HIERARCHICAL IMAGE SEGMENTATION USING THE EARTH MOVER'S DISTANCE

by

Darren MacDonald

Submitted in partial fulfillment of the requirements
for the degree of Bachelor of Computer Science
with Honours

at

Dalhousie University
Halifax, Nova Scotia
April 2005

DALHOUSIE UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Computer Science for acceptance a thesis entitled “HIERARCHICAL IMAGE SEGMENTATION USING THE EARTH MOVER’S DISTANCE” by Darren MacDonald in partial fulfillment of the requirements for the degree of Bachelor of Computer Science with Honours.

Dated: April 8, 2005

Supervisor: _____

Dr. Michael McAllister

Reader: _____

Dr. Qigang Gao

DALHOUSIE UNIVERSITY

DATE: April 8, 2005

AUTHOR: Darren MacDonald

TITLE: HIERARCHICAL IMAGE SEGMENTATION USING THE EARTH MOVER'S DISTANCE

DEPARTMENT OR SCHOOL: Department of Computer Science

DEGREE: BCSc Hon. CONVOCATION: May YEAR: 2005

Hierarchical Image Segmentation using the Earth Mover's Distance

Table of Contents

Table of Figures

Abstract

Image segmentation methods often operate with the goal of partitioning the image into regions of uniform colour. Because real-world objects can have many different colours, we hypothesize that a segment uniformity measure based on colour distributions, instead of mean colour for example, is more appropriate for object segmentation. To demonstrate this, a histogram comparison metric known as the Earth Mover's Distance is used with a graph contraction algorithm in a new segmentation technique. Graph contraction is used for its efficiency and its ability to generate multiple-level segmentations called pyramids. The advantages of pyramidal object segmentation for object recognition and stereoscopy, two prevalent problems in computer vision, are discussed. Output is shown for images from the Berkeley Segmentation Data Set and compared against manual image segmentations.

1 Introduction

Image segmentation is the process of subdividing an image into groups of related pixels. The need for accurate segmentation of an image into a number of regions was recognized early in digital image processing. Segmentation reduces the number of elements in an image, summarizing thousands of pixels into a manageable number of segments. Where the pixel values of a digital image represent only atomic information, pixel groups have additional information including shape, size, topology, colour patterns, and texture. Because of this, segmentation is a good starting point for understanding what is in the image. Image segments can be used in many higher-level processes which must make decisions based on the content of the image. Counting and measuring features for production control, recognition and classification in medical imaging [HDF99], and video compression [KS01] are a few examples of applications. The focus of this approach is for computer vision.

If segmentation is performed as a precursor for some other process or computation, evaluation of a segmentation method must consider how the output will be used. In general, it is desirable that the segments come as close as possible to delineating actual objects in the scene. Breaking down a scene by objects is called object segmentation. Haralick and Shapiro [HSh85] say of desirable segment traits: “Regions of an image segmentation should be uniform and homogeneous with respect to some characteristic such as gray tone or texture. Region interiors should be simple and without many small holes. Adjacent regions of a segmentation should have significantly different values with respect to the characteristic on which they are uniform. Boundaries of each segment should be simple, not ragged, and must be spatially accurate.”

In some cases, it is desirable to have the additional quality of multiple levels of detail. A structure known as a ‘pyramid’ is able to represent multi-level segmentations. At high levels in the pyramid, there are only a few large segments. Lower levels have many small segments, which are subdivisions of high level segments. The detail levels exist simultaneously and segments are organized in a hierarchy. Ideally, the segments which

exist at any particular level are as visually representative of the scene as possible. The pyramidal output provides higher-level processes with both generalized and detailed information about the image. The advantages of pyramids to specific computer vision problems are discussed in Section 1.1.

The contribution of this paper is the use of a metric known as the Earth Mover's Distance with an established segmentation method called dual-graph contraction. The goal is a segmentation technique which uses colour distributions to differentiate between objects in the image. We argue that a segmentation where neighbouring segments have dissimilar colour histograms is a good approach to object segmentation. Histogram similarity measures are reviewed in Section 2.2. We use the Earth Mover's Distance based on its success in image retrieval studies. Existing segmentation methods are covered in Section 2.1. Dual-graph contraction is used due it its efficiency, its ability to easily incorporate colour distribution comparison, and its inherent pyramidal output.

The integration of the Earth Mover's Distance and dual-graph contraction is presented in Chapter 3. Results of the algorithm are shown in Chapter 4 and are compared qualitatively against manual segmentation data. Concluding remarks are given in Chapter 5.

1.1 Motivation

Vision is a prime example of a task where computers fail to match human capabilities. Even with unlimited computational resources, it is very hard for a computer to 'understand' digital imagery. Vision is the most important input device for human perception – it provides us with the most information about the world around us. Sight enables countless tasks which involve our physical surroundings but do not require higher thinking; these are commonly the tasks we want to automate.

Work in the broad area of image processing has assembled a vast array of filters, transforms, and other tools for measuring or extracting well-defined data from imagery. Some image processing applications, such as text recognition and quality control, are

programmed to find and measure predefined features. Others, such as satellite photo manipulation in GIS, may require a human operator. The term ‘computer vision’ refers to the problem of automatically constructing an internal model of the physical world by interpreting two-dimensional projections of the scene. Providing a computing device with a high-bandwidth input mechanism about the physical world would allow it to base its decisions on features in its environment instead of on the raw data, much as we do. Processes which can be built on top of such a vision system span automation and robotics.

A computer vision system should be able to accomplish two tasks. One task is to discriminate between objects in the scene in order to determine their shape, location and pose. Some systems can recognize feature patterns and relate them to a supplied model [Low04]. Object recognition methods can be based on matching object models in either two or three dimensions. Object recognition can therefore be performed before or after the second task, which is to recover, at least partially, the third dimension [JKS95]. The third dimension is recovered in a combination of ways by the human visual system, using cues such as occlusion and shadowing. However the method that shows most promise for computer systems is stereoscopy, the interpretation of a scene by comparison of different views. If the location of the viewer is known, the location of features in view can be calculated from simple geometric laws. The different views can be generated either simultaneously from multiple cameras or from a single moving camera. The greatest difficulty in stereoscopy is the identification and correspondence of features in the different images.

Segmentation can be useful to both of these tasks. Size, shape, colour, texture, and other segment attributes can help the system recognize objects in the image. The same attributes qualify segments as features which can be tracked across different images for stereoscopy – a process known as image registration. For these reasons image segmentation is a good place to begin for vision systems.

The algorithm presented in this paper generates a hierarchical segmentation pyramid. In this model, the top level represents a single segment that covers the entire image. Descending the pyramid, large segments are broken down into smaller ones, until

at the bottom level each segment covers only a single pixel. Though this model has greater memory requirements than a single planar segmentation, it has practical advantages stemming from eliminating the compromise between a coarse or fine segmentation. Also, because the pyramid extends down to the pixel level, no information is lost during the segmentation. Pyramids output more information than is in the original image, and subsume planar segmentations.

Higher-level computer vision processes can benefit from pyramids for several reasons. First, the model is able to reflect the hierarchical nature of physical objects, which vary widely in size and complexity. Using a pyramid, a multi-part object can be represented in its entirety at a high level, and still be decomposed into its individual components at a lower level in the pyramid. This flexibility is advantageous for a process attempting to interpret objects from the segments. Second, the ancestry of a segment in a pyramid gives it context which can be used for image registration. This task involves matching features in different images, which becomes increasingly difficult as the number of features, in this case segments, increases. The hierarchy allows registration first at a coarse scale, for a rough estimate of the disparity between the images, before attempting to correlate smaller segments. In theory, this approach should enable feature matching to converge down to the pixel level.

A hierarchical segmentation is advantageous to a computer vision system in other ways. If the segmentation is done top-down, in such a way that the coarse segmentation is available first, it can be made available to higher-level processes so that they may begin processing the information before it is all available. The feature correspondence process can begin matching the high-level segments, for example. The object recognition process can begin searching its object database for matches with objects in the scene. It is even conceivable for a high-level process to report back to the segmentation process, guiding the segmentation. After recognizing a feature of interest, a high-level process could focus the attention of the segmentation process on a particular region of interest in the image.

