-based Retrieval

In this section, we show some results of using the lower bounds CBOX, $\mathrm{PMAX}_{\mathrm{FSBL}}$, $\mathrm{PAMAX}_{\mathrm{FSBL}}$, and $\mathrm{PASUM}_{\mathrm{FSBL}}$ in the color-based retrieval system described in [65]. This system summarizes an image by a distribution of dominant colors in the CIE-Lab color space, where the weight of a dominant color is equal to the fraction of image pixels which are classified as that color. The input to the system is a query and a number R of nearest images to return. The system computes the EMD between the query distribution and each of the database distributions. The ground distance is $d = L_2$.

If the query is a full image (e.g. an image in the database), then the query distribution and all the database distributions will have total weight equal to one. In this query-by-example setting, the system first checks the distance between distribution centroids before performing an exact EMD computation. If the centroid distance is larger than the Rth smallest distance seen before the current comparison, then the system does not compute the EMD and simply considers the next database image. An R-nearest neighbor database image to the query cannot be missed by this algorithm because the centroid distance is a lower bound on the EMD between equal-weight distributions. When the query is a partial query (such as "give me all the images with at least 20% sky blue"), the system in [65] performs an exact EMD computation between the query and every database image.

To use the CBOX lower bound for partial queries, some additional preprocessing is needed. At database entry time, the distribution $\mathbf{x} = (X, w)$ of an image is computed and stored, as well as the centroid bounding boxes $B^{\alpha}(\mathbf{x})$ for $\alpha = 0.05k$, $k = 1, \ldots, 20$. Given a query distribution $\mathbf{y} = (Y, u)$ of weight $u_{\Sigma} \leq w_{\Sigma}$, let $\alpha_{\mathbf{y}}$ denote the largest sample α value which does not exceed the total weight ratio u_{Σ}/w_{Σ}. The system computes the distance between $\overline{\mathbf{y}}$ and the nearest point in $B^{\alpha_{\mathbf{y}}}(\mathbf{x})$. This is the CBOX lower bound. To use the $\mathrm{PMAX}_{\mathrm{FSBL}}$ lower bound, a set V of L (specified later) random projection directions and the L position-sorted projections of each database distribution along the directions in V are computed and stored at database load time. At query time, the query distribution is also projected along the directions in V. To use the $\mathrm{PAMAX}_{\mathrm{FSBL}}$ and $\mathrm{PASUM}_{\mathrm{FSBL}}$ lower bounds, the K position-sorted projections of each database distribution along the space axes are computed and stored at database entry time. At query time, the same axis projections are performed on the query distribution.

There are many factors that affect the performance of our lower bounds. The most obvious is the database itself. Here, we use a Corel database of 20000 color images which is dominated by outdoor scenes. The order in which the images are compared to the query is also important. If the most similar images to a query are processed first, then the Rth smallest distance seen will be relatively small when the dissimilar images are processed, and relatively weak lower bounds can prune these dissimilar images. Of course, the purpose of the query is to discover the similar images. Nonetheless, a random order of comparison may help ensure good performance over a wide range of queries. Moreover, if a certain type of query is more likely than others, say, for example, queries with large amounts of blue and green (to retrieve outdoor images containing sky and grass), then it would be wise to pre-determine a good comparison order to use for such queries. In the results that follow, however, the comparison order is the same for all queries, and the order is *not* specialized for any particular type of query.

The number R of nearest images to return is yet another factor. For a fixed comparison order and query, the number of exact EMD calculations pruned is inversely related to the size of R. This is because the Rth smallest distance (against which a lower bound is compared) after comparing a fixed number images is an increasing function of R. In all the upcoming experiments, the number of nearest images returned is fixed at $R = 20$. In terms of the actual lower bounds, a system may be able to achieve better query times by using more than one bound. For example, a system might apply the CBOX lower bound first, followed by the more expensive $\text{PASUM}_{\text{FSBL}}$ bound if CBOX fails, followed by an even more expensive exact EMD computation if $\text{PASUM}_{\text{FSBL}}$ also fails. The hope is that the lower bound hierarchy of CBOX, $\text{PASUM}_{\text{FSBL}}$, and EMD speeds up query times in much the same way that the memory hierarchy of primary cache, secondary cache, and main memory speeds up memory accesses. Our experiments, however, apply one lower bound per query. For the $\text{PMAX}_{\text{FSBL}}$ lower bound, the number L of random directions must be specified. This parameter trades off between pruning power and computation speed. The more directions, the greater the pruning power, but the slower the computation. In our work, we use the heuristic $L = 2K$ (without quantifiable justification), where K is the dimension of the underlying point space (so $L = 6$ in the color-based system).

All experiments were conducted on an SGI Indigo^2 with a 250 MHz processor, and query times are reported in seconds (s). The exact EMD is computed by the transportation simplex method as described by Hillier and Lieberman in [32]. The color signature of a typical database image has eight to twelve points. The time for an EMD calculation between two such images varies roughly between half a millisecond and one millisecond (ms). The EMD computation time increases with the number of points in the distributions, so EMD computations involving a partial query distribution with only a few points are, in general, faster than EMD computations between two database images. The time for an EMD computation between a database image and a partial query with three or fewer points is typically about 0.25ms.

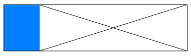
We begin our experiments with a few very simple queries. Each of these queries consists of a distribution with exactly one color point in CIE-Lab space. The results of the three queries

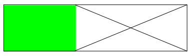
(a) 

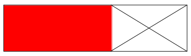
(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	2.210
CBOX	19675	0.193
$\mathrm{PMAX}_{\mathrm{FSBL}}$	19715	0.718
$\mathrm{PAMAX}_{\mathrm{FSBL}}$	19622	0.441
$\mathrm{PASUM}_{\mathrm{FSBL}}$	18969	0.536

Figure 5.3: Query C.1.1 – 20% Blue. (a) query results. (b) query statistics.

C.1.1 at least 20% (sky) blue  ,

C.1.2 at least 40% green  , and

C.1.3 at least 60% red 

are shown in Figure 5.3, Figure 5.4, and Figure 5.5, respectively. In these examples, all the lower bounds result in query times which are less than the brute force query time, and avoid a large fraction of exact EMD computations. The CBOX and $\mathrm{PASUM}_{\mathrm{FSBL}}$ bounds gave the best results on these three queries.

The next set of examples consists of randomly generated partial queries. The results for the five queries

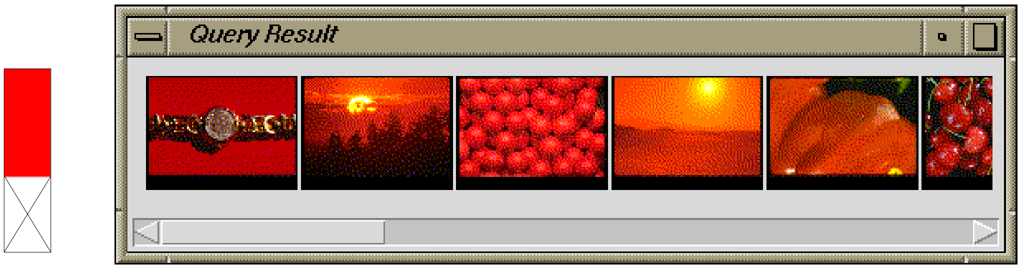
(a)

(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	3.043
CBOX	19634	0.233
$\text{PMAX}_{\text{FSBL}}$	10172	2.552
$\text{PAMAX}_{\text{FSBL}}$	16222	1.124
$\text{PASUM}_{\text{FSBL}}$	18424	0.754

Figure 5.4: Query C.1.2 – 40% Green. (a) query results. (b) query statistics.

(a)

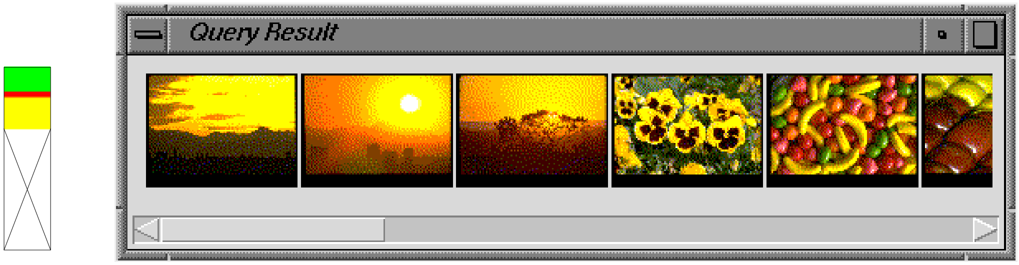


(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	2.920
CBOX	19621	0.240
$\text{PMAX}_{\text{FSBL}}$	15903	1.505
$\text{PAMAX}_{\text{FSBL}}$	17125	0.871
$\text{PASUM}_{\text{FSBL}}$	18182	0.785

Figure 5.5: Query C.1.3 – 60% Red. (a) query results. (b) query statistics.

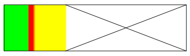
(a)

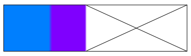


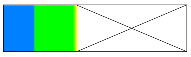
(b)

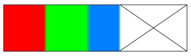
Lower Bound	# Pruned	Query Time (s)
NONE	0	4.240
CBOX	18704	0.496
$\text{PMAX}_{\text{FSBL}}$	17989	1.323
$\text{PAMAX}_{\text{FSBL}}$	17784	1.035
$\text{PASUM}_{\text{FSBL}}$	18418	0.832

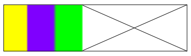
Figure 5.6: Query C.2.1 – 13.5% Green, 3.4% Red, 17.8% Yellow. The total weight of the query is $u_\Sigma = 34.7\%$. (a) query results. (b) query statistics.

C.2.1 13.5% green, 3.4% red, 17.8% yellow  ,

C.2.2 26.0% blue, 19.7% violet  ,

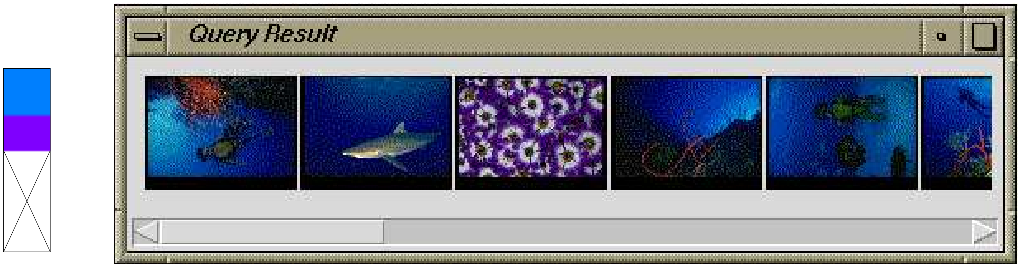
C.2.3 16.8% blue, 22.2% green, 1.8% yellow  ,

C.2.4 22.8% red, 24.2% green, 17.3% blue  , and

C.2.5 13.2% yellow, 15.3% violet, 15.3% green 

are shown in Figure 5.6 through Figure 5.10, respectively. The CBOX lower bound gives the best results for queries C.2.1 and C.2.2, but its performance drops by an order of magnitude for C.2.3, and it is completely ineffective for C.2.4 and C.2.5. Indeed, the CBOX lower bound pruned only 1 of 20000 database images for query C.2.5. The CBOX behavior can be explained in part by the locations of centroids of the query distributions and the database distributions. See Figure 5.11. Roughly speaking, the effectiveness of the CBOX bound is directly related to the amount of separation between the database distributions and the query distribution, with larger separation implying a more effective bound. The

(a)



(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	3.812
CBOX	18631	0.453
$\text{PMAX}_{\text{FSBL}}$	16472	1.452
$\text{PAMAX}_{\text{FSBL}}$	17032	1.010
$\text{PASUM}_{\text{FSBL}}$	17465	1.037

Figure 5.7: Query C.2.2 – 26.0% Blue, 19.7% Violet. The total weight of the query is $u_\Sigma = 45.7\%$. (a) query results. (b) query statistics.

(a)

(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	4.073
CBOX	1631	3.999
$\text{PMAX}_{\text{FSBL}}$	10550	3.235
$\text{PAMAX}_{\text{FSBL}}$	11690	2.648
$\text{PASUM}_{\text{FSBL}}$	15386	1.612

Figure 5.8: Query C.2.3 – 16.8% Blue, 22.2% Green, 1.8% Yellow. The total weight of the query is $u_\Sigma = 40.8\%$. (a) query results. (b) query statistics.

(a)

(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	3.969
CBOX	26	4.158
$\text{PMAX}_{\text{FSBL}}$	3606	4.342
$\text{PAMAX}_{\text{FSBL}}$	3399	4.010
$\text{PASUM}_{\text{FSBL}}$	12922	2.324

Figure 5.9: Query C.2.4 – 22.8% Red, 24.2% Green, 17.3% Blue. The total weight of the query is $u_\Sigma = 64.3\%$. (a) query results. (b) query statistics.

(a)

(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	3.375
CBOX	1	3.560
$\text{PMAX}_{\text{FSBL}}$	9608	2.924
$\text{PAMAX}_{\text{FSBL}}$	10716	2.381
$\text{PASUM}_{\text{FSBL}}$	15562	1.492

Figure 5.10: Query C.2.5 – 13.2% Yellow, 15.3% Violet, 15.3% Green. The total weight of the query is $u_\Sigma = 43.8\%$. (a) query results. (b) query statistics.

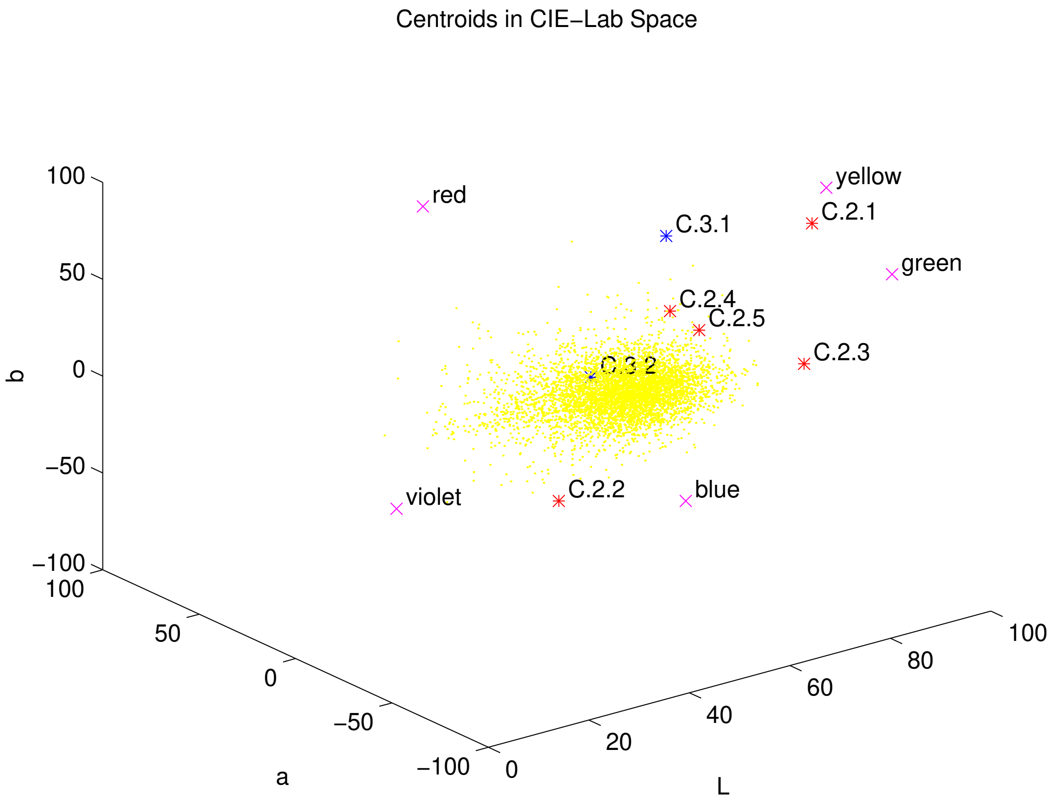
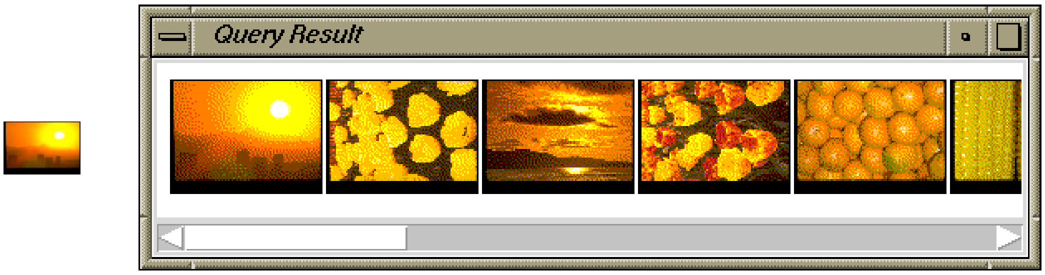


Figure 5.11: Distribution Centroids for Corel Database Images and Example Queries. The centroids of the color signature distributions of a random subset of 5000 images in the Corel database are plotted as dots, and the centroids for the queries C.2.* and C.3.* are plotted as stars. The locations of blue (C.1.1), green (C.1.2), red (C.1.3), yellow, and violet are plotted as x's.

query C.2.1 consists almost entirely of green and yellow. As one can see from Figure 5.11, the centroid of C.2.1 is very isolated from the database centroids. The approximately equal amounts red, green, and blue in query C.2.4 result in a centroid which is close to a large number of database centroids. The same statement holds for query C.2.5 which has green and yellow in one corner of the CIE-Lab space, and violet at the opposite corner.

The distances of the centroids for C.2.2 and C.2.3 to the database centroids are (i) about the same, and (ii) are smaller than the distance for C.2.1 and larger than the distances for C.2.4 and C.2.5. Observation (ii) helps explain why the performance of CBOX on C.2.2 and C.2.3 is worse than the performance on C.2.1, but better than the performance on C.2.4 and C.2.5. Observation (i) might lead one to believe that the CBOX performance should be about the same on C.2.2 and C.2.3. The statistics, however, show that this is not the case. To understand why, we must remember that the queries are partial queries. The relevant quantity is not the centroid of a database distribution, but rather the locus of the centroid of all sub-distributions with weight equal to the weight of the query. Consider images with significant amounts of blue and green, and other colors which are distant from blue and green (such as red). The other colors will help move the distribution centroid away from blue and green. However, a sub-distribution of such an image which contains only blue and green components will have a centroid which is close to blue and green, and hence close to the centroid of C.2.3. The distance between the query centroid and this image centroid may be large, but the CBOX lower bound will be small (and, hence, weak). From Figure 5.11 and the results of C.2.2 and C.2.3, one can infer that there are many more images that contain blue, green, and significant amounts of distant colors from blue and green than there are images that contain blue, violet, and significant amounts of distant colors from blue and violet. The centroid is a measure of the (weighted) average color in a distribution, and the average is not an accurate representative of a distribution with high variance (i.e. with colors that span a large portion of the color space).

The projection-based lower bounds $\mathrm{PMAX}_{\mathrm{FSBL}}$, $\mathrm{PAMAX}_{\mathrm{FSBL}}$, $\mathrm{PASUM}_{\mathrm{FSBL}}$ compare two distributions by comparing the distributions projected along some set of directions. There is hope that these bounds will help when the CBOX bound is ineffective. In queries C.2.3, C.2.4, and C.2.5, the projection-based lower bounds prune far more EMD calculations than the CBOX bound. However, pruning a large number of EMD calculations does *not* guarantee a smaller query time than achievable by brute force because of the overhead of computing a lower bound when it fails to prune an EMD calculation. In all the random partial queries C.2.*, the query times for $\mathrm{PMAX}_{\mathrm{FSBL}}$, $\mathrm{PAMAX}_{\mathrm{FSBL}}$, and $\mathrm{PASUM}_{\mathrm{FSBL}}$ were less than the query times for brute force processing, except for the $\mathrm{PMAX}_{\mathrm{FSBL}}$ and $\mathrm{PAMAX}_{\mathrm{FSBL}}$ bounds in query C.2.4. In particular, the $\mathrm{PASUM}_{\mathrm{FSBL}}$ bound performed very

(a) 

(b)

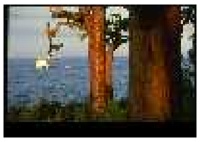
Lower Bound	# Pruned	Query Time (s)
NONE	0	15.768
CBOX	19622	0.535
$\mathrm{PMAX}_{\mathrm{FSBL}}$	19635	1.522
$\mathrm{PAMAX}_{\mathrm{FSBL}}$	19548	1.062
$\mathrm{PASUM}_{\mathrm{FSBL}}$	18601	1.847

Figure 5.12: Query C.3.1 – Sunset Image. (a) query results. (b) query statistics.

well for all the queries. Since the projection-based lower bounds are more expensive to compute than the CBOX lower bound, they must prune more exact EMD calculations than CBOX in order to be as effective in query time.

The queries in the final two examples of this section are both images in the Corel database. The results of the queries

C.3.1  and

C.3.2 

are shown in Figure 5.12 and Figure 5.13, respectively. The distributions for queries C.3.1 and C.3.2 contain 12 and 13 points, respectively. Notice that the brute force query time for the C.3.* queries is much greater than the brute force query time for the C.1.* and C.2.* queries. The difference is that both the query and the database images have a "large" number of points for the C.3.* queries. All the lower bounds perform well for query C.3.1, but the CBOX lower bound gives the lowest query time. Recall that the CBOX lower bound reduces to the distance between distribution centroids for equal-weight distributions. The centroid distance pruned many exact EMD calculations for C.3.1 because most of the

(a)

(b)

Lower Bound	# Pruned	Query Time (s)
NONE	0	14.742
CBOX	9571	8.106
$\text{PMAX}_{\text{FSBL}}$	15094	5.893
$\text{PAMAX}_{\text{FSBL}}$	13461	6.741
$\text{PASUM}_{\text{FSBL}}$	17165	3.343

Figure 5.13: Query C.3.2 – Image with Trees, Grass, Water, and Sky. (a) query results. (b) query statistics.

weight in the distribution is around yellow and orange, far from the centroids of the database images (as one can see in Figure 5.11). The blue, green, and brown in query C.3.2 span a larger part of the color space than the colors in C.3.1, the query centroid is close to many database centroids (once again, see Figure 5.11), and the centroid distance lower bound does not perform as well as for C.3.1. The projection-based lower bounds, however, each give a better query time for query C.3.2 than the centroid-distance bound. Recall that the lower bounds $\text{PMAX}_{\text{FSBL}}$, $\text{PAMAX}_{\text{FSBL}}$, and $\text{PASUM}_{\text{FSBL}}$ reduce to the stronger lower bounds PMAX, PAMAX, and PASUM for equal-weight distributions. The $\text{PASUM}_{\text{FSBL}}$ lower bound yields a tolerable query time for query C.3.2.

Chapter 6

The EMD under Transformation Sets

A major challenge in image retrieval applications is that the images we desire to match can be visually quite different. This can happen even if these images are views of the same scene because of illumination changes, viewpoint motion, occlusions, etc.. Consider for example, recognizing objects by their color signatures. A direct comparison of color histograms or an EMD between color signatures of imaged objects does not account for lighting differences. In [28], Healey and Slater show that an illumination change results in a linear transformation of the image pixel colors (under certain reasonable assumptions). In a similar result, sensor measurements of multispectral satellite images recorded under different illumination and atmospheric conditions differ by an affine transformation ([27]).

The general problem of comparing features modulo some transformation set also arises in texture-based and shape-based image retrieval. In [69], the texture content signature of a single texture image is a collection of spatial frequencies, where each frequency is weighted by the amount of energy at that frequency. If the frequency features are represented in log-polar coordinates, then scaling the texture results in a feature translation along the log-scale axis, while rotating the texture results in a feature translation along the cyclic orientation axis. These translations must be taken into account by a texture distance measure which is invariant to scaling and/or rotation.

For shape-based retrieval, suppose we summarize the shape content of an image as a collection of curves or feature points in the image. Then changes in viewpoint and/or viewing distances will result in changes in the coordinates of the extracted features, even though we are looking at an image of the same scene. Allowing for a transformation is necessary, for example, in matching point features in stereo image pairs. Even if the viewpoint and viewing distance for two images of the same scene are roughly the same, the images may have been acquired at different resolutions or drawn with different drawing programs which use different units and have different origin points. Direct comparison of summary feature coordinates is not likely to capture the visual similarity between the underlying images.

The *Earth Mover's Distance under a transformation set* is the minimum EMD between one distribution and a transformed version of the other distribution, where transformations are chosen from a given set. The allowable transformations of a distribution are dictated by the application. Sets of distribution transformations can be divided into three classes: those with transformations that change (I) only the weights, (II) only the points, and (III) both the weights and the points of a distribution. This chapter focuses on class (II) transformation sets. The previously mentioned applications of lighting-invariant object recognition, scale and orientation-invariant texture retrieval, and point feature matching in stereo image pairs use class (II) sets.

Some applications may call for class (III) sets. Suppose, for example, that a distribution point captures the location and properties of an image region, and that its corresponding weight is the region area. The EMD between two such distributions implicitly defines similar images as those in which regions of similar size and properties are close to one another. This measure will not capture visual similarities present at different scales within two images unless we allow for a transformation of both region locations and areas. Such transformations change both the points and weights of a distribution.

We have already seen an application involving a class (I) set. The scale estimation problem described in section 4.5 is formulated as the EMD under transformation ($\mathrm{EMD}_{\mathcal{G}}$) problem

$$c^0 = \max\arg \boxed{\min_{g_c \in \mathcal{G}} \ \mathrm{EMD}(\mathbf{x}, g_c(\mathbf{y}))}\,, \tag{6.1}$$

where $\mathcal{G} = \{\, g_c \ : \ g_c(\mathbf{y}) = g_c(Y,u) = (Y, cu),\ 0 < c \leq 1 \,\}$. In words, $\mathcal{G}$ consists of transformations g_c that scale down the weights of a distribution by a factor c. The $\mathrm{EMD}_{\mathcal{G}}$ problem is the boxed minimization in (6.1). Analysis of the function $E(c) = \mathrm{EMD}(\mathbf{x}, (Y, cu))$ in section 4.5 revealed a lot of structure: $E(c)$ decreases as c decreases until it becomes constant for all c less than or equal to some c^0. The scale estimate is c^0, which is the largest weight scale c that minimizes the EMD between $\mathbf{x}$ and (Y, cu). Please refer back to section 4.5 for more details and an efficient solution to (6.1) which takes full advantage of the structure of $E(c)$.

This chapter is organized as follows. In section 6.1, we give some basic definitions and notation, including a formal definition of the Earth Mover's Distance under a transformation set and related measures. In section 6.2, we give a direct, but inefficient, algorithm to compute a globally optimal (flow,transformation) pair that yields the EMD under class (II) transformation sets. In section 6.3, we give an iteration for class (II) sets that always converges monotonically, although it may converge to a transformation which is only locally (as opposed to globally) optimal. The FT iteration is applicable to any transformation set for which there is an algorithm to minimize a weighted sum of distances from one point set to a transformed version of the other. This *optimal transformation problem*, which is also a required subroutine for the direct algorithm, is the subject of section 6.4. In section 6.5, we show how the FT iteration may still be applied for a useful class (III) transformation set.

In section 6.6, we consider some specific combinations of transformation set, ground distance function, and feature space for which a globally optimal transformation can be computed directly, without the aid of our iteration. We return to the FT iteration in section 6.7, where we consider questions of convergence to only a locally optimal transformation. In section 6.8, we cover two miscellaneous topics: the tradeoffs in choosing between the L_2^2 and L_2 ground distances, and the growth rate of the EMD with respect to transformation parameters. Although the former topic is discussed in the context of the $\mathrm{EMD}_{\mathcal{G}}$ problem, other criteria are also considered. Finally, in section 6.9 we apply the FT iteration to the problems of (i) illumination-invariant object recognition, and (ii) point feature matching in stereo image pairs.

In [12], we proposed the previously mentioned iteration for the case of translation.

6.1 Definitions and Notation

The *Earth Mover's Distance under transformation set* $\mathcal{G}$ is defined as

$$\mathrm{EMD}_{\mathcal{G}}(\mathbf{x},\mathbf{y}) = \min(\min_{g\in\mathcal{G}} \mathrm{EMD}(\mathbf{x}, g(\mathbf{y})), \min_{g\in\mathcal{G}} \mathrm{EMD}(g(\mathbf{x}), y)), \tag{6.2}$$

where $g(\mathbf{x})$ is the result of applying the transformation $g \in \mathcal{G}$ to the distribution $\mathbf{x}$. In the case that

(i) the transformations in $\mathcal{G}$ only modify the distribution points,

(ii) $\mathcal{G}$ is a transformation group (and therefore every element of $\mathcal{G}$ has an inverse element), and

(iii) the ground distance d satisfies $d(x, g(y)) = d(g^{-1}(x), y)$ for all $g \in \mathcal{G}$,

we have $\mathrm{EMD}(\mathbf{x}, g(\mathbf{y})) = \mathrm{EMD}(g^{-1}(\mathbf{x}), \mathbf{y})$, and definition (6.2) reduces to

$$\mathrm{EMD}_{\mathcal{G}}(\mathbf{x},\mathbf{y}) = \min_{g\in\mathcal{G}} \mathrm{EMD}(\mathbf{x}, g(\mathbf{y})) = \min_{g\in\mathcal{G}} \mathrm{EMD}(g(\mathbf{x}), \mathbf{y}). \tag{6.3}$$

Condition (ii) is satisfied, for example, when $\mathcal{G} = \mathcal{T}$, the group of translations, and when $\mathcal{G} = \mathcal{E}$, the group of Euclidean transformations (rotation plus translation). It is not satisfied by the set of similarity transformations $\mathcal{S}$ (uniform scaling plus rotation plus translation), linear transformations $\mathcal{L}$, and affine transformations $\mathcal{A}$ (linear plus translation). Transformations that shrink a point set to a single point (scale parameter is zero) do not have inverses. Condition (iii) is satisfied, for example, with $\mathcal{G} = \mathcal{T}$ and ground distance d equal to any L_p distance function, and with $\mathcal{G} = \mathcal{E}$ and the ground distance d equal to the L_2 distance function.

We can combine the partial matching allowed by EMD^γ and the transformations allowed by $\mathrm{EMD}_\mathcal{G}$. The *partial Earth Mover's Distance under transformation set* $\mathcal{G}$ is defined as

$$\mathrm{EMD}_\mathcal{G}^\gamma(\mathbf{x},\mathbf{y}) = \min(\min_{g\in\mathcal{G}} \mathrm{EMD}^\gamma(\mathbf{x}, g(\mathbf{y})), \min_{g\in\mathcal{G}} \mathrm{EMD}^\gamma(g(\mathbf{x}), y)). \tag{6.4}$$

In the case that conditions (i), (ii), and (iii) hold, definition (6.4) reduces to

$$\mathrm{EMD}_\mathcal{G}^\gamma(\mathbf{x},\mathbf{y}) = \min_{g\in\mathcal{G}} \mathrm{EMD}^\gamma(\mathbf{x}, g(\mathbf{y})) = \min_{g\in\mathcal{G}} \mathrm{EMD}^\gamma(g(\mathbf{x}), \mathbf{y})). \tag{6.5}$$

Using the partial EMD under a transformation set may be useful, for example, in matching point features in stereo image pairs. The fraction parameter γ compensates for the fact that only some features appear in both images, and the set parameter $\mathcal{G}$ accounts for the appropriate transformation between corresponding features.

We shall now prove that the EMD under a transformation set is a metric when conditions (i), (ii), and (iii) are satisfied and the EMD itself is a metric. Recall that the EMD is a metric on distributions when the ground distance is a metric and the distributions have equal total weight. For a precise statement of the theorem, we need to define equivalence classes on distributions under the group $\mathcal{G}$. Two distributions $\mathbf{x}$ and $\mathbf{y}$ are in the same $\mathcal{G}$-equivalence class iff $\mathbf{x} = g(\mathbf{y})$ for some $g \in \mathcal{G}$. The equivalence class $E(\mathbf{x})$ that contains distribution $\mathbf{x}$ is

$$E(\mathbf{x}) = \{\, g(\mathbf{x}) : g \in \mathcal{G} \,\}.$$

We say that two distributions $\mathbf{x}$ and $\widehat{\mathbf{x}}$ are *$\mathcal{G}$-equivalent* iff $\mathbf{x}, \widehat{\mathbf{x}} \in E(\mathbf{x})$, and we denote this equivalence as $\mathbf{x} \sim \widehat{\mathbf{x}}$. Note that

$$\mathrm{EMD}_\mathcal{G}(\mathbf{x},\mathbf{y}) = \mathrm{EMD}_\mathcal{G}(\widehat{\mathbf{x}},\widehat{\mathbf{y}}) \quad \forall \mathbf{x} \sim \widehat{\mathbf{x}}, \mathbf{y} \sim \widehat{\mathbf{y}}.$$

The EMD under a transformation group is then well defined on $\mathcal{G}$-equivalence classes by

$$\mathrm{EMD}_{\mathcal{G}}(E_1, E_2) = \mathrm{EMD}_{\mathcal{G}}(\mathbf{x}, \mathbf{y}), \quad \forall \mathbf{x} \in E_1, \mathbf{y} \in E_2. \tag{6.6}$$

Theorem 12 *The EMD under a transformation group $\mathcal{G}$ is a metric on distribution $\mathcal{G}$-equivalence classes when conditions (i), (ii), and (iii) are satisfied and the EMD itself is a metric on distributions.*

Proof. Obviously, $\mathrm{EMD}_{\mathcal{G}}(E_1, E_2) \geq 0$ for every pair of $\mathcal{G}$-equivalence classes E_1 and E_2 because the EMD is nonnegative. We need to show that $\mathrm{EMD}_{\mathcal{G}}(E_1, E_2) = 0$ iff $E_1 = E_2$. The nontrivial direction is to show that $\mathrm{EMD}_{\mathcal{G}}(E_1, E_2) = 0$ implies $E_1 = E_2$. Fix equivalence class representatives $\mathbf{x} \in E_1$ and $\mathbf{y} \in E_2$. If $\mathrm{EMD}_{\mathcal{G}}(E_1, E_2) = 0$, then by definitions (6.3) and (6.6) there exists $g \in \mathcal{G}$ such that $\mathrm{EMD}(\mathbf{x}, g(\mathbf{y})) = 0$. Since the EMD is a metric, we must have $\mathbf{x} = g(\mathbf{y})$. It follows that $E_1 = E_2$. The symmetry of $\mathrm{EMD}_{\mathcal{G}}$ follows from the symmetry of the EMD. Finally, we need to show that the triangle inequality holds. Suppose E_1, E_2, and E_3 are equivalence classes with representative distributions $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$, and that

$$\begin{aligned} \mathrm{EMD}_{\mathcal{G}}(E_1, E_2) &= \mathrm{EMD}(g_x(\mathbf{x}), \mathbf{y}) \quad \text{and} && (6.7) \\ \mathrm{EMD}_{\mathcal{G}}(E_2, E_3) &= \mathrm{EMD}(\mathbf{y}, g_z(\mathbf{z})). && (6.8) \end{aligned}$$

Here g_x and g_z are the transformations of $\mathbf{x}$ and $\mathbf{z}$ that yield the minimum value in (6.3). Since $g_x(\mathbf{x}) \sim \mathbf{x}$ and $g_z(\mathbf{z}) \sim \mathbf{z}$, it follows from (6.3) and (6.6) that

$$\mathrm{EMD}_{\mathcal{G}}(E_1, E_3) \leq \mathrm{EMD}(g_x(\mathbf{x}), g_z(\mathbf{z})).$$

But the EMD is a metric between distributions, so it obeys the triangle inequality and

$$\mathrm{EMD}(g_x(\mathbf{x}), g_z(\mathbf{z})) \leq \mathrm{EMD}(g_x(\mathbf{x}), \mathbf{y}) + \mathrm{EMD}(\mathbf{y}, g_z(\mathbf{z})).$$

Combining the previous two inequalities with (6.7) and (6.8) gives the triangle inequality

$$\mathrm{EMD}_{\mathcal{G}}(E_1, E_3) \leq \mathrm{EMD}_{\mathcal{G}}(E_1, E_2) + \mathrm{EMD}_{\mathcal{G}}(E_2, E_3)$$

that we desire. ∎

For simplicity, we write about $\mathrm{EMD}_{\mathcal{G}}(\mathbf{x}, \mathbf{y})$ in the remaining sections of this chapter as if it were just $\min_{g \in \mathcal{G}} \mathrm{EMD}(\mathbf{x}, g(\mathbf{y}))$.

6.2 A Direct Algorithm

The transformed distribution $g(\mathbf{y}) = (g(Y), u) \in \mathbf{D}^{K,n}$ has the same weights as the original distribution $\mathbf{y}$. Thus $\mathcal{F}(\mathbf{x},\mathbf{y}) = \mathcal{F}(\mathbf{x}, g(\mathbf{y}))$ and

$$\mathrm{EMD}_{\mathcal{G}}(\mathbf{x},\mathbf{y}) = \frac{\min_{g\in\mathcal{G}, F\in\mathcal{F}(\mathbf{x},\mathbf{y})} \mathrm{WORK}(F,\mathbf{x},g(\mathbf{y}))}{\min(w_\Sigma, u_\Sigma)}. \tag{6.9}$$

Clearly, it suffices to minimize the work $h(F,g) = \mathrm{WORK}(F,\mathbf{x},g(\mathbf{y}))$ over the region $R(\mathbf{x},\mathbf{y}) = \{\ (F,g)\ :\ F \in \mathcal{F}(\mathbf{x},\mathbf{y}), g \in \mathcal{G}\ \} = \mathcal{F}(\mathbf{x},\mathbf{y}) \times \mathcal{G}$.

The function $h(F,g)$ is linear in F. It follows that for g fixed, the minimum value $\min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y})} h(F,g)$ is achieved at one of the vertices (dependent on g) of the convex polytope $\mathcal{F}(\mathbf{x},\mathbf{y})$. If we let $V(\mathbf{x},\mathbf{y}) = \{\ v_1(\mathbf{x},\mathbf{y}),\ \ldots,\ v_N(\mathbf{x},\mathbf{y})\ \}$ denote this finite set of vertices, then

$$\min_{(F,g)\in R(\mathbf{x},\mathbf{y})} h(F,g) = \min_{k=1,\ldots,N}\ \min_{g\in\mathcal{G}}\ h(v_k(\mathbf{x},\mathbf{y}), g). \tag{6.10}$$

Assuming that we can solve the innermost minimization problem on the right-hand side of (6.10), we can compute the numerator in (6.9) by simply looping over all the vertices in $V(\mathbf{x},\mathbf{y})$. Only a finite number of flow values must be examined to find the minimum work.

Although this simple strategy guarantees that we find a globally optimal transformation, it is not practical because N is usually very large even for relatively small values of m and n. The worst case complexity of the number of vertices in the feasible convex polytope for a linear program is exponential in the minimum of the number of variables and constraints.[1] The beauty of the simplex algorithm for solving a linear program is that it provides a method for visiting vertices of the feasible polytope in such a way that the objective function always gets closer to its optimal value, and the number of vertices visited is almost always no larger in order than the maximum of the number of variables and the number of constraints ([58]). In the next section, we give an iterative algorithm that generates a sequence of (flow,transformation) pairs for which the amount of work decreases or remains constant at every step.

6.3 The FT Iteration

The work function $h(F,g)$ to minimize depends on both a flow vector F and a transformation g. Given either variable, we can solve for the optimal value of the other. This leads us to an iteration which alternates between finding the best flow for a given transformation, and

[1] For a balanced transportation problem with m suppliers and n demanders, there are mn variables and $m+n$ constraints (not including the nonnegativity constraints on the variables).

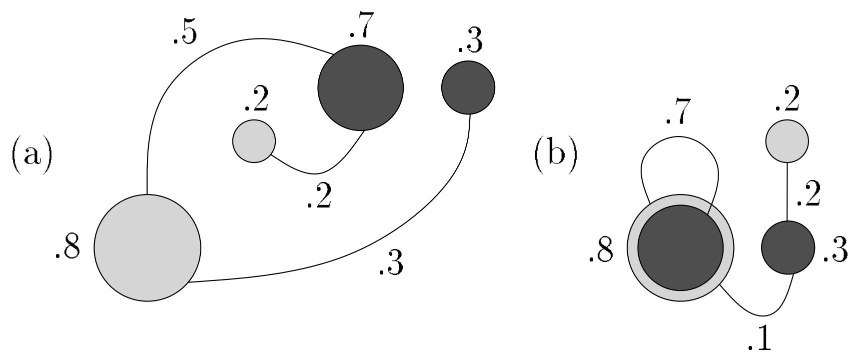


Figure 6.1: FT Iteration Example. See the text for an explanation.

the best transformation for a given flow. The flow step establishes correspondences that minimize the work for a fixed configuration of distribution points, while the transformation step moves the distribution points around so that the work is minimized for a given set of correspondences. By alternating these steps we obtain a sequence of (flow,transformation) pairs for which the amount of work decreases or remains constant at every step.

In this section, we consider distribution transformations that alter only the points of a distribution, leaving distribution weights unchanged. If g is such a transformation, then $\mathcal{F}(\mathbf{x}, g(\mathbf{y})) = \mathcal{F}(\mathbf{x}, \mathbf{y})$ since $\mathbf{y}$ and $g(\mathbf{y})$ have the same set of weights. Consider the following iteration that begins with an initial transformation $g^{(0)}$:

$$F^{(k)} = \arg\left(\min_{F \in \mathcal{F}(\mathbf{x}, g^{(k)}(\mathbf{y})) = \mathcal{F}(\mathbf{x}, \mathbf{y})} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} d\left(x_i, g^{(k)}(y_j)\right)\right), \tag{6.11}$$

$$g^{(k+1)} = \arg\left(\min_{g \in \mathcal{G}} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^{(k)} d(x_i, g(y_j))\right). \tag{6.12}$$

The minimization problem on the right-hand side of (6.11) is the familar transportation problem. For now, we assume that there is an algorithm to solve for the optimal transformation in (6.12). This problem is the subject of section 6.4. Since this iteration alternates between finding an optimal F̲low and an optimal T̲ransformation, we refer to (6.11) and (6.12) as the *FT* iteration. It can be applied to equal-weight and unequal-weight distributions.

Figure 6.1(a) shows an example with a dark and a light distribution that we will match under translation starting with $g^{(0)} = 0$. The best flow $F^{(0)}$ for $g^{(0)}$ is shown by the labelled arcs connecting dark and light weights. This flow matches half (.5) the weight over a large distance. We should expect the best translation for $F^{(0)}$ to move the .7 dark weight closer to the .8 light weight in order to decrease the total amount of work done by $F^{(0)}$. Indeed, $g^{(1)}$ aligns these two weights as shown in Figure 6.1(b). The best flow $F^{(1)}$ for this translation matches all of the .7 dark weight to the .8 light weight. No further translation improves the work – $g^{(2)} = g^{(1)}$ and the FT iteration converges.

The flow and transformation iterates $F^{(k)}$ and $g^{(k)}$ define the WORK and EMD iterates

$$\begin{aligned}
\mathrm{WORK}^{(k)} &= \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^{(k)} d\left(x_i, g^{(k)}(y_j)\right) = \mathrm{WORK}\left(F^{(k)}, \mathbf{x}, g^{(k)}(\mathbf{y})\right), \\
\mathrm{EMD}^{(k)} &= \frac{\mathrm{WORK}^{(k)}}{\min(w_\Sigma, u_\Sigma)}.
\end{aligned}$$

The order of evaluation is

$$\underbrace{g^{(0)} \longrightarrow F^{(0)}}_{\mathrm{WORK}^{(0)},\, \mathrm{EMD}^{(0)}} \longrightarrow \underbrace{g^{(1)} \longrightarrow F^{(1)}}_{\mathrm{WORK}^{(1)},\, \mathrm{EMD}^{(1)}} \longrightarrow \cdots .$$

By (6.11), we have

$$\begin{aligned}
\mathrm{WORK}^{(k+1)} &= \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^{(k+1)} d\left(x_i, g^{(k+1)}(y_j)\right) \\
&\le \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^{(k)} d\left(x_i, g^{(k+1)}(y_j)\right). \qquad (6.13)
\end{aligned}$$

In detail, $F^{(k)} \in \mathcal{F}(\mathbf{x},\mathbf{y}) = \mathcal{F}(\mathbf{x}, g^{(k)}(\mathbf{y}))$, while $F^{(k+1)} \in \mathcal{F}(\mathbf{x},\mathbf{y}) = \mathcal{F}(\mathbf{x}, g^{(k+1)}(\mathbf{y}))$ is optimal for $g^{(k+1)}$ over all flows in $\mathcal{F}(\mathbf{x},\mathbf{y})$. Therefore using $F^{(k+1)}$ with $g^{(k+1)}$ results in less work than using $F^{(k)}$ with $g^{(k+1)}$. From (6.12), we know

$$\begin{aligned}
\mathrm{WORK}^{(k)} &= \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^{(k)} d\left(x_i, g^{(k)}(y_j)\right) \\
&\ge \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^{(k)} d\left(x_i, g^{(k+1)}(y_j)\right). \qquad (6.14)
\end{aligned}$$

Combining (6.13) and (6.14),

$$\mathrm{WORK}^{(k+1)} \le \mathrm{WORK}^{(k)}.$$

The decreasing sequence $\mathrm{WORK}^{(k)}$ is bounded below by zero, and hence it converges ([38], pp. 49–50). There is, however, no guarantee that the work iteration converges to the global minimum of $h(F, g) = \mathrm{WORK}(F, \mathbf{x}, g(\mathbf{y}))$.

