INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

A HIERARCHICAL, MULTISCALE TEXTURE SEGMENTATION ALGORITHM FOR REAL-WORLD SCENES

A Dissertation

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

Doctor of Philosophy

By

VALLIAPPA LAKSHMANAN

Norman, Oklahoma

2001

UMI Number: 3028801

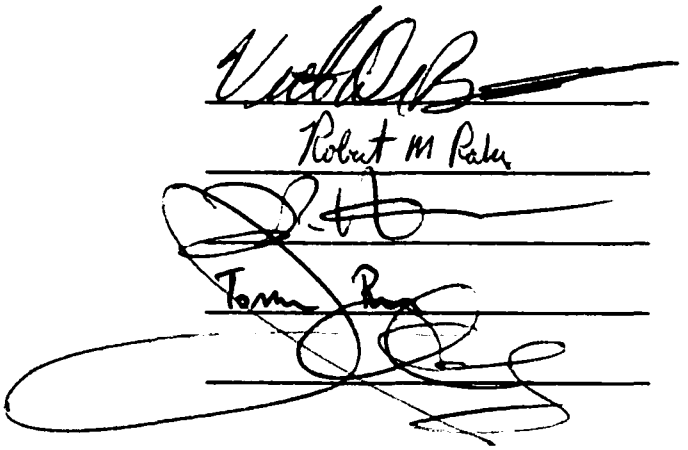
UMI Microform 3028801

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

A HIERARCHICAL, MULTISCALE TEXTURE SEGMENTATION ALGORITHM FOR REAL-WORLD SCENES

A Dissertation APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY



ACKNOWLEDGMENTS

Thanks to my parents for stressing the importance of an education and to my wife, Abi, for not-so subtly hinting that instead of twiddling my thumbs, I could work on a PhD.

Bob Rabin has been of utmost help to me in the work that led to this thesis. He patiently waited weeks between the spurts in which I could do the work, and then quickly determined the usefulness and flaws in the various approaches I implemented. He also suggested different ways of displaying the data to provide as much information as possible to the forecaster, but in as concise a way as possible. If this work leads to a useful real-time algorithm, that would be because of Bob's constant help. It was Victor DeBrunner, my advisor, who pointed me toward texture segmentation when it seemed no approach would segment satellite data properly. He was a constant source of ideas and someone I could bounce my ideas off of.

Mike Eilts, erstwhile chief of the storm-scale research division at NSSL, encouraged me to start on a PhD. Thanks to Kurt Hondl, leader of a couple of the teams I've worked in during these three years, for allowing me to pursue funding opportunities that would support my thesis work.

Different parts of the work done as part of this thesis were supported by the National Science Foundation, by the Federal Aviation Authority and by the National Severe Storms Laboratory. The segmentation of weather data was implemented and visualized within the NSSL Warning Decision Support System algorithm framework.

CONTENTS

ABSTRACT

A novel method of performing multiscale segmentation of images using texture properties is introduced. Various methods of segmentation at a single scale, including texture segmentation, are described and compared with the K-Means clustering of texture vectors used in this thesis.

We survey the state of the art in multiscale segmentation and identify some drawbacks in the traditional approach to multiscale segmentation – image pyramids and quad-tree decomposition, especially in typical applications that make use of the segmented results. We then introduce the idea of multiscale segmentation within the context of the segmented regions themselves instead of, as is traditional, working in the context of the original image. It is shown that this new multiscale approach can be incorporated into the K-Means clustering technique as a steady relaxation of inter-cluster distances.

We also develop a way of objectively evaluating texture segmentation algorithms on natural and synthetic texture patches. Finally, our multiscale segmentation approach is demonstrated on several families of real-world images. It is shown that quality of the segmented results at the different scales is significantly improved.

1. INTRODUCTION

1.1 Segmentation

Splitting an image into several components, by assigning one of these components to each pixel in the image, is termed image segmentation. The photograph (from the University of Groningen image database mentioned in [1]) in the left panel of Figure 1.1, for example, can be segmented into different components as shown in the right panel of the same figure. The components here are the windows of the building, the building itself and the sky.

A good segmentation may be recognized from the characteristics of its output components: each component should be spatially cohesive as well as spatially accurate while different components should be dissimilar [2, 3].

Many image processing algorithms and techniques lend themselves to a concept of scale – that the results of the analysis would be different if one were concerned with a different level of detail. For example, the segmentation of the photograph in Figure 1.1 should, depending on the level of detail desired, yield components that are the various windows and floors, or just two components – building and sky. Multiscale segmentation can, then, be thought of as the ability to segment a given image into different sets of components depending

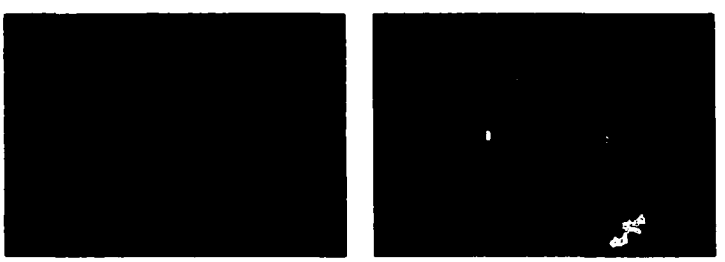


Fig. 1.1: Segmentation: The photograph of a building on the left has been segmented and the result, colored such that each component is a different color, is shown on the right.

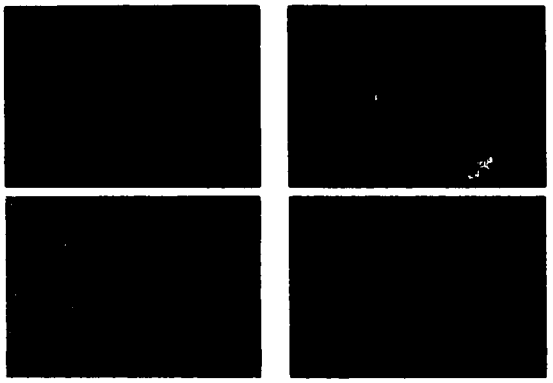


Fig. 1.2: Multiscale segmentation: The photograph of a building on the top left has been segmented using the multiscale segmentation algorithm described in this dissertationand the results at each scale, colored such that each component is a different color, are shown. Top-right is the most detailed segmentation. The second row contains, left-to-right, the segmentation at successively coarser scales.

on the scale.

The segmentation of the building photograph shown in Figure 1.2 demonstrates the results of a multiscale segmentation. At the highest level of detail (top-right panel), we find that the windows are components, but that at lower levels of detail, the window components are subsumed in the building itself.

This is, however, not the way multiscale segmentation is commonly approached. Rather, multiscale segmentation usually refers to segmentation performed on images that have been blurred to different degrees.

Traditionally, multiscale segmentation is done in one of two ways:

1. Image pyramids where wavelets or filter banks are employed to obtain the image at different scales (with the original image as the most fine resolution available). Each of these images is then segmented.

2. Quad-tree decomposition where the entire image is assumed to be a single region, then split into smaller regions, on each of which the process is repeated. Similar regions are merged at each stage.