2 Background

2.1 Segmentation Methods

Two complementary approaches have emerged in segmentation, which will be discussed in turn. Edge detection searches the image for features which suggest the boundary of two objects, and delineates segments using these boundaries. Region detection searches the image for regions which are homogenous and do not suggest the presence of object boundaries. As edges can be obtained from segments and vice versa, the two are often considered as 'chicken and egg' problems. A recent trend is to combine elements of both approaches.

Edge detection is one of the oldest image processing operations. The most elementary methods pass a filter across the image that computes the image intensity derivative [MH80]. Examples of these filters are the Sobel, Prewitt, Roberts, Kirsch and Canny [Can86] operators (Figure 1). Points with a high derivative correspond to sections with a step or high gradient in colour, as is encountered at the boundary of one object and the next in the scene. After deciding which edges actually reflect object boundaries in the image, the edges are output as a line drawing of the physical scene. If the lines have appropriate topology [Jac96] they can be superimposed on the original image to create segmentation.

Figure 1. Edges of an image found using the Canny operator [Fis96].

More advanced edge detectors attempt to extract contours using less localized information. ‘Snakes’ are seeded contours which evolve over several iterations into a position which minimizes an energy function based on image intensity gradient and shape (Figure 2). Snake-based edge detectors have shown better connectivity and resilience to image noise than local filters [SM04]. Examples are Geodesic active contours [CKS97], Gradient Vector Flow [XP98], and EdgeFlow [MM00].

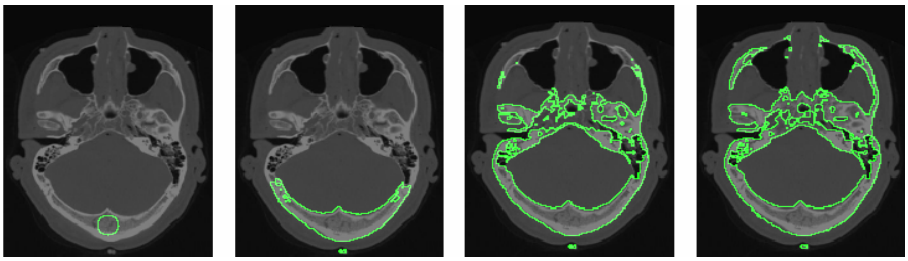

Figure 2. Active contour segmentation of bone in a CAT scan [Was05].

The position taken in this paper is that homogenous regions are the primary features, and that more accurate object segmentations are possible by looking for pixel neighbourhoods with similar, instead of dissimilar, visual qualities. Two examples of such region-based techniques are split-and-merge and region growing [JKS95]. Split-and-merge techniques begin with a single segment covering the entire image. If the segment is not uniform, it is split into sub-segments (commonly, four square segments). Adjacent uniform segments are then joined, and this process is repeated until some criterion is satisfied. Similarly, region growing begins with one or more seed locations, which expand by absorbing neighbouring pixels that satisfy the uniformity criterion. Seeded

Figure 3. Segmentation of a baseball scene using a region-merging approach [FH04].

region growing begins with user-defined seed locations, which normally requires an operator or advance knowledge of the image content. Automatically seeded regions are more desirable but less effective in the general case. Dual-graph contraction [Kro95] is a member of this category (Figure 3).

Other methods take a less localized view of region-based segmentation and group pixels by global criteria (Figure 4). These relate closely to clustering problems and are generally more adaptive [Dub93][PF99]. In these techniques, features are measured over the surface of the image and then organized in feature space. Often the measured features are the output from Fourier transformations or Gabor filters [JCH04], which are good at generating unique feature-space signatures for textures. Grouping can then be accomplished using one of many clustering techniques, such as k-means. Notable variations include spanning-tree methods [KC97], neural networks [Muh02], and eigenvector-based clustering [Wei99].

Figure 4. Segmentation based on clustering in feature-space [PF99].

Hybrid techniques have represented a large proportion of recent publications [JCH04] [MBS99] [KS01], perhaps because of the limited capabilities of any one method even after many years of research. Some methods perform each in turn, perhaps iteratively. For example, Fan, Yah, and Elmagarmid [FYE01] begin with an edge detector and use the centroids between the edges as seeds for region growing.

The importance of hierarchical segmentation has been recognized by scientists pursuing different directions in segmentation. Shi and Malik [ShM97] presented a method to build a pyramid top-down using a popular segmentation criterion called the normalized cut and an eigenvector based clustering method for computing it. Graph contraction

methods implicitly build a pyramid bottom-up during the merging process. This has been leveraged by Fuh, Cho, and Essig [FCE00], for example, who use the hierarchy for content-based image retrieval. Dual-graph contraction and the value of irregular pyramidal representation are well covered by Haxhimusa and Kropatsch [HK04]. Graph theory for the subject is analyzed by Kropatsch [Kro95].

Dual-graph contraction is based on merging adjacent segments which are similar according to some criteria. An adjacency graph is used for efficiently representing the segments and their connectivity. Dual-graph contraction is so-called because the adjacency graph is the dual of a graph in which faces represent segments. Because we use only the adjacency graph, the algorithm we use will hereafter be referred to as simply 'graph contraction'. Edge weight is determined by some measure or combination of measures pertaining to the similarity of the two neighbouring regions. This measure is known as the decimation kernel. The graph begins as a grid of single-pixel segments. Iteratively, the lowest weight edge is contracted, which merges the neighbouring segments. Felzenszwalb and Huttenlocher's graph-based method [FH04] compares the colour variability between the regions with the variability within the region. If the inter-segment variance is large compared to the intra-segment variance, there is evidence for a boundary between the regions. They found that this results in a fast algorithm which makes greedy decisions, yet still captures global properties. Similar measures are used by Haximusa and Kropatsch [HK04] and Fuh, Cho, and Essig [FCE00].

2.2 Histogram Comparison Methods

The decimation kernel used in this paper measures the similarity of the colour histograms of neighbouring segments. The basis for using this measure is discussed in Section 4.1. This section will review some of the many ways to evaluate histogram similarity [ZL03].

The **Minkowski-Form** sums the difference of the histogram values on a bin-by-bin basis. It is based on the L_p norm, and is defined as:

$$L_p(H, K) = \left(\sum_{i=0}^{N-1} |H_i - K_i|^p \right)^{1/p}$$

where $H = \{H_0, H_1, \ldots H_{N-1}\}$ and $K = \{K_0, K_1, \ldots K_{N-1}\}$ are fixed-bin histograms and $1 \leq p \leq \infty$. The L_1 norm, sometimes called the city-block distance, is a simple difference summation and is popular for image retrieval. L_2 (Euclidean distance) and $L\infty$ (maximum distance) norms are also used. Greater p values results in greater sensitivity to outliers.

The **Histogram Intersection** [SB91] method finds the intersection volume on a bin-by-bin basis. It is defined as:

$$d_{\cap}(H, K) = 1 - \frac{\sum_{i=0}^{N-1} \min(H_i, K_i)}{\min(|H|, |K|)}$$

where $H = \{H_0, H_1, \ldots H_{N-1}\}$ and $K = \{K_0, K_1, \ldots K_{N-1}\}$ are fixed-bin histograms, and $|H|$ and $|K|$ are the total weights of H and K. This measure is useful for measuring histograms of different total masses. When the histogram sizes are equal, it degrades to the L_1 difference.

X^2 Statistics [MBS99] is defined as:

$$d_{X^2}(H, K) = \sum_{i=0}^{N-1} \frac{(H_i - m_i)^2}{m_i}$$

where $H = \{H_0, H_1, \ldots H_{N-1}\}$ and $K = \{K_0, K_1, \ldots K_{N-1}\}$ are histograms, and $m_i = (H_i + K_i)/2$. The X^2 Statistic measures the likelihood that one distribution was drawn from the other's population.

These three measures have the weakness that they can only compare histograms on a bin-by-bin basis. Figure 5a shows one of the many possible instances where bin-by-bin measures fail to represent perceptual differences. It is desirable to have a histogram comparison formula where bin differences are better quantified than 'equal' and 'not equal'. The following measures use cross-bin measurements.