Using the exact same iteration with $\mathcal{F}^\gamma(\mathbf{x}, \mathbf{y})$ in place of $\mathcal{F}(\mathbf{x}, \mathbf{y})$ will also yield a decreasing sequence of WORK values (and, hence, a decreasing sequence of EMD values). This is because $\mathcal{F}^\gamma(\mathbf{x}, g(\mathbf{y})) = \mathcal{F}^\gamma(\mathbf{x}, \mathbf{y})$ when g does not change distribution weights. Therefore, the FT iteration can also be used in an attempt to compute the partial EMD under transformation when the transformations do not change the weights of a distribution. Please refer back to section 4.4.1 for the details of the partial EMD.

6.3.1 Similar Work

The FT iteration is similar to the ICP (Iterative Closest Point) algorithm ([5]) used to register 3D shapes. The computation of the optimal flow between distributions in the FT iteration plays the role of the computation of closest "model shape" points to the "data shape" points in the ICP iteration. Both these steps determine correspondences used to compute a transformation that improves the EMD/registration. There are, however, a number of important differences between the two algorithms and the contexts in which they are applied.

As we noted in section 4.3.1, the EMD provides a distance between point sets (which are distributions in which all weights are equal to one) as well as general distributions. The ICP algorithm is used to register a data shape defined as a point set with a model shape defined by a set of geometric primitives such as points, line segments, curves, etc.. If the model shape is also defined as a set of points, then the ICP algorithm also seeks to minimize the distance between two point sets under a transformation. The notion of point set distance defined by the EMD, however, is different than the notion of distance used in the ICP formulation. The ICP algorithm's correspondence step computes the nearest neighbor in the model shape for each data shape point, and sums up all these distances. The same model shape point may be the nearest neighbor of many data shape points in the distance sum computation. Therefore, the ICP algorithm uses a Hausdorff-like distance between point sets. In contrast, when $\text{EMD}_{\mathcal{G}}$ is used to match point sets under a transformation, the constraints that define the EMD imply a one-to-one matching of the points (see section 4.3.1). The correspondence step for the FT iteration requires the solution of an assignment problem, whereas the correspondence step in the ICP algorithm matches each data shape point independently to its closest model shape point. The unconstrained matching in the ICP algorithm will obviously be faster to compute than the constrained matching specified by the EMD, but the two iterations are trying to minimize different point set distance metrics.

Using our EMD framework instead of the ICP framework to match point sets under a transformation has the advantage that it can find the best subsets (with size specified by γ) of the given sets to match. In fact, the partial match parameter γ can be used to align different sub-distributions (subsets) of two distributions (point sets) using different transformations. For example, one could find the transformation that works well for $\gamma = 20\%$ of the data, remove the matched data, and repeat. The ICP algorithm could also be applied in this piecewise manner, but the user must select the subsets of the data shape to be matched, not just the subset size as in the partial EMD case.

Limiting our comparison of the FT iteration and the ICP iteration to point sets is unfair to the FT iteration since the EMD can be used to match distributions of mass which are more general than point sets. The mass at a point in one distribution can be matched to the mass at many points in the other distribution, and vice-versa. The FT iteration in general provides a many-to-many matching of distribution points, while the ICP iteration (applied to a model shape point set) gives a many-to-one matching of model shape points to data shape points, and the FT iteration applied to point sets gives a one-to-one matching of points in the two sets. Furthermore, the amount of mass matched between two points is used to weight the distance between the points; all the point distances have weight one in the ICP iteration and the FT iteration applied to point sets. The many-to-many matching specified by the EMD is constrained by the distribution masses in such a way that the matching process represents a morphing of one distribution into the another. As we shall see in section 6.5, the FT iteration as presented thus far can be modified to handle the case in which transformations are allowed to modify both the distribution points and their corresponding masses.

Another well-known application of the alternation idea is the *Expectation-Maximization* or EM algorithm ([47, 48]) for computing mixture models in statistics. In this problem, observed data are assumed to arise from some number of parametrized distributions. The goal is to determine which data come from which distributions and to compute the parameter values of the distributions. The EM algorithm alternates between finding the expected assignments[2] of the data to the distributions with fixed defining parameters (the E-step), and finding the maximum likelihood estimate of the parameter values given the expected assignments (the M-step). The function to be optimized in this case is a likelihood function, and the optimization problem is a maximization.

[2]The E-step computes the expected value $E[Z_{ij}] = P(Z_{ij} = 1)$, where $Z_{ij} = 1$ if the jth observation arises from the ith distribution, and $Z_{ij} = 0$ otherwise.

6.3.2 Convergence Properties

One way for the WORK iteration to converge is if $F^{(k)}$ is returned in step (6.11) as an optimal flow for $g^{(k)}$, and $g^{(k+1)} = g^{(k)}$ is returned in step (6.12) as an optimal transformation for $F^{(k)}$. Denote the indicator function for this event as $\text{MUTUAL}\left(F^{(k)}, g^{(k)}\right)$, since $F^{(k)}$ is optimal for $g^{(k)}$, and $g^{(k)}$ is optimal for $F^{(k)}$. It is clear that

$$\text{MUTUAL}\left(F^{(k)}, g^{(k)}\right) \quad \Rightarrow \quad \begin{cases} g^{(k)} & = & g^{(k+1)} & = & \cdots, \\ F^{(k)} & = & F^{(k+1)} & = & \cdots, \\ \text{WORK}^{(k)} & = & \text{WORK}^{(k+1)} & = & \cdots. \end{cases} \quad \text{and}$$

The WORK iteration converges to either a local minimum or a saddle point value if $\text{MUTUAL}\left(F^{(k)}, g^{(k)}\right)$ is true.[3]

Now suppose that the routine that solves the linear program (LP) in (6.11) always returns a vertex of $\mathcal{F}(\mathbf{x}, \mathbf{y})$. The simplex algorithm and the transportation simplex algorithm, for example, always return a vertex of the feasible polytope. This is possible since there is always a vertex of the feasible polytope at which a linear objective function achieves its minimum. With the assumption that the flow iterates are always vertices of $\mathcal{F}(\mathbf{x}, \mathbf{y})$, there will be only a finite number of points (F, t) that the WORK iteration visits because there are a finite number of flow iterates, and each transformation iterate (other than the initial transformation) must be an optimal transformation returned for one of the flow iterates. It follows that there are only a finite number of WORK values generated. Since the WORK iteration is guaranteed to converge, the WORK iterates must stabilize at one of these WORK values. Suppose

$$\text{WORK}^{(k)} = \text{WORK}^{(k+1)} = \cdots . \tag{6.15}$$

Since there are only a finite number of pairs (F, t) visited, condition (6.15) implies that there must be a repeating cycle of pairs:

$$\left(F^{(k)}, g^{(k)}\right), \ \ldots, \left(F^{(k+r-1)}, g^{(k+r-1)}\right), \left(F^{(k+r)}, g^{(k+r)}\right) = \left(F^{(k)}, g^{(k)}\right), \ \ldots .$$

[3]If $g^{(k)}$ occurs in the interior of $R(F^{(k)}) = \left\{ g : F^{(k)} \in \arg\min_{F \in \mathcal{F}} \text{WORK}(F, x, g(y)) \right\}$, then $(F^{(k)}, g^{(k)})$ cannot be a saddle point and the WORK iteration converges to a local minimum of $\text{WORK}(F, \mathbf{x}, g(\mathbf{y}))$. The argument is toward the end of section 6.7.4, the paragraph beginning "Let us now explicitly connect ...". In general, the possibility convergence to a saddle point cannot be eliminated. All we know is that along the F axis though $g^{(k)}$, the minimum occurs at $F^{(k)}$, and along the g axis though $F^{(k)}$, the minimum occurs at $g^{(k)}$. This does not imply anything about the value of WORK close to $(F^{(k)}, g^{(k)})$ along a "diagonal" through $(F^{(k)}, g^{(k)})$.

For $r > 1$, the WORK iteration converges even though the flow and transformation iterations do not converge. However, such a nontrivial (flow,transformation) cycle is unstable in the sense that it can be broken (for any real problem data) by perturbing one of the transformation iterates by a small amount. In practice, the WORK iteration almost always converges because a length $r = 1$ cycle occurs. A cycle of length $r = 1$ starting at $\left(F^{(k)}, g^{(k)}\right)$ is exactly the condition MUTUAL $\left(F^{(k)}, g^{(k)}\right)$, and we previously argued that the WORK iteration converges to a critical value in this case.

Finally, let us note that the WORK sequence will stabilize at the global minimum once $F^{(k)} = F^*$, where (F^*, g^*) is optimal for some g^*. This is because $g^{(k+1)}$ and g^* both solve (6.12), so $h(F^{(k)}, g^{(k+1)}) = h(F^*, g^*)$, which is the global minimum of the WORK function. Since h can only decrease or remain the same with successive flow and transformation iterates, and h can never be less than the global minimum of the WORK function, we must have

$$h(F^*, g^*) = \mathrm{WORK}^{(k+1)} = h(F^{(k+1)}, g^{(k+1)}) = \mathrm{WORK}^{(k+2)} = h(F^{(k+2)}, g^{(k+2)}) = \cdots.$$

Similarly, if the transformation iteration ever reaches a transformation $g^{(k)} = g^*$ at which the minimum value of WORK occurs with some flow F^*, then the WORK iteration converges to the global minimum. Here we need the fact that $F^{(k)}$ and F^* both solve (6.11).

Let us summarize the results of this section. The WORK iteration always converges. We can arrange to have all flow iterates at the vertices of $\mathcal{F}(\mathbf{x}, \mathbf{y})$. In this case, the (flow,transformation) iterates must cycle. A cycle of length $r > 1$ will almost never occur, and a cycle of length $r = 1$ implies that the (flow,transformation) sequence converges to a critical point and, therefore, that the WORK sequence converges to either a local minimum or a saddle point value. Thus, in practice the WORK iteration almost always converges to a critical value. If the flow iteration ever reaches a vertex at which the minimum WORK occurs with a suitable choice of transformation, then the WORK iteration converges to the global minimum. Global convergence will also occur if the transformation iteration ever reaches a transformation at which the minimum WORK occurs with a suitable choice of flow.

Although we do not explore the possibility here, perhaps convergence of the FT iteration can be accelerated by using the EMD values at the past few transformation iterates to predict the transformation at which the EMD will be minimized. If the predicted transformation causes an increase in the EMD, we could discard the prediction and just use the solution to (6.12) as usual to define the next transformation iterate. This approach successfully accelerated the convergence of the previously mentioned ICP iteration ([5]).

6.4 The Optimal Transformation Problem

Now consider the problem (6.12) of solving for the optimal transformation for a fixed $F = (f_{ij})$:

$$\min_{g \in \mathcal{G}} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d(x_i, g(y_j)). \tag{6.16}$$

If we let

$$\begin{aligned}
[a_1 \cdots a_N] &= [x_1 \cdots x_1 x_2 \cdots x_2 \cdots x_m \cdots x_m], && (6.17)\\
[b_1 \cdots b_N] &= [y_1 \cdots y_n y_1 \cdots y_n \cdots y_1 \cdots y_n], \quad \text{and} && (6.18)\\
[c_1 \cdots c_N] &= [f_{11} \cdots f_{1n} f_{21} \cdots f_{2n} \cdots f_{m1} \cdots f_{mn}], && (6.19)
\end{aligned}$$

where $N = mn$, then the *optimal transformation problem* (6.16) can be rewritten as

$$\min_{g \in \mathcal{G}} \sum_{k=1}^{N} c_k d(a_k, g(b_k)). \tag{6.20}$$

In this form, the optimal transformation problem can be stated as follows: given a weighted correspondence between point sets, find a transformation of the points in one set that minimizes the weighted sum of distances to corresponding points in the other set.[4] We now discuss the solution of the optimal transformation problem for translation, Euclidean, similarity, linear, and affine transformations with d equal to the L_2-distance squared, as well as the optimal translation problem with the L_2-distance, the L_1-distance, and a cyclic L_1-distance (to be explained shortly).

6.4.1 Translation

Suppose that $\mathcal{G} = \mathcal{T}$, the group of translations. If

$$d(x_i, y_j + t) = d(x_i - y_j, t), \tag{6.21}$$

then (6.20) can be written as

$$\min_{t \in \mathbf{R}^K} \sum_{k=1}^{N} c_k d(a_k - b_k, t).$$

[4]Note that some structure is lost in the rewrite from (6.16) to (6.20). In the setting of the FT iteration, the N points a_k consist of n copies of the set $\{x_i\}_{i=1}^{m}$, and the N points b_k consist of m copies of the set $\{y_j\}_{j=1}^{n}$.

Note that condition (6.21) holds for any L_p distance function d, as well as $d = L_2^2$. This minimization problem asks for a point t which minimizes a sum of weighted distances to a given set of points. We show how to solve this *minisum* problem when d is the L_2-distance squared, the L_1-distance, and the L_2-distance in sections 6.4.1.1, 6.4.1.2, and 6.4.1.3, respectively.

In section 6.4.1.4, we solve the optimal translation problem when points are located on a circle, and ground distance is the length of the shorter arc connecting two points. This problem arises, for example, when applying the FT iteration to compute an orientation-invariant EMD between texture signatures in log-polar frequency space ([69, 68]). The polar coordinates for spatial frequency (f_x, f_y) are (s, θ), where the scale $s = \sqrt{f_x^2 + f_y^2}$ and the orientation $\theta = \arctan(f_y, f_x)$. Roughly speaking, distributions are of the form $\mathbf{x} = \{((\widehat{s}_i, \theta_i), w_i)\}_{i=1}^m$, where $\widehat{s}_i = \log s_i$ and the ith element indicates that the texture has a fraction w_i of its total energy at scale s_i and orientation θ_i.

If we denote the distribution for another texture as $\{((\widehat{\alpha}_j, \psi_j), u_j)\}_{j=1}^n$, then the texture distance $\min_{\Delta\psi} \mathrm{EMD}(\mathbf{x}, \{((\widehat{\alpha}_j, \psi_j + \Delta\psi), u_j)\})$ is invariant to texture orientation. The optimal translation problem is

$$\begin{aligned} \min_{\Delta\psi} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\left(|\widehat{s}_i - \widehat{\alpha}_j| + |\theta_i - (\psi_j + \Delta\psi)|_{2\pi}\right) = \\ \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}|\widehat{s}_i - \widehat{\alpha}_j| + \min_{\Delta\psi} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}|\theta_i - (\psi_j + \Delta\psi)|_{2\pi}, \end{aligned} \tag{6.22}$$

where the absolute value subscript indicates distances are measured modulo 2π. The minimization problem on the right-hand side of (6.22) is the subject of section 6.4.1.4.

Notice that using the logarithm of the scale in the texture signatures implies that a scale change by factor k results in a shift by $\widehat{k} = \log k$ along the log-scale axis. A scale-invariant texture distance measure is $\min_{\widehat{k}} \mathrm{EMD}(\mathbf{x}, \{((\widehat{\alpha}_j + \widehat{k}, \psi_j), u_j)\})$. The optimal translation problem in this case is

$$\begin{aligned} \min_{\widehat{k}} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\left(|\widehat{s}_i - (\widehat{\alpha}_j + \widehat{k})| + |\theta_i - \psi_j|_{2\pi}\right) = \\ \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}|\theta_i - \psi_j|_{2\pi} + \min_{\widehat{k}} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}|\widehat{s}_i - (\widehat{\alpha}_j + \widehat{k})|. \end{aligned} \tag{6.23}$$

The L_1 minimization problem on the right-hand side of (6.23) is solved in section 6.4.1.2.

Finally, the texture distance $\min_{\widehat{k},\Delta\psi} \mathrm{EMD}(\mathbf{x}, \{((\widehat{\alpha}_j + \widehat{k}, \psi_j + \Delta\psi), u_j)\})$ is invariant to both texture scale and orientation. The associated optimal translation problem

$$\begin{aligned}
&\min_{\widehat{k},\Delta\psi} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}(|\widehat{s}_i - (\widehat{\alpha}_j + \widehat{k})| + |\theta_i - (\psi_j + \Delta\psi)|_{2\pi}) = \\
&\qquad \min_{\widehat{k}} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}|\widehat{s}_i - (\widehat{\alpha}_j + \widehat{k})| + \min_{\Delta\psi} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}|\theta_i - (\psi_j + \Delta\psi)|_{2\pi}
\end{aligned} \tag{6.24}$$

can be solved by solving the minimization problems on the right-hand side of (6.24) separately. In general, of course, the separability of the L_1 distance into a sum of component distances means that we can solve the optimal translation problem under L_1 in any number of dimensions, where each dimension may have a different wrap around period or no wrap around at all.

6.4.1.1 Minimizing a Weighted Sum of Squared L_2 Distances

If d is the L_2-distance squared, then it is well-known that the unique optimal translation is given by the centroid difference

$$t^*_{L_2^2} = \overline{a} - \overline{b} = \frac{\sum_{k=1}^{N} c_k a_k}{c_\Sigma} - \frac{\sum_{k=1}^{N} c_k b_k}{c_\Sigma}.$$

This result is easily proven using standard calculus.

6.4.1.2 Minimizing a Weighted Sum of L_1 Distances

In this section, we consider the minisum problem when d is the L_1-distance. The minimization problem is

$$\begin{aligned}
\min_p \sum_{i=1}^{n} w_i ||p - p_i||_1 &= \min_p \sum_{i=1}^{n} w_i \sum_{k=1}^{K} \left|p^{(k)} - p_i^{(k)}\right| \\
&= \min_p \sum_{k=1}^{K} \left(\sum_{i=1}^{n} w_i \left|p^{(k)} - p_i^{(k)}\right| \right) \\
\min_p \sum_{i=1}^{n} w_i ||p - p_i||_1 &= \sum_{k=1}^{K} \left(\min_{p^{(k)}} \sum_{i=1}^{n} w_i \left|p^{(k)} - p_i^{(k)}\right| \right),
\end{aligned} \tag{6.25}$$

where $p^{(k)}$ and $p_i^{(k)}$ are the kth components of p and p_i, respectively.[5] Thus, a solution to the problem in one dimension gives a solution to the problem in K dimensions by simply collecting the optimal location for each of the one-dimensional problems into a K-dimensional vector. We shall see that an optimal location for the one-dimensional problem in dimension k is the (weighted) median of the values $p_1^{(k)}, \ldots, p_n^{(k)}$. A point p at which the minimum (6.25) is achieved is thus called a *coordinate-wise median* of $p_1, \ldots, p_n$ (with weights $w_1, \ldots, w_n$).

Now suppose $p_1 \leq p_2 \leq \cdots \leq p_n$ are points along the real line, and we want to minimize

$$g(p) = \sum_{i=1}^{n} w_i |p - p_i|.$$

Let $p_0 = -\infty$ and $p_{n+1} = +\infty$. Then

$$g(p) = \sum_{i=1}^{l} w_i (p - p_i) + \sum_{i=l+1}^{n} w_i (p_i - p) \qquad \text{for } p \in [p_l, p_{l+1}], \;\; l = 0, \ldots, n.$$

Over the interval $[p_l, p_{l+1}]$, $g(p)$ is affine in p:

$$g(p) = \left(\sum_{i=1}^{l} w_i - \sum_{i=l+1}^{n} w_i \right) p + \left(\sum_{i=l+1}^{n} w_i p_i - \sum_{i=1}^{l} w_i p_i \right) \qquad \text{for } p \in [p_l, p_{l+1}].$$

If we let

$$m_l = \sum_{i=1}^{l} w_i - \sum_{i=l+1}^{n} w_i \tag{6.26}$$

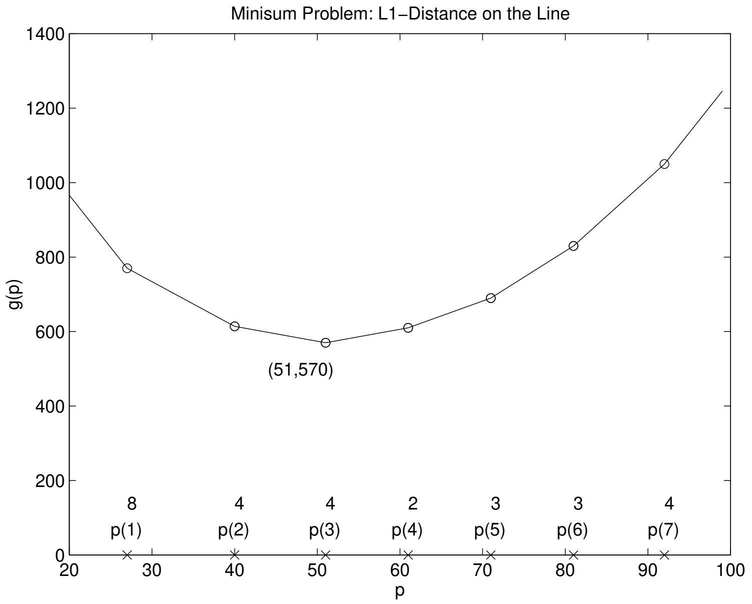
denote the slope of $g(p)$ over $[p_l, p_{l+1}]$, then $-w_\Sigma = m_0 < m_1 < \cdots < m_n = w_\Sigma$ (assuming $w_i > 0 \; \forall i$), and $m_{l+1} = m_l + 2w_{l+1}$. The function $g(p)$ is a continuous piecewise linear function with slope increasing from a negative value at $-\infty$ to a positive value at $+\infty$, and as such it obviously has a minimum value at the point when its slope first becomes nonnegative. Let

$$l^* = \min \{ \, l \; : \; m_l \geq 0 \, \}.$$

If $m_{l^*} \neq 0$, then the unique minimum value of $g(p)$ occurs at p_{l^*}. Otherwise, $m_{l^*} = 0$ and the minimum value of $g(p)$ is achieved for $p \in [p_{l^*}, p_{l^*+1}]$. See Figure 6.2. In the special case of equal-weight points, the minimum value occurs at the ordinary median value of the points. If $w_i \equiv w$, then it follows easily from (6.26) that $m_l = w(2l - n)$. If n is odd, then $l^* = \lceil n/2 \rceil$, $m_{l^*} > 0$, and the unique minimum of $g(p)$ occurs at the median point $p_{\lceil n/2 \rceil}$.

[5] In this section and the next, weights are denoted by w_i instead of c_k, the total number of points is denoted by n instead of N, and the summation of weighted distances is over the variable i instead of k. The points p_i are the differences $a_k - b_k$.

(a)



(b)

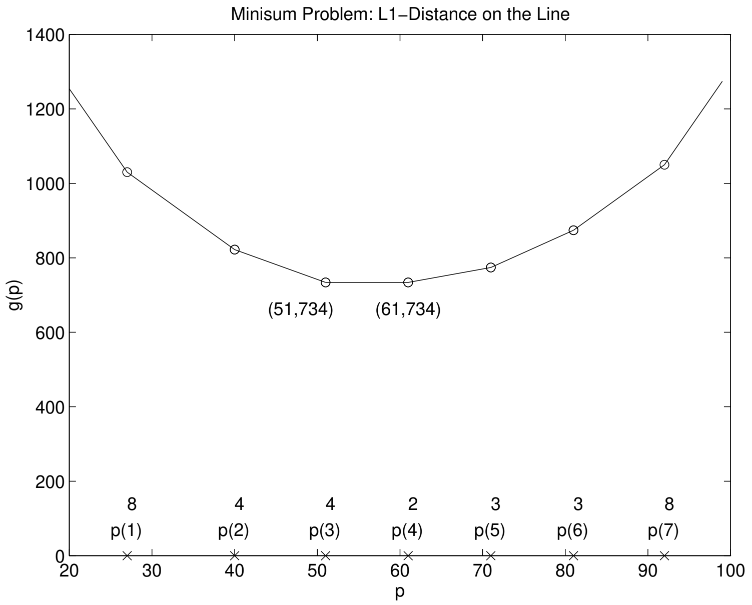
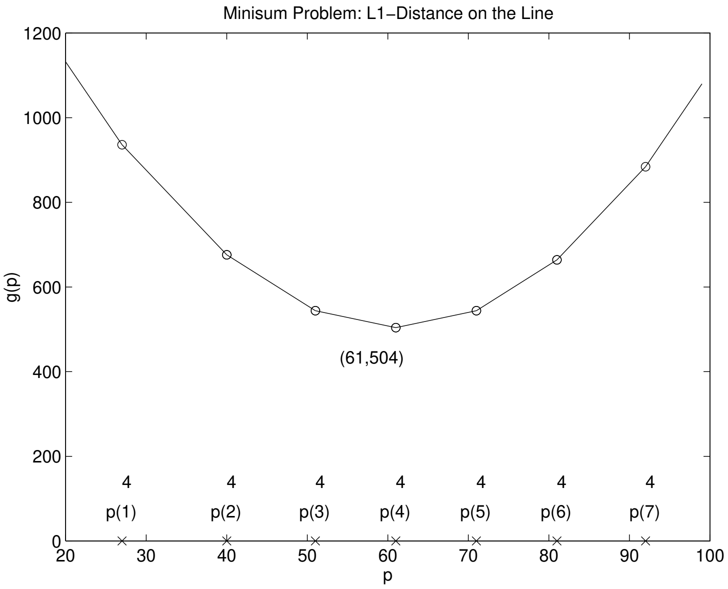


Figure 6.2: The Minisum Problem on the Line with Unequal Weights. (a) $p = [27, 40, 51, 61, 71, 81, 92]$, $w = [8, 4, 4, 2, 3, 3, 4]$: $l^* = 3$, $m_{l^*} > 0$, and there is a unique minimum at $p_3 = 51$. (b) $p = [27, 40, 51, 61, 71, 81, 92]$, $w = [8, 4, 4, 2, 3, 3, 8]$: $l^* = 3$, $m_{l^*} = 0$, and the minimum occurs at every value in $[p_3, p_4] = [51, 61]$.

(a)



(b)

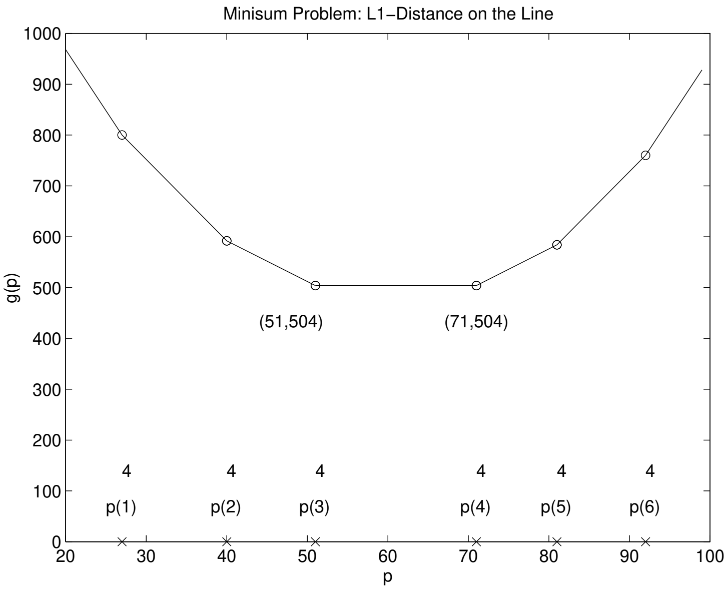


Figure 6.3: The Minisum Problem on the Line with Equal Weights. (a) $p = [27, 40, 51, 61, 71, 81, 92]$, $w = [4, 4, 4, 4, 4, 4, 4]$: $l^* = 4$, $m_{l^*} > 0$, and there is a unique minimum at the ordinary median $p_4 = 61$. (b) $p = [27, 40, 51, 71, 81, 92]$, $w = [4, 4, 4, 4, 4, 4]$: $l^* = 3$, $m_{l^*} = 0$, and the minimum occurs at every value in the interval $[p_3, p_4] = [51, 71]$.

See Figure 6.3(a). If n is even, then $l^* = n/2$, $m_{l^*} = 0$, and the minimum value of $g(p)$ is attained for every point in the interval $[p_{n/2}, p_{(n/2)+1}]$. See Figure 6.3(b).

6.4.1.3 Minimizing a Weighted Sum of L_2 Distances

The next minisum problem that we consider is when d is the L_2-distance function:

$$\min_p \sum_{i=1}^{n} w_i ||p - p_i||_2 \tag{6.27}$$

A point p at which this minimum is achieved is called a *spatial median* of the points $p_1, \ldots, p_n$ (with weights $w_1, \ldots, w_n$). The minimization problem (6.27) has a long history ([87]), and has been referred to by many names, including the *Weber problem*, the *Fermat problem*, the *minisum problem*, and the *spatial median problem*. In [87], Wesolowsky suggests the *Euclidean Minisum Problem*.

A basic iteration procedure that solves (6.27) was proposed in 1937 by Weiszfeld ([84]). Consider the objective function

$$g(p) = \sum_{i=1}^{n} w_i ||p - p_i||_2.$$

If the points $p_1, \ldots, p_n$ are not collinear, then $g(p)$ is strictly convex and has a unique minimum. If $p_1, \ldots, p_n$ are collinear, then an optimal point must lie on the line through the given points (if not, one could project the claimed optimal point onto the line, thereby decreasing its distance to all the given points, to obtain a better point). In this case, the algorithm given in section 6.4.1.2 for points on the real line can be used (the L_2-distance reduces to the absolute value in one dimension).

The objective function is differentiable everywhere except at the given points:

$$\frac{\partial g}{\partial p}(p) = \sum_{i=1}^{n} \frac{w_i(p - p_i)}{||p - p_i||_2} \qquad \text{if } p^{(k)} \neq p_1, \ldots, p_n.$$

Setting the partial derivative to zero results in the equation

$$\sum_{i=1}^{n} \frac{w_i(p - p_i)}{||p - p_i||_2} = 0,$$

which cannot be solved explicitly for p. The Weiszfeld iteration replaces the p in the numerator by the $(k+1)$st iterate $p^{(k+1)}$ and the p in the denominator by the kth iterate $p^{(k)}$, and solves for $p^{(k+1)}$:

$$p^{(k+1)} = \begin{cases} \frac{\sum_{i=1}^{n} w_i ||p^{(k)} - p_i||_2^{-1} p_i}{\sum_{i=1}^{n} w_i ||p^{(k)} - p_i||_2^{-1}} & \text{if } p^{(k)} \neq p_1, \ldots, p_n \\ p_i & \text{if } p^{(k)} = p_i \end{cases} .$$

Here are some facts about this iteration (assuming the input points are not collinear).

- The iteration always converges. ([42])
- If no iterate $p^{(k)}$ is equal to one of the given points, then the iteration converges to the global minimum location of $g(p)$. ([42])
- The iteration can fail to converge to the global minimum location for a continuum of starting values $p^{(0)}$ because some iterate $p^{(k)}$ becomes equal to a non-optimal given point. ([7])
- If the optimal location is *not* at one of the given points, then convergence will be linear. ([41])
- If the optimal location *is* at one of the given points, then convergence can be linear, superlinear, or sublinear. ([41])

Since convergence to the global minimum location is not guaranteed, the iteration should be run more than once with different starting points.

It is conjectured in [7] that if the starting point is within the affine subspace P spanned by the given points, then the Weiszfeld iteration is guaranteed to converge to the global minimum location for all but a finite number of such starting points. If this conjecture is true, then the iteration will converge with high probability to the optimal location if one chooses a random starting point in P. Note that P is the entire space $\mathbf{R}^d$ if the $n-1$ vectors $p_n - p_1, p_n - p_2, \ldots, p_n - p_{n-1}$ span all of $\mathbf{R}^d$. If the given points are random, this event is very likely to occur if $n - 1 \geq d$. In regards to speeding up convergence, see [18] for an accelerated Weiszfeld procedure.

6.4.1.4 Minimizing a Weighted Sum of Cyclic L_1 Distances

In this section, we study the optimal translation problem on the real line when the feature domain is circular. In other words, we assume the feature points are real numbers which are defined only modulo T. Also, we assume the ground distance is the *cyclic L_1-distance*

$$d_{L_1,T}(x, y) = \min_{k \in \mathbf{Z}} |(x + kT) - y|.$$

If we identify feature values with arclengths on a circle of perimeter T, then $d_{L_1,T}(x, y)$ measures the smaller of the two arclengths that connect x to y along the circle. It is easy to prove that $0 \leq d_{L_1,T}(x, y) \leq T/2$. The intuition here is that a point should never have to travel more than half the circle to arrive at another point. Suppose, for example, that features are angles in radians ($T = 2\pi$), $x = \pi/4 = 45°$, and $y = 11\pi/6 = 330°$. Then $d_{L_1,T}(x, y) = d_{L_1,T}(\pi/4, 11\pi/6) = 5\pi/12 = 75°$, where the minimum is achieved at $k = 1$. It should also be clear that $d_{L_1,T}$ is cyclic with period T in both arguments: $d_{L_1,T}(x + T, y) = d_{L_1,T}(x, y + T) = d_{L_1,T}(x, y)$.

In order to apply the FT iteration for translation with the cyclic L_1-distance, we need to minimize

$$\mathrm{WORK}(F, \mathbf{x}, \mathbf{y} \oplus t) = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{L_1,T}(x_i, y_j + t).$$

over t. As is the L_1 case in section 6.4.1.2, the multidimensional case in $\mathbf{R}^K$ can be solved by solving K one-dimensional problems.[6] Therefore, consider the minimization problem

$$\min_{t \in \mathbf{R}} \mathrm{WORK}(F, \mathbf{x}, \mathbf{y} \oplus t) = \min_{t \in \mathbf{R}} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} \min_{k \in \mathbf{Z}} |(x_i + kT) - (y_j + t)| \tag{6.28}$$

given a fixed flow F. Since $d_{L_1,T}$ is cyclic with period T, the WORK function is cyclic in t with period T: $\mathrm{WORK}(F, \mathbf{x}, \mathbf{y} \oplus (t+T)) = \mathrm{WORK}(F, \mathbf{x}, \mathbf{y} \oplus t)$. Therefore, for every feasible flow F there will be a WORK minimizing translation $t \in [0, T)$. We can also assume that $x_i, y_j \in [0, T)$ since these numbers need only be defined up to a multiple of T when using ground distance $d_{L_1,T}$.

The inner minimization of (6.28) can be trivially rewritten as

$$\min_{k \in \mathbf{Z}} |(x_i + kT) - (y_j + t)| = \min_{k \in \mathbf{Z}} |kT - (y_j + t - x_i)|.$$

If we restrict $x_i, y_j, t \in [0, T)$, then $(y_j + t - x_i) \in (-T, 2T)$. The above minimum will never be achieved outside the set $\{ -1, 0, 1, 2 \}$, so

$$\min_{k \in \mathbf{Z}} |(x_i + kT) - (y_j + t)| = \min_{k \in \{ -1,0,1,2 \}} |(x_i + kT) - (y_j + t)| \quad \text{for } x_i, y_j, t \in [0, T).$$

If we let

$$h(t) = h(F, \mathbf{x}, \mathbf{y}, t) = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} \min_{k \in \{ -1,0,1,2 \}} |kT - (y_j + t - x_i)|, \tag{6.29}$$

then we have argued that

$$\min_{t \in \mathbf{R}} \mathrm{WORK}(F, \mathbf{x}, \mathbf{y} \oplus t) = \min_{t \in [0,T)} h(t) \qquad \text{if } x_i, y_j \in [0, T).$$

We now consider the function $h(t)$ in more detail.

---

[6] The period can be different in each dimension.

In order to better understand $h(t)$, we can partition the real line into intervals over which $\arg\min_{k\in\{-1,0,1,2\}} |kT - (y_j + t - x_i)|$ is constant. If we let $z_{ij} = x_i - y_j$, then $|kT - (y_j + t - x_i)| = |kT + z_{ij} - t|$, and

$$\arg\min_{k\in\{-1,0,1,2\}} |kT + z_{ij} - t| = \begin{cases} -1 & \text{if } -\infty \le t < -\frac{1}{2}T + z_{ij} \\ 0 & \text{if } -\frac{1}{2}T + z_{ij} \le t < \frac{1}{2}T + z_{ij} \\ 1 & \text{if } \frac{1}{2}T + z_{ij} \le t < \frac{3}{2}T + z_{ij} \\ 2 & \text{if } \frac{3}{2}T + z_{ij} \le t < +\infty \end{cases} .$$

The plan now is to divide each of the above intervals into two parts: one in which $kT+z_{ij} < t$, and the other in which $kT+z_{ij} \ge t$. This will allow us to express $\min_{k\in\{-1,0,1,2\}} |kT+z_{ij}-t|$ as a linear function in t over these subintervals (i.e. we can eliminate the min operator and the absolute value function). Toward this end, define the intervals

$$\begin{aligned} I_{ijkl} &= [(k+\tfrac{l}{2})T + z_{ij}, (k+\tfrac{l+1}{2})T + z_{ij}) && i=1,\ldots,m, j=1,\ldots,n, \\ & && k=-1,0,1,2, l=-1,0, \\ & && (k,l)\ne(-1,-1), (k,l)\ne(2,0), \\ I_{ij(-1)(-1)} &= [-\infty, -T+z_{ij}) && i=1,\ldots,m, j=1,\ldots,n, \quad \text{and} \\ I_{ij20} &= [2T+z_{ij}, +\infty) && i=1,\ldots,m, j=1,\ldots,n. \end{aligned}$$

Here

$$\begin{aligned} I_{ij(-1)(-1)} \cup I_{ij(-1)0} &= [-\infty, -\frac{1}{2}T + z_{ij}], \\ I_{ijk(-1)} \cup I_{ijk0} &= [(k-\frac{1}{2})T + z_{ij}, (k+\frac{1}{2})T + z_{ij}) \quad k=0,1, \quad \text{and} \\ I_{ij2(-1)} \cup I_{ij20} &= [\frac{3}{2}T + z_{ij}, +\infty), \end{aligned}$$

so that $\arg\min_{k\in\{-1,0,1,2\}} |kT + z_{ij} - t| = k^*$ for $t \in I_{ij(k^*)(-1)} \cup I_{ij(k^*)0}$. Furthermore,

$$\min_{k\in\{-1,0,1,2\}} |kT + z_{ij} - t| = \begin{cases} k^*T + z_{ij} - t & \text{if } t \in I_{ij(k^*)(-1)} \\ t - k^*T - z_{ij} & \text{if } t \in I_{ij(k^*)0} \end{cases} .$$

The above notation will now be used to rewrite $h(t)$ in a form that makes its structure more apparent.

We can get rid of the absolute value and minimization in (6.29) with the following algebraic manipulations:

$$
\begin{aligned}
h(t) &= \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \min_{k\in\{\,-1,0,1,2\,\}} |kT + z_{ij} - t| \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \sum_{k=-1}^{2}\sum_{l=-1}^{0} [t \in I_{ijkl}]|kT + z_{ij} - t| \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \sum_{k=-1}^{2} ([t \in I_{ijk(-1)}](kT + z_{ij} - t) + [t \in I_{ijk0}](t - kT - z_{ij})) \\
h(t) &= \left(\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \sum_{k=-1}^{2} ([t \in I_{ijk0}] - [t \in I_{ijk(-1)}])\right) t + \qquad (6.30) \\
&\quad \left(\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \sum_{k=-1}^{2} ([t \in I_{ijk(-1)}] - [t \in I_{ijk0}])(kT + z_{ij})\right).
\end{aligned}
$$

If we let $e_{ijkl} = (k + \frac{l}{2})T + z_{ij}$, then the breakpoint set

$$E = \{\, e_{ijkl} \;:\; i \in [1..m], j \in [1..n], k \in [-1..2], l \in [-1..0], (k,l) \neq (-1,-1) \,\}$$

divides the real line into intervals over which the coefficient of t and the coefficient of $1 = t^0$ in (6.30) are constant. On each such interval, the equation of $h(t)$ is that of a line. Therefore, $h(t)$ is a piecewise linear function of t.

The minimum of $h(t)$ over $[0, T)$ can be computed by visiting the breakpoints e_{ijkl} in sorted order, updating the line equation for $h(t)$ as we go along. The line function for $h(t)$ at $t = -\infty$ is given by

$$h(-\infty) = \left(-\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\right) t + \left(-T\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} z_{ij}\right).$$

This follows from (6.30) and the fact that $-\infty \in I_{ij(-1)(-1)}$. Thus the sweep algorithm sets the initial slope m to m_0 and the initial intercept b to b_0, where

$$m_0 = -\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} \qquad \text{and} \qquad b_0 = -T\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} z_{ij}.$$

There are two types of elementary steps over a breakpoint $t = e_{ijkl}$. In the case $l = -1$, the sweep line moves from $I_{ij(k-1)0}$ into $I_{ijk(-1)}$. By subtracting the $I_{ij(k-1)0}$ terms and adding in the $I_{ijk(-1)}$ terms in (6.30), we see that the updates to the slope and intercept when $l = -1$ are

$$\begin{aligned} m &\leftarrow m - f_{ij} + (-f_{ij}) = m - 2f_{ij} \quad \text{and} &(6.31)\\ b &\leftarrow b - (-f_{ij}((k-1)T + z_{ij})) + f_{ij}(kT + z_{ij}) \\ &= b + f_{ij}((2k-1)T + 2z_{ij}). &(6.32)\end{aligned}$$

In the case $l = 0$, the sweep line moves from $I_{ijk(-1)}$ into I_{ijk0}. From (6.30), we see that the updates to the slope and intercept when $l = 0$ are

$$\begin{aligned} m &\leftarrow m - (-f_{ij}) + f_{ij} = m + 2f_{ij} \quad \text{and} &(6.33)\\ b &\leftarrow b - f_{ij}(kT + z_{ij}) + (-f_{ij}(kT + z_{ij})) \\ &= b - 2f_{ij}(kT + z_{ij}). &(6.34)\end{aligned}$$

The sweep algorithm maintains the minimum value seen so far as it proceeds from $t = -\infty$ to $t = \infty$. The value of the function $h(t)$ is checked at any breakpoint $0 \leq t = e_{ijkl} < T$ at which the slope of the line equation for $h(t)$ changes from negative to positive. The locations of such sign changes in slope are local minimum locations for $h(t)$ in $[0, T)$. Computing h at a local minimum location $t = e_{ijkl}$ is done via $h(e_{ijkl}) = me_{ijkl} + b$, where m and b are the slope and intercept *after* the update for passing e_{ijkl}. Since we want to compute the minimum of $h(t)$ over $t \in [0, T)$, we must also check the value of $h(0)$ when we have the formula for $h(t)$ over the interval that contains zero. Finally, we can stop the sweep once we reach a breakpoint $e_{ijkl} \geq T$.

One final note to make is that at most $m + n - 1$ of the mn values f_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$, are nonzero if $F = (f_{ij})$ is an optimal vertex flow, as is returned by the transportation simplex algorithm ([32]). There is no reason to stop the sweep at e_{ijkl} for which $f_{ij} = 0$ since the values of m and b do not change at these points (see the update formulae (6.31)–(6.34)). This is obvious since the summation for $h(t)$ in (6.29) is the same with or without the (i, j)th term when $f_{ij} = 0$. The desired minimum can be computed by sweeping over the set

$$E' = \{\ e_{ijkl}\ :\ f_{ij} \neq 0, i \in [1..m], j \in [1..n], k \in [-1..2], l \in [-1..0], (k, l) \neq (-1, -1)\ \}$$

instead of the set E. Note that $|E| = 7mn$, while $|E'| \leq 7(m + n - 1)$. The sorting of the points in E' takes time $O((m + n)\log(m + n))$, and then the sweep over the points in E' takes time $O(m + n)$ since only a constant amount of work needs to be done at each elementary step.