Typically, the relationship between the segmented regions at the different scales are of no interest. If they are, then components at different scales have to be associated in some, often heuristic, manner.

1.2 Texture

In this dissertation, the word texture will be used to refer exclusively to image texture. Intuitively, texture refers to a pattern of closely placed elements, in such a manner that the pattern somehow repeats itself.

Texture has long been an active area of research in the image processing community. In 1979, Haralick wrote a survey of the main approaches to texture [4], in which he cited papers written as early as 1955 [5]. In spite of this long history of research, perhaps because texture is such an intuitive concept, there has been no universally acknowledged definition of the term. In a 1990 paper on texture, for example, Bovik [6] noted, "an exact definition of texture either as a surface property or as an image property has never been adequately formulated."

The difficulty of reaching an exact definition of the word "texture" stems from the wide variety of images in which the intuitive meaning of the term takes different meanings. A fuzzy word, then, is the cause of the common frustration as expressed by Greenspan et. al. [7]:

> Although texture analysis has been a subject of intense study by many researchers, it is as yet an open challenge to achieve a high percentage classification rate on all the above textures within one framework.

This Holy Grail is one we do not attempt to capture in this dissertation: we claim only that the approach proposed in this dissertation provides a multiscale segmentation of real-world scenes that possess a particular type of texture.

Although researchers [6] have often noted the lack of a common definition, the literature in the field has long reached a consensus on what texture is, how it can be analyzed within an image and when texture analysis is useful in image processing. Texture in the literature always refers to properties at a scale larger than that of a pixel. Most commonly, texture analysis is employed when there is significant variation between the intensities of adjacent pixels even in the absence of an "edge" between the pixels. It is recognized that there is a difference in the meaning of texture depending on the nature of the images themselves. For example, it is recognized that natural textures tend to be random but artificial textures tend to be regular and periodic [8].

Image texture analysis methods use different descriptors of texture. The descriptors used capture a differ-

ent part of the intuitive understanding of texture. Suggested descriptors include hidden Markov models [9], Markov Random Fields [10], image moments [11], co-occurrence [4] and correlation matrices [12] and filtering methods [13, 14, 15]. There exists no consensus as to which of these approaches provides the optimal texture vector. Havlicek [16] points out that several approaches ([17, 18, 19], for example) developed as a way to emulate the human visual system.

The methods that make use of statistical properties assume that the pixels in a texture region are realizations of a two-dimensional random process. Thus, statistical properties estimated within a texture region should be different from those of another texture. The statistical approach, also referred to as the stochastic approach, assumes that texture is characterized by the gray value pattern in a neighborhood surrounding the pixel [20]. Local coherence and orientation estimates [21], Gabor filters banks [22], statistics of Gabor coefficients [23, 24], amplitude envelopes of band-pass filters [25, 26] and multiple components' frequency estimates [16, 27] have been used successfully.

Since an image can be completely reconstructed from its wavelet decomposition, the texture within an image is part of this decomposition [28]. Thus, texture at different scales can be captured by the wavelet coefficients of the image at different stages of the decomposition. The working assumption here, of course, is that the variation between wavelet coefficients within the same texture region is much less than that between wavelet coefficients of different regions. In this sense, the wavelet decomposition approach to texture is simply the statistical approach, only at multiple scales.

The periodic primitive placement approach follows most directly from the intuitive understanding of texture. Its popularity derives from the ubiquitous data set of artificial pattern images [29] used in the literature. The reliable periodicity of many features in synthetic images leads directly to this approach. Commonly referred to as the structural approach, the requirement of periodicity makes it unsuitable for real-world scenes. In many real-world scenes, the texture in consideration is statistical, i.e. there are no primitives of a fractal nature within the image. Hence, in this dissertation, we will not consider the structural approach, or features such as the gray-level coöccurrence matrix that are associated with the structural approach, any further.

Texture segmentation is simply image segmentation where the labeling of pixels is based on some measure of texture. Texture segmentation algorithms, then, differ on the actual form of texture used in the measure-

ment. Structural methods rely on placement rules and will not be considered in this dissertation. That leaves feature vector-based methods, where a vector of features is computed at each pixel. The segmentation is done based on this vector.

We make use of the stochastic approach to texture in this dissertation, computing local texture vectors from statistics of the pixel's neighborhood. However, we do not segment these texture vectors directly. Instead, in this dissertation, we offer a novel way to approach the multiscale problem – from the segmented regions rather than from the spatial features of the image. To do that, we have to pick the scale at which the most detailed segmentation is done. Thus, our segmentation technique fits mostly into the stochastic approach to texture but we will show how the technique can be extended to incorporate elements of the wavelet decomposition approach as well.

1.3 Motivation: Segmentation of Satellite Infrared Images

Identifying and tracking storms is very useful for meteorological algorithms [30]. Storms thus identified and tracked may be used for visualization in a "storm-relative" sense, to study the evolution of storms, project storm movement in a short time period [31] and as inputs to feature detection algorithms [32]. For storm tracking to be useful, it should satisfy a few requirements:

1. The identification and tracking algorithm should be completely automated. A few tracking algorithms require some environmental parameters to be updated daily, but in any case, algorithms that require user intervention any more often would be ignored in an operational context.

2. The identification algorithm should not require training, i.e. to expect to see examples of all the "objects" it should identify.

3. Storm "cells" (small scale features) should be capable of being identified.

4. The identification and tracking scheme should be robust across frames, by permitting association of regions in one frame with identified regions in the next.

5. Identified storms may split or merge in the time to come; the tracking scheme should be capable of

handling these cases.

6. Because the notion of scale is natural in the storm tracking context, we would like to add the requirement that storms at various scales be identified, with their hierarchical structure intact. A multiscale tracking algorithm would be a significant improvement over current tracking schemes which concentrate either on small scales [30] or on large scales [33].

Successful storm cell identification and tracking schemes have been implemented for radar reflectivity data [30], and for mesoscale (relatively large) convective systems using the infrared channel of satellite data [34]. There have been no successful approaches when it comes to identifying and tracking storms at finer scales than the mesoscale on satellite infrared imagery.

Satellite infrared imagery provides a different view of the storm from that provided by the national weather radar network. The view is of the storm tops, something which ground-based radar miss. Radar imagery also has to be mosaic-ed to provide a larger geographic coverage while it is natural in the satellite context. Tracking on satellite can, therefore, be carried out over a large geographic area without any regard to inter-sensor differences of mosaic-ed fields. Thus, any successful segmentation technique that is useful for identifying storms at small scales from infrared imagery would be a useful contribution to the applied meteorology community. A multiscale segmentation algorithm for weather imagery, whether radar reflectivity or satellite infrared, would be an advance over the current methods.

1.4 Thesis Objective

In the work that led up to this dissertation, our aim was to develop a method of multiscale segmentation that would yield a hierarchically arranged tree such that the relationships between regions at different levels of the segmentation would be explicit. Such a multiscale segmentation should be useful for tracking regions in a sequence of images, and therefore should be robust to minor changes in the regions themselves.

In this dissertation, we will detail our approach to multiscale segmentation in the context of the segmented regions themselves and show a natural texture-vector implementation using K-means clustering, region growing and inter-cluster differences. We will also introduce an objective way to measure the performance of

texture segmentation algorithms. Finally, we will demonstrate the results of using this multiscale method on synthetic as well as on real-world scenes.