Quadratic Form is defined as:

$$d_A(H, K) = \sqrt{((\mathbf{h}-\mathbf{k})^T A\ (\mathbf{h}-\mathbf{k}))}$$

where **h** and **k** are vectors of histogram bin values from H and K respectively and A is a matrix of distance values between each pair of bins. The Quadratic Form distance has the effect of summing the distance-mass from each pair of bins (see Figure 5b). This measure tends to over-estimate the distance of histograms which do not have pronounced peaks.

Match is defined as:

$$d_M(H, K) = \sum_{i=0}^{N-1} | \mathbf{H}_i - \mathbf{K}_i |$$

where **H** and **K** are cumulative histograms, in which bins are defined as

$$\mathbf{H}_i = \sum_{j=0}^{i} H_j$$

and likewise for $\mathbf{K}_i$. The match distance finds the minimum cost matching between two one-dimensional histograms (see Figure 5a). Unfortunately the efficient algorithm which evaluates to the L_1 difference of the cumulative histograms is not computationally applicable to multiple dimensions. A comparison method for colour space histograms must be able to handle three dimensions.

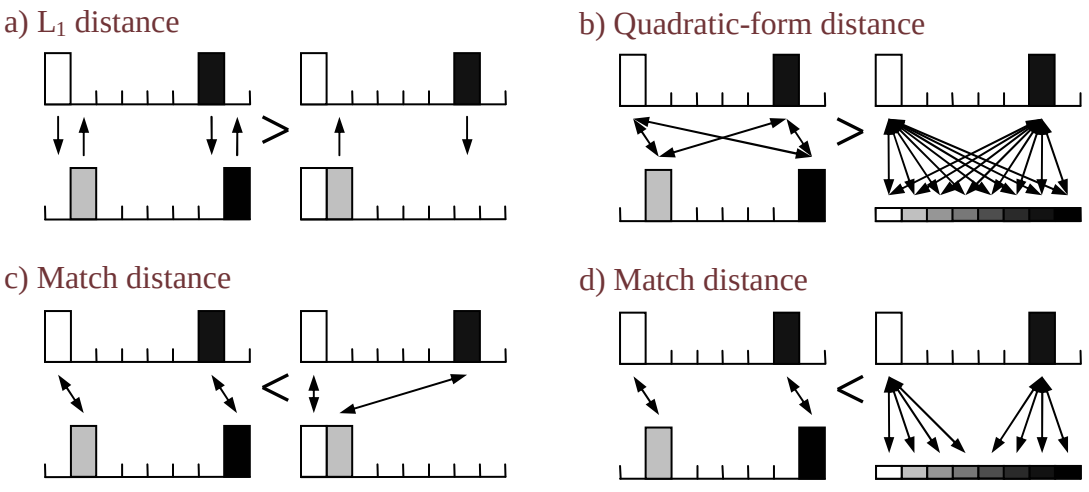

Figure 5. Examples where match distance better approximates relative histogram perceptual dissimilarity than a) L1 distance and b) Quadratic-form distance [RTG00].

Werman, Peleg and Rosenfeld [WPR85] explain how match distance in multiple dimensions, though more difficult to compute, is still an intuitive and useful comparator for histograms. It is also a metric, provided the bin distances are metrics. Rubner, Tomasi and Guibas [RTG20] apply multi-dimensional match distance to colour distributions for the purpose of image retrieval. They call this the Earth Mover's Distance (EMD), because it measures the minimum amount of work required to transform one histogram (a conceptual mound of earth) into another (conceptually, holes in the ground which must be filled). Because it incorporates bin distance, and compares distributions in a minimal-energy transformation sense, it is successful in matching perceptual distances. Rubner et al. report better retrieval performance with Earth Mover's Distance than with the above-mentioned comparators.

The Earth Mover's Distance is beginning to find uses in different areas of computer vision [CG99]. Ruzon and Tomasi's compass operator [RT99] [MB03] uses EMD for edge detection. They argue that edge detectors should not assume that regions have constant colour. By computing the EMD of the colour distributions on either side of the filter, the compass operator reports less false negatives than with classical edge detectors. Grauman and Darrell use EMD for shape comparison [GD04]. Their method finds the minimum-cost matching of two contours.

3 Graph Contraction using the Earth Mover's Distance

All object segmentation methods must define, in one way or another, what measurable qualities are best for discriminating between objects in an image. Haralick and Shapiro [HSh85] state that adjacent regions of a segmentation should have significantly different values with respect to the characteristic on which they are uniform, however the measure that best characterizes objects in an image is not immediately obvious. Many homogeneity criteria have been proposed, comprising such values as mean colour intensity, colour variance, texture values, and optical flow (in video segmentation).

The best way to partition the image such that the uniformity within the regions is high, but between the regions is low, is also not obvious. Edge-detection methods find image regions with the absence of steep gradient in the uniformity measure. Clustering methods find partitions of features space which reflect grouping trends. Graph contraction techniques use the uniformity criterion directly, for determining dissimilarity measures for neighbouring segments.

Shi and Malik [ShM97] summarize image segmentation with two questions:

1) What is the precise criterion for a good partition?

2) Can such a partition be computed efficiently?

These questions will be addressed in Sections 3.1 and 3.2 respectively.

3.1 Partitioning Criterion

This work proposes the use of colour distributions as the quality used to distinguish objects in an image. The colour distribution of a group of pixels can be summarized with a histogram. A colour histogram divides the colour space into finite bins and counts how often a colour is present in the sample. Colour histograms have been investigated for content-based retrieval from image databases, where they have been found to be a good representative measure for image content [SB91][RTG00][ZL03]. That is, given a query

image, they are able to retrieve images containing the same objects by comparing the colour content. This suggests colour distributions are good at characterizing objects, and is a good motivation for using colour histograms in the context of image segmentation.

Assuming objects in a scene have distinctive colour distributions, similar colour histograms in adjacent segments suggests they belong to the same object. Conversely, dissimilar histograms suggest the opposite. We cannot evaluate a single histogram to determine if it represents only a single object, as an object may have any conceivable colour histogram. Therefore the criterion we wish to optimize has nothing to do with the uniformity within a segment, but instead focuses on the non-uniformity between segments. This differs from many methods, such as Felzenszwalb and Huttenlocher [FH04], who define their criterion with an internal and an external component. In our case, the internal component would be redundant – if there were a segment with low internal uniformity then it could be subdivided into segments which have low uniformity between them.

The colour space in which digital images are encoded is RGB, which is the combination of red, green, and blue intensity values required to generate the desired colour in an additive red, green, and blue palette. Unfortunately, distances in the RGB colour space do not correspond accurately to perceptual distance. The CIELAB colour space was developed by the Commission Internationale de l'Eclairages to meet this purpose. CIELAB is also three-dimensional: **L** describes luminance or lightness, **A** represents the red-greenness value, and **B** describes the yellow-blueness value. It is a non-linear transformation of RGB, and approximates the relative perceptual distance of colours. By using the CIELAB colour space for perceptually accurate bin distances, we expect perceptually accurate histogram distances.

We hypothesize that a segmentation with high colour histogram distances between adjacent segments will accurately delineate objects in the scene. Therefore, our predicate for a good partition is a function defined between neighbouring segments. Earth Mover's Distance (EMD) will be used to compare histograms, based on findings of Rubner et al

[RTG00]. The Earth Mover's Distance is elaborated upon in Section 3.2.2. The higher the EMD distances between the segments, the better the segmentation.

One of the points on which Rubner et al. favour EMD is that it handles partial matches very naturally, which is important when comparing images of different sizes. This is useful in image retrieval when searching for a particular item in a database of cluttered images. Partial matching is not as desirable for our purposes. Our basis for merging adjacent segments is that if they have similar colour distributions, they are likely part of the same object. Assuming they are part of the same object, both segments should have relatively consistent total distributions. As it is the shape of the histogram that determines the segment's colour signature, histograms are normalized to a total mass of 1.0.

Unfortunately, normalizing histograms to a total mass of 1.0 annuls the important perceptual aspect of segment size. The size of an object has a great deal to do with its importance in the scene. We can have a segment with a very distinct colour pattern, but if the segment is very small, it may not be very important in the scene. Small segments have much less shape and colour information, so very small segments should not appear at high levels in the pyramid.