6.4.2 Euclidean and Similarity Transformations

The optimal transformation problem for $\mathcal{G} = \mathcal{E}$, the group of Euclidean transformations, and $d = L_2^2$ is

$$\min_{(R,t)\in\mathcal{E}} \sum_{k=1}^{N} c_k ||a_k - (Rb_k + t)||_2^2,$$

where R is a rotation matrix. For a fixed R, the optimal translation must be $t^*(R) = \overline{a} - R\overline{b}$. Thus, the optimal Euclidean problem reduces to

$$\min_R \sum_{k=1}^{N} ||\widehat{a}_k - R\widehat{b}_k||_2^2 = \min_R ||\widehat{A} - R\widehat{B}||_{\mathrm{F}}^2, \tag{6.35}$$

where the columns of $\widehat{A}$ and $\widehat{B}$ are the vectors $\widehat{a}_k = \sqrt{c_k}(a_k - \overline{a})$ and $\widehat{b}_k = \sqrt{c_k}(b_k - \overline{b})$, and $||\cdot||_{\mathrm{F}}$ denotes the Frobenius matrix norm ([25]). Here we have also used the assumption that the c_k are nonnegative. The best rotation problem (6.35) is solved completely in [81]. The minimization problem (6.35) is easier to solve if we only require that R is orthogonal (i.e. we drop the requirement $\det(R) = 1$). Under this assumption, (6.35) is known as the *orthogonal Procrustes problem* ([25]). If $U\Sigma V^T$ is an SVD of $\widehat{A}\widehat{B}^T$, then the minimum value is $||\widehat{A}||_{\mathrm{F}}^2 + ||\widehat{B}||_{\mathrm{F}}^2 - 2\mathrm{tr}(\Sigma)$, and is achieved at $R = UV^T$.

The optimal transformation problem for $\mathcal{G} = \mathcal{S}$, the set of similarity transformations, allows for an additional scaling factor:

$$\min_{(s,R,t)\in\mathcal{S}} \sum_{k=1}^{N} c_k ||a_k - (sRb_k + t)||_2^2.$$

The special case of this problem in which $c_k \equiv 1/N$ is solved in [81] using the solution to (6.35). It is not difficult to repeat the analysis for general c_k and solve the optimal similarity problem as we have posed it, but we omit the details.

6.4.3 Linear and Affine Transformations

Finally, we consider the optimal transformation problem for linear and affine transformations with the L_2-distance squared. When $\mathcal{G} = \mathcal{L}$, the set of linear transformations, the optimal transformation problem becomes

$$\min_{L\in\mathcal{L}} \sum_{k=1}^{N} c_k ||a_k - Lb_k||_2^2.$$

Assuming that $c_k \geq 0$, an equivalent formulation is

$$\min_{L \in \mathcal{L}} \sum_{k=1}^{N} ||\widehat{a}_k - L\widehat{b}_k||_2^2 = \min_{L \in \mathcal{L}} ||\widehat{A} - L\widehat{B}||_{\mathrm{F}}^2,$$

where the columns of the matrices $\widehat{A}$ and $\widehat{B}$ are the vectors $\widehat{a}_k = \sqrt{c_k} a_k$ and $\widehat{b}_k = \sqrt{c_k} b_k$. An optimal linear transformation is $L^* = \widehat{A}\widehat{B}^\dagger$, where $\widehat{B}^\dagger$ is the pseudo-inverse ([25]) of $\widehat{B}$. In the case $\mathcal{G} = \mathcal{A}$, the set of affine transformations, the optimal transformation problem allows for an additional translation:

$$\min_{(L,t) \in \mathcal{A}} \sum_{k=1}^{N} c_k ||a_k - (Lb_k + t)||_2^2.$$

The optimal translation for a fixed L is $t^*(L) = \overline{a} - L\overline{b}$. Hence, with $\widetilde{a}_k = a_k - \overline{a}$ and $\widetilde{b}_k = b_k - \overline{b}$,

$$\min_{(L,t) \in \mathcal{A}} \sum_{k=1}^{N} c_k ||a_k - (Lb_k + t)||_2^2 = \min_{L \in \mathcal{L}} \sum_{k=1}^{N} c_k ||\widetilde{a}_k - L\widetilde{b}_k||_2^2,$$

and the affine problem reduces to the linear problem.

6.5 Allowing Weight-Altering Transformations

When a transformation g changes the weights of the distributions that it acts upon, in general $\mathcal{F}(\mathbf{x}, \mathbf{y}) \neq \mathcal{F}(\mathbf{x}, g(\mathbf{y}))$. This is because the constraints that define the feasible flows between two distributions depend on the weights in the distributions. Recall that there were two steps in proving that WORK sequence is decreasing when distribution weights are unchanged: (1) $F^{(k+1)}$ is a better flow for $g^{(k+1)}$ than the flow $F^{(k)}$, and (2) $g^{(k+1)}$ is a better transformation for $F^{(k)}$ than the transformation $g^{(k)}$. The inequality (6.14) which expresses step (2) still holds when distribution weights are not fixed. This is because $g^{(k+1)}$ is optimal for flow $F^{(k)}$ over all allowable transformations, and $g^{(k)}$ is one of the allowable transformations. The inequality (6.13) which expresses step (1), however, may not hold when distribution weights are changed. The flow $F^{(k+1)}$ is optimal for transformation $g^{(k+1)}$ over all flows in $\mathcal{F}(\mathbf{x}, g^{(k+1)}(\mathbf{y}))$, but flow $F^{(k)}$ may not be in the set $\mathcal{F}(\mathbf{x}, g^{(k+1)}(\mathbf{y}))$ – the flow $F^{(k)}$ was chosen from the set $\mathcal{F}(\mathbf{x}, g^{(k)}(\mathbf{y}))$.

It is easy to see that inequality (6.13) will hold if

$$\mathcal{F}(\mathbf{x}, g^{(k+1)}(\mathbf{y})) \supseteq \mathcal{F}(\mathbf{x}, g^{(k)}(\mathbf{y})), \tag{6.36}$$

for then $F^{(k)} \in \mathcal{F}(\mathbf{x}, g^{(k)}(\mathbf{y}))$ implies $F^{(k)} \in \mathcal{F}(\mathbf{x}, g^{(k+1)}(\mathbf{y}))$. Thus when we have an "increasing" sequence of feasible regions as specified by condition (6.36), we are guaranteed to get a decreasing WORK sequence. This, however, is not the end of the story because we are really after a decreasing EMD sequence. Remember that $\mathrm{EMD}^{(k)}$ is equal to $\mathrm{WORK}^{(k)}$ divided by the smaller of the total weights of $\mathbf{x}$ and $g^{(k)}(\mathbf{y})$, and the weight $g^{(k)}(\mathbf{y})$ is no longer constant over k.

The problem outlined above is that the WORK and the EMD sequences are not related by a constant multiplicative factor. We can get around this problem by a change of variables that moves the minimum total weight normalization factor into the definition of the flow. An example of such a change of variables has already been given in section 4.5 on scale estimation, where the change of variables is $\widehat{f}_{ij} = f_{ij}/c$ and c is the total weight of the lighter of the two distributions being compared (in section 4.5, we used h_{ij} instead of $\widehat{f}_{ij}$ as the new variables). This change of variables yielded a collection of transportation problems (one for each c) with increasing feasible regions $\mathcal{F}((X, w/c), \mathbf{y})$ as c is decreased. It followed that $E(c)$, the EMD between $\mathbf{x}$ and the transformed $\mathbf{y}$ as a function of the transformation parameter c, decreases as c decreases. In this case, the distribution transformations are transformations that scale down all the weights in the distribution by a factor c and leave the distribution points unchanged.

The same change of variables allows the FT iteration to be applied with some sets of transformations which alter both the weights and the points of distributions. Such transformations may be needed, for example, if a distribution point contains the position of an image region with some property and the corresponding weight is the region area; applying a similarity transformation with non-unit scale to region positions causes a change in region areas. Next we show how to apply the FT iteration in this case, where a distribution point is (L, a, b, x, y) in a combined CIE-Lab color space and image position space. The feature point (L, a, b, x, y) with weight w is meant to indicate that there is a region in the image plane with area w that has centroid (x, y) and color (L, a, b).

We denote the distribution summaries of the image and the pattern as

$$\begin{aligned}
\mathbf{x} &= \{\, (x_1, w_1), \ldots, (x_m, w_m) \,\} = \{\, ((a_1, p_1), w_1), \ldots, ((a_m, p_m), w_m) \,\}, \quad \text{and} \\
\mathbf{y} &= \{\, (y_1, u_1), \ldots, (y_n, u_n) \,\} = \{\, ((b_1, q_1), u_1), \ldots, ((b_n, q_n), u_n) \,\},
\end{aligned}$$

respectively, where $x_i = (a_i, p_i)$ divides feature point x_i into its color components a_i and position components p_i, and $y_j = (b_j, q_j)$ divides feature point y_j into its color components b_j and position components q_j. We assume that the ground distance d_{cp} in the combined color-position space is given by

$$d_{\mathrm{cp}}(x,y) = d_{\mathrm{cp}}((a,p),(b,q)) = d_{L_2}(a,b) + \lambda d_{L_2^2}(p,q),$$

where the parameter λ trades off between distance in color space and distance in position space.[7] We also assume that the weight normalization $w_\Sigma = u_\Sigma = 1$. Finally, we denote similarity transformations by $g = (s, \theta, t)$. The action of g on distribution $\mathbf{y}$ is defined by

$$g((b_j, q_j), u_j) = ((b_j, sR_\theta q_j + t), \kappa s^2 u_j),$$

where κ is a constant factor relating the scale s in positional units to the corresponding scale in area units (which need not be exactly the square of the position units since we have assumed $u_\Sigma = 1$). In the analysis that follows, we define the area scale $c = \kappa s^2$.

Our EMD under similarity transformation problem is

$$\mathrm{EMD}_{\mathcal{S}}(\mathbf{x},\mathbf{y}) = \min_{g\in\mathcal{S}} \min_{F\in\mathcal{F}(\mathbf{x},g(\mathbf{y}))} \frac{\sum_{i=1}^m \sum_{j=1}^n f_{ij}(d_{L_2}(a_i,b_j) + \lambda d_{L_2^2}(p_i, g(q_j)))}{c}.$$

If we let $\widehat{f}_{ij} = f_{ij}/c$, then the problem becomes

$$\mathrm{EMD}_{\mathcal{S}}(\mathbf{x},\mathbf{y}) = \min_{g\in\mathcal{S}} \min_{\widehat{F}\in\widehat{\mathcal{F}}(\mathbf{x},g(\mathbf{y}))} \sum_{i=1}^m \sum_{j=1}^n \widehat{f}_{ij}(d_{L_2}(a_i,b_j) + \lambda d_{L_2^2}(p_i, g(q_j))),$$

where the feasible region $\widehat{\mathcal{F}}(\mathbf{x}, g(\mathbf{y})) = \mathcal{F}((X, w/c), \mathbf{y})$ as defined in section 4.5 by conditions (4.17), (4.18), and (4.19).

The minimization problems to be solved by the FT iteration are

$$\widehat{F}^{(k)} = \arg\left(\min_{\widehat{F}\in\widehat{\mathcal{F}}(\mathbf{x},g^{(k)}(\mathbf{y}))} \sum_{i=1}^m \sum_{j=1}^n \widehat{f}_{ij}(d_{L_2}(a_i,b_j) + \lambda d_{L_2^2}(p_i, g^{(k)}(q_j)))\right), \quad (6.37)$$

$$g^{(k+1)} = \arg\left(\min_{g\in\mathcal{S}} \sum_{i=1}^m \sum_{j=1}^n \widehat{f}_{ij}^{(k)}(d_{L_2}(a_i,b_j) + \lambda d_{L_2^2}(p_i, g(q_j)))\right). \quad (6.38)$$

Since g does not change the color component of a distribution point, we can compute $g^{(k+1)}$

[7]If colors are represented in CIE-Lab space, then the Euclidean distance is the natural choice for a distance in color space. The analysis that follows, however, does not require the distance in color space to be L_2.

defined in equation (6.38) by solving

$$g^{(k+1)} = \arg\left(\min_{g\in\mathcal{S}} \sum_{i=1}^{m}\sum_{j=1}^{n} \widehat{f}_{ij}^{(k)} d_{L_2^2}(p_i, g(q_j))\right). \tag{6.39}$$

Section 6.4.2 discusses the solution to this optimal similarity transformation problem. Solving for $\widehat{F}^{(k)}$ in (6.37) is still a transportation problem.

If we define

$$\mathrm{EMD}^{(k)} = \sum_{i=1}^{m}\sum_{j=1}^{n} \widehat{f}_{ij}^{(k)} (d_{L_2}(a_i, b_j) + \lambda d_{L_2^2}(p_i, g^{(k)}(q_j))),$$

then, as previously argued, we will get a decreasing EMD sequence if we can guarantee

$$\widehat{\mathcal{F}}(\mathbf{x}, g^{(k+1)}(\mathbf{y})) \supseteq \widehat{\mathcal{F}}(\mathbf{x}, g^{(k)}(\mathbf{y})) \quad \forall k. \tag{6.40}$$

If $g^{(k)} = (s^{(k)}, \theta^{(k)}, t^{(k)})$, then (6.40) will hold if $s^{(k+1)} \leq s^{(k)}$ $\forall k$. So the FT iteration will yield a decreasing EMD sequence if we only allow scale to decrease as the iteration proceeds. For the specific case of allowing a similarity transformation, this can be accomplished as follows. In computing $g^{(k+1)}$, first perform the minimization over the set of similarity transformations as in (6.39) to get a similarity transformation $(s^{(k+1)}, \theta^{(k+1)}, t^{(k+1)})$. If $s^{(k+1)} \leq s^{(k)}$, then set $g^{(k+1)} = (s^{(k+1)}, \theta^{(k+1)}, t^{(k+1)})$. Otherwise, set $g^{(k+1)}$ to the best Euclidean transformation with fixed scale $s^{(k)} : g^{(k+1)} = (s^{(k)}, \theta^{(k+1)}, t^{(k+1)})$ where

$$(\theta^{(k+1)}, t^{(k+1)}) = \arg\left(\min_{g\in\mathcal{E}} \sum_{i=1}^{m}\sum_{j=1}^{n} \widehat{f}_{ij}^{(k)} d_{L_2^2}(p_i, g(s^{(k)} q_j))\right).$$

This optimal Euclidean transformation problem is discussed in section 6.4.2.

6.6 Some Specific Cases

There are some specific cases of transformation set, ground distance function, and feature space that are worth mentioning in our discussion of the EMD under transformation sets.

6.6.1 The Equal-Weight EMD under Translation with $d = L_2^2$

Recall from section 6.4.1.1 that if d is the L_2-distance squared, then the unique optimal translation for a fixed flow is given by the centroid difference

$$t^*_{L_2^2} = \overline{a} - \overline{b} = \frac{\sum_{k=1}^N c_k a_k}{c_\Sigma} - \frac{\sum_{k=1}^N c_k b_k}{c_\Sigma},$$

where the a_k, b_k, and c_k are defined by the distribution points and the fixed flow as in (6.17), (6.18), and (6.19). In terms of the original points and flow vector,

$$t^*_{L_2^2} = \frac{\sum_{i=1}^m \sum_{j=1}^n f_{ij} x_i}{\sum_{i=1}^m \sum_{j=1}^n f_{ij}} - \frac{\sum_{i=1}^m \sum_{j=1}^n f_{ij} y_j}{\sum_{i=1}^m \sum_{j=1}^n f_{ij}}. \tag{6.41}$$

If $\mathbf{x}$ and $\mathbf{y}$ are equal-weight distributions (and $\gamma = 1$), then $\sum_{i=1}^m f_{ij} = u_j$, $\sum_{j=1}^n f_{ij} = w_i$, and $\sum_{i=1}^m \sum_{j=1}^n f_{ij} = w_\Sigma = u_\Sigma$ for any feasible flow $F = (f_{ij})$. Using these facts in equation (6.41) shows that the best translation for any feasible flow $F = (f_{ij})$ is $t^*_{L_2^2} = \overline{\mathbf{x}} - \overline{\mathbf{y}}$. Therefore, the FT iteration described in section 6.3 is not needed in the equal-weight case to compute $\mathrm{EMD}_{\mathcal{T},L_2^2}(\mathbf{x},\mathbf{y})$. Instead, simply translate $\mathbf{y}$ by $\overline{\mathbf{x}} - \overline{\mathbf{y}}$ (this lines up the centroids of $\mathbf{x}$ and $\mathbf{y}$) to get $\widehat{\mathbf{y}}$ and compute $\mathrm{EMD}(\mathbf{x}, \widehat{\mathbf{y}})$.

6.6.2 The Equal-Weight EMD under Translation on the Real Line

In this section, we assume that the ground distance is the absolute value between points on the real line ($d = L_1$). Recall the definition in section 4.3.2 of the CDF flow F^{CDF} between two equal-weight distributions $\mathbf{x} = (X, w)$ and $\mathbf{y} = (Y, u)$ on the real line:

$$f_{ij}^{\mathrm{CDF}} = |[W_{i-1}, W_i] \cap [U_{j-1}, U_j]|,$$

where

$$\begin{array}{rclcll} W_k &=& W(x_k) &=& \sum_{i=1}^k w_i & \text{and} \\ U_l &=& U(y_l) &=& \sum_{j=1}^l u_j. & \end{array}$$

Here the points and corresponding weights in the distributions are numbered according to increasing position along the real line: $x_1 < \cdots < x_m$ and $y_1 < \cdots < y_n$. In Theorem 5, we showed that the CDF flow F^{CDF} is an optimal flow between $\mathbf{x}$ and $\mathbf{y}$ if $d = L_1$. Now denote the translation of $\mathbf{y}$ by t as

$$\mathbf{y} \oplus t = \{\ (y_1 + t, u_1), (y_2 + t, u_2), \ldots, (y_n + t, u_n)\ \}.$$

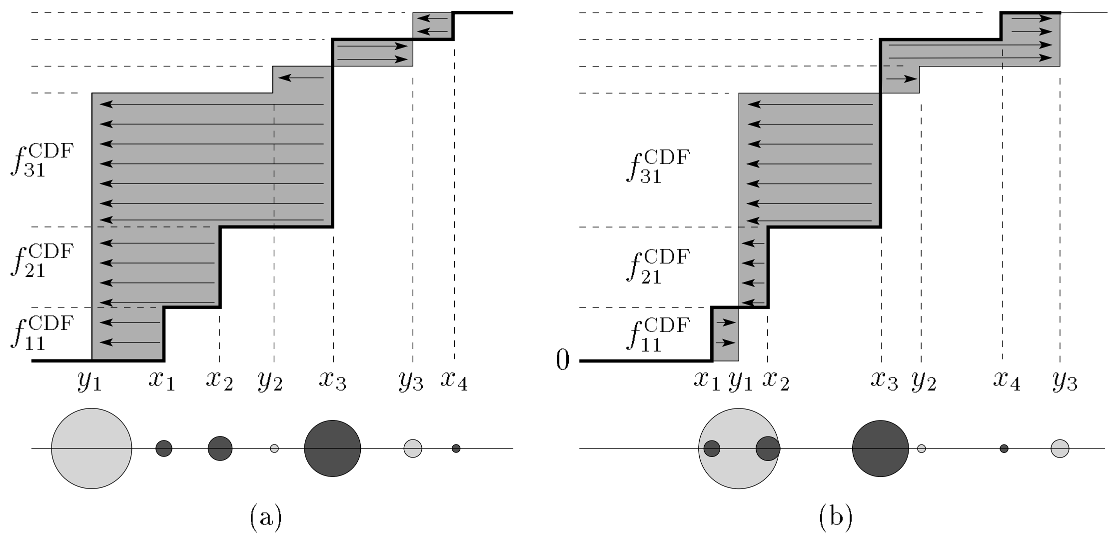


Figure 6.4: The Equal-Weight EMD under Translation in 1D with $d = L_1$. The same flow F^{CDF} is optimal for (a) $\mathbf{x}$ and $\mathbf{y}$, and (b) $\mathbf{x}$ and $\mathbf{y} \oplus t$. We re-use the labels y_j in (b) instead of using $y_j + t$ in order to make all the labels fit in the given space.

Since the sorted order of the points of $\mathbf{y} \oplus t$ is the same as the sorted order of the points of $\mathbf{y}$, and the weights of $\mathbf{y} \oplus t$ are the same as the weights of $\mathbf{y}$, the CDF flow between $\mathbf{x}$ and $\mathbf{y}$ is the same as the CDF flow between $\mathbf{x}$ and $\mathbf{y} \oplus t$. By Theorem 5, this CDF flow is also an optimal flow between $\mathbf{x}$ and $\mathbf{y} \oplus t$. See Figure 6.4 for an example.

Now for fixed t, the optimal transformation step (6.11) in the FT iteration is to compute

$$\delta(t) = \min_{F \in \mathcal{F}(\mathbf{x},\mathbf{y})} \mathrm{WORK}(F^{\mathrm{CDF}}, \mathbf{x}, \mathbf{y} \oplus t).$$

Since the CDF flow is optimal for every $t \in \mathbf{R}$,

$$\delta(t) = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}^{\mathrm{CDF}} |x_i - (y_j + t)|.$$

Rewriting the 2D index as a 1D index, we have

$$\delta(t) = \sum_{k=1}^{N} f_k^{\mathrm{CDF}} |z_k - t|.$$

Functions of this form were studied extensively in section 6.4.1.2 where we gave the solution to this "minisum problem" on the line. This solution gives us the EMD under translation since

$$\mathrm{EMD}_{\mathcal{T}}(\mathbf{x},\mathbf{y}) = \frac{\min_{F\in\mathcal{F}(\mathbf{x},\mathbf{y}),t\in\mathbf{R}} \mathrm{WORK}(F^{\mathrm{CDF}},\mathbf{x},\mathbf{y}\oplus t)}{\min(w_\Sigma,u_\Sigma)} = \frac{\min_{t\in\mathbf{R}}\delta(t)}{\min(w_\Sigma,u_\Sigma)}.$$

The function $\delta(t)$ is piecewise linear with monotonic slope increasing from a negative value at $t=-\infty$ to a positive value at $t=+\infty$ (see Figures 6.2 and 6.3). Thus, $\delta(t)$ is convex and has its minimum at a point t at which the slope first becomes nonnegative.

Once the m points in $\mathbf{x}$ and the n points in $\mathbf{y}$ have been sorted, the CDF flow F^{CDF} can be computed in $\Theta(m+n)$ time using the second algorithm labelled EMD_1 given on page 78 in section 4.3.2. The result of this algorithm is an array of $\Theta(m+n)$ records containing a pair (i,j) and the flow value f_{ij}. Any pair (i,j) not appearing in this array has flow value $f_{ij}=0$. To compute the optimal translation (and the actual value of the EMD under translation) using the results in section 6.4.1.2, we need to find the first index k at which the slope m_k,

$$\begin{aligned} m_0 &= -\sum_{k=1}^{N} f_k^{\mathrm{CDF}} = -w_\Sigma = -u_\Sigma, \\ m_{k+1} &= m_k + 2f_{k+1}^{\mathrm{CDF}}, \end{aligned}$$

is greater than or equal to zero. This can be done by sorting the returned flow pairs (i,j) by increasing 1D index k in time $\Theta((m+n)\log(m+n))$, and then tracking the slope m_k while marching through the previously sorted flow array in time $O(m+n)$ (the array traversal can stop once it reaches k such that $m_k \geq 0$). This algorithm to compute the EMD under translation between equal-weight distributions on the line requires $\Theta((m+n)\log(m+n))$ time.

6.6.3 The Equal-Weight EMD under $\mathcal{G}$ with $m=n=2$

In this section, we consider the problem of matching equal-weight distributions $\mathbf{x}$ and $\mathbf{y}$ with two points each ($m=n=2$) under a general transformation set $\mathcal{G}$, where $g\in\mathcal{G}$ changes only the points in a distribution. Without loss of generality, we assume the unit-weight normalizations $w_\Sigma=u_\Sigma=1$.

The conditions which define the feasible flow set $\mathcal{F}(\mathbf{x},\mathbf{y})=\mathcal{F}(\mathbf{x},g(\mathbf{y}))$ are $f_{11}\geq 0$, $f_{12}\geq 0$, $f_{21}\geq 0$, $f_{22}\geq 0$, and

$$f_{11}+f_{12} = w_1, \tag{6.42}$$

$$f_{21}+f_{22} = w_2 = 1-w_1, \tag{6.43}$$

$$
\begin{aligned}
f_{11} + f_{21} &= u_1, && (6.44)\\
f_{12} + f_{22} &= u_2 = 1 - u_1. && (6.45)
\end{aligned}
$$

Using (6.42)–(6.45), we can write all flow variables in terms of f_{11}:

$$
\left.\begin{aligned}
f_{12} &= w_1 - f_{11}\\
f_{21} &= u_1 - f_{11}, \quad \text{and}\\
f_{22} &= 1 - w_1 - f_{21} = 1 - (w_1 + u_1) + f_{11}.
\end{aligned}\right\} \tag{6.46}
$$

From (6.46), we see that

$$
\max(0, (w_1 + u_1) - 1) \le f_{11} \le \min(u_1, w_1) \tag{6.47}
$$

is a necessary condition for $F = (f_{ij})$ to be feasible since flow variables must be nonnegative. Also, every f_{11} which satisfies (6.47) defines a feasible flow F according to equations (6.46). Thus, we have argued that the set of feasible flows $\mathcal{F}(\mathbf{x}, \mathbf{y})$ is also defined by conditions (6.46) and (6.47).[8]

Using (6.46), we may write the work done by F to match $\mathbf{x}$ and $\mathbf{y}$ as

$$
\begin{aligned}
\mathrm{WORK}(F, \mathbf{x}, \mathbf{y}) &= f_{11}d_{11} + f_{12}d_{12} + f_{21}d_{21} + f_{22}d_{22}\\
&= ((d_{11} + d_{22}) - (d_{12} + d_{21}))f_{11}\\
&\quad + (w_1 d_{12} + u_1 d_{21} + (1 - (w_1 + u_1))d_{22}).
\end{aligned}
$$

When we allow a transformation $g \in \mathcal{G}$, the point distances d_{ij} become functions $d_{ij}(g)$ of g, and we have

$$
\begin{aligned}
\mathrm{WORK}(F, \mathbf{x}, g(\mathbf{y})) &= ((d_{11}(g) + d_{22}(g)) - (d_{12}(g) + d_{21}(g)))f_{11} \qquad (6.48)\\
&\quad + (w_1 d_{12}(g) + u_1 d_{21}(g) + (1 - (w_1 + u_1))d_{22}(g)).
\end{aligned}
$$

Since $w_\Sigma = u_\Sigma = 1$,

$$
\mathrm{EMD}_{\mathcal{G}}(\mathbf{x}, \mathbf{y}) = \min_{F \in \mathcal{F}(\mathbf{x},\mathbf{y}), g \in \mathcal{G}} \mathrm{WORK}(F, \mathbf{x}, g(\mathbf{y})). \tag{6.49}
$$

From (6.48), we see that the minimum in (6.49) must be achieved at one of two feasible

[8]Note that $\mathcal{F}(\mathbf{x}, \mathbf{y})$ defined by (6.47) is nonempty. Since distribution weights are nonnegative, we have $\min(u_1, w_1) \ge 0$. Without loss of generality, suppose $w_1 \le u_1$. Then $\min(u_1, w_1) - ((w_1 + u_1) - 1) = 1 - u_1 \ge 0$ (since $u_\Sigma = 1$, $u_1 \ge 0$ implies $u_1 \le 1$), and hence $\min(u_1, w_1) \ge \max(0, (w_1 + u_1) - 1)$. The case $u_1 \le w_1$ is similar.

flows F^1 or F^2. More precisely, an optimal F for a given g is

$$f_{11} = \begin{cases} f_{11}^1 = \min(u_1, w_1) & \text{if } d_{11}(g) + d_{22}(g) < d_{12}(g) + d_{21}(g) \\ f_{11}^2 = \max(0, (u_1 + w_1) - 1) & \text{if } d_{11}(g) + d_{22}(g) \geq d_{12}(g) + d_{21}(g) \end{cases},$$

where f_{12}, f_{21}, and f_{22} follow from (6.46).

Since we know that the minimum in (6.49) is achieved at one of the flows F^1 or F^2 given above, we can compute

$$\mathrm{EMD}_{\mathcal{G}}(\mathbf{x}, \mathbf{y}) = \min\left(\min_{g \in \mathcal{G}} \mathrm{WORK}(F^1, \mathbf{x}, g(\mathbf{y})), \min_{g \in \mathcal{G}} \mathrm{WORK}(F^2, \mathbf{x}, g(\mathbf{y}))\right)$$

by solving an optimal transformation problem for each of F^1 and F^2.

6.7 Global Convergence in $\mathcal{F} \times \mathcal{G}$?

This section is devoted to the following question: Under what conditions does the FT iteration converge to the global minimum of $\mathrm{WORK}(F, \mathbf{x}, g(\mathbf{y})) : \mathcal{F}(\mathbf{x}, \mathbf{y}) \times \mathcal{G} \longrightarrow \mathbf{R}_{\geq 0}$? There are many parameters here, including (1) the transformation set $\mathcal{G}$, (2) the ground distance d, (3) the dimension of the underlying points, (4) whether or not $\mathbf{x}$ and $\mathbf{y}$ are equal-weight distributions, and (5) the distributions $\mathbf{x}$ and $\mathbf{y}$ themselves. Here we shall only consider transformations which modify the distribution locations but not their corresponding weights.

In section 6.7.1, we consider the problem for unequal-weight distributions $\mathbf{x}$ and $\mathbf{y}$. We call this the "partial matching" case because some of the weight in the heavier distribution will be unmatched. In different regions of $\mathcal{G}$, different parts of the heavier distribution may be used in an optimal flow, and this makes it impossible to prove a guarantee of global convergence.

In section 6.7.2, we argue that the FT iteration is guaranteed to converge to the global minimum of $\mathrm{WORK}(F, \mathbf{x}, g(\mathbf{y}))$ if either (1) there is a transformation g^* which is the unique optimal transformation for every feasible flow, or (2) there is a feasible flow F^* which is the unique optimal flow for every transformation. These may seem like highly constrained situations, but we have already encountered an example of (1), namely the EMD under translation between equal-weight distributions with ground distance $d = L_2^2$. We also discuss the effect of removing the uniqueness requirement from (1) and (2).

In section 6.7.3, we consider the case of matching a distribution to a translated version of itself. The EMD under translation is obviously zero in this perfect matching case. We briefly describe experiments which show that in practice the FT iteration converges to the global minimum of zero.

In section 6.7.4, we demonstrate that there can be transformations which are locally but not globally optimal even for equal-weight comparisons. We give an example of equal-weight distributions in the plane with two points each ($m = n = 2$) for which there are local minima in $\mathcal{F} \times \mathcal{T}$ for the L_2 and L_1 ground distances. We also show that the WORK function with the L_2^2 distance can have local minima if $\mathcal{G}$ is the group of rotations.

If there are local minima in $\mathcal{F} \times \mathcal{G}$, then it is hard to have a guarantee of convergence to the global minimum. If there is a local minimum at (F^0, g^0), then g^0 is locally optimal for F^0, and F^0 is locally optimal for g^0. But holding $g = g^0$ fixed yields a WORK function which is linear in F, so a locally optimal flow for g^0 must be a globally optimal flow for g^0. In many cases there are no local minima of the WORK function in g when F is held fixed (e.g. when $\mathcal{G} = \mathcal{T}$, $d = L_p$), so a locally optimal transformation for F^0 is a globally optimal transformation for F^0. We have already seen that the FT iteration gets stuck at (F, g) when F and g are mutually optimal for each other.

6.7.1 Partial Matching

When one distribution is heavier than the other, some of the mass in the heavier distribution is unmatched in a feasible flow. In different regions of $\mathcal{G}$, different parts of the heavier distribution may be used in an optimal flow. This fact allows one to develop examples that possess local minima.

Imagine a distribution $\mathbf{x}$ composed of two spatially separated sub-distributions $\widehat{\mathbf{x}}$ and $\widetilde{\mathbf{x}}$ which are equal-weight, say $\frac{1}{2}w_\Sigma$ each. Now consider matching $\mathbf{x}$ to a distribution $\mathbf{y}$ with weight $u_\Sigma = \frac{1}{2}w_\Sigma$. Of all the transformations g which place $g(\mathbf{y})$ in the $\widehat{\mathbf{x}}$ part of the point space, there will be an optimal transformation $\widehat{g}^*$. If the sub-distributions $\widehat{\mathbf{x}}$ and $\widetilde{\mathbf{x}}$ are separated enough, the corresponding flows will not involve any mass from $\widetilde{\mathbf{x}}$. Similarly, of all the transformations g which place $g(\mathbf{y})$ in the $\widetilde{\mathbf{x}}$ part of the point space, there will be an optimal transformation $\widetilde{g}^*$. Assuming that $\widehat{g}^*(\mathbf{y})$ does not match $\widehat{\mathbf{x}}$ equally as well as $\widetilde{g}^*(\mathbf{y})$ matches $\widetilde{\mathbf{x}}$, one of $\widehat{g}^*$ and $\widetilde{g}^*$ is only a locally optimal transformation. Figures 6.5 and 6.6 show examples in 1D and 2D, respectively.

We can also create examples in which there are as many local minima as we like if we allow the ratio u_Σ / w_Σ to be arbitrarily small. If $\mathbf{x}$ is L well-separated copies $\mathbf{y} \oplus t_l$ of $\mathbf{y}$, then $\mathrm{EMD}(\mathbf{x}, \mathbf{y} \oplus t_l) = 0$ for $l = 1, \ldots, L$. We can produce $\geq L - 1$ only locally optimal translations by slightly perturbing the points in each copy of $\mathbf{y}$. In general, there may be no overlap in the mass of the heavier distribution used to match the mass in the lighter distribution in different parts of the transformation space.

(a)

(b)

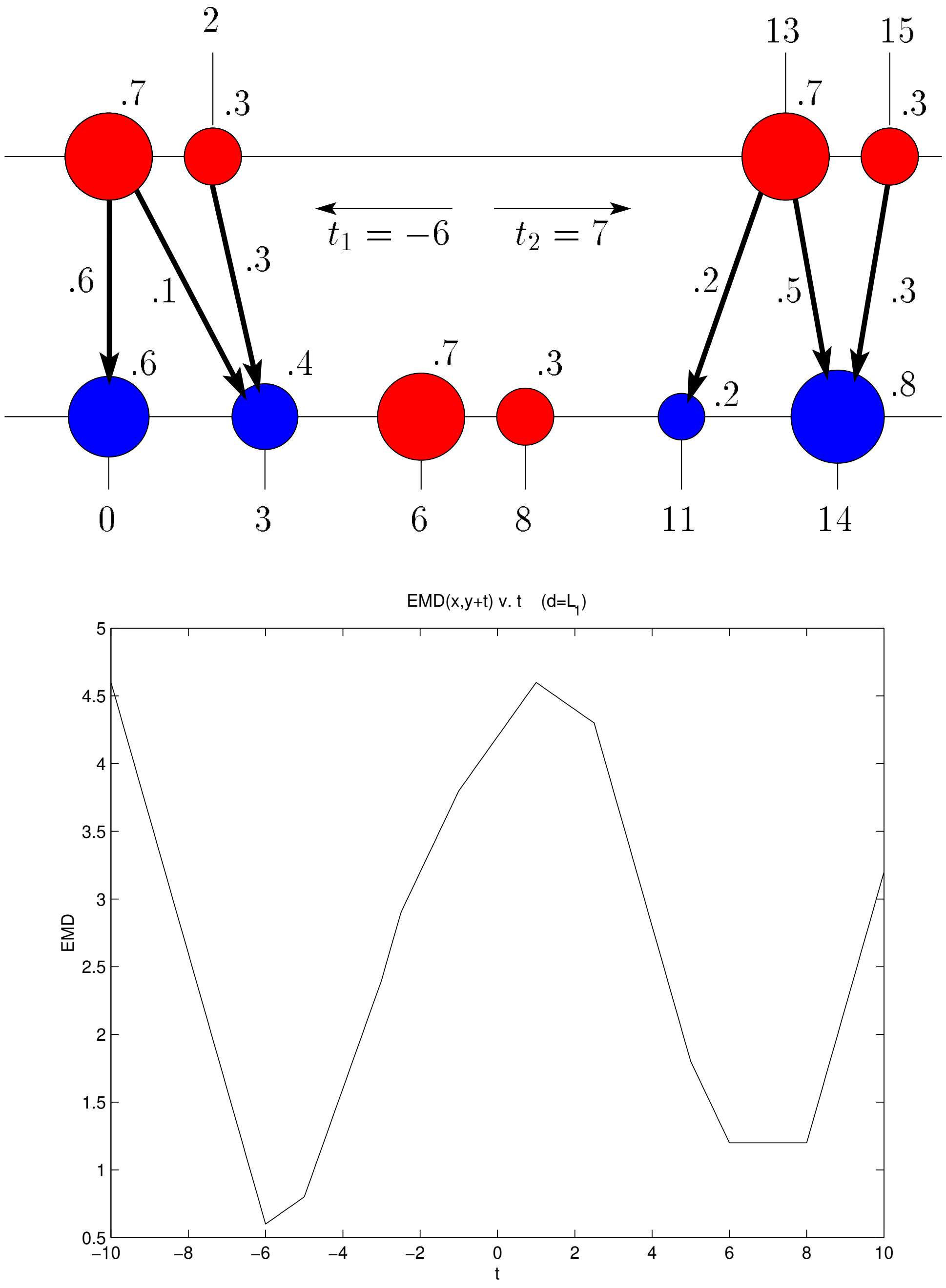


Figure 6.5: A Local Minimum in a 1D Partial Matching Case. (a) Distributions over the real line $\mathbf{x}$ and $\mathbf{y}$ are shown on the bottom line in blue and red, respectively. Translating $\mathbf{y}$ by $t_1 = -6$ gives $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t_1) = .6(0) + .1(3) + .3(1) = .6$, while translating $\mathbf{y}$ by $t_2 = 7$ gives $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t_2) = .2(2) + .5(1) + .3(1) = 1.2$. The translation t_1 yields the global minimum, while the translation t_2 yields a local minimum as one can see from (b) the graph of $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t)$ v. t.

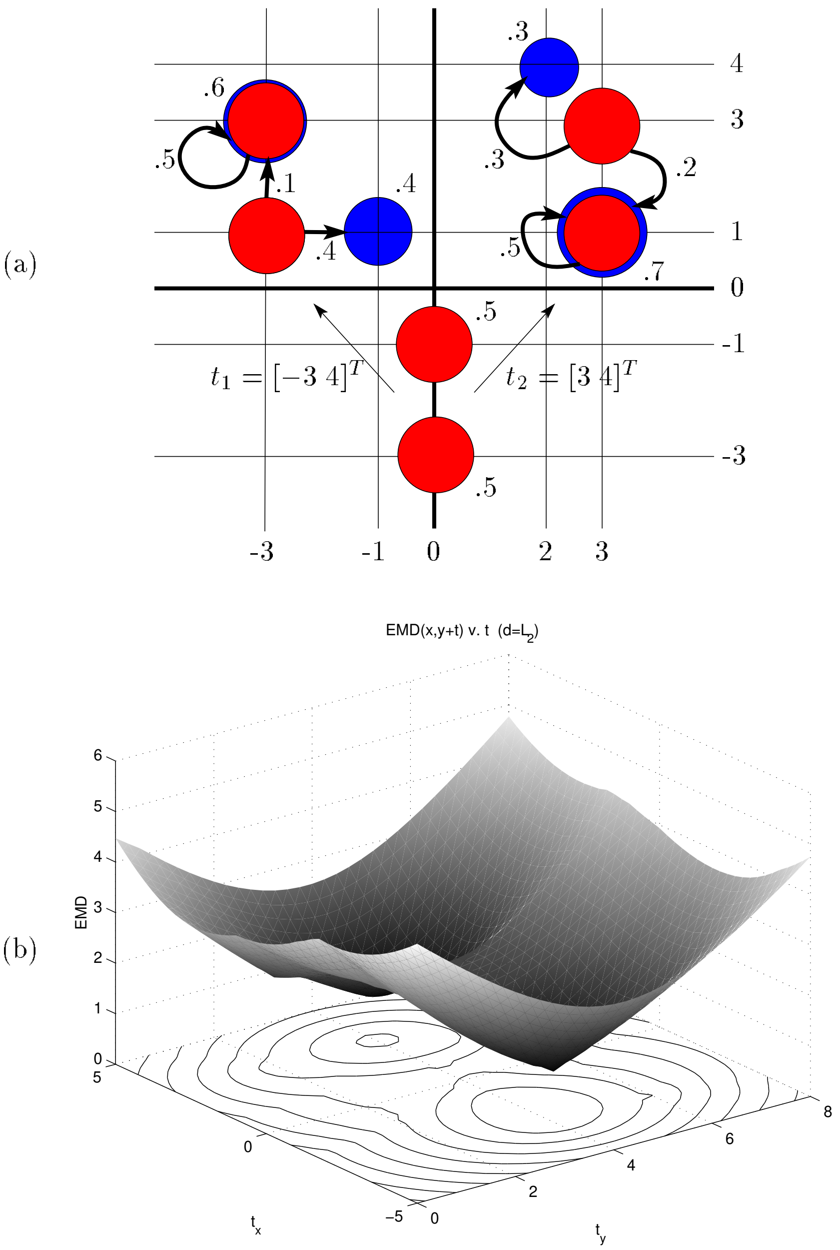


Figure 6.6: A Local Minimum in a 2D Partial Matching Case. (a) Distributions over the plane $\mathbf{x}$ and $\mathbf{y}$ are shown in blue and red, respectively. Translating $\mathbf{y}$ by $t_1 = [-3\ 4]^T$ gives $\mathrm{EMD}(\mathbf{x}, \mathbf{y} \oplus t_1) = .5(0) + .1(2) + .4(2) = 1.0$, while translating $\mathbf{y}$ by $t_2 = [3\ 4]^T$ gives $\mathrm{EMD}(\mathbf{x}, \mathbf{y} \oplus t_2) = .5(0) + .3(\sqrt{2}) + .2(2) \doteq .824$. The translation t_2 yields the global minimum, while the translation t_1 yields a local minimum as one can see from (b) the graph of $\mathrm{EMD}(\mathbf{x}, \mathbf{y} \oplus t)$ v. t.

6.7.2 One Optimal Flow or Transformation

If g^* is the unique optimal transformation for every feasible flow, then the WORK sequence converges to the global minimum work value in only a couple of iterations:

$$g^{(0)} \longrightarrow F^{(0)} \longrightarrow g^{(1)} = g^* \longrightarrow F^{(1)} \longrightarrow g^{(2)} = g^* \longrightarrow F^{(2)},$$

where $\text{WORK}(F^{(1)}, \mathbf{x}, g^{(1)}(\mathbf{y})) = \text{WORK}(F^{(1)}, \mathbf{x}, g^{(2)}(\mathbf{y})) = \text{WORK}(F^{(2)}, \mathbf{x}, g^{(2)}(\mathbf{y}))$ is the global minimum. We have already encountered such a case. For equal-weight distributions with $\mathcal{G} = \mathcal{T}$ and $d = L_2^2$, $t^* = \bar{x} - \bar{y}$ is the unique optimal translation for every feasible flow.

If there is a unique optimal flow F^* for every transformation, then the WORK sequence also converges to the global minimum work value in only a couple of iterations:

$$g^{(0)} \longrightarrow F^{(0)} = F^* \longrightarrow g^{(1)} \longrightarrow F^{(1)} = F^* \longrightarrow g^{(2)} \longrightarrow F^{(2)} = F^*,$$

where $\text{WORK}(F^{(1)}, \mathbf{x}, g^{(1)}(\mathbf{y})) = \text{WORK}(F^{(1)}, \mathbf{x}, g^{(2)}(\mathbf{y})) = \text{WORK}(F^{(2)}, \mathbf{x}, g^{(2)}(\mathbf{y}))$ is the global minimum. We have seen a case which comes close to meeting this requirement. For equal-weight distributions on the real line with $\mathcal{G} = \mathcal{T}$ and $d = L_1$, the CDF flow F^{CDF} is optimal for every $t \in \mathcal{T}$, although it is not necessarily the unique optimal flow for every $t \in \mathcal{T}$. We have been unable to rule out (even in this specific case) the possibility that

- $\widehat{F}$ and F^* are both optimal for some $g^{(k)} = \widehat{g}$,
- $\widehat{F}$ is returned by the transportation problem solver instead of F^*,
- $\widehat{g}$ is optimal for $F^{(k)} = \widehat{F}$,
- $\widehat{g}$ is returned by an optimal transformation solver as optimal for $F^{(k)}$, and
- $(\widehat{F}, \widehat{g})$ is not a globally optimal (flow,transformation) pair.