1.5 Organization of Thesis

The rest of this dissertation is organized as follows. In Chapter 2, we describe some prominent segmentation techniques, including the major approaches to texture segmentation. Multiscale texture segmentation techniques are presented and some of the problems inherent in these techniques identified. The concept of multiscale segmentation in the context of the segmented results, rather than in the context of the segmentation input, is introduced.

In Chapter 3, the method proposed in this dissertation is described in detail. In Chapter 4, synthetic and real-word images, from the image processing literature as well from weather data, are segmented using the techniques described in Chapter 2 and using the method described in this dissertation. In Chapter 5, various parameter choices made in the algorithm as described in Chapter 3 are identified and the effects of varying these parameters as well as the effect of using higher resolution images is explored.

These results are summarized, and future extensions to this research proposed in Chapter 6. In Appendix A, a way of tracking multiscale segmented outputs is described. The hierarchical segmentation approach described in this dissertation is applied to satellite and radar weather images in Appendix B, and the hierarchical tree is used to track storms in a robust manner.

2. SEGMENTATION ALGORITHMS

Segmentation algorithms come in two flavors – supervised and unsupervised. Since our goal is to develop a completely automated algorithm, we will consider only the unsupervised algorithms here.

2.1 Single Scale Segmentation

Traditionally, the underlying techniques for decomposing an image into its components include amplitude thresholding, connected-set labeling, contour and edge-based methods and template matching.

Amplitude thresholding is a simple method that works whenever the pixel values themselves differentiate the different components. The most common amplitude thresholding image segmentation algorithms decompose images into just two components, i.e. they are binary techniques. Whether or not the segmentation is binary, the choice of appropriate amplitude thresholds is critical. Often, the choice is made based on examination of the histogram (for gray-level values with few pixels, but either side of which lie many pixels) or through a probabilistic model of the different objects' gray-level values.

The gray level histogram of the image of a wound shown in Figure 2.1a is shown in Figure 2.2. Since there is a shallow valley between the peaks in the histogram corresponding to the wound and to the undamaged skin i.e. since the histogram is a bimodal distribution, it is possible to threshold the image so that pixels below a certain threshold are said to belong to a wound (See Figure 2.1b).

Contour or edge-based methods attempt to find the boundaries of the components based usually on the gradient of an image and then "fill" in the boundaries with the components (See Figure 2.3).

It is also possible to segment images of controlled scenes by matching the components against a dictionary of known components and extracting these templates.

Fig. 2.1: Amplitude Segmentation. Left to right: (a) Photograph of a leg wound, from [35]. (b) The wound image segmented so that all pixels below a certain threshold are said to belong to the wound. In this image, all black pixels that are part of the leg are said to be a wound.

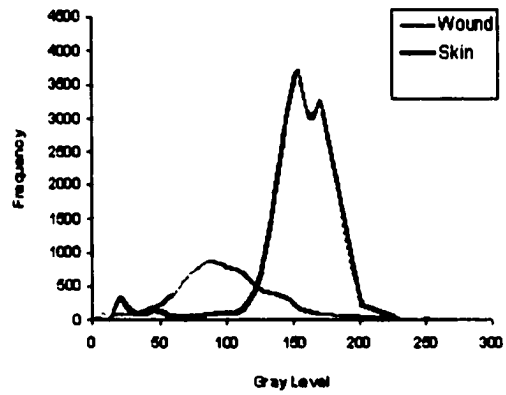


Fig. 2.2: Amplitude Segmentation. The gray-level histogram of the pixels in the photograph of wound in Figure 2.1a. Figure based on that of [35].

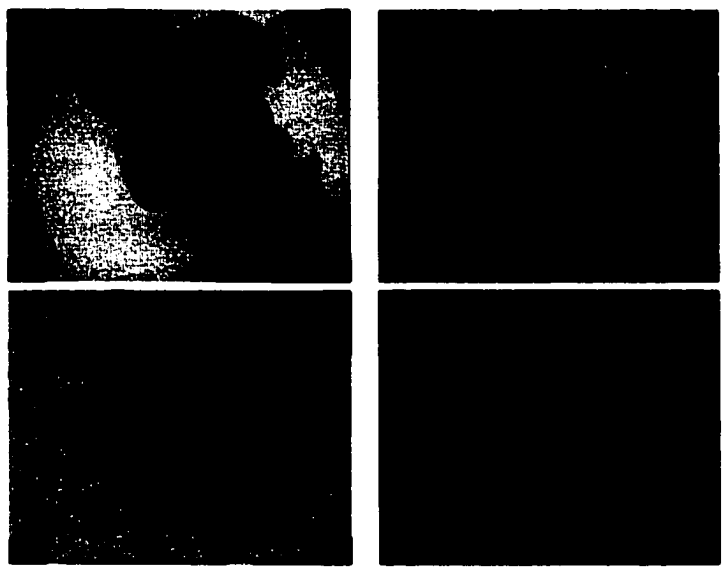


Fig. 2.3: Contour-based segmentation, from [36]. Left to right, top to bottom: (a) Image of fruits (b) Edges detected in image (c) Salient edges chosen and closed based on known models. (d) Models extracted from the image.

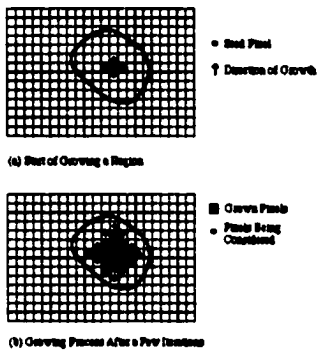


Fig. 2.4: Region growing: When an unlabeled pixel is encountered, the next possible label is assigned to that pixel. Every neighbor of that pixel (a 4-neighborhood here) which should belong to the same region as that pixel is also associated with the new label. This process is carried out recursively until there are no neighbors who should belong to the same region. Figure from [38].

2.1.1 Region-based methods

Connected set labeling assumes that connected pixels with similar characteristics belong to the same component. Run-length analysis may be used to obtain a hierarchical structure among the segmented components [8] while region-growing methods divide an image into connected components based on the pixel values [37].

Having identified pixels that belong to various objects, regions can be built by labeling connected components. This can be done recursively by stepping through an image. When an unlabeled pixel is encountered, the next possible label is assigned to that pixel (See Figure 2.4). Every neighbor of that pixel which should belong to the same region as that pixel is also associated with the new label. This process is carried out recursively until there are no neighbors who should belong to the same region. This process is termed region growing, especially in combination with a process of merging adjacent regions that have sufficiently close characteristics. Merging heuristics might compare the number of weak boundary pixels that lie in the border of the two regions to the total number of boundary pixels of the smaller region or to the total number of pixels on the common boundary of the two regions [8]. The merging heuristic used in the method of this dissertation is to merge two regions if they are sufficiently similar in their texture characteristics, i.e. merge labels i and j if the mean characteristic of the regions, μ_i and μ_j, are such that:

$$\| \mu_i - \mu_j \| < \epsilon \tag{2.1}$$

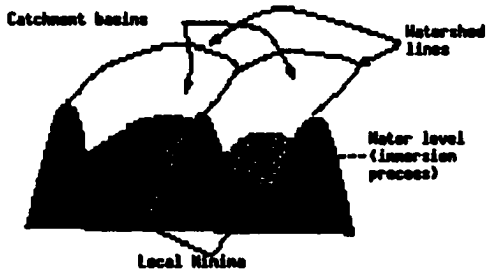


Fig. 2.5: The watershed algorithm treats an image as a topographic relief, with the gray level at a pixel standing for the elevation of that point. Watershed lines separate catchment basins and thus segment the image. Figure based on that of [40].