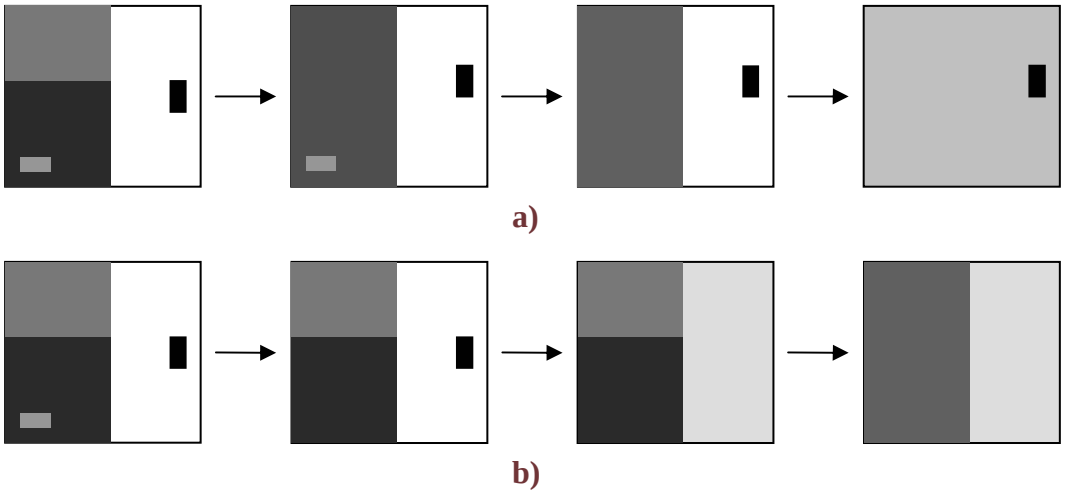

Figure 6. Demonstration of the importance of segment size. Arrows indicate merging order. a) Merging order when considering only EMD between segments. Desired merging order b) is obtained when multiplying EMD distance by smaller segment's size.

As the predicate is a function based on neighbouring segments, we wish to assign low values between small segments. To this end, we define the segmentation predicate as the product of the EMD of two segments and the size of the smaller segment. This way, small segments contribute negatively to the quality of the segmentation.

Discouraging small segments causes segments to be relatively the same size. This may seem divergent from the goal of object segmentation, as objects may be of varied sizes in the image. However it has particular importance for a region-merging algorithm which makes greedy decisions about the similarity of image regions (see Figure 6) – if segments are roughly the same size, it can make a more informed decision about which segments are most similar. Merging small segments is important for generalizing textured or noisy areas into coherent uniform segments. Additionally, similarly-sized segments are favourable computationally because they keep a balanced pyramid.

In summary, the formula for determining the distance between segments S_1 and S_2, with histograms H_1 and H_2 respectively, is:

$$d(S_1, S_2) = EMD(H_1, H_2) \times MIN(|S_1|, |S_2|)$$

The algorithm presented in the next section attempts to segment an images such that this value is high between all neighbouring segments.

3.2 Partitioning Algorithm

After having defined the characteristic we desire of the image segments, we now explain the method used to compute them. The algorithm is based on the graph contraction methods previously discussed. These methods represent the segments and their connectivity with an adjacency graph. Iteratively, the two most similar neighbouring segments are merged. After merging the similarity measures between the combined segment and its neighbours are recomputed. By repeatedly merging the most similar segments, it is hoped that the remaining segments will be very dissimilar.

This is an example of a greedy algorithm because the choice of which segments to merge is made according to what seems best at the time. Greedy algorithms are known for being efficient. Felzenszwalb and Huttenlocher [FH04] report running times nearly linear in the number of image pixels. But because it makes greedy decisions, this tactic does not provide optimality guarantees about the quality of the segmentation. This means that at any stage in the merging process, there is no guarantee that the image partitioning maximizes the inter-segment differences based on the predicate. Such a global optimization would most likely be very hard to compute [Coo98]. Still, results from Felzenszwalb and Huttenlocher suggest it is possible to capture global image characteristics by careful selection of a merging predicate.

The segmentation pyramid is built bottom-up during the merging process. Each time two segments are merged, the combined segment becomes the parent of the two which are merging. By recording the parent-child relationships and the order in which the segments are joined the pyramid can represent the image using any number of segments.

Following is a detailed explanation the data structures and operations used in our work. As running time is an issue, complexity considerations are addressed throughout. Section 3.2.1 will explain the framework for graph contraction and building the pyramid. Section 3.2.2 explains computation of the EMD. A section on running time analysis follows. The test program was written in C++, using CImg [CI05] and ImageMagick [IM05] libraries and Standard Template Library containers.

3.2.1 Graph Contraction

There are three principal data structures in our implementation of the graph contraction algorithm. The adjacency graph G is composed of edge and node objects. We initialize G with four-connected nodes in a regular grid, however this is not imperative for operation of the algorithm (the flexibility of adjacency initialization permits extension into higher dimensions). Each node in G is assigned a segment. Segment objects and their parent-child relationships establish the segment pyramid P. At initialization, segments

comprise only a single pixel. When two segments are merged, they become children of the newly created, merged segment. Hence, the segment belonging to a node in G is at the top of a hierarchy, and its spatial domain and colour distribution are determined by its leaf-level descendents. P is therefore a forest until there is only one remaining node. Finally, a heap of edge pointers H allows fast extraction of the lowest-weight edge.

```
1.   Initialize graph G and heap H. Segment s_n = null.
2.   While H is not empty:
3.      Pop the lowest valued edge e from H. Let n_1 be
        the larger endpoint, and n_2 the smaller.
4.      Let s_1 and s_2 be n_1's and n_2's segments. Create a
        new segment s_n with s_1 and s_2 as children.
5.      Set s_n as n_1's new segment.
6.      For each edge incident on n_2:
7.         Swap endpoint from n_2 to n_1. Reorder opposite
           endpoint's edge list.
8.         Remove if a double edge.
9.      For each edge incident on n_1:
10.        Recompute edge weight.
11.        Move edge up or down in H.
12.     Delete n_2 and e from memory.
13.  Return s_n as the root of P.
```

Figure 7. Pseudocode for the graph contraction algorithm.

Figure 7 outlines the steps used to generate the pyramid. The main loop removes the lowest-weight edge, merges its two endpoints, and updates incident edges. In order to quickly find which edges are affected by a merger, nodes contain a list of incident edges. Because edge lists are ordered by memory location of opposite endpoint node, double edges can be detected easily (step 8). Consequently, edge lists in nodes neighbouring n_2 must be readjusted to maintain correct order, as they contain an edge whose opposite endpoint is swapped from n_2 to n_1 (step 7). The separation of segments and nodes allows the graph to reuse one of the nodes in the merger, reducing the number of edges which must be updated to maintain the correct topology (step 6).

Each time a new value is computed for an edge (step 10) the modified edge is bubbled up or down in the heap, depending on whether its weight was increased or

decreased, until the heap is reestablished (step 11). In order to readjust only the modified edge it must first be located in the heap. To do this efficiently edges keep their heap index (the heap is a random-access container) as an attribute. Maintaining heap indices is accomplished by modifying the STL heap to execute a function object each time one of its elements changes position. The function object reports back to the edge its new index.

3.2.2 Earth Mover's Distance

To recapitulate, the Earth Mover's Distance (EMD) is a metric for measuring the similarity of two distributions. EMD has roots in the transportation problem of graph theory. Given a set of suppliers, each with a surplus of mass, and a set of consumers, each with a deficit, and ground distance between each supplier and consumer, the transportation problem is to find the minimum work required to achieve equilibrium. The transportation graph is a complete bipartite graph to which we must assign edge flows such that each supplier and consumer is satisfied while minimizing the flow-cost product of all edges.

EMD interprets one histogram as suppliers and the other as consumers. Surplus or deficit is determined by the mass in the histogram bin. Ground distances are obtained from CIELAB colour distances between bins. Under this interpretation, the EMD computes the minimum work required to transform one distribution into the other.