In this case, the FT iteration converges to $(\widehat{F}, \widehat{g})$ which is not globally optimal. We have also been unable to rule out the analogous possibility in the case when one transformation g^* is optimal for for every flow, but g^* is not the unique optimal transformation for every flow.

Now suppose that F^* is an optimal flow for every transformation. If the FT iteration ever reaches a transformation $g^{(k)}$ for which F^* is the unique optimal flow, then the iteration will converge to the global minimum work value. Here we are guaranteed that $F^{(k)} = F^*$, and we argued in section 6.3.2 that the FT iteration converges to the global minimum if the flow sequence ever reaches a globally optimal flow. Similarly, if g^* is an optimal transformation for every flow, then the FT iteration converges to the global minimum if it ever reaches a flow $F^{(k)}$ for which g^* is the unique optimal transformation.

6.7.3 A Perfect Match under Translation

It is clear that $\mathrm{EMD}_{\mathcal{T}}(\mathbf{y}, \mathbf{y} \oplus \Delta y) = 0$, where the translation that best aligns $\mathbf{y}$ and $\mathbf{y} \oplus \Delta y$ is $t^* = -\Delta y$. Is the FT iteration guaranteed to converge to the global minimum in this perfect matching case? We begin exploring this question by considering the EMD between $\mathbf{y}$ and $\mathbf{y} \oplus \Delta y$ without allowing a translation.

Theorem 13 *The EMD between a distribution and a translation of the distribution is*

$$\mathrm{EMD}(\mathbf{y}, \mathbf{y} \oplus \Delta y) = \begin{cases} ||\Delta y||_p & \textit{if } d = L_p \ (p \geq 1)\textit{, and} \\ ||\Delta y||_2^2 & \textit{if } d = L_2^2 \end{cases}.$$

Proof. The flow F^{ID} defined by $f_{ij}^{\mathrm{ID}} = \delta_{ij} u_j$, where $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ if $i \neq j$, gives normalized WORK values

$$\frac{\mathrm{WORK}(F^{\mathrm{ID}}, \mathbf{y}, \mathbf{y} \oplus \Delta y)}{u_\Sigma} = \begin{cases} ||\Delta y||_p & \text{if } d = L_p \ (p \geq 1), \text{ and} \\ ||\Delta y||_2^2 & \text{if } d = L_2^2 \end{cases}.$$

Since the EMD is the normalized WORK value for the optimal flow, we have

$$\mathrm{EMD}(\mathbf{y}, \mathbf{y} \oplus \Delta y) \leq \begin{cases} ||\Delta y||_p & \text{if } d = L_p \ (p \geq 1), \text{ and} \\ ||\Delta y||_2^2 & \text{if } d = L_2^2 \end{cases}. \tag{6.50}$$

Is there a feasible flow which requires less work than the size of Δy for either ground distance? The answer is "no". By the centroid lower bound theorems 6 and 7 (and the fact that $\overline{\mathbf{y} \oplus \Delta y} = \overline{\mathbf{y}} + \Delta y$), we know that

$$\mathrm{EMD}(\mathbf{y}, \mathbf{y} \oplus \Delta y) \geq \begin{cases} ||\Delta y||_p & \text{if } d = L_p \ (p \geq 1), \text{ and} \\ ||\Delta y||_2^2 & \text{if } d = L_2^2 \end{cases}. \tag{6.51}$$

The result follows from the opposite inequalities (6.50) and (6.51). ■

It is somewhat surprising that no matter how small or large the shift Δy, there is no better flow than matching a point to its translate.

If two points y_k and y_l have equal weights $u_k = u_l \equiv \alpha$, then F^{ID} will *not* be the unique optimal flow between $\mathbf{y}$ and $\mathbf{y} \oplus \Delta y$ if $\Delta y = y_l - y_k$ and $d = L_p$. In this case, the following slight modification $\widehat{F}$ of F^{ID} is also an optimal feasible flow:

$$\widehat{f}_{ij} = \begin{cases} u_i & \text{if } i = j,\ i \neq k,\ i \neq l, \\ u_k = u_l \equiv \alpha & \text{if } (i,j) = (k,l) \text{ or } (i,j) = (l,k), \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

When $d = L_p$ and $\Delta y = y_l - y_k$, we have

$$\begin{array}{lllll} d_{kk} & = & d(y_k, y_k + \Delta y) & = & \|\Delta y\|_p, \\ d_{ll} & = & d(y_l, y_l + \Delta y) & = & \|\Delta y\|_p, \\ d_{kl} & = & d(y_k, y_l + \Delta y) & = & 2\|\Delta y\|_p, \quad \text{and} \\ d_{lk} & = & d(y_l, y_k + \Delta y) & = & 0. \end{array}$$

In order to match the mass at y_k and y_l, the flow F^{ID} spends an amount of work equal to $\alpha d_{kk} + \alpha d_{ll} = 2\|\Delta y\|_p\alpha$, while the flow $\widehat{F}$ spends the same amount of work $\alpha d_{kl} + \alpha d_{lk} = 2\|\Delta y\|_p\alpha$ in a different way.

If we replace Δy by $\Delta y + t$, then we see that F^{ID} is an optimal flow between $\mathbf{y}$ and $(\mathbf{y} \oplus \Delta y) \oplus t = \mathbf{y} \oplus (\Delta y + t)$ for every translation t. Global convergence of the FT iteration is guaranteed when F^{ID} is the unique optimal flow for every t (see section 6.7.2). From the above discussion, however, we know that F^{ID} will *not* be the unique optimal flow for $t = (y_l - y_k) - \Delta y$ if $u_k = u_l$. On the other hand, F^{ID} is optimal for every t, so it is potentially returned as $F^{(k)}$ for every translation iterate $t^{(k)}$. F^{ID} might be returned even when it is not the unique optimal flow for $t^{(k)}$. Of course, once the FT iteration reaches a flow iterate $F^{(k)} = F^{\mathrm{ID}}$, the corresponding WORK sequence immediately converges to the global minimum WORK of zero.

The transportation simplex algorithm used to compute $F^{(k)}$ is an iterative algorithm. There are a few common rules for computing an initial feasible solution to the transportation problem, including the northwest corner rule, Vogel's method, and Russell's method ([32]). Applying the northwest corner rule to the transportation problem specified by $\mathbf{y}$ and $(\mathbf{y} \oplus \Delta y) \oplus t$ results in F^{ID} as the initial feasible solution for every t. Since F^{ID} is optimal, the transportation simplex algorithm will return F^{ID} for every t when the northwest corner rule is used. In this case, the FT iteration is guaranteed to converge to the global minimum.

The northwest corner rule is faster than Vogel's and Russell's methods, but it produces an initial solution which is usually not as close to optimal. Consequently, more iterations are usually required with the northwest corner rule. In general, the transportation simplex algorithm will find an optimal solution faster using Vogel's or Russell's method because fewer iterations will be required. In constrast to the northwest corner rule, these methods use the transportation problem costs (which are the ground distances in the EMD context) to compute an initial solution.

The EMD code that we use in our work applies Russell's method. The initial solution computed by this method is not necessarily F^{ID}, so there is no guarantee that the transportation simplex algorithm will return F^{ID} if there is another optimal flow. In practice, however, the FT iteration always converged to the global minimum of zero in hundreds of randomly generated, perfect-translation examples with $d = L_2$. In these random examples, the points in $\mathbf{y}$ were chosen with coordinates uniformly distributed in $[0, 1]$, and we varied the point space dimension (1, 2, and higher dimensions), whether points all have the same weight or not (if not, then the weight vector u is random), and whether $\Delta y + t^{(0)}$ is the difference between two points in $\mathbf{y}$ or not (if not, then the initial translation $t^{(0)}$ is random).

6.7.4 Equal-Weight Comparisons with Local Minima

In section 6.7.1, we showed examples of unequal-weight comparisons with local minima. These local minima arose because different parts of the heavier distribution were used in an optimal flow for different areas of the transformation space. In this section, we show that there can be local minima even when all the mass in one distribution must be matched to all the mass in the other distribution everywhere in transformation space. This is the matching requirement imposed by the EMD when the distributions are equal-weight.

The main example of this section consists of two distributions over the plane, each having two points ($m = n = 2$). See Figure 6.7(a). We seek to match these distributions under translation using the L_2 and L_1 ground distances.

In section 6.6.3, we analyzed the $m = n = 2$ matching problem under $\mathcal{G}$. Recall that there are two feasible flows F^1 and F^2 such that

$$
\begin{aligned}
&F^1 \in \arg\min_{F \in \mathcal{F}(\mathbf{x},\mathbf{y})} \text{WORK}(F, \mathbf{x}, g(\mathbf{y})) \quad \text{if } d_{11}(g) + d_{22}(g) \leq d_{12}(g) + d_{21}(g) \quad \text{and} \\
&F^2 \in \arg\min_{F \in \mathcal{F}(\mathbf{x},\mathbf{y})} \text{WORK}(F, \mathbf{x}, g(\mathbf{y})) \quad \text{if } d_{11}(g) + d_{22}(g) \geq d_{12}(g) + d_{21}(g).
\end{aligned}
$$

Here $d_{ij}(g) = d(x_i, g(y_j))$. If $\mathcal{G} = \mathcal{T}$ and $d = L_p$ or $d = L_2^2$, then $d_{ij}(t) = d(x_i, y_j + t) = d(\delta_{ij}, t)$, where $\delta_{ij} = x_i - y_j$.

Now consider the sets

$$
\begin{aligned}
\mathcal{T}_1 &= \{\, t \,:\, d_{11}(t) + d_{22}(t) < d_{12}(t) + d_{21}(t) \,\} \quad \text{and} \\
\mathcal{T}_2 &= \{\, t \,:\, d_{11}(t) + d_{22}(t) > d_{12}(t) + d_{21}(t) \,\}.
\end{aligned}
$$

If $\mathcal{T}_1 = \emptyset$, then F^2 is optimal for every $t \in \mathcal{T}$; if $T_2 = \emptyset$, then F^1 is optimal for every

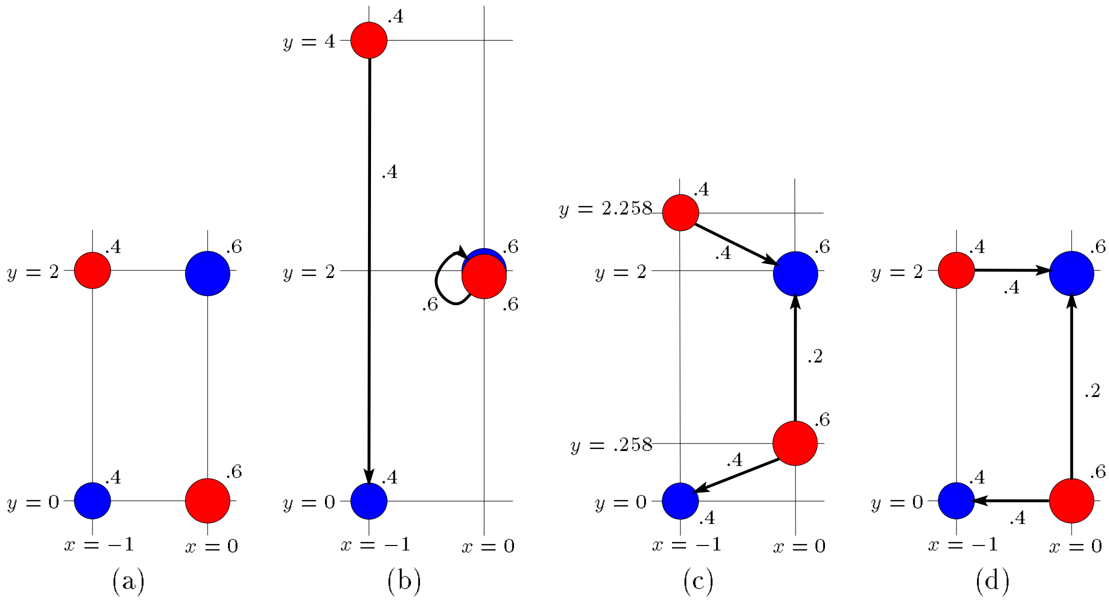


Figure 6.7: A Local Minimum in a 2D Equal-Weight Case. (a) Distributions $\mathbf{x}$ and $\mathbf{y}$ over the plane are shown in blue and red, respectively. (b) The translation $t = [0\ 2]^T$ of $\mathbf{y}$ is locally optimal for both $d = L_2$ and $d = L_1$. The optimal flow for this translation is the same for both ground distances, as is the EMD: $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t) = .6(0) + .4(4) = 1.6$. (c) The globally optimal translation of $\mathbf{y}$ for $d = L_2$ is $t = [0\ .258]^T$. This yields $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t) = .4\sqrt{1^2 + .258^2} + .2(2 - .258) + .4\sqrt{1^2 + .258^2} \doteq 1.175$. (d) The globally optimal translation of $\mathbf{y}$ for $d = L_1$ is $t = [0\ 0]^T$. This yields $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t) = .4(1) + .2(2) + .4(1) = 1.2$.

$t \in \mathcal{T}$. In our example, $x_1 = (-1,0)$, $x_2 = (0,2)$, $y_1 = (0,0)$, $y_2 = (-1,2)$, $\delta_{11} = [-1\ 0]^T$, $\delta_{12} = [0\ -2]^T$, $\delta_{21} = [0\ 2]^T$, and $\delta_{22} = [1\ 0]^T$. For $d = L_2^2$, it is easy to check that $(d_{11}(t) + d_{22}(t)) - (d_{12}(t) + d_{21}(t)) = -6$ for every $t \in \mathcal{T}$.[9] Thus $\mathcal{T}_2 = \emptyset$, and (as expected) the FT iteration converges to the global minimum on this example with $d = L_2^2$.

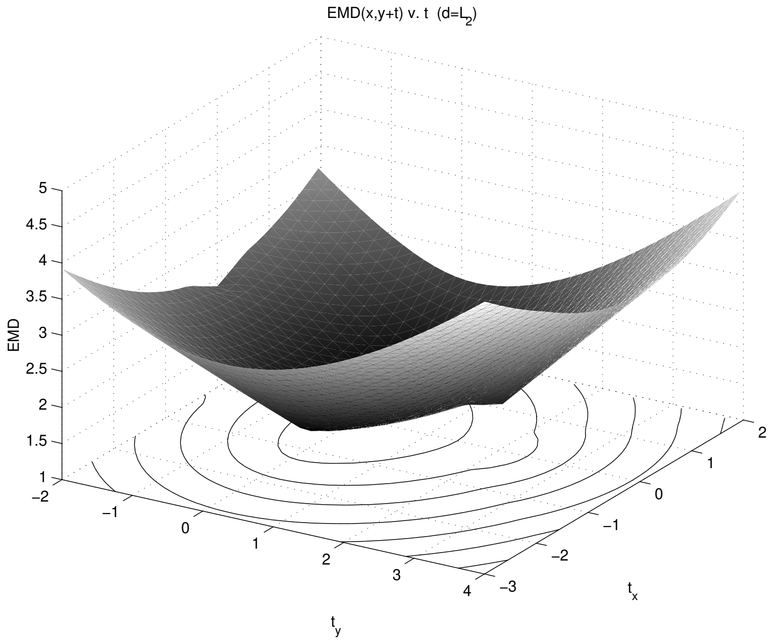
In general with $d = L_2$ or $d = L_1$, both $\mathcal{T}_1$ and $\mathcal{T}_2$ will be nonempty. This, however, does not imply that there are local minima. With $d = L_2$ or $d = L_1$, the functions $\text{WORK}(F^1, \mathbf{x}, \mathbf{y} \oplus t)$ and $\text{WORK}(F^2, \mathbf{x}, \mathbf{y} \oplus t)$ are convex in t. If the global minima of these functions occur in $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively, then the larger of these values is a local minimum of $\text{WORK}(F, \mathbf{x}, \mathbf{y} \oplus t)$ and the smaller is the global minimum. This is precisely what happens in the example of Figure 6.7(a). A local minimum can also exist along the boundary between $\mathcal{T}_1$ and $\mathcal{T}_2$ where $d_{11}(t) + d_{22}(t) = d_{12}(t) + d_{21}(t)$. More generally, a local minimum may occur in the interior of a transformation space region with constant optimal flow, or along the boundary between two such regions.

Figure 6.7(b) shows $\mathbf{y}$ translated by a locally optimal translation $t = [0\ 2]^T$ for both $d = L_2$ and $d = L_1$, along with the corresponding optimal flow for both cases. The globally optimal translations for the L_2 and L_1 distances are given in Figures 6.7(c) and 6.7(d), respectively. Finally, graphs of $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t)$ versus t for the L_2 and L_1 distances are shown in Figures 6.8 and 6.9, repectively. In Figure 6.10, we show that the locally optimal and globally optimal translations occur in regions of the translation space for which there are different optimal flows. We also prove that $t = [0\ 2]^T$ is locally optimal for both the L_2 and L_1 ground distances.

Let us now explicitly connect a local minimum in $\delta(g)$ over $\mathcal{G}$ with a local minimum of $\text{WORK}(F, \mathbf{x}, g(\mathbf{y}))$ over $\mathcal{F} \times \mathcal{G}$. Suppose that a local minimum of $\delta(g)$ occurs at g^0 in the interior of a region $R(F^*) = \{\, g \ :\ F^* \in \arg\min_{F \in \mathcal{F}} \text{WORK}(F, \mathbf{x}, g(\mathbf{y}))\,\}$. In words, g^0 is inside the region of transformation space with constant optimal flow F^*. Then there exists a neighborhood $N_\varepsilon^{\mathcal{G}}(g^0) \in \mathcal{G}$ around g^0 of size $\varepsilon > 0$ such that $N_\varepsilon^{\mathcal{G}}(g^0) \subseteq R(F^*)$ and $\delta(g) \geq \delta(g^0)$ for every $g \in N_\varepsilon^{\mathcal{G}}(g^0)$. For every $(F, g) \in \mathcal{F} \times N_\varepsilon^{\mathcal{G}}(g^0)$, we have $\text{WORK}(F, \mathbf{x}, g(\mathbf{y})) \geq \text{WORK}(F^*, \mathbf{x}, g(\mathbf{y})) = \delta(g) \geq \delta(g^0) = \text{WORK}(F^*, \mathbf{x}, g^0(\mathbf{y}))$. The first inequality follows from the optimality of F^* over $N_\varepsilon^{\mathcal{G}}(g^0)$, whereas the second inequality follows from the local optimality of g^0 over $N_\varepsilon^{\mathcal{G}}(g^0)$. Since $\text{WORK}(F, \mathbf{x}, g(\mathbf{y})) \geq \text{WORK}(F^*, \mathbf{x}, g^0(\mathbf{y}))$ for every $(F, g) \in \mathcal{F} \times N_\varepsilon^{\mathcal{G}}(g^0)$, there is a local minimum of $\text{WORK}(F, \mathbf{x}, g(\mathbf{y}))$ at (F^*, g^0). Of course, this logic depends upon being able to fit an open neighborhood inside $R(F^*)$, where F^* is optimal for g^0. This cannot be done, for example, if $R(F^*)$ is the single point g^0. A similar

[9] The fact that this quantity is independent of t is not specific to the particular example of this section. With $d = L_2^2$, $(d_{11}(t) + d_{22}(t)) - (d_{12}(t) + d_{21}(t)) = -2(x_1^T y_1 + x_2^T y_2 - x_1^T y_2 - x_2^T y_1)$ for every $t \in \mathcal{T}$. It follows that there will be one flow which is optimal for every translation.

(a)



(b)

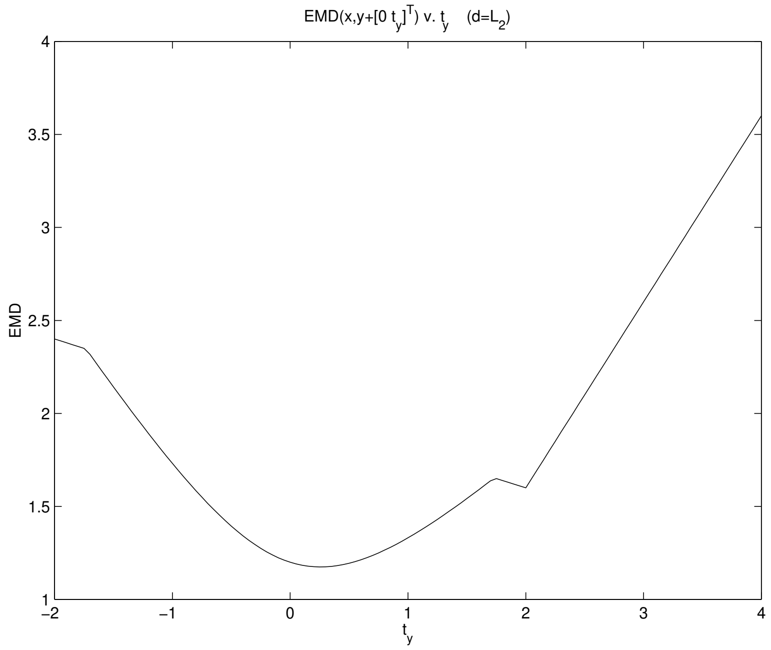
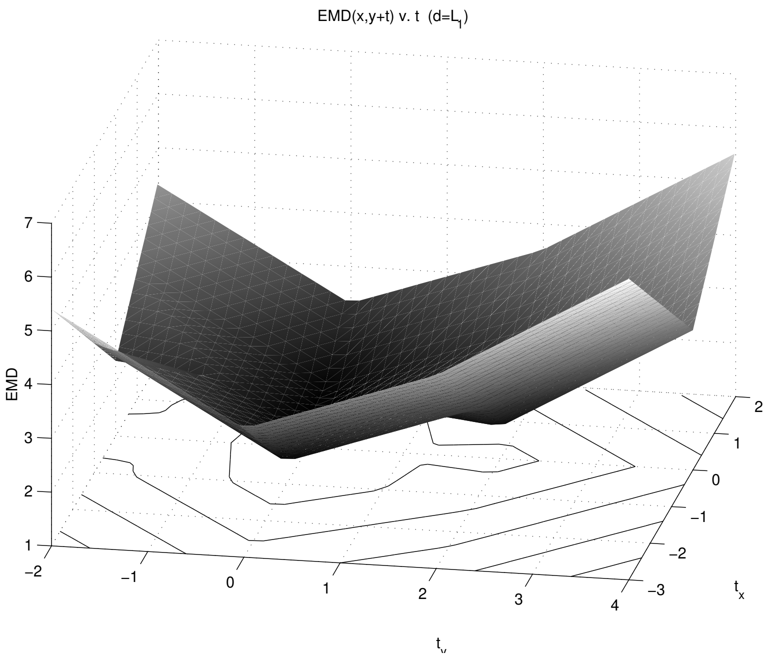


Figure 6.8: Graphs of EMD v. t Showing a Locally Optimal Translation for $d = L_2$. (a) EMD v. t for the 2×2 example shown in Figure 6.7(a) with $d = L_2$. There is a locally optimal translation at $t = [0\ 2]^T$, while the globally optimal translation is $t = [0\ .258]^T$. (b) A slice of the graph in (a) at $t_x = 0$.

(a)



(b)

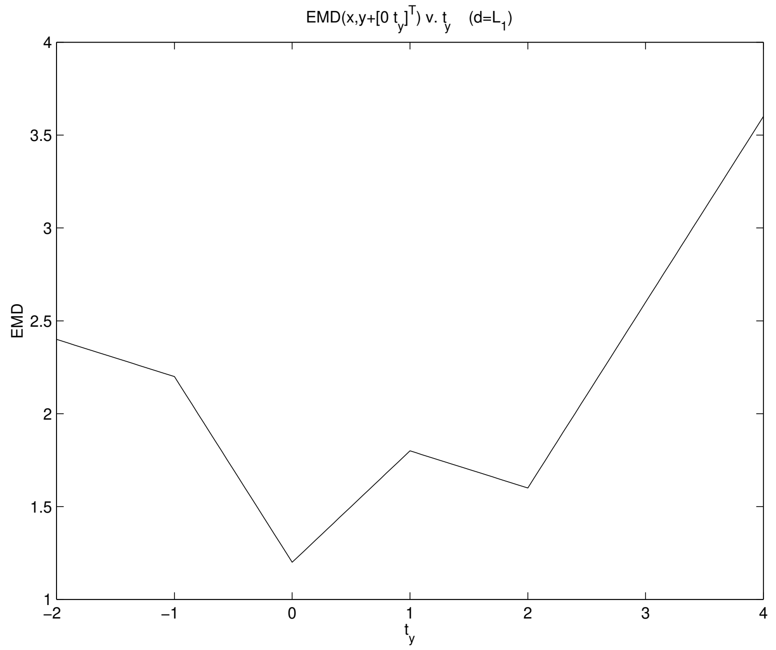


Figure 6.9: Graphs of EMD v. t Showing a Locally Optimal Translation for $d = L_1$. (a) EMD v. t for the 2×2 example shown in Figure 6.7(a) with $d = L_1$. There is a locally optimal translation at $t = [0\ 2]^T$, while the globally optimal translation is $t = [0\ 0]^T$. (b) A slice of the graph in (a) at $t_x = 0$.

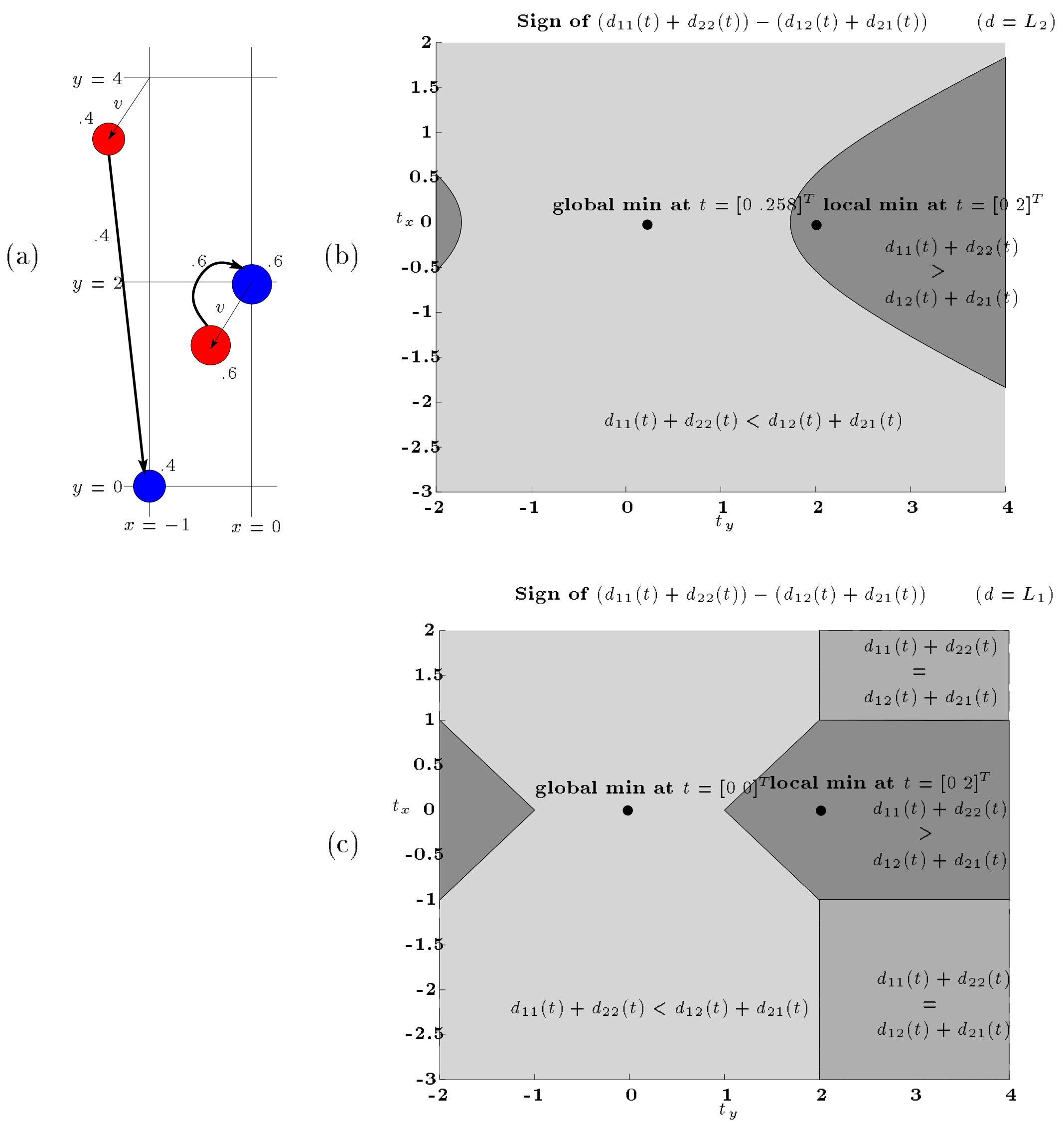


Figure 6.10: A Closer Look at a Locally Optimal Translation. The translation $t = [0\ 2]^T$ is locally optimal in the example shown in Figure 6.7(a) for $d = L_2$ and $d = L_1$. For both these ground distances, $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus t) = 1.6$ (see Figure 6.7(b)). (a) Here we show $\mathbf{y}$ translated by $t + v = [0\ 2]^T + v$, where $||v||$ is small. (b) All translations in the darkest gray area have the same optimizing flow shown in part (a) for $d = L_2$. Here $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus (t + v)) \geq .6||v||_2 + .4(4 - ||v||_2) = .2||v||_2 + 1.6$. Since the EMD is 1.6 when $v = 0$, we see that there is a local minimum at $t = [0\ 2]^T$. The global minimum at $t = [0\ .258]^T$ occurs in an area of translation space where a different flow is optimal. (c) All translations in the darkest gray area have the same optimizing flow shown in part (a) for $d = L_1$. Here $\text{EMD}(\mathbf{x}, \mathbf{y} \oplus (t + v)) \geq .6(|v_x| + |v_y|) + .4(|v_x| + (4 - |v_y|)) = 1.0|v_x| + .2|v_y| + 1.6$. Since the EMD is 1.6 when $v = 0$, we see that there is a local minimum at $t = [0\ 2]^T$. The global minimum at $t = [0\ 0]^T$ occurs in an area of translation space where a different flow is optimal.

connection can be made between a local minimum of $\gamma(F)$ at $F^0 \in \mathcal{F}$ and a local minimum of $\mathrm{WORK}(F, \mathbf{x}, g(\mathbf{y}))$ over $\mathcal{F} \times \mathcal{G}$ if we can fit an open neighborhood inside the set $S(g^*)$ of feasible flows which have g^* as their optimal transformation, where g^* is optimal for F^0.

Of the L_1, L_2, and L_2^2 ground distances, only the L_2^2 distance is guaranteed to yield a WORK function in which a locally optimal translation must be globally optimal. The globally optimal translation for $d = L_2^2$ is the one that lines up the centroids of the two distributions. The centroid of a weighted point set is the point from which the weighted sum of L_2^2 distances to the points in the set is minimized. Recall from section 6.4.1.3 that the spatial and coordinate-wise medians are the points from which the weighted sum of L_2 and L_1 distances, respectively, are minimized. In general, the optimal translation to match two distributions with the EMD, however, is *not* the spatial median for $d = L_2$ and is *not* the coordinate-wise median for $d = L_1$. Indeed, the spatial medians for $\mathbf{x}$ and $\mathbf{y}$ are $(0, 2)$ and $(0, 0)$, respectively, and the coordinate-wise medians are also $(0, 2)$ and $(0, 0)$. In both cases, the locally optimal translation $t = [0\ 2]^T$ lines up the medians, but the globally optimal translation does not.

The magic of the EMD under translation with $d = L_2^2$ does not extend to the EMD under rotation with $d = L_2^2$. In the plane, we seek a rotation angle θ such that $\mathrm{EMD}(\mathbf{x}, R_\theta \mathbf{y})$ is minimized. Even when the L_2^2 ground distance is used, there can be only locally optimal rotation angles. This is clearly shown in Figure 6.11 which contains plots of the EMD versus θ for the example in Figure 6.7(a).

6.8 Odds and Ends

We have not yet discussed the choice of ground distance function used in EMD computations. In section 6.8.1, we consider the tradeoffs in choosing betwen the Euclidean distance and the Euclidean distance squared. One criterion of comparison is solving EMD under transformation problems, although we consider the ground distance choice for other criteria as well. In section 6.8.2, we briefly consider the question: how fast can the EMD between one distribution and a transformed version of another change with respect to the transformation parameters. If the EMD for a given transformation is large, then the EMD for a nearby transformation will also be large if the EMD does not change too quickly. This information may allow a search for an optimal transformation to eliminate a region of the search space without computing the EMD for many transformations in that region.

(a)

(b)

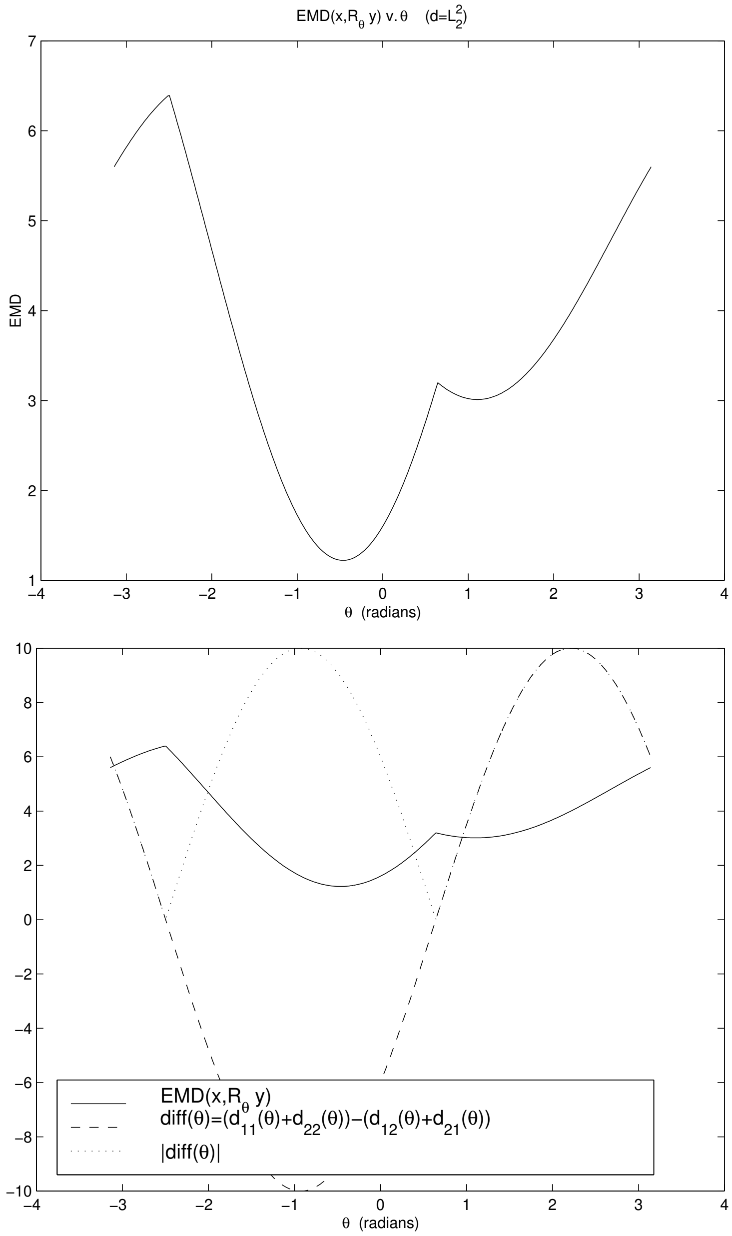


Figure 6.11: Graphs of EMD v. θ Showing a Locally Optimal Rotation for $d = L_2^2$. (a) EMD v. θ for the 2×2 example shown in Figure 6.7(a) with $d = L_2^2$. (b) The globally and locally optimal rotations lie in regions of the rotation space for which there are different optimal flows. The difference function $(d_{11}(\theta) + d_{22}(\theta)) - (d_{12}(\theta) + d_{21}(\theta))$ and its absolute value are the dashed and dotted plots, respectively. The plot of the absolute value shows where the difference function becomes zero and the optimal flow changes.

6.8.1 L_2^2 versus L_2

The EMD takes a "ground distance" function between points and builds upon it a distance function between sets of weighted points. An appropriate ground distance is application-dependent. In the case where the points are located in the CIE-Lab color space, it is natural to use the Euclidean distance as the ground distance since this feature space was specially designed so that perceptual color distance is well-approximated by the L_2 distance. In other feature spaces, the choice may not be so clear. When there is no clear reason to prefer one ground distance over another, it is worth considering the L_2-distance squared even though L_2^2 is not a point metric.

When comparing equal-weight distributions we would like the EMD to be metric so that we cannot have the non-intuitive situation in which two distributions are similar to a third but not to each other. Using an L_p ground distance guarantees that the EMD is a metric between equal-weight distributions. Although this is not the case for L_2^2, the EMD is at most a factor of two away from satisfying the triangle inequality for three given distributions. See section 4.1, formula (4.6).

Another criterion for comparing ground distances is the availability of efficient, effective lower bounds to prune unnecessary EMD computations. In section 5.1.1, we showed that the centroid distance lower bound between equal-weight distributions is valid for both the L_p and L_2^2 ground distances. In section 5.1.2, we used the bound for equal-weight distributions to get a bound on the EMD between unequal-weight distributions. We showed that the minimum ground distance between the centroid of the lighter distribution $\mathbf{y}$ and the centroid of any sub-distribution of $\mathbf{x}$ with the same weight as $\mathbf{y}$ is a lower bound on $\text{EMD}(\mathbf{x}, \mathbf{y})$. Recall that this CLOC lower bound and the more practical CBOX lower bound apply with any ground distance for which the equal-weight centroid bound holds, and this includes $d = L_2^2$ as well as $d = L_2$.

Using L_2^2 also has many advantages in computing the EMD under transformation sets. Consider, for example, the problem of computing the EMD under translation. Applying the FT iteration requires a solution to the optimal translation problem. We gave algorithms for this problem for each of the L_1, L_2 and L_2^2 point distances, but even in the equal-weight case there can be locally optimal translations that are not globally optimal when $d = L_1$ or $d = L_2$ is used. In the equal-weight case with $d = L_2^2$, there are no translations which are only locally optimal. The L_2^2 distance has an even bigger advantage in the FT iteration for higher order transformation sets such as the sets of Euclidean, similarity, linear, and affine transformations. Sum-of-squares optimization problems are well-studied in mathematics, and there are solutions to the optimal transformation problem for each of the listed sets

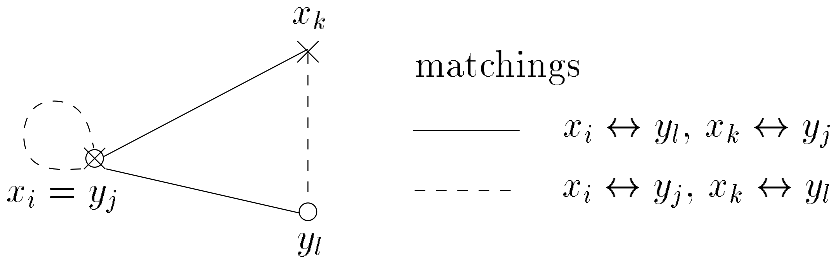


Figure 6.12: Matching Pairs of Points. The two possible matchings of $\{ x_i, x_k \}$ to $\{ y_j, y_l \}$. Here $x_i = y_j$.

(see sections 6.4.2–6.4.3).

The fact that L_2^2 is not a point metric often allows more "natural" optimal flows than for L_p metrics. Consider matching point sets with the EMD. If two points from different sets are on top of each other, then there is always an optimal flow which pairs these two points. This is easily seen with the aid of Figure 6.12. If $d = L_p$, then by the triangle inequality $d(x_i, y_l) + d(x_k, y_j) \geq d(x_k, y_l)$. Thus matching $x_i \leftrightarrow y_l$, $x_k \leftrightarrow y_j$ is at least as expensive as matching $x_i \leftrightarrow y_j$, $x_k \leftrightarrow y_l$. In fact, if $x_i (= y_j)$, x_k, and y_l are not collinear, then the matching with the zero cost correspondence $x_i \leftrightarrow y_j$ is strictly less expensive.

Figure 6.13 gives examples in 1D and 2D to illustrate our previous point. The 1D example in Figure 6.13(a) is due to Jorge Stolfi ([76]). Both flows shown in Figure 6.13(a) are optimal for $d = L_1$ and $d = L_2$, each requiring 6 units of work. The flow on the left costs 36 work units with $d = L_2^2$, while the more natural flow on the right costs 6 work units and is the unique optimal flow for $d = L_2^2$. Figure 6.13(b) shows an example in 2D. The flow on the left is optimal for $d = L_1$ and $d = L_2$, and it is the unique optimal flow since the duplicate point $(0, 0)$ is not involved in a collinearity with two other points from the two sets. The flow on the right is the unique optimal flow for $d = L_2^2$. The two point sets are close to differing by a translation. The correspondences on the right are a lot better for the FT iteration to find the globally optimal translation. Using an L_p ground distance results in "greedy" flows which may give wrong correspondences to the optimal transformation step of the FT iteration.

6.8.2 Sensitivity and Growth Rate

The EMD is insensitive to small perturbations of mass locations. After all, the EMD is a weighted average distance between points, and small changes in point locations result in small changes in inter-point distances. More precisely, we have the following result.

(a)

(b)

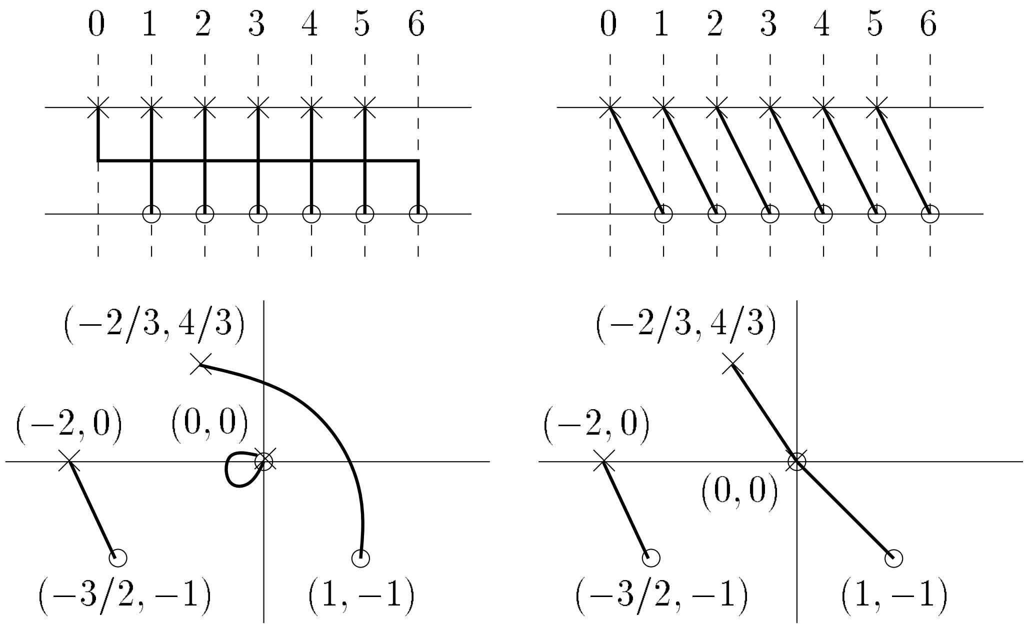


Figure 6.13: Optimal Point Set Matchings under L_2 and L_2^2. The matchings are indicated by dark lines connecting the points. (a) This is a 1D example where the point sets have been offset vertically for clarity. The left and right flows are both optimal for L_2, while the right flow is the unique optimal flow for L_2^2. (b) In this 2D example, the left flow is the unique optimal flow for L_2 and the right flow is the unique optimal flow for L_2^2.