The choice of ϵ is made based on the global characteristics of the currently labeled regions, thus yielding an iterative, hierarchical scheme.

Region-based methods are, in general, less sensitive to noise although the implementation of a region-based method is generally more complex [8]. Boundary-based methods are easier to implement, but are more sensitive to noise. In boundary-based methods, edges are detected in the image and these edges are used to follow boundaries and yield segmented regions. Very often, the segmented regions are not the main goal here – the extracted boundaries are smoothed or fit to curves and used in classification and analysis.

The method of texture segmentation described in this dissertation makes use of both amplitude thresholding and region growing. Amplitude thresholding is utilized to obtain an initial set of clusters to operate upon, although in this case, the amplitude is really the textural feature vector. Connected-set labeling and region-growing are used after clustering at each stage to obtain the component regions.

2.1.2 *Watershed Segmentation*

In watershed segmentation, images are considered to be topographic reliefs, with the gray level at a pixel standing for the elevation of that point. Watershed segmentation is the process of finding watershed lines which separate "catchment basins". A catchment basin is associated with a minimum, such that a drop of water dropped at any pixel in the basin would flow toward that minimum – see Figure 2.5. The automated identification of these catchment basins provides a segmentation of the image [39].

A straight-forward implementation of the watershed technique, introduced in [40, 41], is to simulate an

immersion process. Assume that we have small openings at all local minima within the image, and that we slowly immerse the topographic representation into a lake. The water will slowly rise up through the various minima. At every pixel where water from two different minima merge, a watershed line is drawn (See Figure 2.5). The set of all the watershed lines drawn by the time the complete image has been immersed completely divide up the image into catchment basins, which are the segmented regions.

In the comparisons of the various segmentation methods in Chapter 6, the efficient method of [40] was used to perform the watershed segmentations. This method implements the immersion methodology using breadth-first scannings of the plateaus in a sorted list of the gray levels in an image, enabled by a queue structure. The pixels in the image are sorted in increasing order of gray scale values. The computation of catchment basins is then done by scanning the entire list for pixels that would be flooded at a particular level. This scanning – the flooding step – is implemented in [40] by a queue of pixels, leading to an efficient implementation.

In practice, watershed segmentation performs well only on images where the basins are relatively homogeneous. A highly textured surface causes watershed lines to be drawn every where within an image. Two possible workarounds exist – to smooth the images before doing the watershed segmentation or to reject watershed lines with low "saliency", i.e. those for which one of the basins is not deep enough.

2.2 Texture Segmentation

The underlying assumption in texture segmentation is that the different regions that are the result of segmentation possess different textural attributes. The first step is often to compute the textural attributes at various points in the image. It is these textural attributes, often in conjunction with original pixel values within the image, that are used to segment the image into regions.

The various texture segmentation techniques in the literature differ both in the way the texture attributes are computed and in the way the texture attributes are utilized to obtain the region segmentation. Texture attributes may be computed using a filter bank, as in the Gabor filter approach, using the idea of cliques as in the Markov Random Field (MRF) approaches or using neighborhood statistics.

Given a texture vector at each location, the segmentation into regions may be performed using unsupervised clustering [13], iterative propagation [42], texture classification [43], or statistical tests [44].

In this section, we will briefly look at each of these approaches.

2.2.1 Gabor Filters

A Gabor filter is a Gaussian filter that has been modulated by a sinusoid. Thus, Gabor filters are determined by their scale which is the standard deviation of the Gaussian and their orientation which is the frequency of the sinusoid. A Gabor filter set with three different scales and eight different orientations is shown in Figure 2.6. The amplitudes of the results of local filtering using a set of Gabor filters is considered a feature vector [13, 42] and segmentation is performed using these feature vectors. By choosing the standard deviation of the Gaussian kernel whose orientations define the feature vector, the scale at which the regions are identified may be chosen.

A coarse image segmentation based on local texture gradient may be performed by extracting Gabor texture features from image tiles [42]. Using these texture vectors, a local texture gradient may be computed within the eight-neighborhood of a pixel. Then, pixels where the gradients possess substantially the same direction are said to belong to the same region. Such pixels, all of which are in similar directions, give rise to a texture "flow". A Markov assumption, that the texture flow is the same at neighboring pixels, is imposed on the field and the flow propagated to neighboring pixels unless the directions of the flows at the two pixels are in opposite directions (greater than 90 degrees). This flow propagation is performed until the segmentation becomes stable.

In [42], the edges were identified as those pixels in between regions with opposite gradients. After a region merging step, the resulting regions are assigned different labels.

2.2.2 Markov Random Field

Markov Random Fields (MRFs) are efficient in texture modeling, classification and segmentation [45, 46, 47]. MRF-based texture segmentation algorithms are popular because they provide a simple way of evaluating the efficiency of the segmentation – generate a patch of MRFs and run the segmentation algorithm. This neatly

Fig. 2.6: A Gabor filter set with three different scales and eight different orientations is shown. The scales are set by changing the standard deviation of the Gaussian function, while the orientations are obtained by changing the angle of the sinusoid.

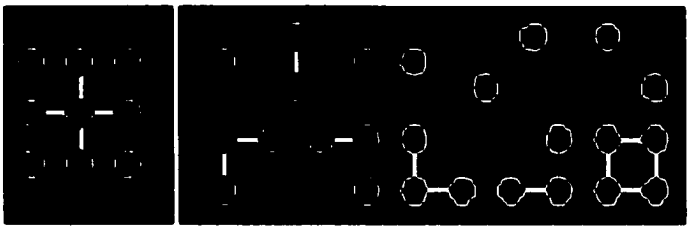


Fig. 2.7: Cliques in the MRF approach: Left to right: (a) An eight neighborhood system (b) The cliques associated with a second order MRF model with a eight-neighborhood. Spatial dependencies characterizing the texture are captured in potential functions associated with these cliques. Figure based on that of [49].

side-steps the question of whether the MRF model captures the intrinsic texture in the image.

However, MRFs in general yield local and efficient texture descriptions [45] on a wide variety of images. The most popular MRF models use "cliques". A clique is a set consisting of the pixel in question and zero or more of its neighbors. For example, in a second order MRF model (which corresponds to a 3x3 neighborhood), there are ten possible clique types – the pixel itself, four clique types with two elements (one horizontally aligned, one vertically aligned, one aligned at 45° and the last aligned at −45°), four clique types with three pixels and one clique type with four pixels – see Figure 2.7. Spatial dependencies characterizing the texture are captured in potential functions that are associated with the cliques. The generalized Ising model [48] assigns non-zero potentials only to the four clique types that have two elements each.