The Earth Mover's Distance is an optimization problem, and can be solved with linear programming. The transportation simplex algorithm [HL95] takes advantage of certain properties of the problem to reduce its complexity. We formulate the transportation simplex algorithm as follows: let $H = \{(h_0, w_{h1}), (h_2, w_{h2}), \ldots, (h_n, w_{hn})\}$ be the first histogram, with h_i the bin designation and w_{hi} the weight of that bin, and similarly for $K = \{(k_1, w_{k1}), (k_2, w_{k2}), \ldots, (k_m, w_{km})\}$. For computational reasons empty bins are not included in the sets H and K. Costs c_{ij} are available for every pair of bins h_i and k_j. We must calculate the flow f_{ij}, between every pair of bins h_i and k_j that minimizes

the total cost:

$$EMD(H, K) = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} c_{ij} f_{ij}$$

Subject to the following constraints:

1) $f_{ij} \geq 0$ for $0 \leq i < n, 0 \leq j < m$

2) $\sum_{j=0}^{m-1} f_{ij} = w_{hi}$ for $0 \leq i < n$

3) $\sum_{i=0}^{n-1} f_{ij} = w_{kj}$ for $0 \leq j < m$

Constraint 1) ensures that mass is transported from H to K and not vice versa. Constraint 2) ensures that the mass flowing from h_i equals its supply, and constraint 3) ensures that the mass delivered to k_j equals its demand. Since both histograms are normalized to a total mass of 1.0, the additional constraint that $\sum_{i=0}^{n-1} \sum_{j=0}^{m-1} f_{ij} = 1.0$ is implied. The variables of this program are the edge flows f_{ij}, which are represented by an m by n flow table. The size of this table results in the EMD being an expensive process to solve. Additionally, the number of pivots can be quite high, in theory, exponential in n + m, though in practice the number does not approach this limit. The complexity of EMD computation for large histograms is prohibitive for real-time applications.

Fortunately, it is not necessary to evaluate the EMD from scratch for each computation. In order to reduce the computation time, it is possible to take advantage of previously solved EMD flows when merging histograms. In Figure 8, the flow table between one of the merging nodes and a neighbour is a large subset of the flow table between that neighbour and the merged pair. This subset can be considered as a pre-initialized section of the flow table. New flow must be initialized between the other merging node and the neighbour before appending the new flow to the existing table. After initializing the new flow independently the tables can be combined and pivoted. This lowers initialization time, as part of the table is already initialized, and pivoting time, because that part of the table is optimal.

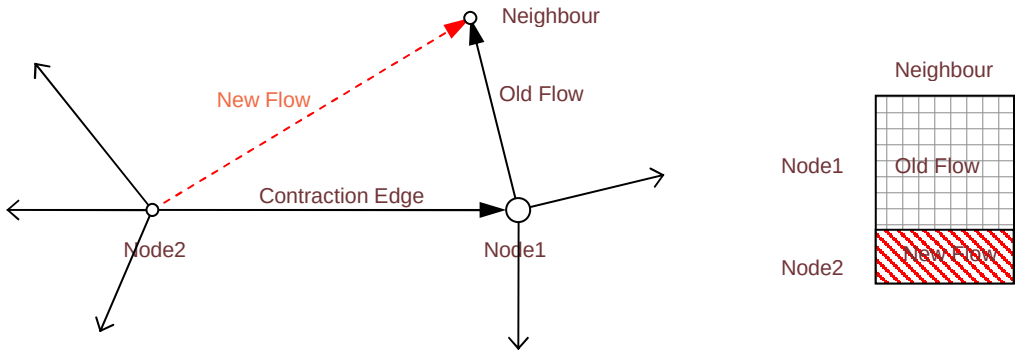

Figure 8. Illustration of the reuse of flow tables when merging Node1 and Node2.

3.2.3 Running Time Analysis

Refer to Figure 7 for the following algorithm references. Problem size n is in terms of image pixels. Because the adjacency graph is planar, and Euler's formula tells us that the number of edges in a planar graph is less than six times the number of nodes, we can reduce terms dependent on the number of edges in the graph to n. Similarly, the number of occupied histogram bins in a segment is bounded above by the number of pixels in the segment, and can therefore also be reduced to n.

Initialization of the data structures and objects (step 1) is dominated by the construction of the heap, which is in O(n log n) time. The main loop (step 2) is executed n times, as each iteration removes one node. Popping the next edge from the heap (step 3) is in O(log n) time. Histograms are sorted containers and can be merged in O(n) time (step 4). The number of iterations of step 6 is dependent on the number of neighbours of n_2, which is upper bounded by n. But because the number of edges in the graph is constant in the number of nodes (because of Euler's Law), we can expect the average number of neighbours per node to be a low constant. We do not expect much deviance from this average because segments are similarly sized. Reordering the edge list in each of n_2's neighbours (step 7) requires an iteration with similar number of comparisons. In brief, we can expect average execution in O(1) time in step 6. By the reasoning we can expect a constant number of iterations of step 9. Within this subloop, we compute the EMD of two

distributions (step 10). Due to the number of flow variables, the EMD runs in $O(n^2)$ time. Step 11 requires a single pass up or down the heap, taking O(log n) time.

Reducing all terms gives an expected asymptotic running time in $O(n^3)$. This is clearly dominated by the time required to compute the EMD. Exchanging the EMD for an edge weight measure that computes in constant time, such as difference in mean colour value, yields a much more acceptable average running time in O(n log n), and in practice executes in time nearly linear in the number of pixels.

In order to lessen the practical EMD computation time, we use histogram bins of a size of four units in each direction, and pivots are arrested after improvements to the objective value function fall below a low threshold. Asymptotic running time after enlarging histogram bins, reducing the number of pivots, and implementing flow table reuse (as illustrated in Figure 8) remains $O(n^3)$, however in practice the improvement in running time is quite noticeable. Running time for a 200 x 200 pixel colour image is on the order of 20 minutes for a commodity Pentium 4 desktop computer. Memory usage for the same test is on the order of 80 megabytes.

4 Results

4.1 Experiment Setup

Since segmentation is usually performed as a pre-processing step for some other operation, the true test of a segmentation algorithm is how well higher-level operations can make use of the information it generates. Some researchers argue that segmentation or other grouping techniques can only be evaluated by applying them to a particular task, such as object recognition [BS97]. This is difficult because applications that use image segments vary greatly, and practical segmentation applications which are tuned for a particular segmentation technique are not useful for unbiased comparison of segmentation methods. To evaluate general-purpose segmentation techniques we often resort to human observation. The Berkeley Segmentation Data Set (BSDS) [MFT01] has been developed specifically for the purpose of benchmarking image segmentations against human perceptual grouping. BSDS provides a diverse set of images as well as ground truth data obtained by having human subjects manually segment each image. While the image segments found by a human subject may not be optimal for any image processing task, they often reflect the same features we are seeking for these tasks, such as important object boundaries.

The algorithm was tested on a variety of images, including a set of 6 randomly sampled from BSDS. Figure 9 shows original images, manual segmentations, and different levels of the segmentation pyramids generated by our algorithm. The manual segmentations are shown with a binary image identifying the most important object boundaries in the image. The image is a superposition of six manual segmentations, therefore stronger lines indicate more important boundaries. We should expect to see the same segment boundaries at high levels in the pyramid.

Errors will be in the form of false positives, where our pyramid has segment boundaries that do not exist in the manual segmentation, and false negatives, where our pyramid has a segment which overlaps a boundary in the manual segmentation. At the

very bottom of the pyramid there will be no false negatives because there will be a boundary between every adjacent pixel. Moving up the pyramid the number of false negatives will increase, and the false positives will decrease. False positives are a somewhat lesser evil because even if an object is over-segmented, we can take the union of those segments to obtain an accurate representation of the object. Therefore we hope to find segmentations with few false positives but almost no false negatives.

The images have been convoluted with a Gaussian edge sharpening operator with a ten-pixel radius to increase sensitivity to object boundaries, based on observations in the preliminary stages of the experiment.

4.2 Observations

Figure 9a shows the segmentation of a scene containing a boat on a dock. High levels of the pyramid show very rudimentary representations of the scene, but capture perceptually important regions; at level four, the segments roughly correspond to the boat, the dock, the water and the sky. Descending the pyramid, more and more detail is presented. By level 32, most of the major segment boundaries are present, though some of the larger regions, particularly the water and sky, are over-segmented.