Theorem 14 *If*

$$|d(x_i, g(y_j)) - d(x_i, y_j)| \leq D(g) \qquad \forall i, j, \tag{6.52}$$

then

$$|\text{EMD}(\mathbf{x}, g(\mathbf{y})) - \text{EMD}(\mathbf{x}, \mathbf{y})| \leq D(g). \tag{6.53}$$

Proof. Condition (6.52) implies

$$\begin{aligned} d(x_i, g(y_j)) &\leq d(x_i, y_j) + D(g) \quad \forall i, j \quad \text{and} & (6.54)\\ d(x_i, y_j) &\leq d(x_i, g(y_j)) + D(g) \quad \forall i, j. & (6.55)\end{aligned}$$

We shall use (6.54) to show that $\text{EMD}(\mathbf{x}, g(\mathbf{y})) - \text{EMD}(\mathbf{x}, \mathbf{y}) \leq D(g)$. Inequality (6.55) implies $\text{EMD}(\mathbf{x}, \mathbf{y}) - \text{EMD}(\mathbf{x}, g(\mathbf{y})) \leq D(g)$ in a completely analogous fashion. Combining these two results yields (6.53).

From (6.54), it follows that

$$\sum_i \sum_j f_{ij} d(x_i, g(y_j)) \leq \sum_i \sum_j f_{ij} d(x_i, y_j) + D(g) \min(w_\Sigma, u_\Sigma) \qquad \forall F \in \mathcal{F}(\mathbf{x}, \mathbf{y}).$$

The left-hand side of the inequality decreases with the replacement of F by an optimal flow $F^*(g) = (f^*_{ij}(g))$ between $\mathbf{x}$ and $g(\mathbf{y})$. Thus

$$\sum_i \sum_j f^*_{ij}(g) d(x_i, g(y_j)) \leq \sum_i \sum_j f_{ij} d(x_i, y_j) + D(g) \min(w_\Sigma, u_\Sigma) \qquad \forall F \in \mathcal{F}(\mathbf{x}, \mathbf{y}).$$

The result follows by dividing both sides by $\min(w_\Sigma, u_\Sigma)$, and replacing F by an optimal flow between $\mathbf{x}$ and $\mathbf{y}$. ∎

Note that this result holds regardless of whether or not $\mathbf{x}$ and $\mathbf{y}$ have equal total weight.

For any L_p norm, we have the *reverse triangle inequality* $|\, ||A||_p - ||B||_p \,| \leq ||A - B||_p$.[10] In particular,

$$|\, ||x_i - (y_j + t)||_p - ||x_i - y_j||_p \,| \leq ||t||_p.$$

Thus, Theorem 14 implies[11]

$$\left|\text{EMD}^{||\cdot||_p}(\mathbf{x}, \mathbf{y} \oplus t_1) - \text{EMD}^{||\cdot||_p}(\mathbf{x}, \mathbf{y} \oplus t_2)\right| \leq ||t_1 - t_2||_p. \tag{6.56}$$

---

[10] Short proof. Apply the triangle inequality twice: (1) $||A||_p - ||B||_p \leq ||A - B||_p$, and (2) $||B||_p - ||A||_p \leq ||B - A||_p = ||A - B||_p$.

[11] The result (6.56) is of the same form (6.53) if we put $\mathbf{z} = \mathbf{y} \oplus t_2$. Then $\mathbf{z} \oplus (t_1 - t_2) = \mathbf{y} \oplus t_1$ and $t = t_1 - t_2$.

In [36], Huttenlocher et al. use an analogous result for the Hausdorff distance to prune translations during the search for a binary model pattern within a binary image. If $\mathrm{EMD}^{||\cdot||_p}(\mathbf{x}, \mathbf{y} \oplus t_1) \geq \alpha$, then $\mathrm{EMD}^{||\cdot||_p}(\mathbf{x}, \mathbf{y} \oplus t_2) \geq \alpha - ||t_1 - t_2||_p$.

Now consider matching planar distributions with $d = L_2$. How fast can $\mathrm{EMD}(\mathbf{x}, R_\theta \mathbf{y} \oplus t)$ change with respect to the Euclidean transformation (R_θ, t)? Here we have

$$| \, ||x_i - (R_\theta y_j + t)||_2 - ||x_i - y_j||_2 \, | \leq ||y_j - R_\theta y_j - t||_2 \leq ||y_j - R_\theta y_j||_2 + ||t||_2,$$

where the first and second inequalities follow from the reverse and the ordinary triangle inequalities, respectively. But $||y_j - R_\theta y_j||_2 \leq |\theta| \; ||y_j||_2$ because the length of the arc from y_j to $R_\theta y_j$ is at least the distance between these two points. We can therefore apply Theorem 14 with $D = ||t||_2 + |\theta|(\max_j ||y_j||_2)$. The larger the quantity $\max_j ||y_j||_2$, the weaker the bound (6.53). If we do not replace $||y_j||_2$ by $\max_j ||y_j||_2$ in D, then following the proof of Theorem 14 shows[12]

$$\begin{aligned} &\left| \mathrm{EMD}^{||\cdot||_2}(\mathbf{x}, R_{\theta_1}\mathbf{y} \oplus t_1) - \mathrm{EMD}^{||\cdot||_2}(\mathbf{x}, R_{\theta_2}\mathbf{y} \oplus t_2) \right| \leq \\ &\qquad ||t_1 - t_2||_2 + |\theta_1 - \theta_2| \sum_j (u_j/u_\Sigma)||y_j||_2 \qquad \text{if } u_\Sigma \leq w_\Sigma. \end{aligned}$$

Here we pay an average (instead of worst case) rotational penalty, where each point's norm contribution is weighted by its fraction of the total distribution mass.

6.9 Some Applications

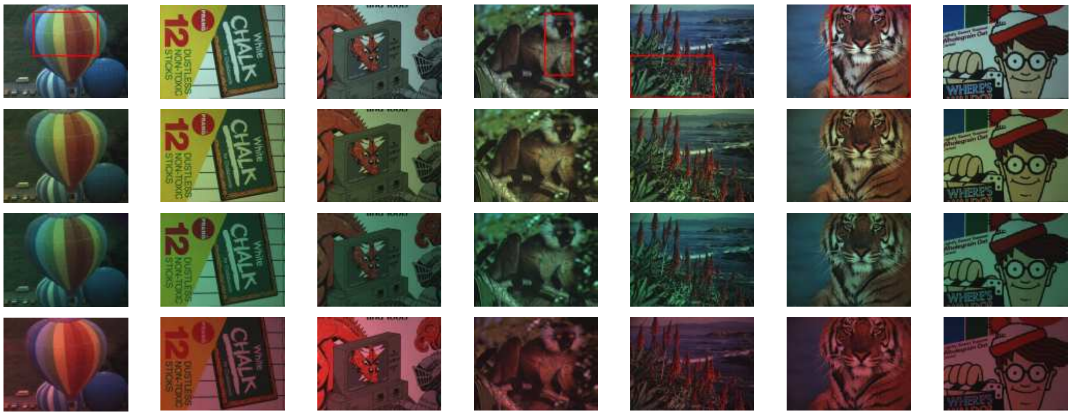
In this section, we apply the FT iteration described in section 6.3 to the problems of illumination-invariant object recognition and point feature matching in stereo image pairs. All experiments were conducted on an SGI Indigo^2 with a 250 MHz processor. The algorithms to solve the transportation problem ([32], pp. 213–229), and the optimal translation, Euclidean, and similarity transformation problems are implemented in C, while the solutions to the optimal linear and affine problems are written in MATLAB.[13]

6.9.1 Lighting-Invariant Object Recognition

Under the assumption of a trichromatic system with a three-dimensional linear model for the surface reflectance functions of object surfaces, Healey and Slater ([28]) showed that

[12] More precisely, use the facts that $| \, ||x_i - (R_{\theta_1} y_j + t_1)||_2 - ||x_i - (R_{\theta_2} y_j + t_2)||_2 \, | \leq ||(R_{\theta_2} y_j - R_{\theta_1} y_j) + (t_2 - t_1)||_2 \leq |\theta_2 - \theta_1| \; ||y_j||_2 + ||t_2 - t_1||_2$, and $\sum_i f_{ij} = u_j$ for any feasible flow $F = (f_{ij})$ when $u_\Sigma \leq w_\Sigma$.

[13] Thanks to Yossi Rubner for providing his transportation problem code.



(B)alloon (C)halk (D)ragon (L)emur (P)lant (T)iger (W)aldo

Figure 6.14: Lighting-Invariant Object Recognition – A Small Object Database. An object database imaged under white, yellow, green, and red light is shown in rows 1, 2, 3, and 4, respectively. For some objects, color signatures are computed over only the area outlined in red as shown in row 1. The color signatures for all images of the same object are computed over the same image area, although we only show the red rectangle in the images taken under white light.

an illumination change results in a linear transformation of image pixel colors.[14] In the following experiment, we use a subset of the objects used in [28]. There are four images of each object, one under nearly white illumination and the other three under yellow, green, and red illumination.[15] Figure 6.14 shows the seven database objects imaged under white, yellow, green, and red light.

As in [69], we summarize each image by a set of dominant colors (without regard to position) obtained by clustering in color space, where a color is weighted by the fraction of image pixels classified as that color. We use the clustering algorithm described in [65] in the RGB color space with a minimum bucket size of 16 units in R, G, and B, and we discard clusters with weight less than 0.5%. This produced color signatures with an average of 27 colors.

Our experiment consists of using each image as the query, where the desired distance between images is the EMD under a linear transformation of the corresponding color signatures (with the L_2-distance squared as the ground distance between points in RGB space). To compare a database signature $\mathbf{x}$ to a query signature $\mathbf{y}$, we applied the FT iteration twice (with $\mathcal{G} = \mathcal{L}$): once to transform $\mathbf{y}$ so that it is as close as possible to $\mathbf{x}$, and once

[14] This result also holds for images of scenes with more than one object if all surfaces of all objects have the same basis reflectance functions.

[15] Thanks to David Slater for providing these images.

	B_W	B_Y	B_G	B_R	C_W	C_Y	C_G	C_R	D_W	D_Y	D_G	D_R	L_W	L_Y	L_G	L_R
W	1	3	2	3	1	4	2	3	1	2	3	2	1	4	6	2
Y	3	1	3	5	4	1	3	2	2	1	$\boxed{2}$	4	4	1	4	4
G	2	2	1	2	2	3	1	4	3	3	1	3	3	2	1	3
R	5	5	4	1	3	2	4	1	4	4	4	1	2	3	2	1
Σ	11	11	10	11	10	10	10	10	10	10	10	10	10	10	13	10

	P_W	P_Y	P_G	P_R	T_W	T_Y	T_G	T_R	W_W	W_Y	W_G	W_R
W	1	4	2	3	1	3	2	2	1	4	3	3
Y	4	1	3	2	3	1	3	3	2	1	2	2
G	2	3	1	5	2	2	1	4	4	2	1	4
R	3	2	5	1	4	4	7	1	3	3	4	1
Σ	10	10	11	11	10	10	13	10	10	10	10	10

Figure 6.15: Lighting-Invariant Object Recognition – Query Results. The label at the top of each column shows the query image. The row labels are the illuminants (W)hite, (Y)ellow, (G)reen, and (R)ed. The entry in position (Z, A_X) is the rank of image A_Z in the result for query image A_X. For example, the dragon image for the yellow illuminant is returned as the second closest image when the query image is the dragon image for the green illuminant ($Z = Y$,$A = D$,$X = G$ – see the boxed entry). The number at the bottom of each column is the total of the ranks in that column, where 10 is the ideal value. The query precision is perfect for 21 of the 28 queries, and the average rank sum is 10.4. One run of the FT iteration required an average of 7.4 steps and 4.6 seconds to converge.

to transform $\mathbf{x}$ so that it is as close as possible to $\mathbf{y}$. Both trials were started with the initial transformation equal to the identity map. We use the smaller of these results as the distance between $\mathbf{x}$ and $\mathbf{y}$. The minimum result is equal to $\text{EMD}_{\mathcal{L}}(\mathbf{x}, \mathbf{y})$ if a globally optimal transformation is found, and is greater than $\text{EMD}_{\mathcal{L}}(\mathbf{x}, \mathbf{y})$ otherwise. Ideally, the closest images to the image of an object are the other three images of the same object.

Figure 6.15 shows the results of our experiment.[16] These results are excellent, but not perfect as in [28]. It is possible that we are not finding the globally optimal transformation in some comparisons. Also, the linear transformation model loses accuracy when we replace the color of a pixel by the centroid of a cluster in color space.

6.9.2 Feature Matching in Stereo Images

As we described in section 6.1, the partial EMD under a transformation set can be used to compute the best partial matching of two point sets when one set is free to undergo

[16] The results obtained with all signatures computed over entire images are very similar to the given results.

some transformation.[17] This is exactly the problem we have in matching features extracted from images of the same scene taken from different viewpoints. The fraction parameter γ compensates for the fact that only some features appear in both images, and the set parameter $\mathcal{G}$ accounts for the appropriate transformation between corresponding features. In our experiments, we extract 50 features of a gray level image using an algorithm due to Shi and Tomasi ([74]).[18] The points in the distribution summary of an image are its feature locations (measured in pixels), and the weight of each point is one.[19] The ground distance is the L_2-distance squared between image coordinates. We set $\gamma = 0.5$, so only 25 of the 50 features per image will be matched. Each of the three examples given below uses a different transformation set $\mathcal{G}$, although the initial transformation used in the FT iteration is the identity map in all cases.

In our first example, we match features in two images of a motion sequence in which the camera moves approximately horizontal and parallel to the image plane. Figure 6.16(top) shows the results of applying the FT iteration with $\mathcal{G} = \mathcal{T}$ in an attempt to minimize the partial EMD under a translation of the point features. For this camera motion, all image points translate along the same direction, but the amount of translation for an image point is inversely proportional to the depth of the corresponding scene point ([80]). Thus, the model of a single translation vector is not accurate in general. It is accurate for a set of features that correspond to scene points with roughly the same depth. In this example, the FT iteration matched features on objects toward the back of the table.

The images in the example depicted in Figure 6.16(middle) are also from a motion sequence. Here, however, the camera motion is a forward motion perpendicular to the image plane. The match results shown are the result of applying the FT iteration with $\mathcal{G} = \mathcal{S}$ in an attempt to minimize the partial EMD under a similarity transformation. In our final example, we match features extracted from images of a toy hotel taken from two different viewpoints. Here we apply the FT iteration with $\mathcal{G} = \mathcal{A}$ in an attempt to minimize the partial EMD under an affine transformation of feature locations. The match results are shown in Figure 6.16(bottom). In all three cases, it appears that the FT iteration converged to a globally optimal transformation. In many examples, however, running the iteration once leads to only a locally optimal solution.

[17] Recall that the same code used to solve the transportation problem can be used to solve the assignment problem and to compute the partial EMD.

[18] Thanks to Stan Birchfield for his implementation of this feature extraction algorithm.

[19] Using the gray levels in a small area around a feature in addition to its location may improve matching results. However, corresponding pixels in images of a scene from different viewpoints may have gray level differences which are not small. Therefore, using gray level information may hurt matching results if we do not account for such differences.

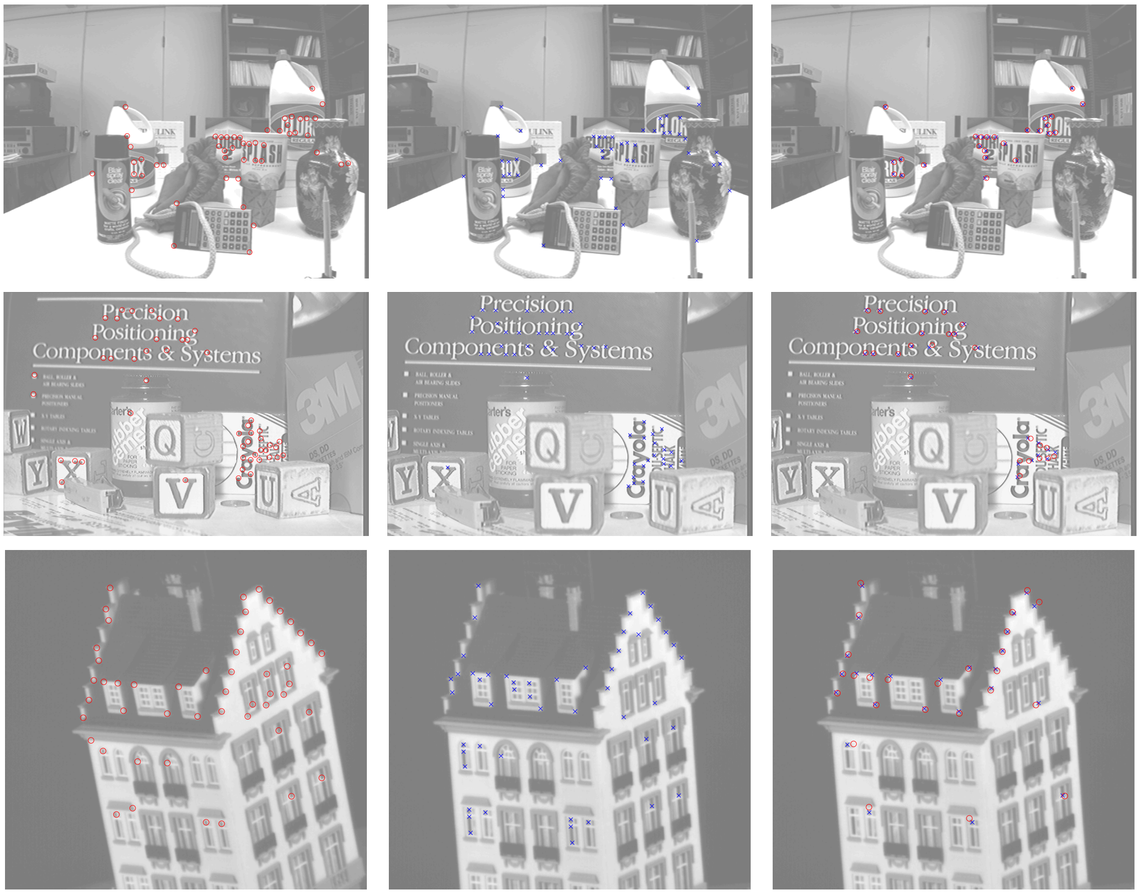


Figure 6.16: Point Feature Matching in Stereo Images – Results. The first two columns in each row show two images and the locations of 50 features in each image. The last column shows the result of matching the features using the FT iteration with an initial transformation equal to the identity map. Here $\gamma = 0.5$, so only 25 features are matched in each example. We report the number of steps S and the time T in seconds (s) for the FT iteration to converge. (top) $\mathcal{G} = \mathcal{T}$, $S = 11$, $T = 1.77$s. (middle) $\mathcal{G} = \mathcal{S}$, $S = 4$, $T = 1.08$s. (bottom) $\mathcal{G} = \mathcal{A}$, $S = 8$, $T = 36.21$s.

Chapter 7

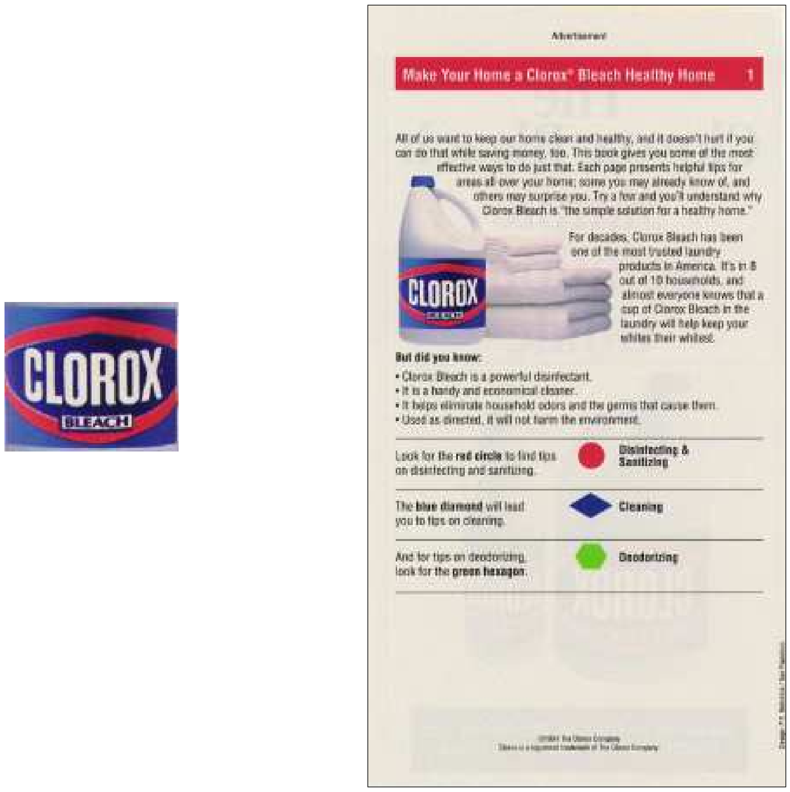
The SEDL Image Retrieval System

The goal of our content-based retrieval system is to find quickly all database images that contain a region similar to a given query pattern. In other words, processing a query consists of solving many pattern problems with the same pattern input. We will illustrate the general strategy described in this chapter with two specific databases: (1) a color database of scanned product advertisements, and (2) a shape database of Chinese character bitmaps. For the color database, the query patterns are product logos. The goal is to find all the ads for the given product and for products with a similar logo. For the shape database, the query patterns are characters that occur repeatedly within other Chinese characters. The goal is to find all the characters that contain strokes similar to those in the query. Examples of a query pattern and a database image that should be returned for the query are shown in Figure 7.1 for both the color and the shape case.

Our image retrieval system is called SEDL, which stands for Scale Estimation for Directed Location. As its name implies, SEDL performs a directed (as opposed to exhaustive) pattern search once it has computed an estimate for the size at which the pattern might occur within an image. If we imagine a search rectangle moving and changing size and orientation over the image, then this rectangle will eventually converge or settle (SEDL) on the image area where the system believes the pattern exists. A few promising initial placements of search rectangles are efficiently computed at query time without having to examine image areas that obviously do not contain the pattern. The initial size of the search rectangles is determined by the scale estimate.

In the shape pattern problem solved by SEDL, images (and patterns) are sets of curves such as might be produced by edgel detection and linking, or by any standard drawing program. For our Chinese character database, the first preprocessing step is to reduce each bitmap character to a set of curves by computing its medial axis.

(a)



(b)

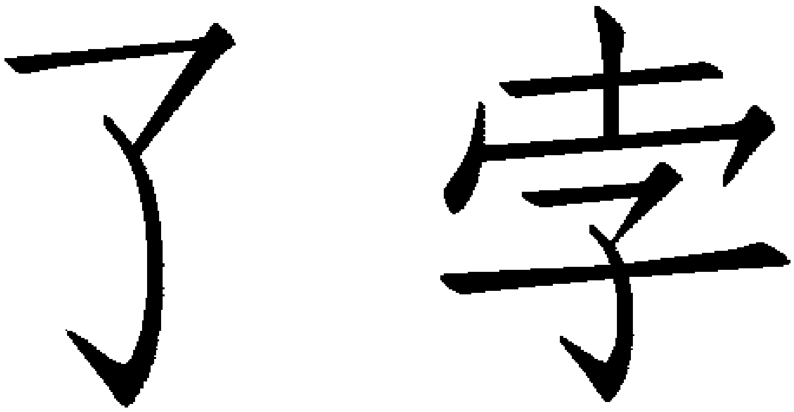


Figure 7.1: Query Patterns and Related Database Images. Here we show examples of a query pattern and a database image that should be retrieved in (a) the color advertisement database, and (b) the shape Chinese character database.

The image signature used in SEDL is a distribution that records what attributes occur where in an image, as well as the amount of the attribute present at the recorded location. For the advertisement database, the attribute is color; for the Chinese character database, the attribute is orientation (of ink along image curves). In general, a pattern "occurs" in an image to the extent that there is a transformation of the pattern attribute locations that aligns similar attributes in the query and image. In the color case, we try to align uniform color regions in the pattern and query with similar colors. SEDL's signature distance function will be small whenever each pattern region is close to an image region of similar color. In the shape case, we try to align pieces of pattern curves with pieces of image curves of similar orientations. SEDL's signature distance function will be small whenever ink along pattern curves is close to ink along image curves of similar orientation. The image signatures and signature distance function used in SEDL are the subject of section 7.1.

There are three phases in SEDL: (1) scale estimation, (2) initial placement selection, and (3) match refinement and verification. See Figure 7.2. In the first phase, only the image and query attributes (and not their locations in the image plane) are used to estimate the scale at which the pattern might occur in the image. The scale estimate is computed using the EMD in attribute space between the image and query signatures marginalized over position. The scale estimation algorithm is discussed in section 7.2. The goal of the initial placement phase is to identify efficiently a handful of promising image regions where the pattern is likely to occur. An image region is "promising" if its color signature is similar to that of the pattern. The size of the regions examined is determined by the scale estimate returned by the previous phase. Only the coarse position information of whether or not an attribute is located in a particular region is used during this phase. The initial placement phase is described in section 7.3. Finally, for each initial placement of the query at the estimated scale, we check for positional consistency of the attributes, modifying the attribute locations by some transformation if this will help improve the match. This last refinement and verification stage is the subject of section 7.4. We will describe the three phases in SEDL mainly as applied to the color pattern problem. The chapter concludes with results of the entire matching process in the advertisement database (section 7.5.1) and the Chinese character database (section 7.5.2).

Figure 7.2: The Three Phases in SEDL.

7.1 SEDL's Image Signatures and Distance Function

The image signature $\mathbf{X}$ is a discrete distribution of mass in a combined attribute-position space:

$$\begin{aligned}\mathbf{X} &= \{ (X_1, W_1), \ldots, (X_M, W_M) \} \\ &= \{ ((A_1, P_1), W_1), \ldots, ((A_M, P_M), W_M) \},\end{aligned}$$

where distribution point $X_I = (A_I, P_I)$ consists of a point A_I in attribute space and a point P_I in image position space. The distribution pair $(X_I, W_I) = ((A_I, P_I), W_I)$ indicates that there is a region of size W_I located at P_I with attribute A_I. For the advertisement database, P_I is the centroid of a region of roughly constant color, its attribute A_I is the average color in the region, and its weight W_I is the area of the region. This idea is conveyed in the example in Figure 7.3. For the Chinese character database, P_I is the centroid of a small curve or segment, its attribute A_I is the average orientation along the curve, and its weight W_I is the length of the curve over which the average is taken. In the shape case, a *region* refers to a segment of a curve.

Throughout this chapter, we denote the signature of a database image by $\mathbf{X}$ as given above, and the signature of a query pattern by $\mathbf{Y}$, where

$$\begin{aligned}\mathbf{Y} &= \{ (Y_1, U_1), \ldots, (Y_N, U_N) \} \\ &= \{ ((B_1, Q_1), U_1), \ldots, ((B_N, Q_N), U_N) \}.\end{aligned}$$

Once again, we assume the normalization $W_\Sigma = U_\Sigma = 1$. A query pattern is similar to part of a database image to the extent that there is a transformation of the pattern region positions $\{ Q_J \}$ that aligns regions in $\mathbf{X}$ and $\mathbf{Y}$ that have similar attributes.

In the color case, a small set of dominant image colors $\{a_i\}_{i=1}^m$ is computed by clustering in color space; in the shape case, a small set of dominant orientations $\{a_i\}_{i=1}^m$ is computed by clustering image curve orientations. In both cases, all A_I are members of the set $\{a_i\}_{i=1}^m$. A similar statement holds for the attributes B_J in the pattern attribute $\times$ position signature. The signature creation processes for the color and shape pattern problems are discussed in sections 7.5.1.1 and 7.5.2.1, respectively.

In order to define a signature distance function, we first define a distance function d_{ap} in the combined attribute-position space. Given a distance function d_{attr} between attributes, we use a linear combination distance

$$d_{\mathrm{ap}}((A, P), (B, Q)) = d_{\mathrm{attr}}^{\tau}(A, B) + \lambda d_{\mathrm{pos}}(P, Q), \tag{7.1}$$

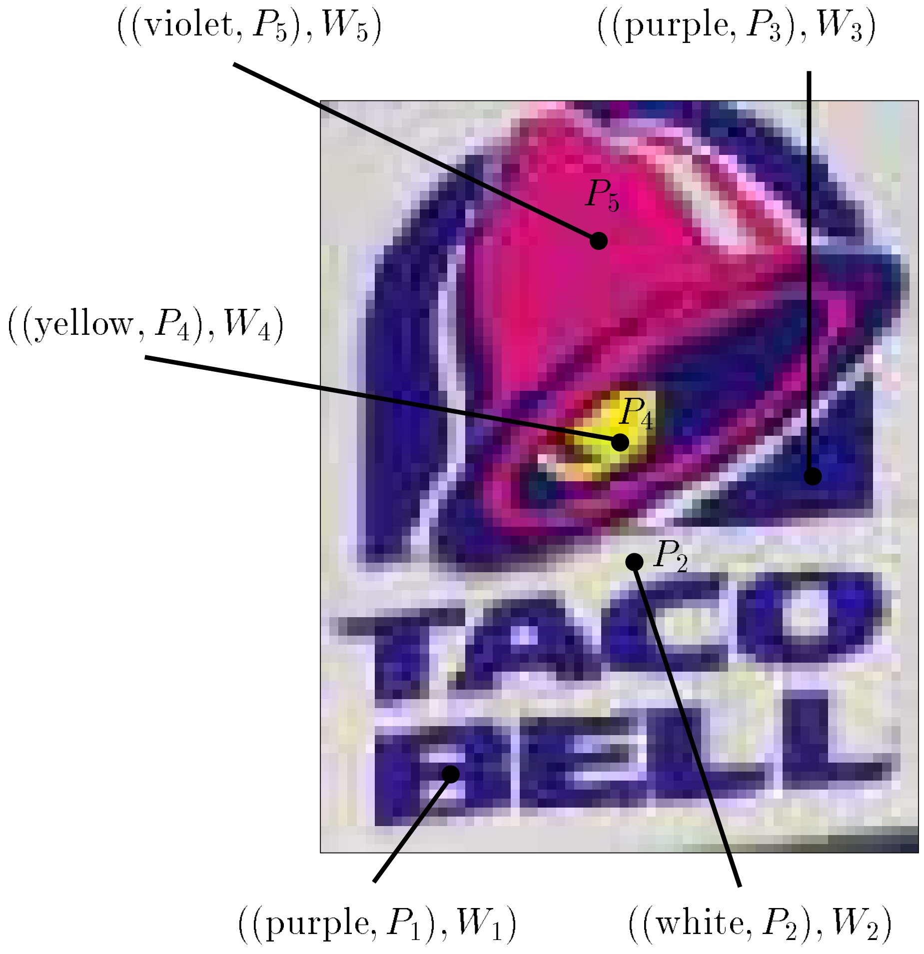


Figure 7.3: Signature in Color $\times$ Position Space. Not all signature elements are labelled.

where

$$d^{\tau}_{\text{attr}}(A,B) = \begin{cases} d_{\text{attr}}(A,B) & \text{if } d_{\text{attr}}(A,B) \le \tau \\ \infty & \text{otherwise} \end{cases} \quad \text{and} \quad d_{\text{pos}}(P,Q) = ||P-Q||_2^2.$$

The attribute distance is set to infinity if it is above some user defined threshold τ. This is to prevent matching of attributes which are very different but happen to be at the same position in the image plane. For colors in CIE-Lab space, we use $d_{\text{attr}}(A,B) = ||A-B||_2$, the Euclidean distance. For orientations in radians, we use $d_{\text{attr}}(A,B) = \min_{k\in\mathbf{Z}} |(A+k\pi)-B|$, the cyclic L_1 distance with period $T=\pi$ radians described in section 6.4.1.4. The pixel locations of region centroids are normalized by dividing by the minimum image dimension. If, for example, a database image has width W and height H pixels with $W \le H$, then the column coordinate $P_x \in [0,1]$ and the row coordinate $P_y \in [0, H/W]$, where $P=(P_x,P_y)$.

The factor λ in front of the position distance d_{pos} is chosen as follows. Given two "equal" distances D_{attr} and D_{pos} in attribute and position space, and the relative importance $\alpha \in (0,1]$ of attribute differences versus position differences, we choose λ so that $(1-\alpha)/\alpha = (\lambda D_{\text{pos}})/D_{\text{attr}}$. We use $\alpha = 25\%$ attribute and $(1-\alpha) = 75\%$ position. We emphasize the position information because it was not used in the scale estimation or initial placement phases, and, if these phases worked correctly, the verification and refinement phase starts in image areas with similar attributes to those in the pattern. On the other hand, we want the attribute values to guide our matcher, so we cannot make α too small. We set $D_{\text{pos}} = (0.10)^2$, which corresponds before normalization to the square of 10% of the minimum image dimension. In the color case, we set $D_{\text{attr}} = 5$ CIE-Lab units; in the shape case, we set $D_{\text{attr}} = 10^\circ \doteq 0.175$ radians.

Armed with the ground distance d_{ap} in the combined attribute-position (what-where) space, we define the signature distance function SD as

$$\text{SD}(\mathbf{X},\mathbf{Y}) = \min_{\psi\in\Psi} D(\psi,\mathbf{X},\mathbf{Y}),$$

where

$$D(\psi,\mathbf{X},\mathbf{Y}) = \sum_{J=1}^{N} U_J d_{\text{ap}}((A_{\psi(J)},P_{\psi(J)}),(B_J,Q_J)),$$

and

$$\Psi = \{\ \psi : [1..N] \to [1..M]\ \}$$

is the set of functions from $[1..N]$ to $[1..M]$. If the image attribute $A_{\psi(J)}$ at position $P_{\psi(J)}$ is matched to the query attribute B_J at position Q_J, then we pay a cost of the distance in the combined attribute-position space times the weight of the query attribute. The allowable matchings Ψ are unconstrained; each query region ((attribute,position) pair) can match any image region. Note that this is very much a one way distance – it is small when every query region is near some image region with a similar attribute value, but not necessarily vice-versa. Also note that the image weights do not appear in our distance function. These weights, however, are used explicitly in the scale estimation and initial placement phases.

As mentioned previously, similarity of a query pattern to a database image is judged modulo a transformation of region locations. Thus we want to compute the distance

$$\begin{aligned} \mathrm{SD}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) &= \min_{g \in \mathcal{G}} \quad \mathrm{SD}(\mathbf{X}, g(\mathbf{Y})) \\ &= \min_{g \in \mathcal{G}, \psi \in \Psi} \quad D(\psi, \mathbf{X}, g(\mathbf{Y})), \end{aligned}$$

where

$$g(\mathbf{Y}) = \{ \ ((B_1, g(Q_1)), U_1), \ldots, ((B_N, g(Q_N)), U_N) \ \}$$

only modifies the region locations, not the region attributes. In full, we have

$$\begin{aligned} \mathrm{SD}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) &= \min_{g \in \mathcal{G}, \psi \in \Psi} \sum_{J=1}^{N} U_J d_{\mathrm{ap}}((A_{\psi(J)}, P_{\psi(J)}), (B_J, g(Q_J))) \\ &= \min_{g \in \mathcal{G}, \psi \in \Psi} \sum_{J=1}^{N} U_J (d_{\mathrm{attr}}^{\tau}(A_{\psi(J)}, B_J) + \lambda d_{\mathrm{pos}}(P_{\psi(J)}, g(Q_J))). \end{aligned}$$

In section 7.4, we show how to find a local minimum of $\mathrm{SD}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y})$ using the same alternation idea behind our EMD iteration. We also discuss how the distance function $\mathrm{SD}_{\mathcal{G}}$ compares with the distance function $D_{\mathcal{G}}^{\mathrm{ICP}}$ used in the ICP registration algorithm (see section 2.5.2) and with the EMD, including our reasons for choosing SD over these other distance measures. As with any iterative optimization scheme in a space with local minima, the key to finding the global minimum is to have a good starting point for the iteration. For us, this means selecting a good initial transformation $g^{(0)} \in \mathcal{G}$. In turn, this means selecting a good initial scale and placement of the pattern within the image. The scale estimation and initial placement steps are the subjects of the next two sections.

7.2 The Scale Estimation Phase

Our scale estimation algorithm considers only the attributes in an image, not their location. It operates on the previously described signature distributions once position has been marginalized out. If we marginalize the attribute-position distributions $\mathbf{X}$ and $\mathbf{Y}$ over position, we get the attribute distributions

$$\begin{aligned} \mathbf{a} &= \{ (a_1, w_1), \ldots, (a_m, w_m) \} \quad \text{and} \\ \mathbf{b} &= \{ (b_1, u_1), \ldots, (b_n, u_n) \}, \end{aligned}$$

respectively, where

$$w_i = \sum_{I:A_I=a_i} W_I \qquad \text{and} \qquad u_j = \sum_{J:B_J=b_j} U_J.$$

This marginalization throws away the positions of attributes and combines the weights for the same attribute at different image locations into a single weight for that attribute. If, for example, the attribute-position image signature $\mathbf{X}$ notes 10% red at the top of an image and 20% red at the bottom, then the attribute signature $\mathbf{a}$ simply notes 30% red in the image. Note that the attribute signatures also have total weight one ($w_\Sigma = u_\Sigma = 1$) since the attribute-position signatures were normalized to have weight one ($W_\Sigma = U_\Sigma = 1$).

Our initial scale estimate is obtained by running the algorithm in section 4.5 on the attribute signatures $\mathbf{a}$ and $\mathbf{b}$. Recall that this estimation algorithm scales down the weights of the query signature $\mathbf{b}$ until there is no further improvement in the EMD between the image signature $\mathbf{a}$ and the modified query signature. The scale c^0 at which this occurs is taken as the scale estimate if $\text{EMD}(\mathbf{a}, c^0\mathbf{b}) \leq \tau$, where τ is a threshold on how well the attributes must match to obtain a visually acceptable match. If not, SEDL assumes that the pattern does not occur within the image, and the initial placement and refinement phases are not performed.

The main property of the scale estimation algorithm is that the error that it makes is roughly equal to the *minimum* amount of background clutter over all query attributes, where the amount of background clutter for an attribute is the amount of that attribute present in the image but not part of the query occurrence. If there is a query attribute that occurs in the image only within the query occurrence, then the scale estimate will be very accurate, regardless of the amount of background clutter for other query attributes. Please refer back to section 4.5 for details and examples from the advertisement database.

7.3 The Initial Placement Phase

All the initial transformations of the query for our iteration will have scale c^0 as returned by the scale estimation step. The selection of initial placements of the query at scale c^0 uses the optimal flow $F^0 = (f^0_{ij})$ returned by $\text{EMD}(\mathbf{a}, c^0\mathbf{b})$. This flow provides important information about what attributes in the image match what attributes in the query. In the color case, for example, the optimal flow tells us which image colors match which query colors. Since the EMD achieved by this flow is small ($\leq \tau$, otherwise this phase would not be considered), the attributes matched by F^0 are similar.

The total weight of query attributes that match attribute i in the image is

$$v_i = \sum_{j=1}^{n} f_{ij}^0,$$

where the total weight of the scaled query signature is c^0. Basically, v defines the query signature in terms of the image attributes. In the color advertisement database, for example, we can think of v_i as the amount of image color i that occurs within the query (assuming that the scale estimate is fairly accurate). The first step in our initial placement selection is to compute the probability or confidence that an image region with attribute i is part of the query pattern (if it occurs within the image). The probability is given by the quotient

$$0 \leq q_i = \frac{v_i}{w_i} \leq 1.$$

The total amount of attribute i in the image is w_i, while the amount of attribute i that occurs within query regions is estimated to be v_i. Thus, if we randomly choose an image pixel from all the pixels with attribute i, the probability that this pixel belongs to a query region is $q_i = v_i/w_i$. The flow feasibility conditions imply $v_i \leq w_i$.

In Figure 7.4, we show an example that illustrates confidences in attributes for the color pattern problem. Yellow is a 98% confidence color since virtually all the yellow in the image is needed to match yellow in the scaled query. The confidence for yellow is not 100% because the pattern is slightly underestimated in scale. About 89% of the purple in the image is matched to purple in the scaled query. Purple is also a high confidence color. The confidence for purple is not 100% because the pattern does not include all the purple on the Ziploc box, and the pattern is slightly underestimated in scale.

For the example in Figure 7.4, white, black, and brown are low confidence colors. Although the white on the Ziploc box within the image is needed to match white in the scaled query, there is a lot of other white in the image (the writings "I will keep my sandwich fresh" and "Ziploc has the lock on freshness", the chalk next to the Ziploc box, and the white on the Ziploc box that is not part of the pattern). The black and brown in the image are not needed to match any color in the pattern. The main idea in the initial placement phase is only to examine image regions that contain (relatively) high confidence colors. In this example, SEDL never needs to examine the upper half of the Ziploc advertisement at

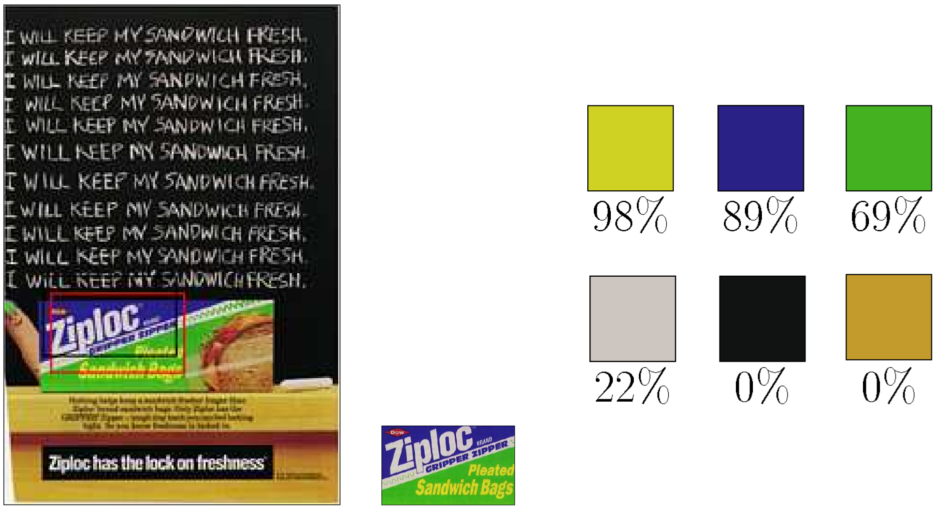


Figure 7.4: An Example of Color Confidences. The left half of the figure shows a Ziploc advertisement, along with the Ziploc query pattern scaled according to SEDL's scale estimate. The right half of the figure shows the confidences computed for various query colors. See the text for further explanation.

query time since it only contains low confidence colors (black and white).

The result of the first step in the initial placement phase is a set I^* of image attributes that have significantly high confidence ($q_i \geq \beta$, with, for example, $\beta = 0.50 = 50\%$). The regions in the image plane with high confidence attributes are good places to check further for initial placements. Pseudocode for this step is shown below.

function $G = \text{InitialTransformations}(\mathbf{X},\mathbf{a},c^0,F^0,l_{\max},\beta)$

$v_i = \sum_{j=1}^{n} f_{ij}^0, \quad i = 1, \ldots, m$

$q = v/w$

$I^* = \{\ i\ :\ q_i \geq \beta\ \} \cup \{\ \arg\max_i q_i\ \}$

$\vdots$

end function

Note that we always include the image attribute of highest confidence in the set I^*, even when this maximum confidence is less than the threshold β.

The second step in the initial placement phase is to compare the image signature in rectangular regions of the image plane to the query signature expressed in image attributes (v). The aspect ratio of the rectangles is equal to the aspect ratio of the bounding box around the query pattern, while the area of the rectangles is c^0 (the scale estimate computed in the previous phase). Let R_0 denote the origin-centered rectangle with these dimensions. This canonical rectangle will be centered at various image attribute positions for signature comparison. If we denote the image signature over window R by w^R, then the similarity measure used is the Histogram Intersection $H(w^R, v)$ between w^R and v:

$$H(w^R, v) = \sum_{i=1}^{m} \min(w_k^R, v_k).$$

The higher this value, the more similar the signature in local area R is to the query signature. Please refer back to section 2.2 for more details concerning Histogram Intersection.