MRF models have been very successful in a supervised context [45]. Unsupervised methods proposed (for example, [45, 50]) involve one or more of these assumptions:

1. The image can be histogram quantified into a small number of gray levels.
2. The number of different textured regions in the image is known.
3. The image can be divided into mostly homogeneous blocks so that texture parameters can be estimated on the blocks.

2.2.3 *Kolmogorov-Smirnov Test*

Various texture statistics are computed in local neighborhoods about a pixel and the vector of measurements is assigned to that pixel. Texture segmentation has been shown to be improved [51, 4] by the use of coöccurrence matrices. So, commonly, the vector of measurements includes both neighborhood statistics, such as mean,

variance and kurtosis, and coöccurrence-based measurements. Other measurements commonly included are the results of filtering the image, such as by a Gabor filter. The segmentation algorithm can then segment the image based on the vector of measurements associated with each pixel rather than simply based on the pixel value.

The Kolmogorov-Smirnov (KS) test can be used to test the hypothesis that an observed distribution function of independent random variables belongs to a specified distribution function at varying confidence levels. This is done by computing the critical value, c, corresponding to the significance level, α, in the observed function from the series expansion: [52]

$$\alpha = 2\sum_{j=1}^{\infty}(-1)^{j+1}e^{-2j^2c^2} \tag{2.2}$$

The KS distance,d, is computed as the maximum of the distances between the cumulative frequency of the observed random variables and that of the specified distribution. Given the KS distance, the KS critical value and the number of samples,n, used to obtain the observed cumulative distribution function, the observations are said to belong to the specified distribution if $d < c/\sqrt{n}$. Unlike other measures of fit such as the chi-square test, the KS test works even if the distributions are not normal.

The problem of segmenting an image based on the local distribution of features can be cast into a KS test [44]. Assume that the segmentation has been initialized in some manner. Then, we can compute the distribution of the features in the pixels within each region. We can then test whether the "local" distribution of these features around a pixel is part of the "global" distribution of these features within a region. If it is part of the global distribution (at the significance assumed), then, this pixel can be added to the region and the global distribution of the region updated.

A separate class of outliers [53, 54, 44, 55] may be maintained. If the KS distance between the distribution at a pixel is too far from the distributions of the current regions, then that pixel is labeled an outlier. Contiguous outliers are combined into new regions, removing the need for the number of regions to be known a priori.

Finding the optimal segmentation is carried in an iterative manner. During an iteration, each pixel of the

image is tested with the labels of its neighbors. Testing a label, p, involves computing the "energy" as the weighted sum of two components

$$U = U_{ks}(p) + 0.5 * U_m(p) \tag{2.3}$$

and replacing the current label with the label p if the energy corresponding to p is less than that corresponding to the current label. The energy, U_{ks} is the energy corresponding to the KS test and is defined as the average result of the KS test applied to each texture feature (with the test returning 1 if it is not a part of the region p and -1 if the pixel is part of the region). The energy, U_m is the energy corresponding to the Markov model and is defined as the average result of a test that returns one if an 8-neighbor is not p and returns -1 if an 8-neighbor has the label p. Since U_{ks} can not be computed if p corresponds to the outlier class (there is no common distribution), U_{ks} is taken to be a constant value for the outlier class. This constant is steadily increased with each iteration, thus slowly discouraging the formation of new outlier regions at later stages of the relaxation. In [55], it was set to be zero on the first iteration and increased by 0.1 each time around.

At the end of each iteration, contiguous outlier pixels are combined and labeled as new regions if they are large enough. This test of size allows the formation of reasonably valid probability distributions [44, 55]. The global distributions of each region are then recomputed (to perform the KS test on the next iteration) and the next iteration started. This is continued until the segmentation is stable. Convergence happened on the set of images considered after 6-8 iterations. At the end of each iteration, contiguous outlier pixels are combined and labeled as new regions if they are large enough. The global distributions of each region are then recomputed (to perform the KS test on the next iteration) and the next iteration started. This is continued until the segmentation is stable. Convergence happened on the set of images considered after 6-8 iterations.

The initial segmentation map is critical since the KS energy depends on the global distribution of the first set of regions [55]. The quality of the segmentation is, to a large extent, dictated by the choice of initial condition. The initialization can be done in one of several ways [55], including:

1. Using only the KS distance after labeling the entire image as one region. Outliers from the distribution of features in the entire image will then be combined in the next iteration of the relaxation algorithm into new regions.

2. Labeling as outliers pixels the majority of whose features do not fall in the most frequent interval of that feature's probability distribution. The other pixels are labeled as a single region.

3. Labeling pixels that have extreme values in the mean (say the top 5% and the bottom 5%) within the image as outliers and the rest of the pixels as a single region.

The first choice is consistent with the rest of the algorithm and was found to work on a wide variety of images [44]. However, the initial labeling varies widely across a sequence of images with small changes from frame to frame [55]. When initialized using pixels that do not lie in the modal interval, the segmentation is robust and only the strongest part of the storms is retained [55]. The third method of initialization was introduced in [55] and found to perform the best, so the results demonstrated in Chapter 6 follow the third method of initialization described above.

2.2.4 K-Means Clustering

K-Means clustering is a clustering technique where the clustering proceeds by computing the affinity of each entity with each of K clusters that already exist. The entity is assigned to the cluster to which it is closest in the measurement space. The means are updated and the process is repeated for the next entity. The entities are cycled through until no entities are moved between clusters. One way to think of this process is this: the entities are represented by a vector of measurements and the K cluster means represent K "representative" vectors in the measurement space. K-Means clustering is then a way of choosing the best possible set of K vectors from a large number of measurements. Thus, traditional K-Means clustering may be thought of as a vector quantization technique where the vectors are chosen adaptively from the image itself. Once the vectors have been chosen, then each pixel can be assigned the vector that it is closest to. In this sense, then, K-Means clustering can be used to requantize an image.

In the discussion that follows, we will refer to the K representative vectors as the cluster means and to the set of measurements taken at each pixel as the texture vectors. There are K cluster means during the process, hence the name of the technique.

The K-Means clustering technique assumes that an initial guess for the cluster means is in place. The

initial guess may be arrived at by dividing up the measurement space into K vectors, or by choosing K pixels at random from the image and using their texture vectors as the cluster means. With this initialization, the K-means iteration is commenced. Thus, K which is the number of clusters in the image has to be known a priori.

In each iteration, every pixel in the image is considered. The label of the cluster mean closest to the pixel's texture vector is assigned to that pixel. After all the pixels in the image have been assigned labels, the cluster means are updated with the mean of the texture vectors of the pixels currently assigned to that cluster. Iterations continue until none of the labels are changed in that iteration. Implementations often terminate the iterations when fewer than 1% of the pixels change labels, to avoid oscillations.