Figure 9b is more complicated image, as object boundaries are not always suggested by local image features. Separation of the two bears is difficult because they have very similar colour distributions, however the algorithm is able to differentiate between the bears and the background as early as level eight. Because we use colour distributions, the background region, which has texture and many different colours, is generalized into large coherent segments at high levels, yet we can still find more subtle snowy or bushy regions lower in the pyramid. The generalization of textured regions can also be seen in the dock in Figure 9a, the building in Figure 9e, and the background grid in Figure 9f.

The contouring that occurs at areas with a smooth gradient in colour, such as in Figure 9c, happens for both natural gradients and photographic effects. These segments falsely suggest the presence of a boundary between objects in the scene, though the region around the boundary line is quite uniform. While this provides information that would not be retrieved by an edge detector, it complicates object recognition.

One of the most notable differences is that image partitioning by hand yields segments in a variety of sizes, where the segmentation pyramid has similarly-sized segments at any particular level. The manual segmentation for Figure 9f shows small segments for the woman's eye and lips, which are adjacent to a large background segment. In the pyramid, most major segments are available by level 16, though the eye and lip features do not appear until level 64, at which point the background is subdivided into many small segments. Merging small segments early on is a compromise required to obtain a good colour representation of a region, as explained in Section 3.1. Users can be confident, however, that small features will emerge eventually at a lower level in the pyramid.

A detrimental aspect of the segmentation generated by this method is the large effect of connectivity. A large object in the scene can be divided into small segments if it is occluded by another object. Even if the occluding object is a single pixel wide, two segments which represent the same occluded object cannot join even if they have very similar colour distributions. Similarly, different objects at different locations in the scene can be joined into one segment if there is a bridge between the two. Both of these cases are undesirable in the context of object segmentation. Connectivity problems are a recognized problem in graph merging techniques for image segmentation. Felzenswalb and Huttenlocher address the issue by initializing the adjacency network according to feature-space neighbourhoods instead of the pixel grid. This joins occluded objects but generates unconnected segments.

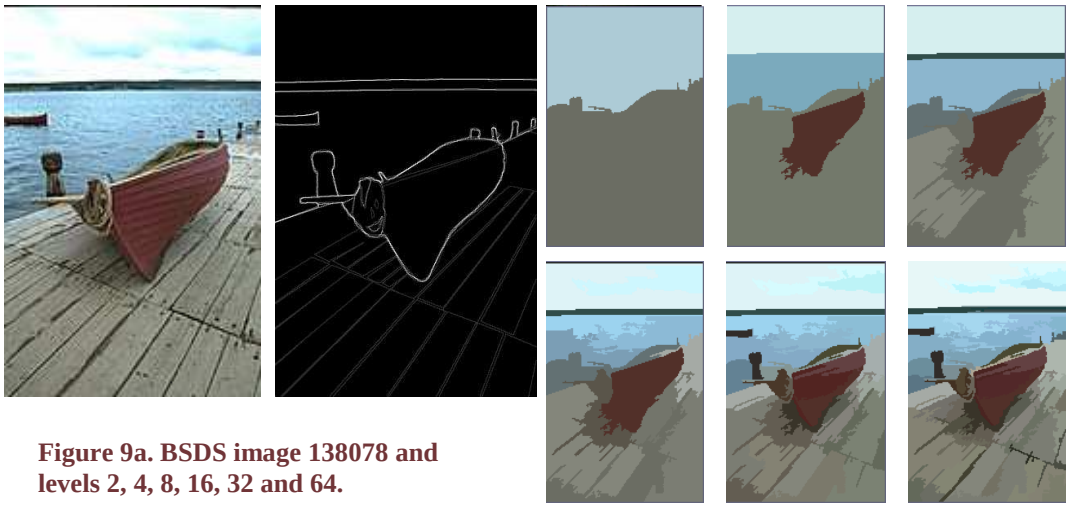

Figure 9a. BSDS image 138078 and levels 2, 4, 8, 16, 32 and 64.

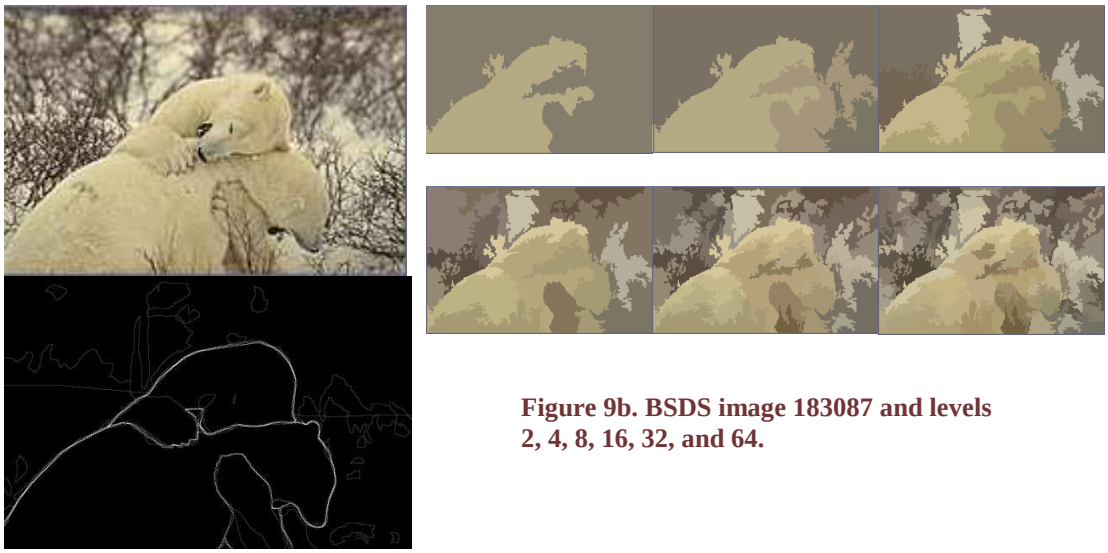

Figure 9b. BSDS image 183087 and levels 2, 4, 8, 16, 32, and 64.

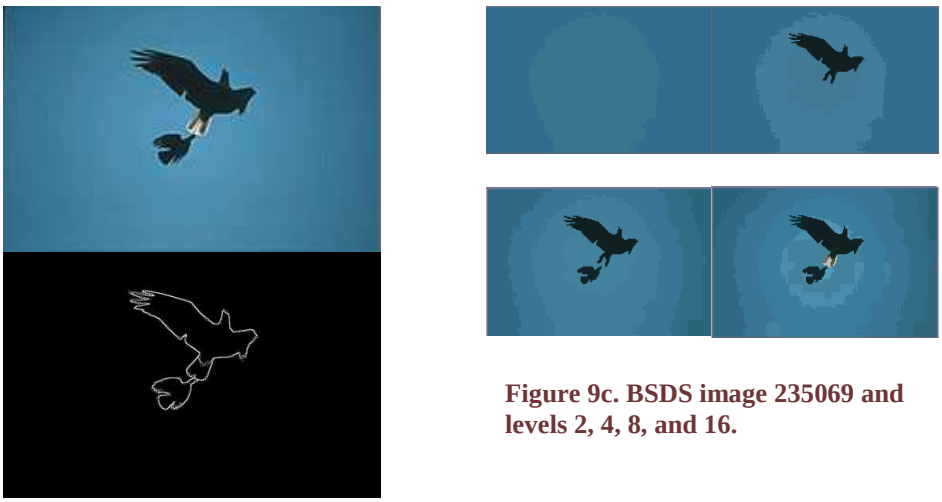

Figure 9c. BSDS image 235069 and levels 2, 4, 8, and 16.

Figure 9d. BSDS image 106025 and levels 2, 4, 8, 16, 32, and 64.

Figure 9e. BSDS image 227040 and levels 2, 4, 8, 16, 32, and 64.

Figure 9f. BSDS image 198023 and levels 2, 4, 8, 16, 32, and 64.