The locations at which to center R_0 are exactly the locations of high confidence image attributes in I^*; we do *not* try every image location. Thus, the search for promising image regions is directed (the D in SEDL) by the relatively high confidence (i.e. low background clutter) colors. We keep track of the best placements in a variable called *optPlacements*. Associated with each placement is a Histogram Intersection value. The user supplies the maximum number $l_{\max}$ of initial similarity transformations to compute. Pseudocode for the algorithm described thus far is shown below.

function G = InitialTransformations($\mathbf{X}$,$\mathbf{a}$,c^0,F^0,$l_{\max}$,β)
  $v_i = \sum_{j=1}^{n} f_{ij}^0, \quad i = 1, \ldots, m$
  $q = v/w$
  $I^* = \{ \, i \, : \, q_i \geq \beta \, \} \cup \{ \, \arg\max_i q_i \, \}$
  optPlacements = {}
  foreach $i^* \in I^*$
    foreach $((A_I, P_I), W_I) \in \mathbf{X}$ such that $A_I = a_{i^*}$
      if (dist(P_I,optPlacements) too small) **continue**
      $R = R_0 \oplus P_I$ /* *center* R_0 *at* P_I */
      $T = \{ \, ((A_K, P_K), W_K) \, : \, P_K \in R \, \}$ /* *rectangular range search* */
      /* *compute image attribute histogram over* R */
      $w^R = 0 \in \mathbf{R}^m$
      foreach $((A_K, P_K), W_K) \in T$
        $\hat{\imath}$ = attribute#(A_K)
        $w_{\hat{\imath}}^R$ += W_K
      end foreach
      /* *compute Histogram Intersection* */
      HI = $H(w^R, v)$
      ⋮
end function

Computing the local histogram w^R requires a 2D rectangular range search for image regions inside R. This can be done in output sensitive time $O(\log M + k)$ using $O(M \log M)$ space or in time $O(\sqrt{M} + k)$ using only $O(M)$ space ([15]), where M is the number of regions in the image attribute-position distribution, and k is the number of such regions with location inside R. Both range searching results referred to above require $O(M \log M)$ preprocessing time. Note that we are really computing an approximation to the local image signature inside R since we count an entire region as inside R if its centroid is in R. Also note that if a placement is too close to a previously chosen good placement, then we do not even try the new placement. This is an efficiency consideration that help keeps the running time low.

It remains to explain exactly why we choose to keep a placement and which previously chosen placement is discarded. Here are the rules. (1) If optPlacements is full, we do not include a placement with Histogram Intersection (HI) value less than the minimum HI value currently in optPlacements. (2) If a placement P is close to a placement $\widehat{P}$ already in optPlacements, and P has a higher HI value than $\widehat{P}$, then we replace $\widehat{P}$ by P. (3) If the HI value of P does not exceed the HI value of all similar placements currently in optPlacements, then we do *not* include P. (4) If we get this far without deciding the fate of P, then its HI value is higher than some currently selected placement and there are no nearby placements in optPlacements. If optPlacements is not full, then we simply add P to optPlacements. Otherwise, we replace the placement with the lowest HI value with P.

For the refinement stage that comes next, we actually need the similarity transformation $g = (s, \theta, t_x, t_y)$ that transforms the query bounding box into the R_{P_*} for each P_* in optPlacements. Just before computing this transformation for a particular P_*, we do a small local search around P_* to see if the placement P_* can be improved. Pseudocode for the whole initial placement phase is shown below.

function G = InitialTransformations($\mathbf{X}$,$\mathbf{a}$,c^0,F^0,$l_{\max}$,β)
  $v_i = \sum_{j=1}^{n} f_{ij}^0, \quad i = 1, \ldots, m$
  $q = v/w$
  $I^* = \{\ i\ :\ q_i \geq \beta\ \} \cup \{\ \arg\max_i q_i\ \}$
  optPlacements = {}
  foreach $i^* \in I^*$
    foreach $((A_I, P_I), W_I) \in \mathbf{X}$ such that $A_I = a_{i^*}$
      if (dist(P_I,optPlacements) too small) **continue**
      $R = R_0 \oplus P_I$ /* *center* R_0 *at* P_I */
      $T = \{\ ((A_K, P_K), W_K)\ :\ P_K \in R\ \}$ /* *rectangular range search* */
      /* *compute image attribute histogram over* R */
      $w^R = 0 \in \mathbf{R}^m$
      foreach $((A_K, P_K), W_K) \in T$
        $\widehat{\imath}$ = attribute#(A_K)

```
            w^R_î += W_K
        end foreach
        /* compute Histogram Intersection */
        HI = H(w^R, v)
        /* accept placement P_K? */
        if ((size(optPlacements) == l_max) and
            (HI ≤ min HI in optPlacements)) continue
        if (HI > HI of similar placement P̂ in optPlacements)
            replace P̂ with P_K /* replace */
            continue
        elseif (HI ≤ HI of all similar placements in optPlacements)
            continue /* do not add */
        end if
        if (size(optPlacements) < l_max)
            add (P_K,HI) to optPlacements /* add */
        else
            remove min HI placement in optPlacements /* replace */
            add (P_K,HI) to optPlacements
        end if
      end foreach
    end foreach
    /* compute similarity transformations to return */
    G = {}
    foreach (P, HI) in optPlacements
      /* check small perturbations of placement P */
      P_ij = P + i × (height(R_0)/4) + j × (width(R_0)/4)    i = −1, 0, 1, j = −1, 0, 1
      P_max = arg max_{i=−1,0,1,j=−1,0,1} HI(placement P_ij)
      add (g : query bounding box ↦ R_0 ⊕ P_max) to G
    end foreach
    return(G)
end function
```

The only expensive step here is the rectangular range search. The number of outer loop iterations and range searches is kept small by only considering locations of high confidence attributes which are significantly different from placements already chosen.

7.3.1 Experiments with the Color Pattern Problem

In this section, we illustrate the performance of the initial placement algorithm for the color pattern problem. In all of the examples in this section, we use $l_{\max} = 2$ placements and $\beta = 0.50 = 50\%$. The placement with the higher histogram intersection score is shown in red, while the other placement is shown in black. For each (image,pattern) pair, we give the attribute-position distribution sizes (M for the image and N for the pattern), the marginal attribute distribution sizes (m colors for the image and n colors for the query), and the running time T_2 for the second phase. The time T_2 does not include the running time of the first phase to produce the scale estimate. The parameters given to the scale estimation algorithm are $c_{\text{min}} = 0.001 = 0.1\%$, $\varepsilon_c = 0.001$, and $\varepsilon_d = 0.0001$ CIE-Lab space units.

The results in Figures 7.5–7.9 are excellent, despite sometimes imperfect scale estimates. Recall that the scale estimation algorithm uses only the amounts of colors present in the image and the pattern, and not their positions. When the pattern appears in more than place in the image, the scale estimate is therefore likely to be greater than the scale of any single pattern occurrence. This is the case for examples shown in Figure 7.7(c) and Figures 7.8(a),(d). In each of these overestimated scale, multiple pattern occurrence examples, the initial placements cover all occurrences of the pattern.

The scale estimate would be guaranteed to be at least the sum of the pattern occurrence sizes if the occurrences in the image had exactly the same colors as the pattern itself and if we did not perform the color clustering efficiency step that represents the image and query as faithfully as possible with as few colors as possible. Figure 7.7(d) shows an example in which the scale is underestimated. In this example, both initial placements are contained within the pattern occurrence in the image.

At least one initial placement in all the examples in Figures 7.5 through 7.8 is very accurate, and in some cases near perfect as in Figure 7.5(c) and Figures 7.6(a),(d). Sometimes the heuristics used for choosing the initial placements without considerable overlap perform poorly, as in Figure 7.7(a). In Figure 7.9, we show some examples in which we search for a logo in advertisements for products with a similar logo to that of the query. In all these examples, the initial placements occur over image regions with color content which is similar to the color content of the query logo.

7.4 The Verification and Refinement Phase

Recall SEDL's notion of visual similarity that each pattern region is close to an image region of similar color or curve orientation (in the shape case). In the verification and refinement phase, SEDL iteratively improves its estimate for the pattern scale, orientation, and location, starting from the scale and locations determined by the scale estimation and initial placement phases. See Figure 7.10 for this main idea illustrated in the color case. The leftmost column contains a pattern that occurs in the image in the rightmost column. The pattern and image signatures in color $\times$ position space are shown in the middle columns. There is a similarity transformation g^* of the pattern regions that aligns pattern regions with image regions of similar colors. Our goal is to find g^* starting from $g^{(0)}$ given by the

(a)

(b)

(c)

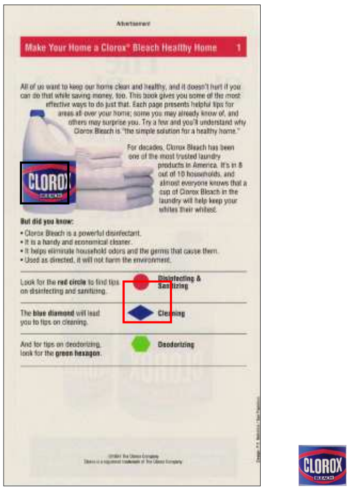
(d)

Figure 7.5: Initial Placement Results – Example Set 1. (a) $M = 491$, $m = 20$, $N = 203$, $n = 14$, $T_2 = 0.012$s. (b) $M = 1002$, $m = 41$, $N = 180$, $n = 18$, $T_2 = 0.010$s. (c) $M = 763$, $m = 35$, $N = 265$, $n = 25$, $T_2 = 0.007$s. (d) $M = 871$, $m = 31$, $N = 84$, $n = 8$, $T_2 = 0.008$s.

(a)

(b)

(c)



(d)

Figure 7.6: Initial Placement Results – Example Set 2. (a) $M = 894$, $m = 29$, $N = 160$, $n = 14$, $T_2 = 0.021$s. The scale estimation and initial placement are near perfect, although the heuristic for selecting initial placements without sizeable overlap performed poorly. (b) $M = 596$, $m = 16$, $N = 127$, $n = 9$, $T_2 = 0.004$s. (c) $M = 1348$, $m = 35$, $N = 325$, $n = 20$, $T_2 = 0.100$s. (d) $M = 382$, $m = 13$, $N = 228$, $n = 19$, $T_2 = 0.007$s. The lower scoring initial placement is only slightly offset from the correct placement. The higher scoring placement is also a good guess at the pattern occurrence given that the exact locations of the colors inside the rectangle are not used.

(a)

(b)

(c)

(d)

Figure 7.7: Initial Placement Results – Example Set 3. (a) $M = 648$, $m = 20$, $N = 436$, $n = 37$, $T_2 = 0.005$s. (b) $M = 533$, $m = 22$, $N = 191$, $n = 12$, $T_2 = 0.005$s. Although the scale is underestimated, the two initial placements are both contained within the pattern occurrence. (c) $M = 545$, $m = 27$, $N = 228$, $n = 19$, $T_2 = 0.004$s. The pattern scale is overestimated due to the occurrence of the pattern at three different image locations. The two initial placements contain all three pattern occurrences. (d) $M = 703$, $m = 21$, $N = 436$, $n = 37$, $T_2 = 0.008$s. As in (b), the scale is underestimated but the two initial placements are both contained within the pattern occurrence.

Figure 7.8: Initial Placement Results – Example Set 4. (a) $M = 898$, $m = 30$, $N = 354$, $n = 36$, $T_2 = 0.023$s. The pattern occurs twice and the scale is overestimated. The higher scoring placement contains both instances of the pattern. (b) $M = 669$, $m = 32$, $N = 180$, $n = 18$, $T_2 = 0.003$s. (c) $M = 873$, $m = 23$, $N = 436$, $n = 37$, $T_2 = 0.003$s. (d) $M = 888$, $m = 43$, $N = 444$, $n = 43$, $T_2 = 0.009$s. Both initial placements overlap one pattern occurrence.

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

Figure 7.9: Initial Placement Results – Example Set 5. In each of these examples, we search for a pattern that does not occur within an image. (a) $M = 410$, $m = 9$, $N = 325$, $n = 20$, $T_2 = 0.008$s. The yellow and red of the Cornpops logo matches the yellow and red of the shredded wheat box. (b) $M = 381$, $m = 9$, $N = 325$, $n = 20$, $T_2 = 0.006$s. The yellow and red of the Cornpops logo matches the yellow and red of the cheerios box and leggos beneath the box. (c) $M = 409$, $m = 9$, $N = 265$, $n = 25$, $T_2 = 0.007$s. The 100% Bran boxes contain white, purple, and a little bit of yellow as in the Taco Bell logo. (d) $M = 420$, $m = 9$, $N = 175$, $n = 10$, $T_2 = 0.009$s. The X14 label is similar to the Comet logo in its color composition.

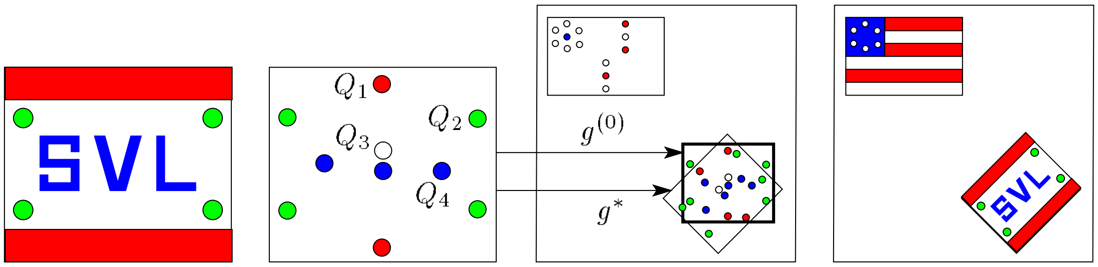


Figure 7.10: The Verification and Refinement Phase. See the text for discussion.

scale estimation and initial placement phases. Notice the importance of starting close to an optimal transformation. The pseudo American flag in the upper left-hand corner of the image contains red, white, and blue just like the query pattern. The search rectangle might be pulled toward this incorrect pattern if the iteration starts too close to the American flag.

We can use the same alternation idea behind the FT iteration to find at least a local minimum of our signature distance function under a transformation set:

$$\mathrm{SD}_{\mathcal{G}}(\mathbf{X},\mathbf{Y}) = \min_{g\in\mathcal{G},\psi\in\Psi} \sum_{J=1}^{N} U_J(d_{\mathrm{attr}}^{\tau}(A_{\psi(J)}, B_J) + \lambda d_{\mathrm{pos}}(P_{\psi(J)}, g(Q_J))).$$

For a fixed transformation $g^{(k)}$, we solve

$$\psi^{(k)} = \min_{\psi\in\Psi} \sum_{J=1}^{N} U_J(d_{\mathrm{attr}}^{\tau}(A_{\psi(J)}, B_J) + \lambda d_{\mathrm{pos}}(P_{\psi(J)}, g^{(k)}(Q_J))). \tag{7.2}$$

This is trivial since Ψ is the set of unconstrained matchings from $[1..N]$ to $[1..M]$. The solution is

$$\psi^{(k)}(J) = \arg\min_{I\in[1..M]} (d_{\mathrm{attr}}^{\tau}(A_I, B_J) + \lambda d_{\mathrm{pos}}(P_I, g^{(k)}(Q_J))) \qquad J = 1,\ldots,N. \tag{7.3}$$

That is, the correspondence step (7.2) involves N nearest neighbor computations over a set of size M. We shall come back to this computation shortly. The transformation step for a fixed matching $\psi^{(k)}$ is to compute the optimal transformation

$$g^{(k+1)} = \arg\min_{g\in\mathcal{G}} \sum_{J=1}^{N} U_J(d_{\mathrm{attr}}^{\tau}(A_{\psi^{(k)}(J)}, B_J) + \lambda d_{\mathrm{pos}}(P_{\psi^{(k)}(J)}, g(Q_J))). \tag{7.4}$$

In performing the minimization (7.4), we need only compute

$$g^{(k+1)} = \arg\min_{g\in\mathcal{G}} \sum_{J=1}^{N} U_J d_{\mathrm{pos}}(P_{\psi^{(k)}(J)}, g(Q_J)) \tag{7.5}$$

since we only allow transformations g of the attribute-position signature that do not modify the attributes. The minimization (7.5) is easy to compute since we use $d_{\mathrm{pos}} = L_2^2$ in SEDL.

If we let $D^{(k)} = D(\psi^{(k)}, \mathbf{X}, g^{(k)}(\mathbf{Y}))$, then it is easy to show that

$$0 \le D^{(k+1)} \le D^{(k)} \quad \forall k. \tag{7.6}$$

Indeed,

$$\begin{array}{lllll} (7.2) & \Rightarrow & D^{(k)} = D(\psi^{(k)}, \mathbf{X}, g^{(k)}(\mathbf{Y})) & \geq & D(\psi^{(k+1)}, \mathbf{X}, g^{(k)}(\mathbf{Y})) \quad \text{and} \\ (7.4) & \Rightarrow & D(\psi^{(k+1)}, \mathbf{X}, g^{(k)}(\mathbf{Y})) & \geq & D(\psi^{(k+1)}, \mathbf{X}, g^{(k+1)}(\mathbf{Y})) = D^{(k+1)}, \end{array}$$

from which (7.6) follows. Hence, we have a monotically convergent sequence $< D^{(k)} >$. The result of the previous initial placement phase is a handful (typically ≤ 5) of initial transformations $g^{(0)}$ from which to begin the above iteration.

Now that we have presented the iteration and proved its convergence, let us return to the correspondence step computation (7.3). A brute force computation of $\psi^{(k)}$ requires computing $d_{\text{ap}}((A_I, P_I), (B_J, Q_J))$ for all MN pairs $(I, J) \in [1..M] \times [1..N]$. This computation time can be greatly improved by reorganizing the computation (7.3) as

$$\begin{aligned} \psi^{(k)}(J) &= \arg \min_{i \in [1..m]} \left(\min_{I \,:\, A_I = a_i} (d^{\tau}_{\text{attr}}(A_I, B_J) + \lambda d_{\text{pos}}(P_I, g^{(k)}(Q_J))) \right) \qquad J = 1, \ldots, N \\ &= \arg \min_{i \in [1..m]} \left(d^{\tau}_{\text{attr}}(a_i, B_J) + \lambda \min_{I \,:\, A_I = a_i} d_{\text{pos}}(P_I, g^{(k)}(Q_J))) \right) \qquad J = 1, \ldots, N. \end{aligned}$$

The point sets $P^i = \{ P_I \,:\, A_I = a_i \}$ can be preprocessed to allow Euclidean nearest neighbor searches in time $O(\log |P^i|)$. This allows us to compute $\psi^{(k)}(J)$ in time $O(\sum_{i=1}^m \log |P^i|)$. Since log is a concave function, Jensen's inequality[1] implies

$$\frac{1}{m} \sum_{i=1}^{m} \log |P^i| \leq \log \left(\frac{1}{m} \sum_{i=1}^{m} |P^i| \right) = \log \left(\frac{M}{m} \right).$$

Therefore, $\sum_{i=1}^m \log |P^i| = O(m \log(M/m))$ and we can compute the entire vector $\psi^{(k)}$ in time $O(Nm \log(M/m))$. If there is any doubt that $m \log(M/m)$ is much smaller than M, just note that $M - m \log(M/m) = m((M/m) - \log(M/m))$.

Combining attribute and position distance in a measure of visual similarity is a very difficult task. The use of a linear combination of the two distances with a constant weighting factor λ is a computationally efficient, reasonably effective solution. We shall discuss this solution and its problems in terms of the color case.

The use of the linear combination d_{ap} (see (7.1)) in the signature distance function $D_{\mathcal{G}}$ captures the fact that two patterns with very similar color regions in very similar locations will appear visually similar. Setting color distances above some threshold to ∞ prevents a small matching cost between nearby regions of very different colors, and heavily penalizes such obvious visual dissimilarities. The threshold τ cannot be too small, however, since we

[1] For a concave function f, Jensen's inequality states that $\sum_{i=1}^m \alpha_i f(x_i) \leq f(\sum_{i=1}^m \alpha_i x_i)$ if $\alpha_i \geq 0 \; \forall i$ and $\sum_{i=1}^m \alpha_i = 1$. With $\alpha_i \equiv \frac{1}{m}$, we get $\frac{1}{m} \sum_{i=1}^m f(x_i) \leq f(\frac{1}{m} \sum_{i=1}^m x_i)$.

need to allow for inexact pattern matches.

The function d_{ap} reasonably trades off small changes in position distance with small changes in color distance, but it does not prevent larger tradeoffs which may lead to a small signature distance without visual similarity; all region pairs $((A, P), (B, Q))$ with the same value $d_{\text{ap}}((A, P), (B, Q)) = d^{\tau}_{\text{attr}}(A, B) + \lambda d_{\text{pos}}(P, Q)$ contribute the same penalty in the signature distance. SEDL's use of a linear combination of color and position distance identifies near exact pattern matches, captures many inexact matches, and identifies obviously dissimilar patterns, but can lead to false positives. Although some of these false positives will be eliminated in the step described below, there is still room for improvement in SEDL's use of color and position information. For example, the perceived color of a region depends on the color of its surrounding regions, but d_{ap} does not take this into account.

Once the alternating iteration (7.2),(7.4) converges, SEDL checks that at least some fraction of the attribute weight inside the final search rectangle R_* can be matched to pattern attribute weight over moderate distances in attribute space. This final check is in recognition of the difficulty of trading off attribute and position distance, and helps eliminate false positives. A visually similar match between areas implies somewhat similar attribute signatures for those areas.

The final check in the verification and refinement phase is performed using the τ-EMD described in section 4.4.2. Recall that the τ-EMD measures the fraction of weight in the lighter distribution which *cannot* be matched to weight in the heavier distribution using only ground distances that do not exceed τ units. Let $\mathbf{x}^{R_*}$ be the approximate image attribute distribution inside R_* (computed by a range search as described in section 7.3), and let $\mathbf{y}_*$ be the pattern attribute distribution scaled by the final scale value c_* (the area of R_*). Then the last step in the verification and refinement phase is to check that $\tau\text{-EMD}(\mathbf{x}^{R_*}, \mathbf{y}_*) < \eta$. If not, then R_* is eliminated from further consideration. We cannot choose τ or η too small since inexact matches require some tradeoffs in attribute and position distance, and we do not want to incorrectly reject such matches. For the product advertisement database, we use $\eta = 0.5$, requiring that at least 50% of the color weight to be matched; for the Chinese character database, we use $\eta = 0.75$, requiring at least 25% of the orientation weight to be matched.

We now discuss our choice of distance function SD in relation to the ICP algorithm used to register shapes. Recall that the ICP algorithm uses the distance function

$$D^{\text{ICP}}(\mathbf{X}, \mathbf{Y}) = \min_{\psi \in \Psi} \sum_{J=1}^{N} ||P_{\psi(J)} - Q_J||_2^2,$$

where $\mathbf{X}$ and $\mathbf{Y}$ are the point sets

$$\mathbf{X} = \{ P_1, \ldots, P_M \} \qquad \text{and} \qquad \mathbf{Y} = \{ Q_1, \ldots, Q_N \}.$$

The ICP iteration seeks to solve the optimization problem

$$D^{\mathrm{ICP}}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) = \min_{g \in \mathcal{G}} D^{\mathrm{ICP}}(\mathbf{X}, \mathbf{Y}) = \min_{g \in \mathcal{G}, \psi \in \Psi} \sum_{J=1}^{N} ||P_{\psi(J)} - g(Q_J)||_2^2.$$

If we eliminate the weights U_J and the attribute part of the ground distance d_{ap}, then our distance function SD reduces to D^{ICP}, and $\mathrm{SD}_{\mathcal{G}}$ reduces to $D^{\mathrm{ICP}}_{\mathcal{G}}$. In fact, the weights U_J are effectively eliminated when our framework is applied to shape-based retrieval since we compute average orientations over equal-sized portions of image curves to produce the image signature, and the weight of an attribute is equal to the (constant) length over which the averages are computed.

The use of orientation attributes which are not modified by any transformation is a very important improvement over the ICP algorithm when registration considers only changes in scale and location (and not orientation). The ICP framework matches only by ink on the page. By using the orientation of the ink, we can avoid matching query ink that is physically close to image ink, but that will not produce a good visual match between the underlying shapes. The point here is applicable in a more general setting than our shape setting: using attributes which are unmodified during iteration gives the matcher something constant upon which to anchor its matching and transformation decisions. In fairness to the ICP algorithm, it is specifically designed to handle differences in orientation.

Another natural question to raise at this time is why we choose to compare the attribute-position distributions $\mathbf{X}$ and $\mathbf{Y}$ with the function SD instead of the EMD. The main answer is speed. We can use the results in section 6.5 on allowing weight-altering transformations to compute the EMD under a transformation set in which the scale changes both the distribution points and the weights (which represent image areas in the color case). This requires several EMD computations for a single comparison under a transformation set. Our distributions are so large, however, that the time for even a single EMD computation per (query,image) pair is too much for a content-based retrieval system which must maintain some interactivity with the user. Even if we could compute the EMD for SEDL's large signatures in an acceptable time, it is still an imperfect measure of image similarity because our representation is imperfect. All masses are concentrated at the centroids of image regions. For large regions, it would be a more accurate representation to spread the mass uniformly over the region. We do not, however, know how to compute the EMD between two such continuous distributions.

SEDL's color $\times$ position signatures are defined (roughly) by the largest possible uniform color regions. If a single red pattern region is represented in the image as three adjacent regions of slightly different reds, there may be a relatively large, unjustified matching penalty using SEDL's signature distance. To help reduce the effects of such problems, region distances are weighted by pattern region area, the theory being that large pattern regions are more stable in appearance and easier to detect as a single region within the image. The excellent results obtained show that this strategy is effective. In the shape case, SEDL avoids such representation problems to a large extent by using a fine sampling of image curves. This strategy is feasible because the shape data is one-dimensional. A fine sampling over the 2D image plane, however, would produce color $\times$ position too large to match in acceptable times for retrieval.

7.5 Results

In this section, we show the results of the entire matching process for the color advertisement database and the shape Chinese character database.

7.5.1 The Product Advertisement Color Database

The advertisement database and product logos used in the experiments in this section were obtained from the web site `http://vis-www.cs.umass.edu/~mdas/color_proj.html` for the FOCUS work by Das et al. ([14]). The twenty five product logos used as queries are shown in Figure 7.11. The database contains 361 advertisements for products with and without corresponding query logos.

7.5.1.1 Creating Signatures

In what follows, a *signel* (signature element) refers to a single element $((A_I, P_I), W_I)$ in the color $\times$ position signature $\mathbf{X} = \{((A_I, P_I), W_I)\}_{I=1}^{M}$ of an image. The signature creation process consists of three steps: (1) clustering in color space to reduce the number of colors in an image, (2) connecting pixels with the same color label to form an initial set of signels, and (3) combining same-colored signels which are close together in the image to form the final set of signels which define the image signature in color $\times$ position space.

The color clustering step uses an algorithm due to Rubner et al. ([65]) which averages pixel colors together when the colors are contained in a small enough rectangular region of the color space. As in [65], we use the perceptual CIE-Lab color space ([88]). The

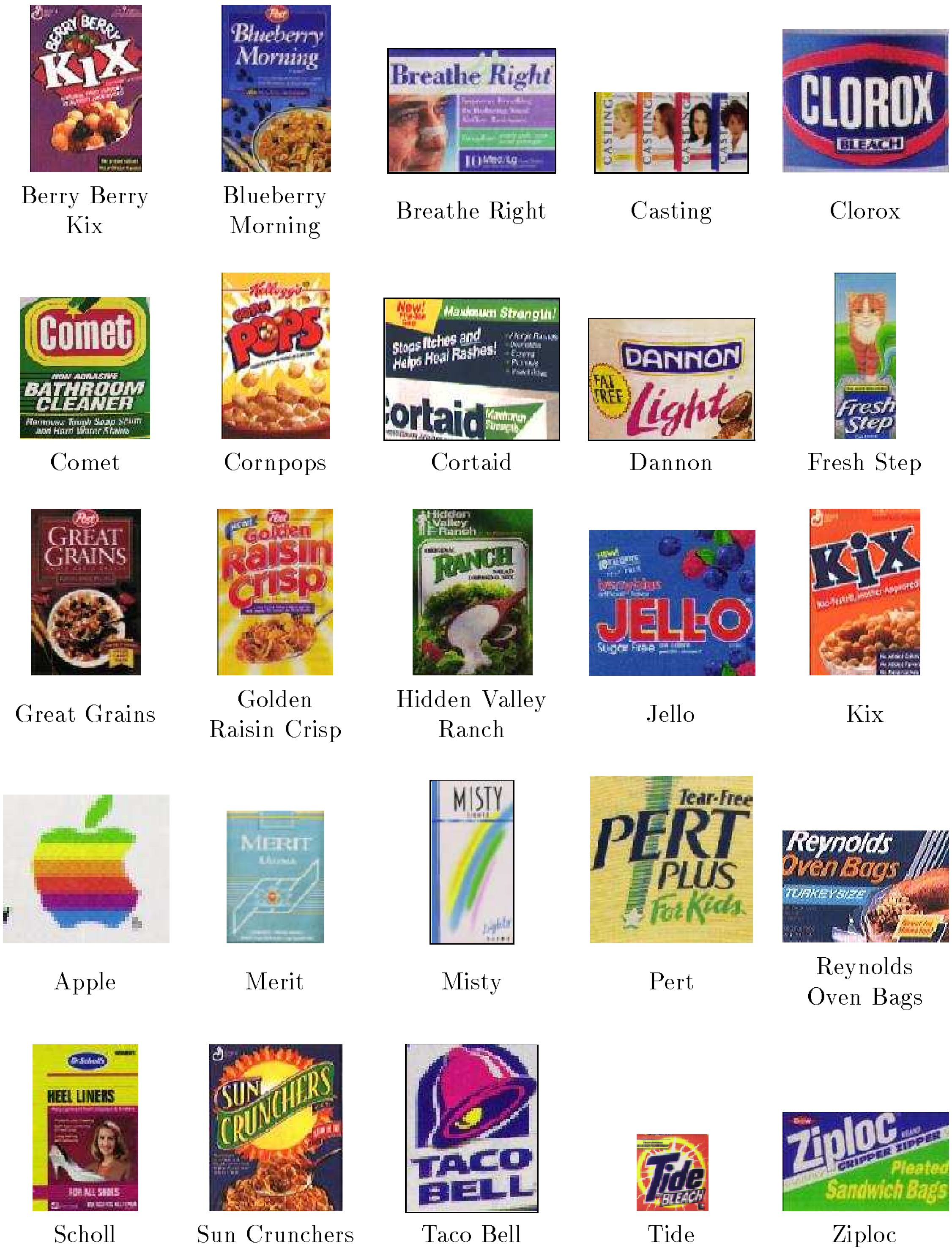


Figure 7.11: Queries for the Color Product Advertisement Database.

L component is a luminance channel, the a component is a red-green channel, and the b component is a blue-green channel. The CIE-Lab space is *perceptual* in the sense that Euclidean distance in this space matches perceptual distance between two colors that are not very different. The initial box which contains all colors ($0 \leq L \leq 100$, $-120 \leq a \leq 120$, $-120 \leq b \leq 120$) is subdivided alternately along each dimension of the color space until the box region size is small enough so that the average color inside the box approximates well all colors inside the box. There is a second stage of the clustering algorithm that merges similar clusters that were broken by fixed boundaries used in the first clustering stage. See [65] for details. By choosing the "small enough" threshold properly, we can usually reduce the image to a small number (5–40) of colors with very little loss of color accuracy. By *reduce*, we mean replace the color at each image pixel with the nearest color cluster representative.

After an image is reduced, we compute all sets of connected pixels with the same color label using Heckbert's seed fill algorithm ([29]). Each of these sets is an initial region in a region-merging process whose ultimate goal is to define the image signature in color $\times$ position space. Associated with each region is a color signature for the pixels contained in the region, along with the position centroid for the region pixels of each color cluster. The color signature for an initial region contains only one color.

When two regions R^i, $i = 1, 2$, are merged into $R = R^1 \bigcup R^2$, we must combine their color signatures and compute the position centroids in R of each color cluster. This is straightforward. Suppose that the image is reduced to n colors $c_1, c_2, \ldots, c_n$, and that region R^i, $i = 1, 2$, has color signature $\mathbf{x}(R^i) = \{ k_1^i, \ldots, k_n^i \}$, where k_j^i is the number of pixels in region R^i with color c_j. Also, let p_j^i denote the centroid of the pixels in R^i with color c_j (if $k_j^i > 0$). Then $\mathbf{x}(R) = \{ k_1, \ldots, k_n \}$, where $k_j = k_j^1 + k_j^2$, and $p_j = (k_j^1 p_j^1 + k_j^2 p_j^2)/(k_j^1 + k_j^2)$ if $k_j^1 + k_j^2 > 0$. Here p_j is the centroid of all pixels in $R = R^1 \bigcup R^2$ with color c_j.

Each final region contributes a number of color $\times$ position signels ((color, position), weight) equal to the number of colors contained within the region. For example, if R as given above is a final region, then we get a color $\times$ position signel $((c_j, p_j), k_j)$ for each $k_j > 0$. The role of a region is to define which image pixels of the same color are okay to combine into one signel. Note that merging two regions with no color clusters in common does not change the color $\times$ position signature derived from the set of all regions.

The region merging process maintains a priority queue of all region adjacencies. The next pair of regions to be merged is the adjacent pair with the smallest priority. If we denote the area of a region R by $\text{area}(R)$, then the priority for region adjacency (R^1, R^2) is

$$\text{priority}(R^1, R^2) = \text{EMD}(\mathbf{x}(R^1), \mathbf{x}(R^2)) \times \min(\text{area}(R^1), \text{area}(R^2)). \tag{7.7}$$

Here we assume that all region color signatures have been normalized to have total weight equal to one before the EMD is computed. Given our color clustering and seed fill steps to produce the initial set of regions, what ordering on region merges does the priority (7.7) imply?

Consider the situation before any region merging has been performed. The connection of pixels with the same color label typically results in many tiny initial regions of a few pixels or less (more about this later). It is even common to have single pixel initial regions. On the other hand, the EMDs between adjacent initial regions cannot be very small. The EMD between two adjacent initial regions cannot, for example, be zero because this implies that the adjacent initial regions have the same color and, therefore, would have been connected by the seed fill algorithm. Furthermore, the colors of the initial regions are the result of clustering in color space in which colors within a certain distance from each other are merged into a single cluster. Therefore, the EMD between two adjacent initial regions cannot be arbitrarily small.

If the EMD between adjacent regions is, for example, at least e units in CIE-Lab space, then the size of the smaller region in any region adjacency with priority at most p is at most p/e. The minimum priority during the region merging process tends to increase as the process proceeds since the EMDs between adjacent regions and the region sizes both tend to increase. While adjacencies with priority less than p are being processed, regions of size more than p/e are very unlikely to be merged with an adjacent region (the merging process would need to create an adjacency with EMD less than e units for this to happen). The result is that, despite the symmetry in the priority (7.7), region adjacencies with a large EMD and a small minimum size are generally eliminated before adjacencies with a small EMD and large minimum size. Of course, among adjacencies with small size the priority (7.7) gives preference to eliminating those with small EMD over those with large EMD. The causes of these small size adjacencies in the initial set of regions are discussed next.

The first region adjacencies that will be eliminated are those with similar colored regions in which at least one of the regions is very small (the EMD factor and the area factor are both small). Such adjacencies typically arise in areas of the image that have a gradual shading change. The color clustering algorithm may choose two color clusters to represent the lighter and darker colors in such an area, but what happens to the colors which are roughly equidistant from these clusters in color space is much less clear. The color clustering algorithm does not use the positions of the colors in the image plane, so it cannot use a connectedness criterion to help make this decision. Furthermore, the Euclidean distance is only an approximation to perceptual distance in the CIE-Lab color space. If a color is close perceptually to two color clusters, the distances in CIE-Lab space to the clusters will be small but might not properly distinguish which cluster is perceptually closer. A conceptual group of same-colored pixels may be disconnected into several very small groups of connected pixels; there may even be some pixels that are not connected to any pixels of the same color label. Such a fragmented group is likely to be contained in a larger region with a similar color. By merging the very small fragment regions into the larger surrounding region, the fragments are combined into one signel as desired.

The next set of region adjacencies that will be eliminated are those with fairly different color descriptors in which at least one of the regions is very small (the EMD factor is relatively large, but the area factor is small). Such adjacencies typically arise in the advertisements when there is very small lettering in the advertisement. Such lettering is usually placed on a relatively high contrast background to aid legibility. For the purposes of our system, summarizing a paragraph of tiny black lettering with one signel of black at the position centroid of the paragraph is a wise compression since we know a priori that we are not interested in such tiny scale features. This compression is exactly what happens as each individual letter (connected black pixels) region is merged with its background region.

A mistake is made if a merge is performed between two regions with large, spatially separated signels of the same color. In this case, the weight of the combined signel in the final signature will be large and its position will be inaccurate. If there are large amounts of the same color in the regions, then the regions themselves must be relatively large and the EMD between the regions will be relatively small. Adjacencies with large regions and small EMD are, as mentioned above, generally processed after those with at least one small region.

The big question, of course, is when to stop the merging process. Currently we use a very simple criterion which halts the region merging once a pre-specified number of regions has been reached. In a general image database setting, we may or may not have the time and resources to pick a different number of regions for each image entered into the database. For all images in the advertisement database, we stop when there are 128 regions. This number was found acceptable for the complexity of a general advertisement. Examples of the final regions for three database images are shown in Figure 7.12. The problem of finding a "natural" stopping point is a very difficult one, even with a specified range of feature scales in which we are interested. For us, "natural" means that the correct conceptual grouping has been done. For all the query logos, we stop the region merging process when there are 32 regions.

It is interesting to see the results of allowing the region merging process to proceed down to two image regions. See Figure 7.13. The priority (7.7) produces a natural hierarchy of

Figure 7.12: Region Merging Results. (left) Original image. (right) The original image reduced to 128 regions. Each region is colored with the average color of its pixels.

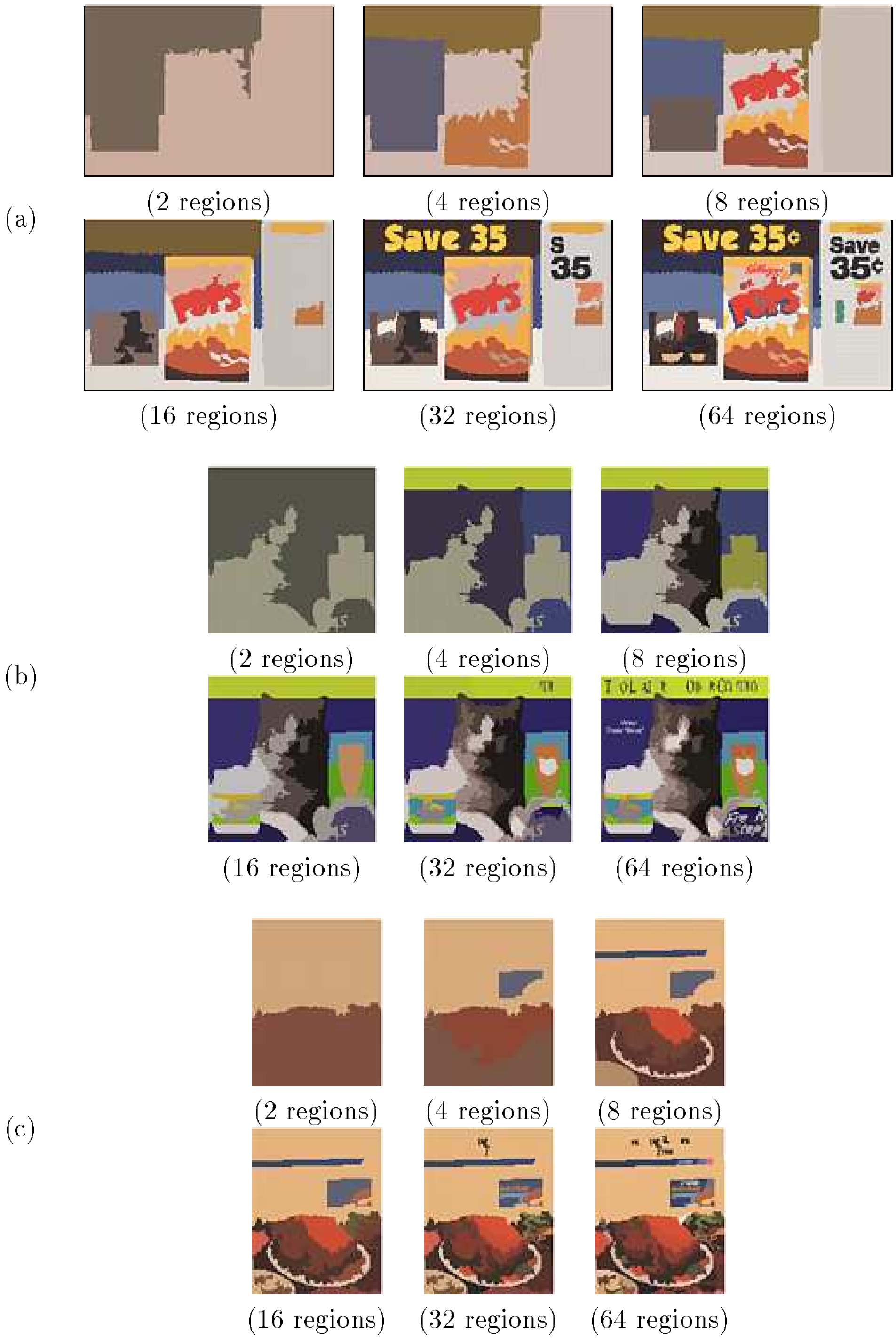


Figure 7.13: More Region Merging Results. Here we show results of allowing the region merging process to proceed down to two image regions. (a) Cornpops advertisement on the top left of Figure 7.12. (b) Fresh Step advertisement on the middle left of Figure 7.12. (c) Reynolds Oven Bag advertisement on the bottom left of Figure 7.12.

segmentations over varying levels of detail (the number of regions).

7.5.1.2 Query Results

We now present some results of SEDL's pattern problem strategy applied to the advertisement database. The "correct" answers for a given product logo are the advertisements for that product. Some exceptions have been made for advertisements that contain a version of the product logo which is substantially different in color from the query logo. See Figure 7.14(a),(b) for example ads that we will not penalize SEDL or FOCUS for missing. At the same time, we retain more challenging cases when the advertisement does not contain the exact logo but is close enough that a good system should be able to handle the small differences. See Figure 7.14(c),(d) for two such examples. The first column in Figure 7.15 shows in parentheses the number of correct answers for each query logo. The FOCUS results were obtained directly from the FOCUS web demo at `http://vis-www.cs.umass.edu/~mdas/color_proj.html` on 8/17/98. See section 2.6 for a review of the FOCUS retrieval system.

The *recall* of a query for a given number R of returned images is the number of correct retrievals in the top R retrievals divided by the number of correct answers. From the first and second columns in Figure 7.15, we see, for example, that SEDL's recall for the Dannon query is $3/4 = 75\%$ and for the Clorox query is $5/5 = 100\%$.

The *precision*, on the other hand, is a measure of the spread of the correct images returned. If there were four correct retrievals in the top twenty, then a query result is more precise if the correct retrieval ranks are 1, 2, 3, and 4 than if the ranks are 3, 7, 10, and 12. Some researchers measure the precision as the number of correct retrievals divided by the rank of the last correct retrieval. This measure, however, does not distinguish between the rank sets 1,2,3,12 and 9,10,11,12; both have precision $4/12 = 33\%$ under the previous definition. We use a definition that accounts for the distribution of ranks by weighing ranks close to one more heavily than ranks closer to R. If there are n correct retrievals in the top R with ranks $1 \leq r_1 < r_2 \cdots < r_n$, then the precision is

$$\rho = \frac{\sum_{i=1}^{n}(R+1-r_i)}{\sum_{i=1}^{n}(R+1-i)}. \tag{7.8}$$

Here we give rank 1 a weight of R, rank 2 a weight of $R-1$, and, in general, rank j a weight of $R+1-j$. The denominator in (7.8) normalizes the precision so that $0 < \rho \leq 1$, with $\rho = 1$ indicating perfect precision ($r_i \equiv i$). If there are no correct retrievals (i.e. $n = 0$), then we set $\rho = 1$ since this bad result is already punished in the recall statistics. The goal, of course, is to obtain simultaneously high recall and precision.