The traditional K-means clustering method uses either the Hamming or the Euclidean distance between the cluster mean and texture vector to determine the distance [56]. However, the traditional method does not take into account the spatial characteristics of the data. Pixels that are spatially contiguous are likely to belong to the same class. K-Means clustering of images is improved if the choice of the vectors, as well as the association of the pixels to a cluster is done after incorporating a contiguity constraint [57], so as to minimize the global energy:

$$E = \lambda D + (1 - \lambda)V \qquad 0 \leq \lambda \leq 1 \tag{2.4}$$

where D is the number of contiguous pixel-pairs in the entire image that have different labels and V is the normalized cluster variance:

$$V = \frac{\sum_{xy} \| T_{xy} - \mu_{S_{xy}} \|^2}{\sum_{xy} \| T_{xy} - \overline{T_{xy}} \|^2} \tag{2.5}$$

T_{xy} is the texture vector at pixel (x, y), S_{xy} the label at the pixel and μ_i the cluster mean of the ith label.

Instead of using global cluster compactness as a measure of distance, we used a local distance measure, but incorporated a Markov assumption for spatial contiguity to form the update rule (See Equation 3.13).

2.2.5 *Optimization Approaches*

When a priori knowledge of the objects in a scene is available, optimization approaches can utilized to find boundaries in the image. This optimization may be performed in the parameter space by modeling the objects

through templates [58, 59] and searching the parameter space for the best fit. The optimization may also be performed in the image space, with texture vectors as the parameters drawn from the image [60]. Where boundaries are of interest, boundary primitives called snakes may be used to cast segmentation as an energy minimization problem [61].

2.3 Multiscale Approaches

The current multiscale techniques approach the multiscale problem from the point of view of the image. It is the image or the texture vector computed from the image that captures the scale. In the image pyramid approach, the images are blurred to different degrees; thus, the blurred images capture scale. In the quad-tree approach, texture vectors are associated with groups of pixels of different sizes. Thus, the texture vectors capture scale.

We argue that a better way of approaching multiscale segmentation is to incorporate scale into the result of the segmentation – into the components themselves. Thus, we will be using only a single image (instead of a set of blurred images) and a single set of texture vectors (computed at each pixel in the image) in our segmentation. We will then arrange the segmented regions in a hierarchical manner – it is the result of our segmentation that incorporates scale. We show, in Section 5.3.1, that it is possible to integrate scale in the traditional sense into our hierarchical segmentation approach, although as we do in this dissertation, it is possible to achieve a hierarchical tree representation using only a single image and a single set of texture vectors.

2.3.1 Image Pyramids

In the image pyramid approach, images are filtered to yield a set of images arranged according to the amount to which the original image was blurred. The most blurred images are used to achieve the coarsest segmentation. Having obtained the different segmentations, the regions in each of the images are associated – this is a matching across scale problem. It is considerably difficult, because the computed textures have been computed on images blurred to different degrees. An adaptive scheme is often used (See Figure 2.8) to do this heuristic association.

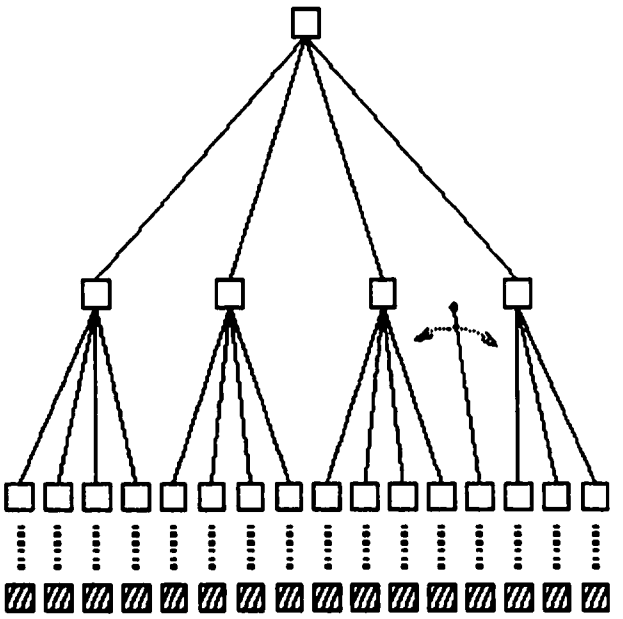


Fig. 2.8: Adaptive pyramid: Depending on the texture characteristics, objects in the detailed segmentation are associated with different objects in the coarse segmentation. Links are moved until a global cost function is minimized. Figure from [62].

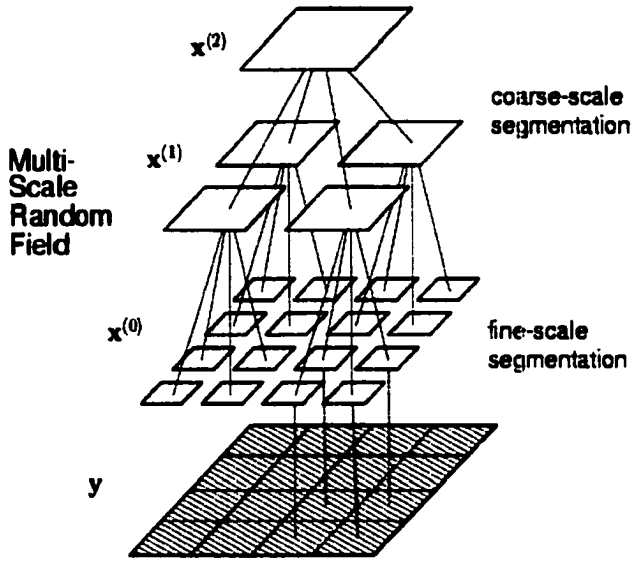


Fig. 2.9: Quad-tree decomposition: In each stage, groups of four quads from the previous grouping are grouped together. The segmentation is performed on different texture vectors, with each texture vector being computed within a quad. Figure from [62].

Obtaining image representations at the different scales required may be done in many ways – Gabor filters, wavelets and morphological open/close operations are often used.

2.3.2 *Quad-tree Decomposition*

In the image pyramid approach, the images are successively blurred and the segmentation is done on the blurred images using texture vectors computed on the blurred images. In the quad-tree approach, the image's pixels are grouped in fours in consecutive stages and texture vectors computed for each of the groups. At each scale, segmentation is performed using the texture vectors associated with the various groups (See Figure 2.9).

2.3.3 *Problems with the traditional approaches*

As image processing algorithms, both the traditional approaches to multiscale segmentation – quad-trees and image pyramids – provide acceptable performance. On a sequence of images, however, there are problems associated with both these approaches.

In the image pyramid approach, images filtered so as to choose different scales are segmented separately

and the regions are then associated across scale [63, 64]. This yields a pyramid (or hierarchical tree) of regions that represents the regions in the image. This method requires that texture vectors computed at different scales be compared for the purposes of matching regions across scale. Matching across scale is a difficult problem and the adaptive schemes that are often used are very sensitive. Thus, even small variations in the images of a sequence can cause the hierarchical tree that is built to look very different. In a tracking problem, this sensitivity causes problems of associating regions across time.

The quad-tree decomposition approach, because it uses only a single set of texture vectors, is less sensitive to small variations in the image when matching across scale [65, 66]. However, the structured manner in which pixels are considered (see Figure 2.9) poses limitation on the object boundaries. On real-world images, objects that span multiple scales may fall into more than one quad-tree region, and the quad-tree representation forces a choice of which region is represented by the image. The technique is well-suited for a patch of Brodatz textures, however, and it is on such applications that the method is reported most often.