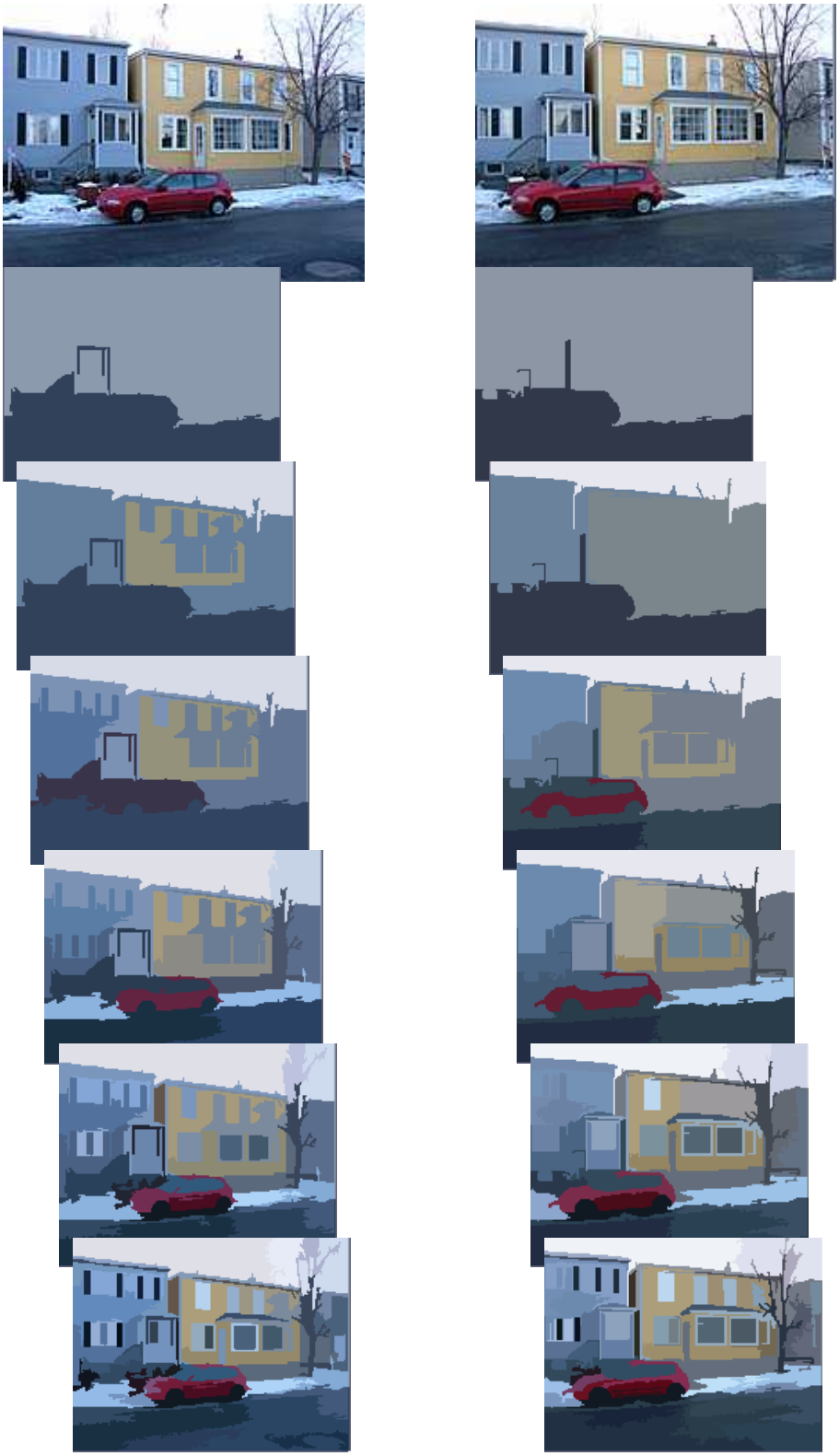

Figure 10. Street scene from different viewpoints. The segmentations at levels 2, 4, 8, 16, 32, and 64 are shown for each image.

4.3 Analysis

Image segments from our algorithm do not always match one-to-one with the segments from the BSDS. However, we can often obtain these segments by taking the union of segments at some reasonable level in the pyramid. Correspondence with Berkeley data does not guarantee that this method will be effective for computer vision but it does provide an indication as to how close we are to object segmentation.

Results are largely consistent with our desire for hierarchical semantic representation though we cannot rely on segments to represent one and only one object. Nevertheless, many of the recognition methods that are based on two dimensional contour model matching could be applied to this segmentation method [EC88][CH02], as well as (or in addition to) models based on texture or colour model matching. Also, the hierarchical representation may be beneficial in improving search accuracy [FCE00] and speed [AH97].

The situation is similar for image registration. Graph contraction algorithms are generally not scale, rotation and translation invariant, which are desirable qualities for registration. Moreover, it is reasonable to expect that changing the view only a little can objects to be segmented in very different ways (for example, the houses is Figure 10). This will make it very difficult to register the image on a segment-by-segment basis. Still, object boundaries, especially for pronounced objects, tend to be represented consistently through multiple views. Notice the similarity between sections of the boundary lines at high levels of the pyramids in Figure 10. Fortunately contour matching has been studied extensively for stereoscopy [Bro92]. Again, the hierarchical representation may be very useful for improving accuracy and speed [EL95][GPK02].

Until an efficient algorithm is devised for finding the optimal segmentation based on colour distribution difference between segments, this algorithm should be considered a good approximation. That is, as long as it can be considered efficient. The restrictive component of the algorithm is the time required to compute the EMD of large histograms. Computation time increases polynomially with the size of the histograms, which can

number in the thousands even using large histogram bins. Rubner, Guibas and Tomasi [RGT00] warn against using EMD to compare histograms directly. When applying EMD to image retrieval, they use colour signatures instead of histograms. Signatures represent colour content, much like a histogram, however they reduce the distribution into the most important handful of colours. Rubner et al. report no loss in retrieval quality by using signatures rather than histograms. Assuming they can be computed quickly, implementing signatures into this segmentation method would improve performance drastically.

5 Conclusions

This dissertation presents a new image segmentation algorithm that combines region merging with histogram comparison using Earth Mover's Distance (EMD). This amalgamation is based on the view that if two adjacent regions in an image have similar colour distributions, it is a good indication that they belong to the same object. In this light, segmenting an image into regions with high pairwise colour distribution differences is a good approach to object segmentation.

Because they make greedy decisions, graph contraction algorithms are asymptotically and practically fast. However, this advantage is annulled in this case by the high cost of computing the EMD between large histograms. A more compact colour distribution representation would resolve this issue. Still, this paper demonstrates the quality that can be achieved by using colour distribution similarity as the decimation kernel in a graph contraction algorithm.

An intrinsic advantage of region merging algorithms is that they can output segments at arbitrary detail levels. This allows them to build segmentation pyramids which concurrently represent an image at different levels of resolution. Using a carefully chosen merging predicate, the most perceptually important segments exist at the higher levels while small details reside at lower levels. Though it requires more space, this representation is superior to a single planar segmentation, which must compromise between detail and generality.

5.1 Future Directions

Future enhancements to this algorithm will substitute histograms for a more compact representation. Also, based on findings in other EMD applications, texture distributions can also be used to discriminate between objects in an image [RTG00][MB03]. It is easy to imagine a situation where segments are defined by texture patterns more so than colour patterns (for example, the grid behind the woman in Figure

9f). Textures can be compared by computing the EMD of distributions based on the coarseness and directionality responses from texture receptors.

Hopefully, the results of this experiment will encourage the use of colour histogram similarity as the homogeneity criterion in different approaches to image segmentation. It would be interesting to see the results of an algorithm that optimizes this measure.

References:

[AH97] Y. Abe and M. Hagiwara, “Hierarchical Object Recognition from a 2D Image using a Genetic Algorithm,” in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, vol. 3, pp. 2549-2554, 1997.

[Bro92] L. G. Brown, “A Survey of Image Registration Techniques,” *ACM Computing Surveys*, vol. 24, no. 4, pp. 325-376, 1992.

[BS97] S. Borra and S. Sarkar, “A Framework for Performance Characterization of Intermediate-Level Grouping Modules,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 11, pp. 1306-1312, 1997.

[Can86] J. Canny, “A Computational Approach to Edge Detection,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698, 1986.

[CG99] S. Cohen and L. Guibas, “The Earth Mover’s Distance under Transformation Sets,” in *Proceedings of IEEE International Conference on Computer Vision, ICCV 99*, pp. 1-8.

[CH02] O. Carmichael and M. Hebert, “Object Recognition by a Cascade of Edge Probes,” in *Proceedings of British Machine Vision Conference, BMVC 2002*, pp. 103-112.