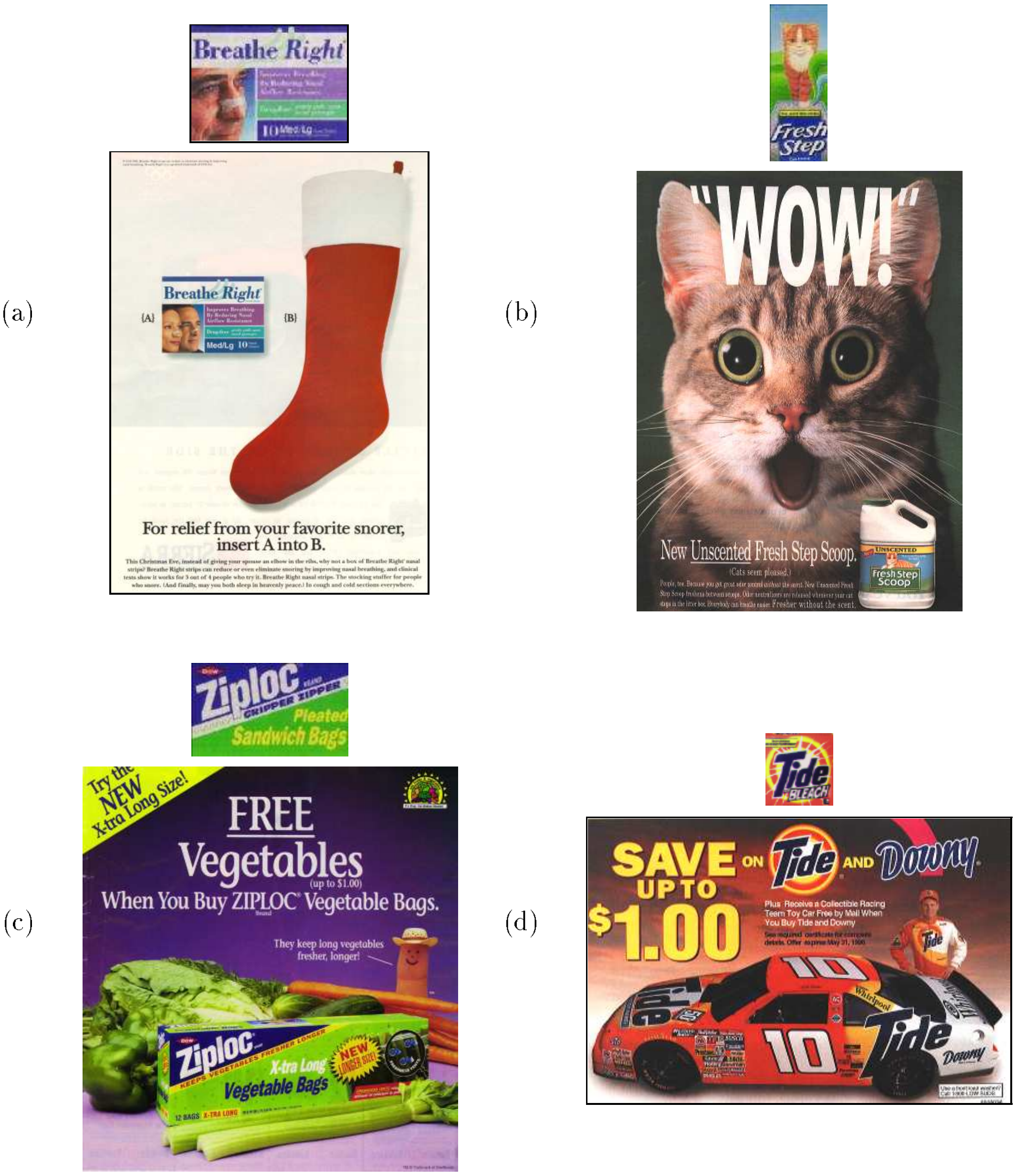


Figure 7.14: Tough Advertisements to Retrieve. (a) The layout of the Breathe Right logo occurrence is the same as that of the query logo, but the colors are different. (b) The advertisement is for "Fresh Step Scoop", while the logo is for "Fresh Step". The layout for the labels of these two products is quite different. (c) The labels for "Ziploc Sandwich Bags" (the query) and "Ziploc Vegetable Bags" (in the advertisement) are very similar but not exactly the same. (d) The Tide advertisement does not contain the Tide box (the query), but it does contain the core Tide logo at the top center.

Query	SEDL Ranks	FOCUS Ranks
Berry Berry Kix (2)	8	2, 4
Blueberry Morning (4)	1, 3, 20	1, 3, 4, 9
Breathe Right (3)	1, 2, 3	1, 2, 3
Casting (4)	1, 5, 13	1, 2
Clorox (5)	1, 3, 4, 7, 8	1, 2, 3
Comet (2)	1, 2	—
Cornpops (2)	1, 3	12, 13
Cortaid (2)	1, 2	2, 8
Dannon (4)	1, 3, 5	1
Fresh Step (2)	1, 2	1, 3
Great Grains (2)	2, 4	3
Golden Raisin Crisp (2)	1, 15	1, 2
Hidden Valley Ranch (9)	1, 2, 6, 11	1, 5
Jello (2)	1, 2	1
Kix (3)	1, 5, 11	1
Apple (3)	1, 4	1, 2, 9
Merit (6)	1, 2, 4, 5, 12, 13	1
Misty (6)	1, 2, 3, 14	—
Pert (2)	1, 11	—
Reynolds Oven Bags (5)	1, 2, 3, 9	1, 2, 3, 5
Scholl (4)	1, 2, 9	—
Sun Crunchers (3)	1	1, 9
Taco Bell (2)	1, 2	1, 4
Tide (7)	1, 2, 8, 12	1, 6, 15, 24, 39
Ziploc (4)	1, 2, 14	1, 2, 4

Figure 7.15: Query Results for the Color Advertisement Database. The FOCUS ranks are phase two results. An "—" entry means that there were no correct answers in the returned images.

In the experiments described in this section, the database searched by SEDL is a subset of the database used in the FOCUS web demo. We selected 361 advertisements of the 1200 advertisements and nature images in the FOCUS database.[2] In all the examples in section 7.3.1, using $l_{\max} = 2$ initial transformations was sufficient to find the pattern occurrences. There are, however, some examples that require a couple more initial placements to find the pattern. Here we use $l_{\max} = 4$ initial placements. As in the scale estimation experiments in section 4.5.1, we set the smallest scale in which we are interested to $c_{\min} = 0.001 = 0.1\%$. Finally, for efficiency reasons we do not proceed with the match verification for a particular initial transformation if the first distance iterate $D^{(0)}$ is very large. If this happens, we simply discard the initial placement.

For SEDL's recall and precision statistics, we use $R = 20$ images (approximately 5.5% of the SEDL database). To be fair to FOCUS, we use $R = 40$ images (about 3.3% of the FOCUS database)[3] Figure 7.15 shows the ranks of the correct retrievals for both SEDL and FOCUS after phase two, while Figure 7.16 gives the summary recall and precision statistics. SEDL's total recall of 77.8% is higher than the FOCUS total recall of 53.3%. Both systems achieved high precision with R which is very small compared to the database size. The average FOCUS precision was 96%, while SEDL's average precision was 88%. The final column in Figure 7.16 shows the time SEDL takes for each query. SEDL's average query time is 40.0 seconds. FOCUS has a sizeable advantage over SEDL in speed, requiring less than a second on average for a query in their larger database of 1200 images. There is more to say, however, about the differences between SEDL and FOCUS than just comparing statistics. We return to this comparison in section 7.5.1.3 after we show some SEDL query results.

Figures 7.17–7.24 show the first five or ten images returned by SEDL for several of the query logos. These figures also show the scale, orientation, and location where SEDL believes that the pattern occurs in some of the correct retrievals. Many more examples of where SEDL finds the pattern are given in Figures 7.25–7.27. In general, SEDL is very accurate in localizing the patterns. Most of the pattern occurrences are *not* rotated from their logo form. By allowing an orientation with this database, we are making SEDL's job more difficult. Sometimes the orientation is far from correct (e.g. see the Taco Bell advertisement in the bottom-right of Figure 7.24). No comparison of accuracy with FOCUS

[2] For the query logos, 18 of the 25 are exactly the same as those used in the FOCUS demo. Small modifications were made to the other 7 logos to eliminate areas not part of the actual logo and to make a rectangular pattern. The seven logos which are different are Comet, Cortaid, Dannon, Apple, Merit, Misty, and Tide. The differences are small and the results are still fair to compare.

[3] Only two ranks for correct retrievals by FOCUS were greater than 20, with the maximum being 39 (see the ranks for the Tide query in Figure 7.15).

Query	SEDL Recall	FOCUS Recall	SEDL Precision	FOCUS Precision	SEDL Query Time (secs)
Berry Berry Kix	1/2	2/2	0.65	0.96	38.2
Blueberry Morning	3/4	4/4	0.68	0.95	20.0
Breathe Right	3/3	3/3	1.00	1.00	41.7
Casting	3/4	2/4	0.77	1.00	37.0
Clorox	5/5	3/5	0.91	1.00	33.6
Comet	2/2	0/2	1.00	1.00	33.6
Cornpops	2/2	2/2	0.97	0.72	58.5
Cortaid	2/2	2/2	1.00	0.91	26.6
Dannon	3/4	1/4	0.97	1.00	118.2
Fresh Step	2/2	2/2	1.00	0.98	32.0
Great Grains	2/2	1/2	0.92	0.95	37.3
Golden Raisin Crisp	2/2	2/2	0.67	1.00	35.8
Hidden Valley Ranch	4/9	2/9	0.86	0.96	28.2
Jello	2/2	1/2	1.00	1.00	14.7
Kix	3/3	1/3	0.81	1.00	34.2
Apple	2/3	3/3	0.95	0.95	3.1
Merit	6/6	1/6	0.85	1.00	25.9
Misty	4/6	0/6	0.86	1.00	40.2
Pert	2/2	0/2	0.77	1.00	60.2
Reynolds Oven Bags	4/5	4/5	0.93	0.99	110.8
Scholl	3/4	2/4	0.89	1.00	7.4
Sun Crunchers	1/3	2/3	1.00	0.91	33.7
Taco Bell	2/2	2/2	1.00	0.97	21.6
Tide	4/7	5/7	0.82	0.63	103.0
Ziploc	3/4	3/4	0.81	0.99	4.4
average	70/90 $\doteq$ 77.8%	48/90 $\doteq$ 53.3%	0.88	0.96	40.0

Figure 7.16: Recall, Precision, and Timing Results for the Color Advertisement Database. The FOCUS statistics are for phase two ranks.

is possible since FOCUS does not compute the scale or location of the pattern.

7.5.1.3 SEDL versus FOCUS

In FOCUS, adjacency graph vertices and edges are computed based on a grid imposed on an image, as described in detail in section 2.6. All the blue, for example, in a single grid cell is treated as one region, and there are edges between different color vertices within the same cell and neighboring cells. This reduces the number of graph vertices, but leads to false adjacencies which can cause false matches. The FOCUS authors did an excellent job of choosing the cell size so that the graphs are small enough to match quickly, yet still descriptive enough to yield excellent precision.

SEDL uses the amounts of colors, the sizes of regions, and (roughly) the centroids of uniform color regions, while FOCUS uses only the existence of colors and the possibility that two different color regions are adjacent. We believe that at least the amount of information used in SEDL will be necessary for large databases, and we speculate that FOCUS will need much better position localization, which means fewer false adjacencies in larger graphs requiring more time to match, to maintain high precision. Perhaps the shapes of regions (not used in SEDL or FOCUS) will also be needed as the database size increases. Of course, using too much information may reduce the recall rate since inexact matches need to be allowed. The issue of which and how much information needed for large databases is far from settled.

Finally, recall that the SEDL framework is general enough to be applied to both the color and shape pattern problems, whereas the FOCUS system is specific to the color case.

7.5.2 The Chinese Character Shape Database

In this section, we apply our general pattern retrieval strategy to find shape patterns within a database of 2000 Chinese characters. A small sample of the images in this database is shown in Figure 7.28. This collection of Chinese characters is an ideal database to test our ideas because there are many patterns that occur throughout the database at different scales and locations. Our shape summary of a character is the *medial axis* of the set of black pixels which define the character. The results shown in Figure 7.29 were computed using algorithms and software by Ogniewicz and Kübler ([52]). The skeletons are a very good one-dimensional summary of the characters.

Figure 7.17: Advertisement Query Result – Clorox. (top) Clorox query logo. (middle) The top ten images returned by SEDL. The Clorox advertisements are at positions 1, 3, 4, 7, and 8. (bottom-left) The match is excellent (rank=1st). (bottom-right) One of the Clorox logo occurrences is found, but the final scale is slightly too big (rank=3rd).

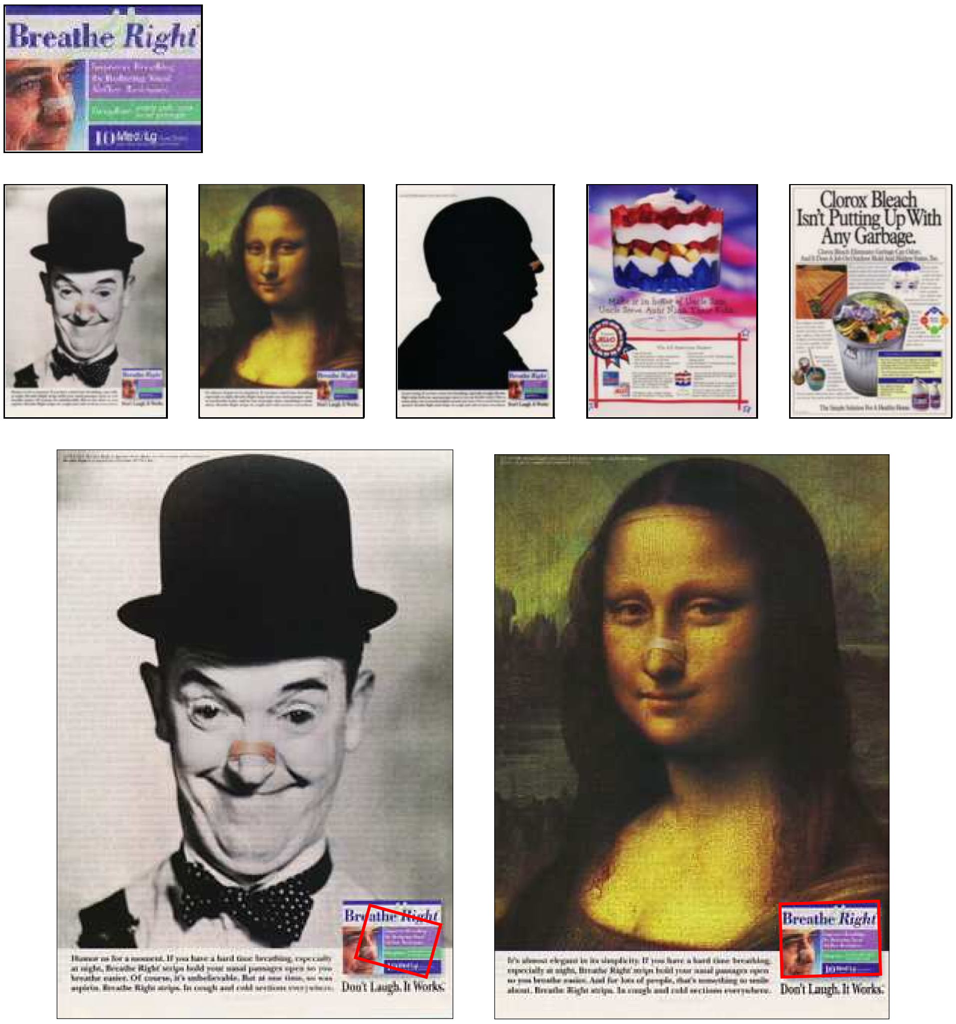


Figure 7.18: Advertisement Query Result – Breathe Right. (top) Breathe Right query logo. (middle) The top five images returned by SEDL. The Breathe Right advertisements are at positions 1, 2, and 3. (bottom-left) The pattern occurence is correctly located, but the orientation is incorrect (rank=1st). (bottom-right) The match is perfect (rank=2nd).

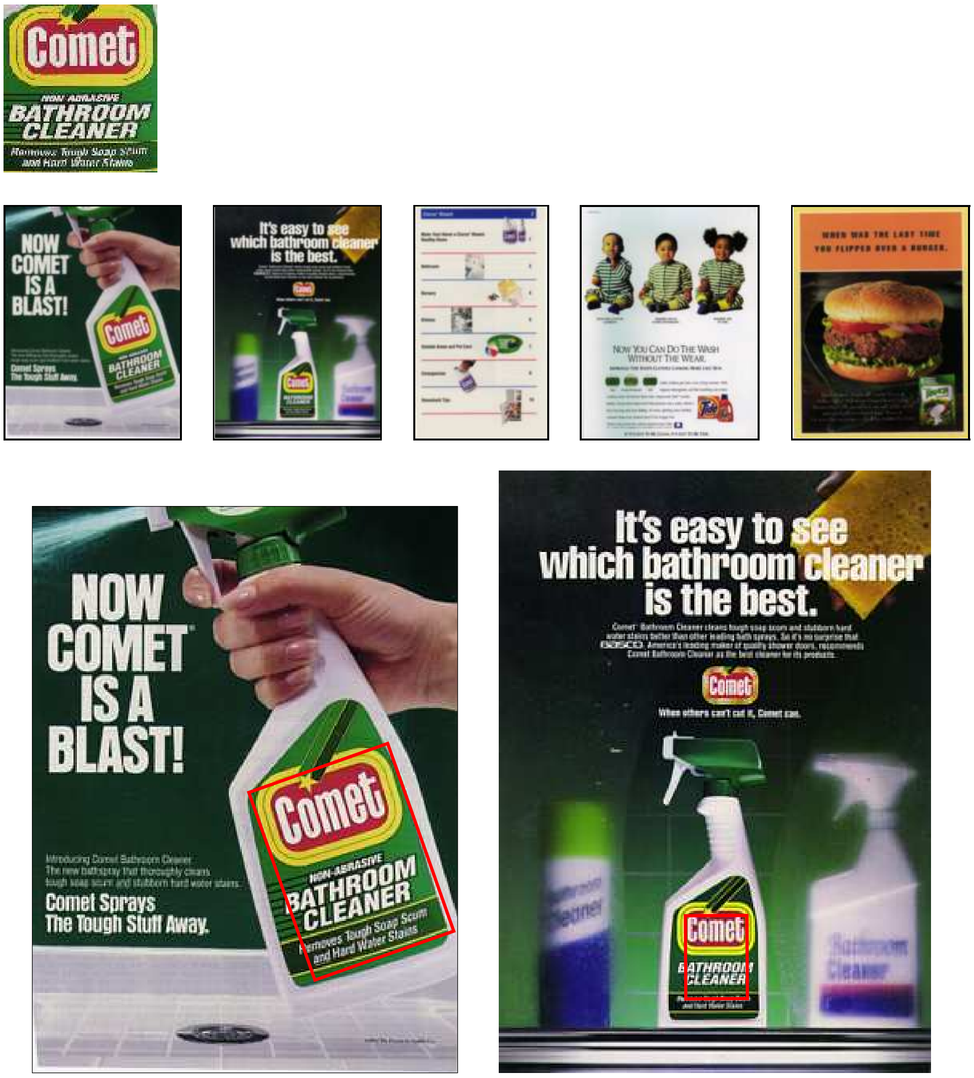


Figure 7.19: Advertisement Query Result – Comet. (top) Comet query logo. (middle) The top five images returned by SEDL. The Comet advertisements are at positions 1 and 2. (bottom-left) The final pattern placement has the correct orientation and is nearly perfect (rank=1st). (bottom-right) The pattern is correctly located, but the final scale should be a little bit larger (rank=2nd).

Figure 7.20: Advertisement Query Result – Fresh Step. (top) Fresh Step query logo. (middle) The top five images returned by SEDL. The Fresh Step advertisements are at positions 1 and 2. (bottom-left) The match is excellent (rank=1st). (bottom-right) The match is good, but the orientation is incorrect (rank=2nd).

Figure 7.21: Advertisement Query Result – Jello. (top) Jello query logo. (middle) The top five images returned by SEDL. The Jello advertisements are at positions 1 and 2. (bottom-left) The match is perfect (rank=1st). (bottom-right) The final placement overlaps a significant amount of the pattern occurrence, but the scale is too small (rank=2nd).

Figure 7.22: Advertisement Query Result – Apple. (top) Apple query logo. (middle) The top five images returned by SEDL. The Apple advertisements are at positions 1 and 4. (bottom-left),(bottom-right) SEDL's algorithm located even these very small scale occurrences of the pattern (ranks=1st,4th).

Figure 7.23: Advertisement Query Result – Reynolds Oven Bags. (top) Reynolds Oven Bags query logo. (middle) The top ten images returned by SEDL. The Reynolds Oven Bags advertisements are at positions 1, 2, 3, 9. (bottom-left) The match is excellent (rank=1st). (bottom-right) The final pattern placement is within the pattern occurrence at a scale which too small (rank=2nd).

Figure 7.24: Advertisement Query Result – Taco Bell. (top) Taco Bell query logo. (middle) The top five images returned by SEDL. The Taco Bell advertisements are at positions 1 and 2. (bottom-left) As in the previous Reynolds Oven Bags query, the pattern is correctly located but with an underestimated scale (rank=1st). (bottom-right) The final pattern placement overlaps a large fraction of the pattern occurence, but the orientation is incorrect (rank=2nd).

Figure 7.25: Verification and Refinement Results – Example Set 1. (top-left) Pert query. The final transformation has the correct scale and translation, but the orientation is slightly off (rank=1st). (top-right) Pert query. The identifed image area has a similar overall color to the Pert logo, but this final pattern placement is incorrect (rank=11th). (bottom-left) Cornpops query. The position and orientation are correct, but the scale is too small (rank=3rd). (bottom-right) Cornpops query. The final transformation is nearly perfect (rank=1st).

Figure 7.26: Verification and Refinement Results – Example Set 2. (top-left) Misty query. The final pattern placement is perfect (rank=1st). (top-right) Misty query. The pattern occurrence is correctly located, but the scale is too small and the orientation does not follow the slope of the box (rank=2nd). (bottom-left) Casting query. The match is nearly perfect (rank=1st). (bottom-right) Casting query. The final scale is too small, but the final placement has the correct 90° orientation (rank=5th).

Figure 7.27: Verification and Refinement Results – Example Set 3. (top-left) Dannon query. The final position is between two occurrences of the Dannon logo (rank=1st). (top-right) Hidden Valley Ranch query. The match is excellent (rank=1st). (bottom-left) Scholl query. This match is also excellent, although the scale should be a little greater and the orientation is slightly off (rank=1st). (bottom-right) Tide query. The final position is between two occurrences of the Tide logo (rank=1st).

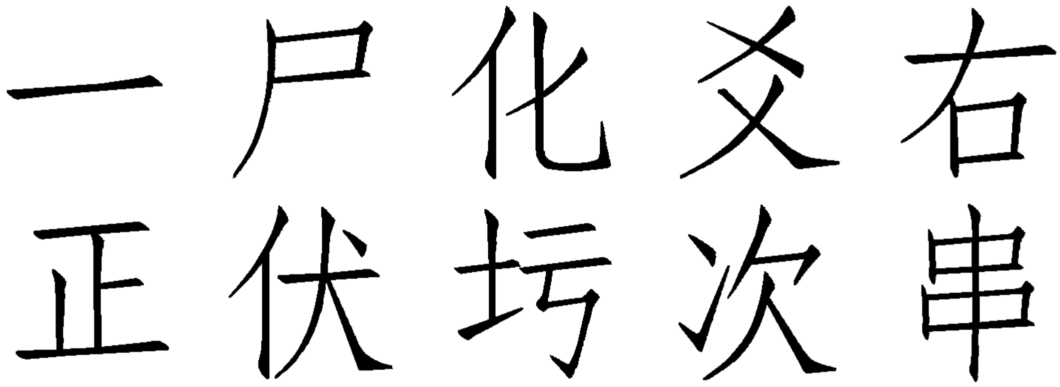


Figure 7.28: Sample Images from the Chinese Character Database. The images are bitmaps.

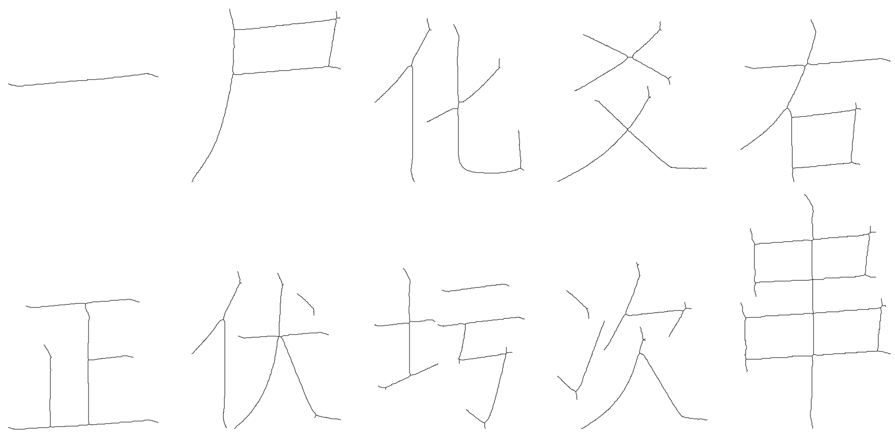


Figure 7.29: Medial Axis Shape Summaries. The shape summary of a Chinese character bitmap is the medial axis of the set of black pixels which define the character. Here we show the summaries for the characters in Figure 7.28.

7.5.2.1 Creating Signatures

The signature creation process operates on the medial axis shape summaries of the characters. The medial axis of a character is represented as a collection of polylines in the image plane. For each polyline, we compute the average position and average orientation of nonoverlapping pieces or samples of a fixed length. The average orientations for an image are clustered to produce a small set of representative orientations $\{ a_1, \ldots, a_n \}$. These orientations are the points in the orientation signature of an image. The weight w_i of each orientation cluster a_i is the total polyline length which is classified as having orientation a_i. The orientation of a polyline sample is the orientation cluster which is closest to the average sample orientation. The weight W_I of (A_I, P_I) = (orientation cluster, average position) in the orientation $\times$ position signature is the sample length, which is the same for every sample except possibly those at the end of a polyline. The orientation signature is the orientation $\times$ position signature marginalized over position.

The same sample length $L = 10$ points is used for every image in the database, where each database image is contained in a square of size 256 points $\times$ 256 points. This results in a relatively dense sampling of the medial axis polylines of a character. Ideally, a database image and a query image are sampled so that the query sample intervals correspond exactly to the sample intervals for the query occurrence within the image. If this were the case, then the average image and query positions would differ by exactly a scaling and translation (assuming that an exact pattern copy with the same orientation appears within the image). Of course this scale and translation is precisely the information that we seek, so it is not possible to obtain the perfect signatures for matching. The average number of orientation clusters is 21.8, while the average number of points in the orientation $\times$ position signature is 119.8.

7.5.2.2 Query Results

The query results shown in this section are the result of matching image and query signatures under the transformation set $\mathcal{G}$ which allows changes in scale and location but *not* in orientation, and ground distance $d = L_{1,\pi}$, the cyclic L_1 distance with period $T = \pi$. We use a cyclic distance so that there is a small distance between an orientation close to 0 and an orientation close to π. The minimum scale parameter is set to 0.25=25% of the total length of the character medial axis. We choose two initial transformations from which to begin the phase 3 iteration that verifies positional consistency and adjusts the scale and location of the match. Finally, we do not allow matches between signels whose orientations differ by more than $\tau = 20°$.

Each of Figures 7.30–7.44 shows the results of a single query into the Chinese character database. The query pattern and its top thirty matches are shown in top row of these figures. In the bottom row, we show where SEDL believes that the query occurs in a selected subset of the matches. Overall, the results of our pattern retrieval algorithm in the Chinese character database are excellent. Almost all the top matches for a query contain the query pattern or a pattern which is very similar to the query pattern. Occurrences of a query are usually well localized, although there are a few examples in which the pattern scale is too large (see e.g. Figure 7.30(iv) and Figure 7.36(viii)) and/or the pattern location is slightly in error (see e.g. Figure 7.32(i) and Figure 7.39(ii)).

There are two points to keep in mind when examining the query results. First, there is no penalty for extra ink on the page that obscures the pattern occurrence; one may have to look closely to see that the pattern is indeed present. This is the case, for example, in Figure 7.30(viii), Figure 7.33(iii), and Figure 7.33(viii). Second, SEDL just matches ink on the page and does not take note of whether or not strokes are connected, or how strokes are connected. Consider the example shown in Figure 7.31(vii) in which the scale of the pattern occurrence is overestimated. This overestimation is understandable because the boxed region contains ink in a very similar layout to that of the query pattern. SEDL does not know that the right part of the query pattern should be one continuous stroke. It only sees that there is ink of the correct orientation in roughly the correct position, and it pays a small price for the incorrect location. A similar example is given in Figure 7.31(vi). The fact that a pattern which is similar to the query pattern appears is merely a coincidence of the juxtaposition of two separate parts of the retrieved character. Neither part alone contains the query. Of course, there are examples in which no matter how hard one looks, nothing similar to the query pattern is present. Examples of such errors are shown in Figure 7.30(vi) and Figure 7.36(vii). Our combination of orientation and position distances produces a small overall distance even though the higher level structure of the pattern is clearly not present in these cases.

The running time for each of the queries in Figures 7.30–7.44 is shown in Figure 7.45. The average time for a single query into our 2000 character database is 95.7 seconds. Thus the average time to compare one (query,image) pair is approximately 0.05 seconds.

7.5.2.3 Possible Modifications

SEDL's signature distance function is asymmetric. The distance is small whenever each piece of all pattern curves is near a similarly-oriented piece of an image curve. There is no penalty for image ink which does not match pattern ink. As we saw in the previous section,

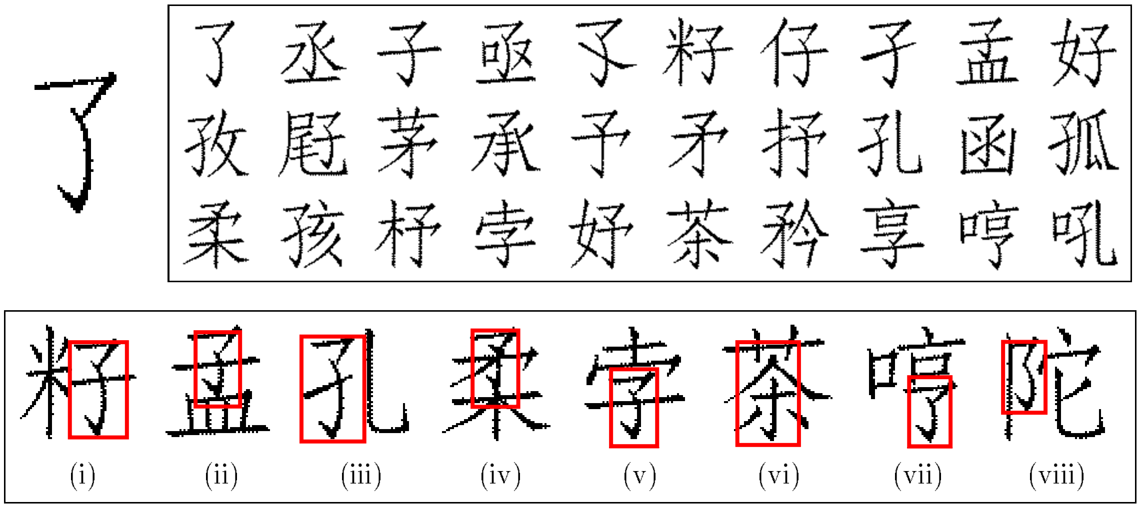


Figure 7.30: Chinese Characters Query Result – Example 1. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 5th, (ii) 9th, (iii) 18th, (iv) 21st, (v) 24th, (vi) 26th, (vii) 29th, (viii) 34th (not shown in (top)).

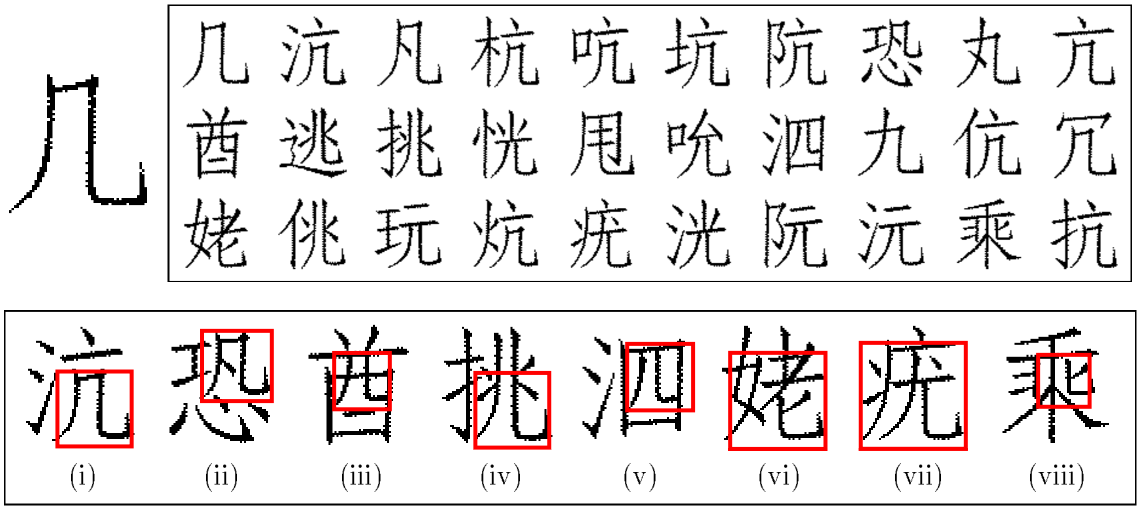


Figure 7.31: Chinese Characters Query Result – Example 2. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 2nd, (ii) 8th, (iii) 11th, (iv) 13th, (v) 17th, (vi) 21st, (vii) 25th, (viii) 29th.

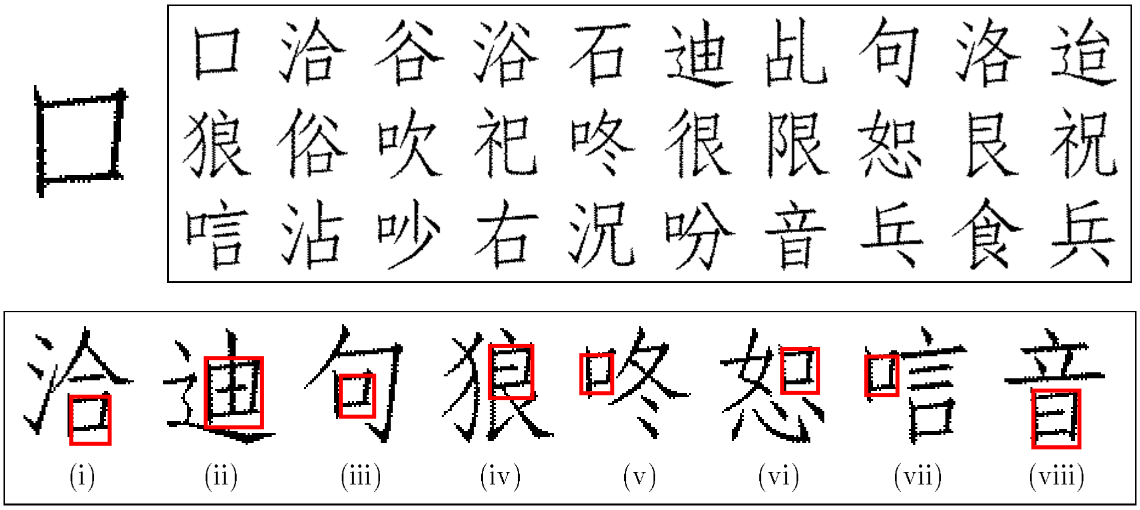


Figure 7.32: Chinese Characters Query Result – Example 3. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 2nd, (ii) 6th, (iii) 8th, (iv) 11th, (v) 15th, (vi) 18th, (vii) 21st, (viii) 27th. The scale is slightly overestimated in (ii). In this case, the query pattern appears five times (the large square and the four smaller squares contained inside the big one) and SEDL finds the largest occurrence.

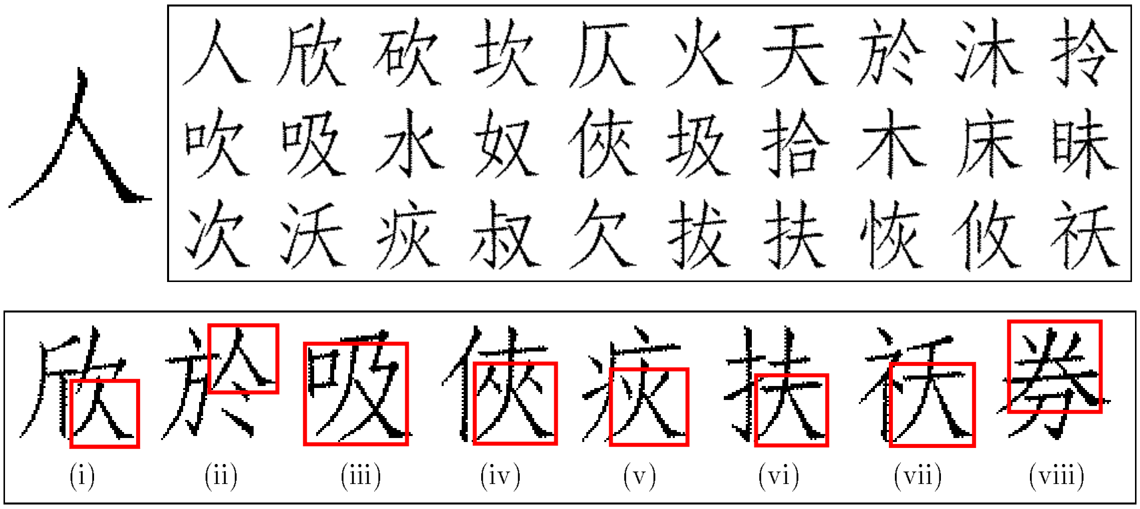


Figure 7.33: Chinese Characters Query Result – Example 4. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 2nd, (ii) 8th, (iii) 12th, (iv) 15th, (v) 23rd, (vi) 27th, (vii) 30th, (viii) 37th (not shown in (top)).

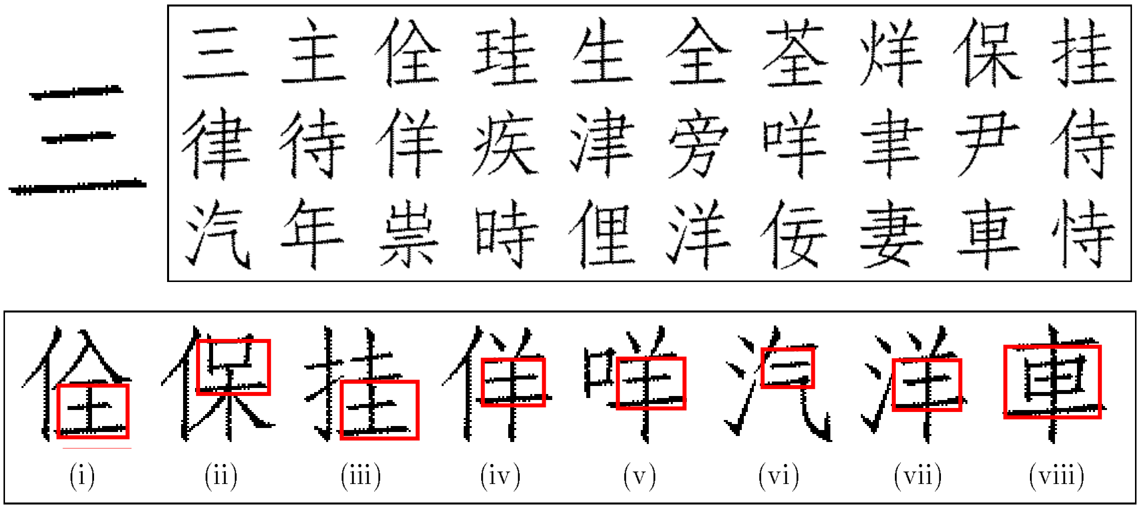


Figure 7.34: Chinese Characters Query Result – Example 5. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 3rd, (ii) 9th, (iii) 10th, (iv) 13th, (v) 17th, (vi) 21st, (vii) 26th, (viii) 29th.

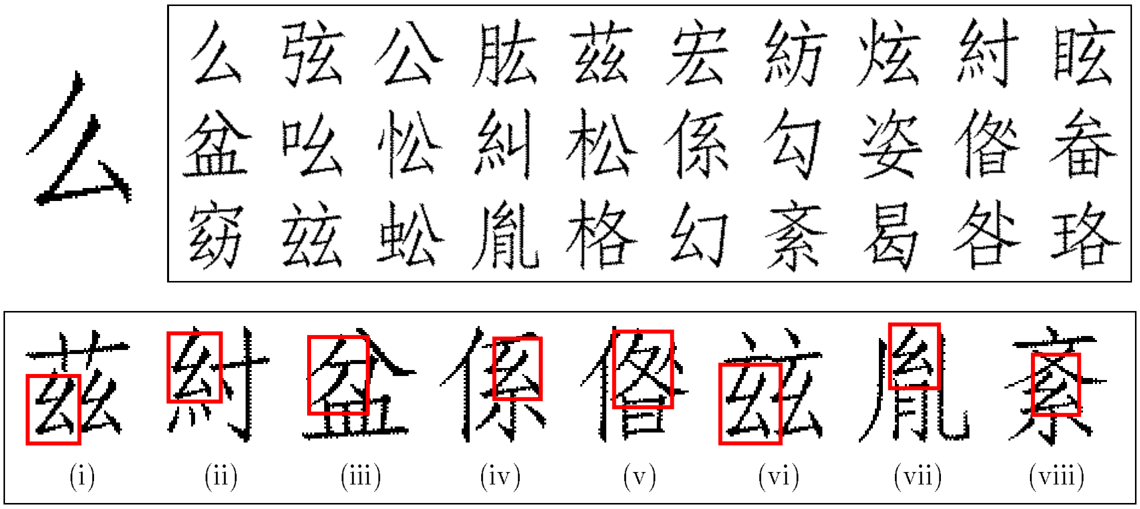


Figure 7.35: Chinese Characters Query Result – Example 6. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 5th, (ii) 9th, (iii) 11th, (iv) 16th, (v) 19th, (vi) 22nd, (vii) 24th, (viii) 27th. A pattern similar to the query occurs in (iii) and (v) because of the juxtaposition of two separate parts of the characters. In both cases, the surrounding ink obscures the pattern appearance.

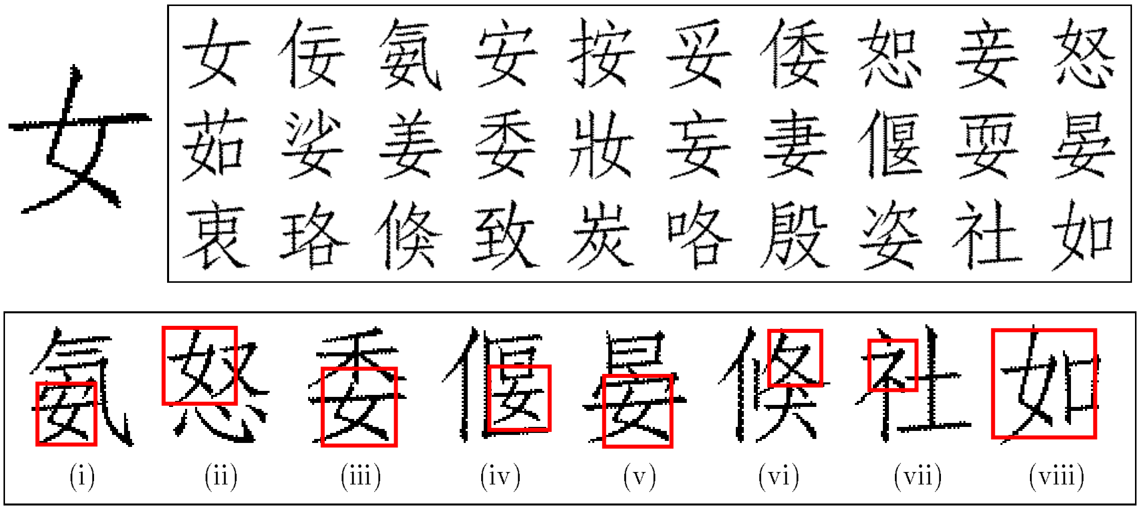


Figure 7.36: Chinese Characters Query Result – Example 7. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 3rd, (ii) 10th, (iii) 14th, (iv) 18th, (v) 20th, (vi) 23rd, (vii) 29th, (viii) 30th.