The hierarchical multiscale segmentation approach we propose uses only a single set of texture vectors, thus providing a way of matching across scale that is not sensitive to small variations in the images of a sequence. However, the pixels are considered not structurally, but adaptively as in the image pyramid approach.

3. A HIERARCHICAL CLUSTERING ALGORITHM

Clustering algorithms try to find structure in data, to find a convenient and valid organization of the data. A cluster is commonly understood to mean a set of entities that are alike, such that entities from different clusters are not alike [67, 68].

Clustering methods require that a measure of affinity can be computed between any pair of entities. In image segmentation, the entities in question are the pixels of the image to be segmented. Thus, a measure of affinity between any pair of pixels needs to be defined. A common measure of affinity is the Euclidean distance between the pixels in a pattern space. The data may be binary, discrete or continuous. In digital image processing of gray-level images, the data are discrete, but not binary. The affinity measure, based as it is on the image data, is usually also discrete.

A hierarchical clustering method is a procedure for transforming an affinity matrix into a sequence of partitions that are nested [67]. In clustering, a set of objects is partitioned based on how similar the objects are to one another (given by the affinity matrix). Hierarchical clustering is a multi-step process where the set of partitions from the previous step is used to form the partitions at the current step. The end result is a sequence of partitions such that each partition is nested into the next partition in the sequence. This can be done in two ways: (a) *agglomerative* where the clustering algorithm at each stage merges two or more trivial clusters, thus nesting the earlier partition into a smaller number of clusters and (b) *divisive* where the clustering algorithm at each stage divides up the current clusters into smaller clusters. The result of clustering is often represented by a dendrogram where layers of nodes each represent a cluster. The method proposed in this dissertation is an agglomerative hierarchical clustering algorithm.

3.1 Hierarchical K-Means Clustering of Texture Vectors

Images are segmented using an iterative texture segmentation method that yields a hierarchical representation of the regions at different scales. We drew on existing segmentation approaches that used K-means clustering on texture vectors [58, 69, 57, 70] but our use of inter-cluster distances leads naturally to a hierarchically arranged tree of identified regions. We follow the K-means clustering stage with a region growing step where connected pixels belonging to the same cluster are labeled the same. This ensures that we need not know the number of textures before hand, since K is only an intermediate step to finding the number of regions in the image. A final step uses the Euclidean distance between the mean texture vectors of statistically unsound regions and their statistically sound neighbors to yield a robust segmentation at the finest level of detail.

Having obtained the most detailed segmentation, we use the available inter-cluster distance measurements between every pair of adjoining regions in a dyadic manner to successively merge regions. Thus, the merge is done based purely on texture characteristics.

3.2 Texture Vector Computation

A vector of measurements taken in the neighborhood of a pixel is associated with that pixel. There is little consensus as to which set of measurements are optimal, or what the standards for the choice of measurements should be. Descriptors based on statistical, structural and spectral properties of images have been utilized to form sets of discriminant features. Suggested descriptors include neighborhood statistics [71, 55], hidden Markov models [9], Markov Random Fields [10], image moments [11], co-occurrence and correlation matrices [12] and filtering methods [13]. Since there exists no consensus as to which of these approaches provides the optimal texture vector, iterative feature extraction algorithms [72] have been devised to choose the basis and to reduce the dimensionality of the texture vectors. The suggested approach, then, is to include as many characteristics as possible and to utilize a dimensionality reduction algorithm to choose the characteristics that will actually be used in the segmentation.

Since we are utilizing the texture vectors to separate the various clusters, we could utilize only those components that simultaneously maximize inter-cluster distance and minimize the within-cluster distance.

Following [73], we can define the inter-cluster distance as:

$$d = \sum_{i=1}^{K} \frac{N_i}{N} \| \mu_i - \mu \| \tag{3.1}$$

and the within-cluster distance of the i^{th} cluster as:

$$d_i = \sum_{j=1}^{N_i} \| \mu_i - v_j \| \tag{3.2}$$

and define a positive cost function that combines d and d_i, for example,

$$J = \frac{d}{\sum_{i=1}^{K} \frac{N_i}{N} d_i} \tag{3.3}$$

where N_i is the number of pixels in the i^{th} cluster, and μ_i the mean texture vector of that cluster. The value v_j is the texture value at the j^{th} pixel. The values N and μ represent the total number of pixels and the mean texture value over the entire image. Among the choices possible for the texture vectors, we should choose the texture vector set that has the highest J.

Texture is defined, not at the pixel level, but over the neighborhood of a pixel. The size of the neighborhood depends on the texture under consideration. Within an image, there may be a variety of textures with different extents. One common solution to this problem is to choose a number of neighborhood sizes and to compute the texture statistics in all the neighborhoods, thus encompassing a variety of textures [74]. In Chapter 6, results are demonstrated with texture vectors computed within a fixed neighborhood, a $7x7$ one, of the pixel. We will explore the effects of neighborhood size in Chapter 5.

Fisher's multiple linear discriminant functions [75] are the optimal solution to the problem of choosing the best set of texture vectors and the right size of neighborhood to compute the texture vectors. However, this requires knowledge of the means and covariance matrices of the different image regions. An iterative technique that reduces the feature set based on matrices computed at different resolutions (neighborhood sizes) can provide a reasonable approximation to the Fisher optima [76].

Thus, the optimal choice of texture vectors and neighborhood sizes is image dependent. The right choice

should be made for the application by computing a variety of statistics at various neighborhood sizes. Then, using either the within- and between-cluster statistics of [73] or the iterative techniques of [72, 76], a subset of those features should be chosen.

In this work, we describe an untrained segmentation technique that can be used regardless of which texture vector is optimal. We used a single set of texture vectors for all the results discussed in this dissertation. With a set of texture vectors that is tuned to the image being segmented (rather than the general purpose set used here), results may be improved. The following are the components of the vector T_{xy}, the texture vector of the image at the pixel (x, y), that are used to generate the results described in Chapter 6.

1. Grey-level value of the pixel, I_{xy}.

2. Mean in the neighborhood of the pixel, $\overline{I_{xy}}$, defined as:

$$\overline{I_{xy}} = \frac{\sum_{i \epsilon N_{xy}} I_i}{card(N_{xy})} \tag{3.4}$$

3. Standard deviation in the neighborhood of the pixel, defined as:

$$s_{xy} = \sqrt{\frac{\sum_{i \epsilon N_{xy}} (I_i - \overline{I_{xy}})^2}{card(N_{xy})}} \tag{3.5}$$

4. Coefficient of variation, computed as:

$$cvar_{xy} = \frac{s_{xy}}{\overline{I_{xy}}} \tag{3.6}$$

5. Skew

$$skew_{xy} = \frac{\sum_{i \epsilon N_{xy}} (\frac{I_{xy} - I_i}{s_{xy}})^3}{card(N_{xy})} \tag{3.7}$$

6. Kurtosis

$$kurt_{xy} = \frac{\sum_{i \epsilon N_{xy}} (\frac{I_{xy} - I_i}{s_{xy}})^4}{card(N_{xy})} \tag{3.8}$$

7. Contrast

$$contrast_{xy} = \frac{\sum_{i \epsilon N_{xy}} (\frac{I_{xy} - I_i}{I_{xy}})^2}{card(N_{xy})} \tag{3.9}$$

8. Homogeneity

$$hom_{xy} = \frac{\sum_{i \epsilon N_{xy}} \frac{1}{1+(\frac{I_{xy}-I_i}{I_{xy}})^2}}{card(N_{xy}) - 1} \tag{3.10}$$

where N_{xy} is the set of pixels in the 7x7 neighborhood of the pixel at (x, y) in the image. The effect of the size of the neighborhood, as well as the choice of all or a subset of these components, will be explored in Chapter 5. Since the results of segmentation are influenced by the choices of the texture parameters, a few of these measurements were omitted for the weather data sets. For example, in satellite images, we omitted skew, kurtosis and contrast.