[CI05] D. Tschumperlé, “The CImg Library – C++ Template Image Processing Library,” 2005, http://cimg.sourceforge.net.

[CKS97] V. Caselles, R. Kimmel, and G. Sapiro, “Geodesic Active Contours,” *International Journal of Computer Vision*, vol. 22, no. 1, pp. 61-79, 1997.

[Coo98] M. Cooper, “The Tractability of Segmentation and Scene Analysis,” *International Journal of Computer Vision*, vol. 30, no. 1, pp. 27-42, 1998.

[Dub93] R. Dubes, “Cluster Analysis and Related Issues,” in *Handbook of Pattern Recognition and Computer Vision*, C. Chen, L. Pau, and P. Wang, Eds. Singapore: World Scientific Publishing, 1993, pp. 3-32

[EC88] K.-B. Eom and X. Chen, “Maximum likelihood decision for recognition of noisy shapes,” in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, ICASSP-88*, vol. 2, pp. 972-975.

[EL95] J. Ens and Z.-N. Li, “Real-time Motion Stereo on SFU Pyramid,” *Real-time Imaging*, vol. 1, no. 6, pp. 385-396, 1995.

[FCE00] C.-S. Fuh, S.-W. Cho, and K. Essig, “Hierarchical Color Image Regions Segmentation for Content-Based Image Retrieval System,” *IEEE Transactions on Image Processing*, vol. 9, no. 1, pp. 156-162, 2000.

[Few55] D. R. Fewer, “Design Principles for Junction Transistor Audio Power Amplifiers,” *IRE Transactions on Audio*, vol. 3, no. 6, pp. 183-201, 1955.

[FH04] P. Felzenszwalb and D. Huttenlocher, “Efficient Graph Based Image Segmentation,” *International Journal of Computer Vision*, vol. 59, no. 2, pp 167-181, 2004.

[Fis96] B. Fisher, “Edges: The Canny Edge Detector,” 1996, http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MARBLE/low/edges/canny.htm.

[FYE01] J. Fan, D. Yah, A. Elmagarmid, and W. Aref, “Automatic Image Segmentation by Integrating Color-Edge Extraction and Seeded Region Growing,” *IEEE Transactions On Image Processing*, vol. 10, no. 10, pp. 1454-1466, 2001.

[GD04] K. Grauman and T. Darrell, “Fast Contour Matching Using Approximate Earth Mover’s Distance,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, CVPR 2004*, vol. 1, pp.220-227.

[GPK02] R. Glantz, M. Pelillo and W. Kropatsch, “Matching Hierarchies of Segmentations,” *Computer Vision Winter Workshop, CVWW 02*, pp. 149-158, 2002.

[HDF99] T. Heinonen, P. Dastidar, H. Frey and H. Eskola, “Applications of MR Image Segmentation,” *International Journal of Bioelectromagnetism*, vol. 1, no. 1, pp. 35-46, 1999.

[HK04] Y. Haxhimusa and W. Kropatsch, “Segmentation Graph Hierarchies,” *Lecture Notes in Computer Science, LNCS 3138,* A. Fred, T. Caelli, R. Doing, A. Campilho, and D. de Ridder, Eds. Lisbon, Portugal: Springer, 2004, pp. 343-351.

[HL95] F. Hillier and G. Lieberman, *Introduction to Mathematical Programming, 2ed.* New York: McGraw-Hill, 1995.

[HSh85] R. Haralick and L. Shapiro, “Image Segmentation Techniques,” *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 1, pp. 100-132, 1985.

[HSt88] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” in *Proceedings of The Fourth Alvey Vision Conference,* 1988, pp. 147-151.

[IM05] ImageMagick Studio, “ImageMagick: Convert, Edit, and Compose Images,” 2005, http://www.imagemagick.org.

[Jac96] D. Jacobs, “Robust and Efficient Detection of Salient Convex Groups,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 1, pp. 23-37, 1996.

[JCH04] Y. Ji, K. Chang, C. Hung, “Efficient Edge Detection and Object Segmentation Using Gabor Filters,” in *ACM Southeast Regional Conference, ACMSE 2004*, pp. 454-459.

[JKS95] R. Jain, R. Kasturi, and B. Schunk, *Machine Vision*. New York: McGraw-Hill, 1995.

[KC97] S. Kwok and A. Constantides, “A Fast Recursive Shortest Spanning Tree,” *IEEE Transactions on Image Processing*, vol. 6, no. 2, pp. 328-332, 1997.

[Kro95] W. Kropatsch, “Building Irregular Pyramids by Dual-Graph Contraction,” in *IEE Proceedings – Vision, Image and Signal Processing*, vol. 142, no. 6, pp. 366-374. 1995.

[KS01] S. Kahn and M. Shah, “Object Based Segmentation of Video Using Colour, Motion and Spatial Information,” in *IEEE International Conference on Computer Vision and Pattern Recognition, CVPR 2001*, pp. 746-751.

[Low04] D. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[MB03] B. Maxwell and S. Brubaker, “Texture Edge Detection Using the Compass Operator,” in *Proceedings of British Machine Vision Conference, BCMV 2003*, vol. 2, pp. 549-558.

[MBS99] J. Malik, S. Belongie, J. Shi, and T. Leung, “Textons, Contours and Regions: Cue Integration in Image Segmentation,” in *Proceedings of IEEE International Conference on Computer Vision, ICCV 1999*, pp. 918-925.

[MFT01] D. Martin, C. Fowlkes, D. Tal and J. Malik, “A Database of Human Segmented Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics,” in *Proceedings of IEEE International Conference on Computer Vision, ICCV 2001*, pp. 416-423.

[MH80] D. Marr and E. Hildreth, “Theory of Edge Detection” in *Proceedings of the Royal Society of London*, 207A, pp. 187-217, 1980.

[MM00] W.-Y. Ma, B. Manjunath, “EdgeFlow: A Technique for Boundary Detection and Image Segmentation,” *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1375-1388, 2000.

[Muh02] H. Muhammed, “Unsupervised Image Segmentation Using New Neuro-Fuzzy Systems,” in *Swedish Society for Automated Image Analysis Symposium, SSAB 2002*, pp. 83-87.

[PF99] E. Pauwels and G. Frederix, “Cluster-Based Segmentation of Natural Scenes,” in *Proceedings of IEEE International Conference on Computer Vision, ICCV 1999*, pp. 997-1002.

[RT99] M. Ruzon and C. Tomasi, “Color edge detection with the compass operator,” in *Proceedings of IEEE Conferenceon Computer Vision and Pattern Recognition, CVPR 1999*, vol. 2, pp.160-165.

[RTG00] Y. Rubner, C. Tomasi, and L. Guibas, “The Earth Mover’s Distance as a Metric for Image Retrieval,” *International Journal of Computer Vision*, vol. 40, no. 2, pp 99-121, 2000.

[SB91] M. Swain and D. Ballard, “Color Indexing”, *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11-32, 1991.

[ShM97] J. Shi and J. Malik, “Normalized Cuts and Image Segmentation,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR 1997*, pp. 731-737.

[SM04] B. Sumengen and B. S. Manunath, “A Comparison of Variational Image Segmentation Methods,” 2004, http://vision.ece.ucsb.edu/~sumengen/documents/variational_eval.pdf

[Was05] M, Wasilewski, “Active Contours using Level Sets for Medical Image Segmentation,” 2005, http://www.cgl.uwaterloo.ca/~mmwasile/cs870/.

[Wei99] Y. Weiss, “Segmentation using eigenvectors: a unified view,” in *Proceedings of IEEE International Conference on Computer Vision, ICCV 1999*, pp. 975-982.

[WPR85] M. Werman, S. Peleg, and A. Rosenfeld, “A Distance Metric for Multidimensional Histograms,” *Computer Vision, Graphics, and Image Processing*, vol. 32, pp. 328-336, 1985.

[XP98] C. Xu and J. Prince, “Snakes, Shapes, and Gradient Vector Flow,” *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 359-369, 1998.

[ZL03] D. Zhang and G. Lu, “Evaluation of Similarity Measurement for Image Retrieval,” in *Proceedings of IEEE International Conference on Neural Networks and Signal Processing, ICNNSP 2003*, pp. 928-931.