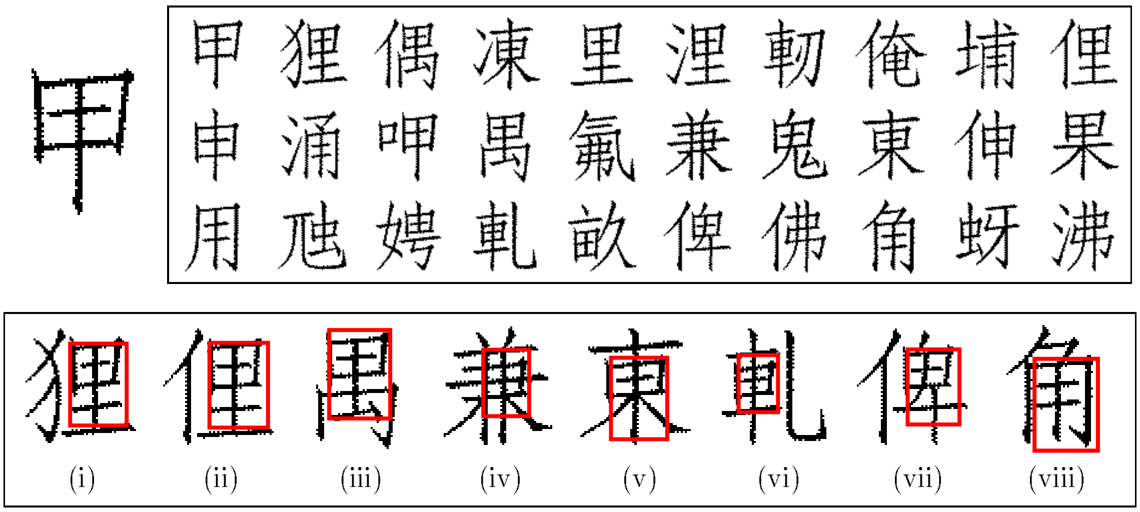


Figure 7.37: Chinese Characters Query Result – Example 8. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 2nd, (ii) 10th, (iii) 14th, (iv) 16th, (v) 18th, (vi) 24th, (vii) 26th, (viii) 28th. It is difficult to see the pattern occurrence in (iv) and (viii) because of its surroundings.

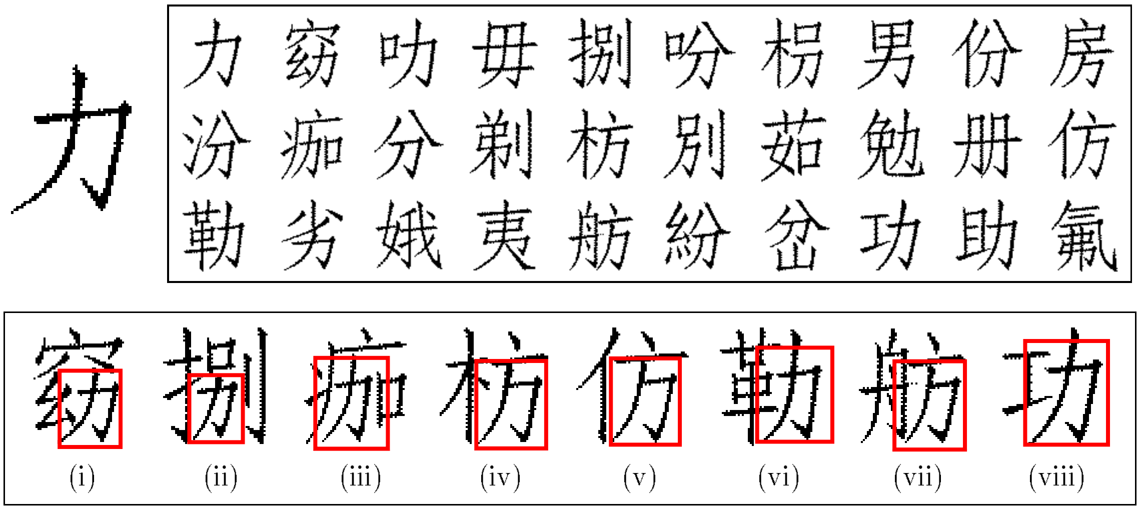


Figure 7.38: Chinese Characters Query Result – Example 9. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 2nd, (ii) 5th, (iii) 12th, (iv) 15th, (v) 20th, (vi) 21st, (vii) 25th, (viii) 28th.

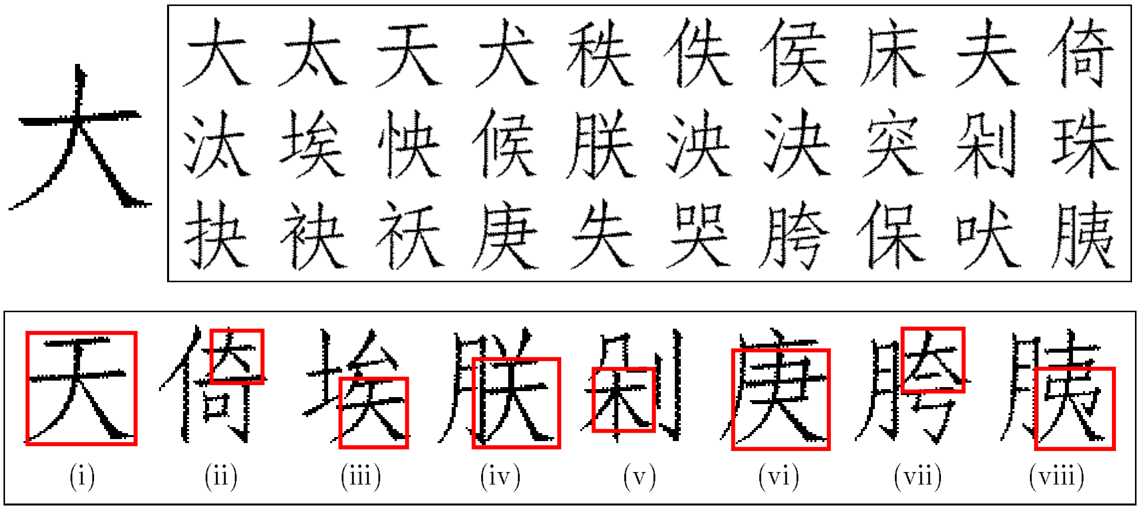


Figure 7.39: Chinese Characters Query Result – Example 10. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 3rd, (ii) 10th, (iii) 12th, (iv) 15th, (v) 19th, (vi) 24th, (vii) 27th, (viii) 30th.

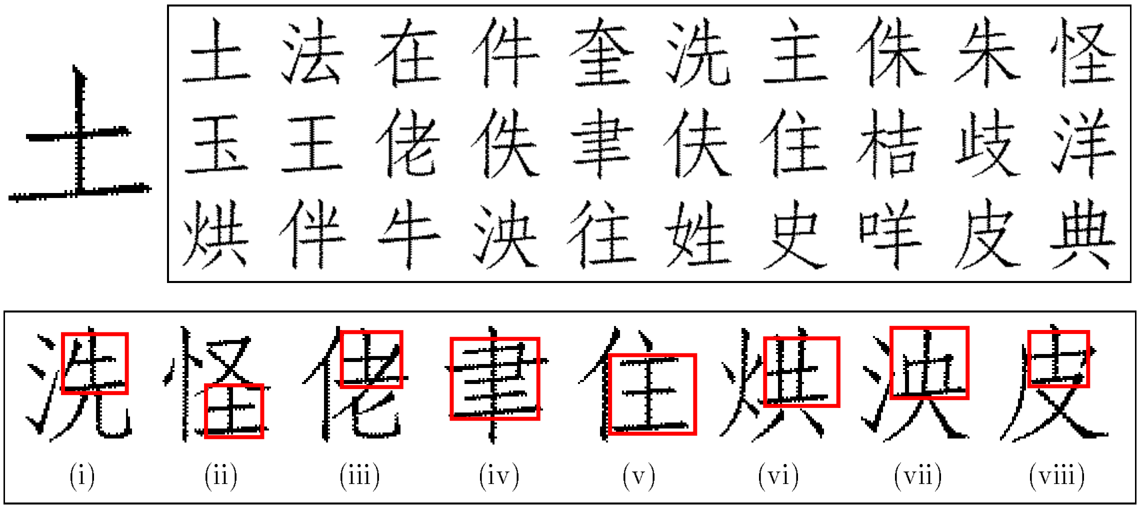


Figure 7.40: Chinese Characters Query Result – Example 11. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 6th, (ii) 10th, (iii) 13th, (iv) 15th, (v) 17th, (vi) 21st, (vii) 24th, (viii) 29th. The extra horizontal lines in (iv) make it difficult to see the pattern occurrence.

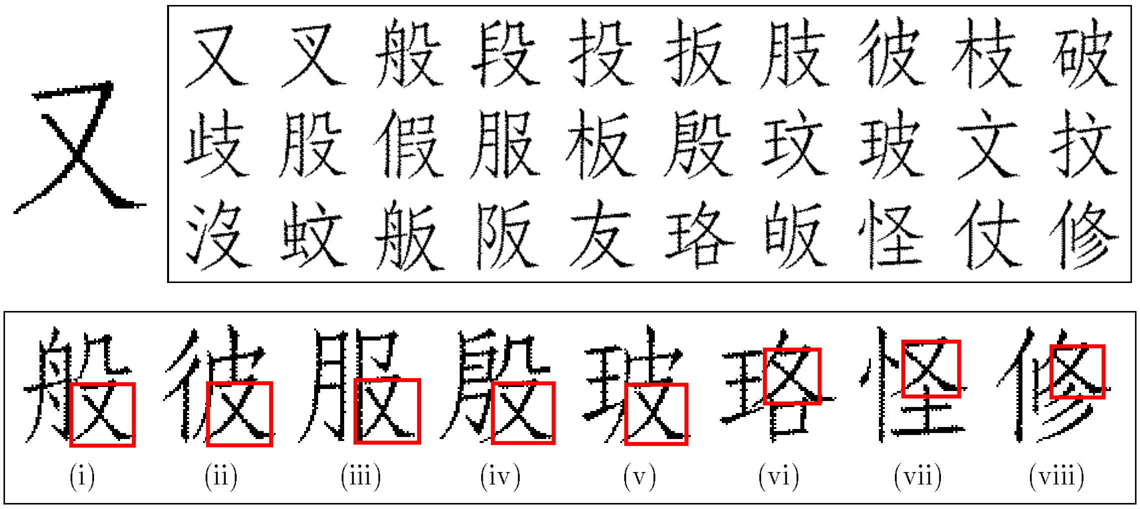


Figure 7.41: Chinese Characters Query Result – Example 12. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 3rd, (ii) 8th, (iii) 14th, (iv) 16th, (v) 18th, (vi) 26th, (vii) 28th, (viii) 30th.

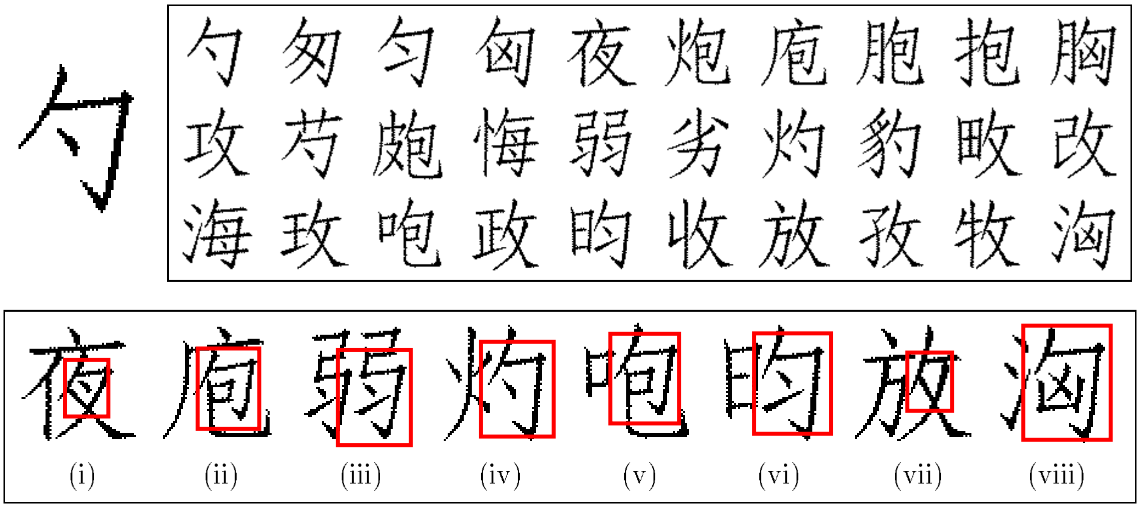


Figure 7.42: Chinese Characters Query Result – Example 13. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 5th, (ii) 7th, (iii) 15th, (iv) 17th, (v) 23rd, (vi) 25th, (vii) 27th, (viii) 30th. Most of these examples show inexact matches of the query and a region of a database character. The differences in appearance are relatively small except for example (vii).

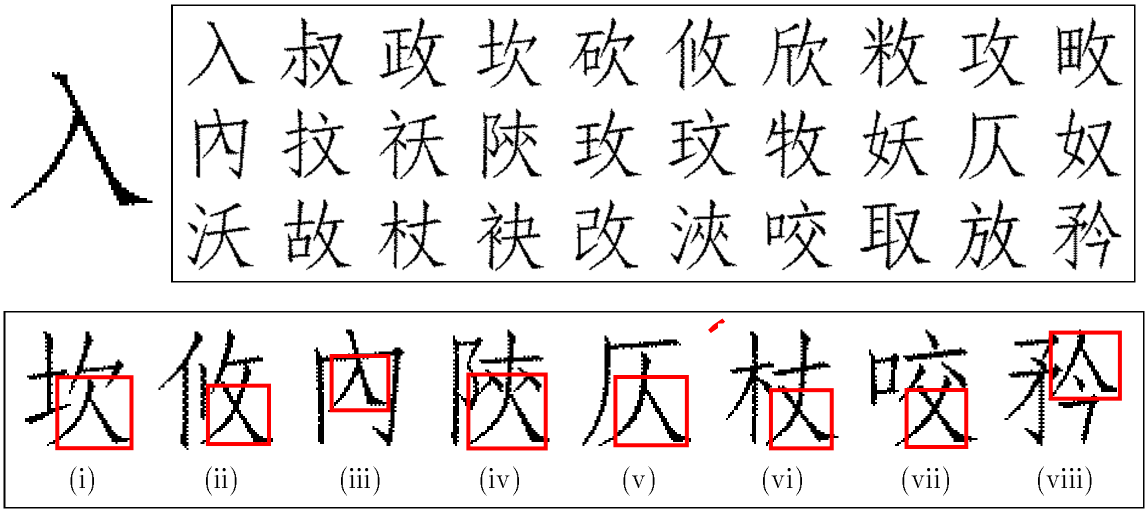


Figure 7.43: Chinese Characters Query Result – Example 14. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 4th, (ii) 6th, (iii) 11th, (iv) 14th, (v) 19th, (vi) 23rd, (vii) 27th, (viii) 30th.

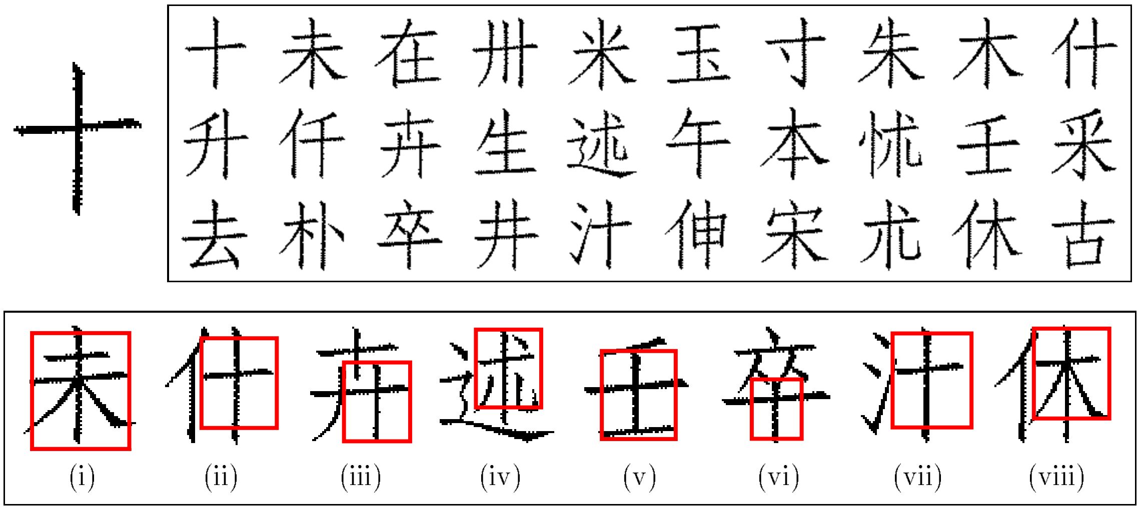


Figure 7.44: Chinese Characters Query Result – Example 15. (top) the query pattern and the top thirty matches. (bottom) some selected query localization results. The ranks of these selected retrievals in (top) are: (i) 2nd, (ii) 10th, (iii) 13th, (iv) 15th, (v) 19th, (vi) 23rd, (vii) 25th, (viii) 29th.

such "extra" ink in the vicinity of the pattern occurrence can obscure the visual similarity between that area of the image and the pattern. The local image signature computed during the final τ-EMD check of the verification and refinement phase can be used to solve this extra ink problem if it is indeed considered a problem for a particular application. A penalty can be added to the signature distance which is proportional to the difference between the total amount of image ink inside the final search rectangle and the amount of ink in the scaled pattern. An implementation of this idea should err on the side of smaller rather than larger penalties since the computed pattern scale and location will not be perfect.

A big difference between the shape and color cases is that, as currently implemented, SEDL does not find rotated versions of a given shape pattern within an image. Instead, it uses curve orientations to direct or guide its search for a possibly scaled version of the given pattern. A possible solution to find rotated shape patterns within the SEDL framework is to use relative orientations instead of absolute orientations ([33]). For example, we might compute the attribute portion of a signel by taking the difference in tangent orientations at the endpoints of a short curve piece instead of the average orientation over the curve piece. SEDL's search for a possibly rotated, scaled version of a given shape pattern can be directed by the relative orientation attribute since relative orientations will not change when a similarity transformation is applied to the pattern curves.

Query	Time (secs) Phase 1	Phase 2	Phase 3	Total
了	39.7	7.6	37.2	84.5
几	40.4	9.8	77.5	127.7
口	16.2	13.9	86.2	116.3
人	40.0	8.4	30.3	78.7
三	14.1	12.6	33.2	59.9
么	36.4	9.4	37.8	83.6
女	46.6	8.9	72.4	127.9
甲	26.1	14.4	103.4	143.9
力	40.4	9.6	75.0	125.0
大	46.5	8.4	62.7	117.6
土	24.3	11.6	50.7	86.6
又	45.3	8.9	54.1	108.3
勺	41.3	9.0	60.5	110.8
入	32.8	8.7	30.5	72.0
十	22.7	7.9	31.7	62.4
Average				95.7

Figure 7.45: Query Times for the Chinese Character Database.

Chapter 8

Conclusion

Image matching for content-based image retrieval (CBIR) applications is fundamentally different from image matching in the more traditional computer vision areas of stereopsis and tracking. In stereo, most of the information in one image has matching information in the other image (missing information results when part of the scene is visible from one viewpoint but not the other). The search for a correspondence to a feature in one image can be limited to the associated epipolar line in the other image. Also, the lighting for the two images is usually the same. In tracking, consecutive frames are nearly identical. Estimates of feature velocity can be used to predict where in the next frame a feature will appear, thus limiting the search space in finding correspondences. As in stereo, the lighting is usually constant during tracking. In contrast, the images we desire to match in CBIR can be visually quite different because most of the information in one image may not have any matching information in the other; images are not usually of the same scene. Also, a region in one image might match any region in another. The matching process may be further complicated by differences in illumination that cause the same object to appear differently in two images.

It is now time to take a step back and see what we have done, how it fits into the broader picture of image retrieval and image comparisons, and what remains to be done for the difficult image matching problems in CBIR.

8.1 Thesis Summary and Discussion

The pattern problem is difficult because partial matching and transformations are allowed. The scale component of a transformation plays a critical role here because it determines how much information in the image to compare to the pattern. A system might incorrectly conclude that a pattern is not present at some image location if the system's scale estimate is very inaccurate. We believe that a good estimate for pattern scale is essential for the efficiency and correctness of a pattern problem solution, and we developed a novel scale estimation algorithm that uses the Earth Mover's Distance (EMD) between two attribute distributions.

The image and pattern signatures used throughout this thesis are distributions of mass in some feature space, where the amount of mass placed at a feature space point is the amount of that feature present in the image. We used a combined color-position feature space for the color pattern problem, and a combined orientation-position feature space for the shape pattern problem. In our scale estimation algorithm, we used a color feature space and an orientation feature space for the color and shape cases, respectively. These choices reflect our general strategy to obtain fast scale estimates by matching attribute-position distributions after marginalizing away position information.

The EMD is a general tool to measure the distance between distributions. Because we believe that mass distributions in a feature space are excellent image signatures for the pattern problem, we devoted a large part of this dissertation to the EMD, including modifications to aid in partial matching, lower bounds to aid in CBIR query efficiency, and computation under transformation sets. We made the EMD more amenable to partial matching with changes that (i) force only part of the mass in both distributions to be matched (the partial EMD), and (ii) measure the amount of mass that cannot be matched if there is a limit on the distance in feature space between allowable matches (the τ-EMD). We extended the centroid lower bound to the partial matching case, and we developed projection-based lower bounds which are also applicable to partial matching. Finally, we made an extensive study of computing the EMD under transformation sets, including theoretical analysis, documentation of the difficulty of the problem, the FT iteration to compute at least a locally optimal transformation (for a large class of transformation sets), cases with special structure that allow us to compute directly a globally optimal transformation, and the previously mentioned scale estimation algorithm that finds the scale of one distribution that minimizes its EMD to another.

The use of an iteration to compute the EMD under a transformation set reflects our decision on a major choice in the design of a pattern problem algorithm. Motivated by the importance of the pattern problem in CBIR, we believe that it is more important to solve many pattern problems quickly with the chance of a small number of errors than to solve every pattern problem correctly but more slowly. Although the FT iteration is not guaranteed to find a globally optimal transformation, its chances are greatly improved if the initial transformation is close to optimal. The combination of SEDL's scale estimation and initial placement methods is an efficient, very effective algorithm for computing a small set of promising pattern regions within an image.

SEDL uses a simpler distance function than the EMD for efficiency reasons. This distance still allows partial matching under a transformation set, but it gives up the notion of morphing one distribution into a subset of the other in order to handle large image and pattern signatures. Our ideal distance measure involves this morphing notion, but also requires a change in the distribution representation. Currently, each mass in feature space is placed at a single point in that space. A region which is mainly blue, for example, is summarized by a mass at the point (blue, region centroid) in color $\times$ position space. Our ideal representation places masses over continuous regions in the feature space. Instead of summarizing a blue segmentation region as above, we could uniformly distribute mass over the entire extent of the region in position space. This is a more faithful representation of the blue region than mass concentrated at a single (color, position) point.

Given image signatures which are continuous mass distributions in feature space, we need a continuous version of the EMD to compute morphing distances. In section 2.4, we mentioned work in computing the EMD between two normal distributions and between two uniform distributions over elliptical volumes. We are, however, unaware of more general work on the continuous EMD which is capable of matching distributions such as those we have just proposed.

In an effort to understand why the continuous formulation is better than the discrete one, let us consider matching two distributions of color clusters with the EMD. In one image, a clustering algorithm might produce two clusters of somewhat similar reds, while in another image the matching red may be represented as a single cluster which is roughly the average red color of the corresponding two clusters. The EMD is not sensitive to such a non-canonicality of its input. In this example, it only pays a small cost to match the red mass because all three clusters are located close to one another in color space. Of course, the EMD would pay zero cost if both representations had one cluster with identical reds. As long as the representation is accurate, it affects the efficiency of the EMD computation but not the correctness of the result.

Using continuous mass distributions in attribute $\times$ position space with a continuous EMD aims for the same effect as above. Suppose, for example, that there are two matching green areas from two different images. The effect of representing the area as one region of green in the first image and two adjacent regions of similar greens in the other will be negligible under the continuous EMD. This effect may not be negligible if mass is placed only at region centroids because the centroid of the green region in the first image may be far away from the centroids of the green regions in the other image if the regions are large. To help reduce the impact of such problems, SEDL weighs region distances by pattern region area, the theory being that large pattern regions are more stable in appearance and easier to detect as a single region within the image. The excellent results obtained show that this strategy is effective, but it is still desirable to have a distance function which is provably robust to non-canonicalities of the representation, yet fast enough to maintain reasonable interactivity with the user. In the shape case, SEDL avoids such representation problems to a large extent by using a fine sampling of image curves. This strategy is feasible because the shape data is one-dimensional, but a fine sampling over the 2D image plane, however, would produce color signatures too large to match in acceptable times for retrieval.

An extreme example in the color case is comparing (for the same image) a segmentation in which every pixel is a region and one which aims for the largest possible "uniform" color regions. The EMD between continuous distributions in color $\times$ position space derived from these segmentations should be small. With a continuous formulation, the segmentation used to produce a color $\times$ position signature is an efficiency issue, but not a correctness issue. It may be possible to match two continuous distributions of masses more quickly if there are fewer masses, but the EMD should be roughly the same if one distribution is replaced by another whose masses cover the same areas of the feature space. Computing the EMD between continuous distributions, or a good approximation if the exact answer is too expensive to compute, is a difficult but worthwhile future research problem.

8.2 Future Work

Developing partial matching distance measures that allow for scale-altering transformations is a crucial problem in CBIR because semantic image similarity often follows from only a partial match of unknown size. The road toward practically useful CBIR systems, however, still has a number of interesting and difficult challenges ahead.

Speed and Scalability to Large Databases

Users will demand semantic retrieval ability, but will not wait more than a few seconds for query results. Simple matching schemes on global image statistics will be fast, but are unlikely to return images which are semantically related to a given query. Speed issues must be addressed, but correctness issues are more important since there is no point in computing undesirable answers rapidly. Perhaps a breakthough will come when algorithms find complicated patterns as quickly as they find simple ones ([78]). This is not the case in SEDL, but intuitively a complicated pattern is more distinctive than a simpler one, and this distinctiveness should make it easier to find the pattern or discover that it is not present. It is not straightforward to avoid explicit query-image comparisons via clustering database images (and comparing a query to cluster representatives) because partial match distance measures do not obey the triangle inequality and may be asymmetric.

Object Recognition in Cluttered Scenes

The experiments discussed in this thesis allow for changes in either camera pose or lighting, but not both at the same time. For example, our object recognition experiments use images of an object under different illuminants but taken with the same camera pose. In the color pattern retrieval experiments, we allow certain changes in object pose but do not account for changes in lighting. SEDL's pattern search is directed by the colors of regions, so big differences in lighting for the database and query images would cause a problem. A completely general pattern problem algorithm would allow for both photometric and geometric factors. In fact, some viewpoint differences may mean that different parts of an object may be visible in the query and database images. Thus, a pattern problem algorithm may also have to allow for partial matching of the query pattern. Allowing all these factors at once makes it difficult to rule out a pattern occurrence at a particular location. Maybe the pattern is present but the system's scale estimate is inaccurate, or all the information in the query should not be matched, or a color correction must be made to account for lighting.

Combination of Different Search Modalities

Another important issue in pattern problem algorithms is the use of more than one modality in judging visual similarity, for example region shapes and colors. Although SEDL is a uniform framework for the color and shape pattern problems, it does not use both color and shape information together. A good pattern problem algorithm that uses both color and shape should know when to use which information. Consider, for example, searching for a grayscale Apple logo within a color advertisement. The outline of the apple will match, but the colors will not. The problem of combining different types of information to measure perceptual similarity is a difficult one. For example, how much do the shapes of regions contribute to the perceptual similarity or dissimilarity of two color patterns?

Shape Representations and Distance Measures

We used the word *shape* in this thesis to mean a salient image curve or region boundary, and we considered the problem of measuring distances between individual shapes and sets of shapes. In the former problem, we represented shapes by their arclength versus turning angle curves and measured distance as sums of orientation differences between correponding points along two curves. Of course, there are many other possible representations and distance measures, including comparison of coefficients in a Fourier decomposition, control points in a B-spline representation, moments in an area-based representation, or the amount of energy to deform one shape into another, just to name a few. We represented a set of curves as a set of (curve point, orientation) pairs obtained by a dense sampling of the curves. As in the single shape case, the distance between two sets of shapes was measured as a sum of orientation differences between corresponding points.

Our choices of representation and distance measures were motivated by the need to handle partial matching and transformations, but there may be better representations and distances for CBIR. The best choice may depend upon the application domain; nature images, product advertisements, Chinese characters, textbook drawings, and CAD models, for example, might all require different representations and distances. Perhaps the continuous EMD can be used as a common distance measure for different representations of both single shapes and sets of shapes, where mass is spread uniformly along curves or over regions. A B-spline representation, for example, describes the distribution of mass as a collection of uniform distributions over Bézier arcs. Is the morphing distance provided by the continuous EMD a perceptual one? For comparisons of one shape to another, maybe using the EMD to compare distributions of energy in frequency space will yield a perceptual measure of shape similarity, as it does in the texture case ([69, 68, 66]).

Image Browsing

We did not touch on the subject of navigating in the space of database images, but this is also an important mode of user interaction with a system. A user may not know exactly what he/she wants, but instead would like to browse the images. Navigation according to global color similarity is accomplished quite effectively by embedding images in a low (two or three) dimensional space such that distances in this space approximate well Earth Mover's Distances between image color signatures ([67, 69, 65]). How can this be done when there is no inherent metric structure imposed by the image distance function as for partial match distance measures, and when there are many dimensions along which images differ. Perhaps a solution to the general image matching problem of finding all pairs of similar regions from two images can help. Imagine a graph structure on database images where there are many links connecting two images, one for each pair of similar regions. These links may help navigate locally, but how can we give a user the big picture of database contents when the underlying image space has high dimension?

The Image Matching Problem

Solutions to the more general image matching problem will take us further toward semantic image retrieval than solutions to the pattern problem. As mentioned in the introduction, an algorithm for the pattern problem may be useful as a subroutine within an algorithm for the image matching problem. But a solution to the image matching problem is far from the end of the story. It is also a major challenge to interpret the results to provide semantic retrieval. This involves looking at positions of similar regions within the database and query images. For example, blue above green in a nature image likely indicates a scene of grass against sky, while blue below green probably means a scene of grass leading to water. Perhaps artificial intelligence knowledge representations and learning algorithms will prove useful here.

8.3 Final Thoughts

In the midst of all the technical details and interesting problems that arise in content-based image retrieval, we should not forget that the ultimate goal is to allow users to find information reliably and efficiently. Image-based and text-based search should complement one another to advance toward this goal. At the present, there is little collaboration between the traditional, text-oriented database community and the image-oriented computer vision and image understanding communities, but this will need to change if we are to produce the best possible information retrieval systems.

Bibliography

[1] Helmut Alt, Bernd Behrends, and Johannes Blömer. Approximate matching of polygonal shapes. In *Proceedings of the Seventh Annual ACM Symposium on Computational Geometry*, pages 186–193, 1991.

[2] Helmut Alt and Michael Godau. Measuring the resemblance of polygonal curves. In *Proceedings of the Eighth Annual Symposium on Computational Geometry*, pages 102–109, 1992.

[3] Esther M. Arkin, L. Paul Chew, Daniel P. Huttenlocher, Klara Kedem, and Joseph S. B. Mitchell. An efficiently computable metric for comparing polygonal shapes. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 129–137, 1990.

[4] Marshall Bern, David Eppstein, Leonidas Guibas, John Hershberger, Subhash Suri, and Jan Wolter. The centroid of points with approximate weights. In *Proceedings of Third Annual European Symposium on Algorithms*, pages 460–472, 1995.

[5] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February 1992.

[6] W. S. Chan and F. Chin. Approximation of polygonal curves with minimum number of line segments or minimum errors. *International Journal of Computational Geometry & Applications*, 6(1):59–77, March 1996.

[7] R. Chandrasekaran and A. Tamir. Open questions concerning Weiszfeld's algorithm for the Fermat-Weber location problem. *Mathematical Programming, Series A*, 44(3):293–295, November 1989.

[8] L. Paul Chew, Dorit Dor, Alon Efrat, and Klara Kedem. Geometric pattern matching in d-dimensional space. In *Proceedings of Third Annual European Symposium on Algorithms*, pages 264–279, 1995.

[9] L. Paul Chew, Michael T. Goodrich, Daniel P. Huttenlocher, Klara Kedem, Jon M. Kleinberg, and Dina Kravets. Geometric pattern matching under euclidean motion. In *Proceedings of the Fifth Canadian Conference on Computational Geometry*, pages 151–156, 1993.

[10] Fernand S. Cohen, Zhaohui Huang, and Zhengwei Yang. Invariant matching and identification of curves using B-splines curve representation. *IEEE Transactions On Image Processing*, 4(1):1–10, January 1995.

[11] Scott D. Cohen and Leonidas J. Guibas. Shape-based illustration indexing and retrieval - some first steps. In *Proceedings of the ARPA Image Understanding Workshop*, pages 1209–1212, February 1996.

[12] Scott D. Cohen and Leonidas J. Guibas. The earth mover's distance: Lower bounds and invariance under translation. Technical Report STAN-CS-TR-97-1597, Stanford University, October 1997. Currently available online at `http://elib.stanford.edu/`.

[13] Scott D. Cohen and Leonidas J. Guibas. Partial matching of planar polylines under similarity transformations. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 777–786, January 1997.

[14] Madirakshi Das, Edward M. Riseman, and Bruce A. Draper. FOCUS: Searching for multi-colored objects in a diverse image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 756–761, June 1997.

[15] Mark de Berg, Marc van Kreveld, Mark Overmars, and Otried Schwarzkopf. *Computational Geometry: Algorithms and Applications*, chapter 5, pages 93–117. Springer-Verlag, 1997.

[16] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

[17] D. C. Dowson and B. V. Landau. The Fréchet distance between multivariate normal distributions. *Journal of Multivariate Analysis*, 12(3):450–455, 1982.

[18] Zvi Drezner. A note on the Weber location problem. *Annals of Operations Research*, 40(1–4):153–161, 1992.

[19] Herbert Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, 1987.

[20] Herbert Edelsbrunner. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Transactions On Graphics*, 9(1):66–104, January 1990.

[21] Herbert Edelsbrunner and Leonidas J. Guibas. Topologically sweeping an arrangement. *Journal of Computer and System Sciences*, 38(1):165–194, February 1989.

[22] François Ennesser and Gérard Medioni. Finding Waldo, or focus of attention using local color information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):805–809, August 1995.

[23] A. Etemadi. Robust segmentation of edge data. In *International Conference on Image Processing and its Applications*, pages 311–314, April 1992.

[24] David Eu and Godfried T. Toussaint. On approximating polygonal curves in two and three dimensions. *CVGIP: Graphical Models and Image Processing*, 56:231–246, May 1994.

[25] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1989.

[26] Yihong Gong, Guido Proietti, and Christos Faloutsos. Image indexing and retrieval based on human perceptual color clustering. In *Proceedings of Computer Vision and Pattern Recognition*, pages 578–583, June 1998.

[27] Glenn Healey and Amit Jain. Retrieving multispectral satellite images using physics-based invariant representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):842–848, August 1996.

[28] Glenn Healey and David Slater. Global color constancy: recognition of objects by use of illumination-invariant properties of color distributions. *Journal of the Optical Society of America A*, 11(11):3003–3010, November 1994.

[29] Paul S. Heckbert. A seed fill algorithm. In Andrew S. Glassner, editor, *Graphics Gems*, pages 275–277,721–722. Academic Press, Inc., 1990.

[30] D. R. Heisterkamp and P. Bhattacharya. Matching of 3d polygonal arcs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):68–73, January 1997.

[31] D. R. Heisterkamp and P. Bhattacharya. Matching 2d polygonal arcs by using a subgroup of the unit quaternions. *Computer Vision and Image Understanding*, 69(2):246–249, February 1998.

[32] Frederick S. Hillier and Gerald J. Lieberman. *Introduction to Mathematical Programming*, pages 202–229. McGraw-Hill, 1990.

[33] David Hoffman. Personal communication, 1999.

[34] Jiawei Hong and Xiaonan Tan. Recognize the similarity between shapes under affine transformation. In *Second International Conference on Computer Vision*, pages 489–493, 1988.

[35] Daniel P. Huttenlocher, Klara Kedem, and Micha Sharir. The upper envelope of Voronoi surfaces and its applications. *Discrete and Computational Geometry*, 9(3):267–291, 1993.

[36] Daniel P. Huttenlocher, Gregory A. Klanderman, and William J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.

[37] Hiroshi Imai and Masao Iri. Polygonal approximations of a curve – formulations and algorithms. In Godfried T. Toussaint, editor, *Computational morphology : A computational geometric approach to the analysis of form*, pages 71–86. Elsevier Science Publishers, 1988.

[38] Richard Johnsonbaugh and W. E. Pfaffenberger. *Foundations of Mathematical Analysis*. Marcel Dekker, Inc., 1981.

[39] Alan Kalvin, Edith Schonberg, Jacob T. Schwartz, and Micha Sharir. Two-dimensional model-based, boundary matching using footprints. *The International Journal of Robotics Research*, 5(4):38–55, Winter 1986.

[40] Behzad Kamgar-Parsi, Avraham Margalit, and Azriel Rosenfield. Matching general polygonal arcs. *CVGIP: Image Understanding*, 53(2):227–234, March 1991.

[41] I. Norman Katz. Local convergence in Fermat's problem. *Mathematical Programming*, 6(1):89–104, February 1974.

[42] Harold W. Kuhn. A note on Fermat's problem. *Mathematical Programming*, 4:98–107, 1973.

[43] Yehezkel Lamdan, Jacob T. Schwartz, and Haim J. Wolfson. Object recognition by affine invariant matching. In *Proceedings of Computer Vision and Pattern Recognition*, pages 335–344, 1988.

[44] Yehezkel Lamdan, Jacob T. Schwartz, and Haim J. Wolfson. On recognition of 3-d objects from 2-d images. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 1407–1413, 1988.

[45] D.G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

[46] W. Y. Ma and B. S. Manjunath. NETRA: A toolbox for navigating large image databases. In *Proceedings of the IEEE International Conference on Image Processing*, volume 1, pages 568–571, 1997. Currently available online at `http://maya.ece.ucsb.edu/Netra/`.

[47] Geoffrey J. McLachlan and Kaye E. Basford. *Mixture Models : Inference and Applications to Clustering*. Marcel Dekker, 1989.

[48] Geoffrey J. McLachlan and Thriyambakam Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., 1997.

[49] Avraham Melkman and Joseph O'Rourke. On polygonal chain approximation. In Godfried T. Toussaint, editor, *Computational morphology : A computational geometric approach to the analysis of form*, pages 87–95. Elsevier Science Publishers, 1988.

[50] Gaspard Monge. Mémoire sur la théorie des déblais et des remblais, 1781.

[51] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The QBIC project: querying images by content using color, texture, and shape. In *Proceedings of the SPIE*, volume 1908, pages 173–187, 1993.

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

[53] I. Olkin and F. Pukelsheim. The distance between two random vectors with given dispersion matrices. *Linear Algebra and its Applications*, 48:257–263, 1982.

[54] S. K. Parui, S. Eswara Sarma, and D. Dutta Majumder. How to discriminate shapes using the shape vector. *Pattern Recognition Letters*, 4(3):201–204, July 1986.

[55] E. J. Pauwels, T. Moons, L. J. Van Gool, and A. Oosterlinck. Recognition of planar shapes under affine distortion. *International Journal of Computer Vision*, 14(1):49–65, January 1995.

[56] T. Pavlidis. *Structural Pattern Recognition.* Springer-Verlag, 1977.

[57] Shmuel Peleg, Michael Werman, and Hillel Rom. A unified approach to change of resolution: Space and gray-level. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):739–742, July 1989.

[58] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C.* Cambridge University Press, second edition, 1992.

[59] Svetlozar T. Rachev. The Monge-Kantorovich mass transference problem and its stochastic applications. *Theory of Probability and Its Applications*, 29(4):647–676, 1984.

[60] Svetlozar T. Rachev and Ludger Rüschendorf. *Mass Transportation Problems*, volume I: Theory. Springer, 1998.

[61] Svetlozar T. Rachev and Ludger Rüschendorf. *Mass Transportation Problems*, volume II: Applications. Springer, 1998.

[62] Paul L. Rosin and Geoff A.W. West. Nonparametric segmentation of curves into various representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1140–1153, December 1995.

[63] Günter Rote. Computing the minimum Hausdorff distance between two point sets on a line under translation. *Information Processing Letters*, 38(3):123–127, 1991.

[64] Yossi Rubner. *Perceptual Metrics for Image Database Navigation.* PhD thesis, Stanford University, 1999.

[65] Yossi Rubner, Leonidas J. Guibas, and Carlo Tomasi. The earth mover's distance, multi-dimensional scaling, and color-based image retrieval. In *Proceedings of the APRA Image Understanding Workshop*, pages 661–668, May 1997.

[66] Yossi Rubner and Carlo Tomasi. Texture metrics. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, October 1998. To appear.

[67] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. Adaptive color-image embeddings for database navigation. In *Proceedings of the 1998 Asian Conference on Computer Vision*, pages 104–111, January 1998.

[68] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. The earth mover's distance as a metric for image retrieval. Technical Report STAN-CS-TN-98-86, Computer Science Department, Stanford University, September 1998.

[69] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. A metric for distributions with applications to image databases. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 59–66, January 1998.

[70] William J. Rucklidge. Locating objects using the Hausdorff distance. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 457–464, 1995.

[71] William J. Rucklidge. Efficient guaranteed search for gray-level patterns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 717–723, June 1997.

[72] Brian Scassellati, Sophoclis Alexopoulos, and Myron Flickner. Retrieving images by 2d shape: a comparison of computation methods with human perceptual judgments. In *Proceedings of the SPIE*, volume 2185, pages 2–14, 1994.

[73] Jacob T. Schwartz and Micha Sharir. Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves. *The International Journal of Robotics Research*, 6(2):29–44, Summer 1987.

[74] Jianbo Shi and Carlo Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, June 1994.

[75] Allen W. Spivey and Robert M. Thrall. *Linear Optimization*, chapter 6, pages 213–243. Holt, Rinehart, & Winston, 1970.

[76] Jorge Stolfi. Personal communication, 1994.

[77] Michael J. Swain and Dana H. Ballard. Indexing via color histograms. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 390–393, 1990.

[78] Carlo Tomasi. Personal communication, 1999.

[79] Godfried T. Toussaint. On the complexity of approximating polygonal curves in the plane. In *Proceedings of the IASTED International Symposium on Robotics and Automation*, pages 59–62, 1985.

[80] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall, Inc., 1998.

[81] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, April 1991.

[82] V. V. Vinod and Hiroshi Murase. Focused color intersection with efficient searching for object extraction. *Pattern Recognition*, 30(10):1787–1797, 1997.

[83] V. V. Vinod, Hiroshi Murase, and Chie Hashizume. Focussed color intersection with efficient searching for object detection and image retrieval. In *Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems*, pages 229–233, 1996.

[84] Endre Vazonyi Weiszfeld. Sur le point par lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematics Journal*, 43:355–386, 1937.

[85] Michael Werman and Shmuel Peleg. Halftoning as optimal quantization. In *Eighth International Conference on Pattern Recognition*, pages 1114–1116, October 1986.

[86] Michael Werman, Shmuel Peleg, and Azriel Rosenfeld. A distance metric for multidimensional histograms. *Computer Vision, Graphics, and Image Processing*, 32(3):328–336, December 1985.

[87] George O. Wesolowsky. The Weber problem: History and perspectives. *Location Science*, 1(1):5–23, May 1993.

[88] Gunter Wyszecki and Walter S. Styles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley, 1982.