Although we use a fixed size neighborhood on only the original image to compute the texture statistics, it is possible to compute these statistics on images at different scales and use these different texture vectors in the hierarchical segmentation process (See Section 5.3.1).

3.3 K-Means Clustering

Using the texture vectors associated with every pixel in the image, the images were requantized to a fixed number of levels using K-Means clustering. It should be emphasized that this fixed number of levels ("K" in the K-means clustering) is not the number of regions in the resulting segmentation. It is the number of levels into which the image is requantized. The requantization is an iterative process that makes use of K-Means clustering to partition the image values into the K bins. We will explore the effect of K further in Chapter 5.

The measurement space (the gray level of the images) was divided up into K equal intervals and each pixel was initially assigned to the interval in which its gray level value lay. A Markov assumption, that a pixel belongs to the same interval as its neighbors, was imposed. In each iteration, the best label for each pixel in the image was chosen based on a cost factor that incorporated two measures. The first measure is the Euclidean distance, $d_m(k)$, between the texture vector at that pixel and the cluster mean of the candidate k, given by:

$$d_m(k) = \| \mu_k^n - T_{xy} \| \tag{3.11}$$

where μ_k^n is the cluster mean of the k^{th} cluster at the n^{th} iteration and T_{xy} the texture vector at the pixel

(x, y). The second measure is a contiguity measure, $d_c(k)$, that measures the number of neighbors whose labels differed from the candidate label k. We can formally express the distance $d_c(k)$ as:

$$d_c(k) = \sum_{ij \epsilon N_{xy}} (1 - \delta(S_{ij}^n - k)) \tag{3.12}$$

where S_{ij}^n is the label of the pixel (i, j) at the n^{th} iteration and N_{xy} is the set of 8-neighbors of the pixel (x, y). Then the choice of the label for the pixel (x, y) in the $(n + 1)^{th}$ iteration, S_{xy}^{n+1}, is given by the label $k \epsilon S_{N_{xy}}^n$ for which the energy, $E(k)$. given by

$$E(k) = \lambda d_m(k) + (1 - \lambda) d_c(k) \qquad 0 \leq \lambda \leq 1 \tag{3.13}$$

is minimum. We used $\lambda = 0.6$ for all the images, but will explore the effect of this parameter in Chapter 5. The candidates that were considered were the labels at the n^{th} iteration of the pixels within the 8-neighborhood of (x, y). At the end of each iteration, the cluster attributes (the μ_k's) were updated based on all the pixels that were labeled as belonging to the cluster at that time.

The requantization, then, consists of these steps:

1. Initialize the K means somehow. The optimal initialization is image dependent [55]. Here, we simply divided up the measurement space into equal intervals.
2. Assign the closest mean to each pixel.
3. Step through the image and at each pixel,
   (a) Take as candidates all the labels in the 8-neighborhood of the pixel.
   (b) Compute the contiguity distance $d_c(k)$ that would result if the label were changed to k.
   (c) Compute the distance between the mean of the kth cluster and the pixel's texture vector, $d_m(k)$.
   (d) Assign to the pixel the label for which E(k) is minimum. If this causes a change in the label of the pixel, another pass through the image is required.
4. Iterate until no changes result to the labels of any of the pixels.

3.4 Hierarchical segmentation

At this point, the image has been requantized, but the quantization has taken the spatial arrangement of pixel values into account. A region growing algorithm is employed to build a set of connected regions, where each region consists of 8-connected pixels that belong to the same K-Means cluster. If a connected region is too small, then its cluster mean (the mean of the texture vectors at each pixel in the region) is compared to the cluster means of the adjoining regions and the small region is merged with the closest mean. This process is repeated until the regions are such that all cluster means have reliable statistics. In practice, we considered a region too small if it had less than 10 contributing textural measurements. We explore the effects of changing this limit in Chapter 5.

The result of the K-Means segmentation, region growing and region merge steps is the most detailed segmentation of the image. From this point onwards, we work exclusively in the domain of the segmented regions. The inter-cluster distances of all adjacent clusters (or regions) in the image are computed. A threshold is set such that half the pairs fall below this threshold. An iterative region merging is carried out whereby if a pair of clusters differ by less than this threshold, they are merged. More or less than half the clusters in the image may get merged because the cluster means are updated at the end of each merge, resulting in a different number of pairs which are closer than the threshold. The region merges are stopped when none of the resulting pairs of adjoining regions are closer than the threshold. The segmentation result at this point is the next coarser segmentation.

Because the results of segmentation at the second stage are formed by region merges only, every region in the coarse segmentation completely contains one or more regions in the detailed segmentation. Thus, there is a hierarchy of containment between the segmented results at these two scales. The inter-cluster distance threshold is relaxed steadily, set at each iteration to be of a value such that half the cluster pairs are closer to each other than the threshold. This process is repeated until the segmented results are stable. The result of the segmentation at each stage gives one level of the hierarchical tree (see Figure 3.1).

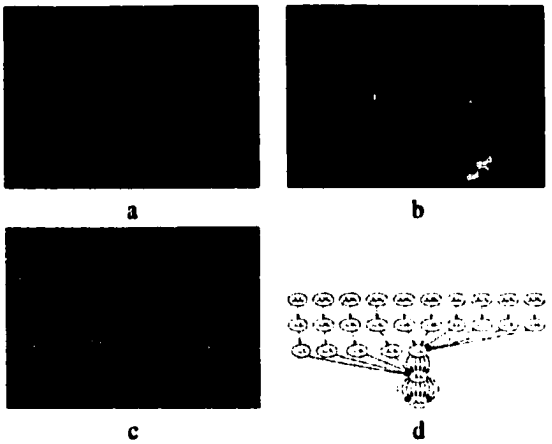


Fig. 3.1: Hierarchical segmentation: Left to right, top to bottom: (a) A photograph of a building from the U. Groningen database [1] (b) Most detailed segmentation, using the multiscale segmentation algorithm described in this dissertation, colored such that each component is a different color. (c) A coarser level of segmentation. (d) A hierarchical tree representation of the segmentation results. Regions at the top of the graph (the results of more detailed segmentation) are contained within the regions at lower levels in the graph.

3.5 Comparison to Similar Work

In this dissertation, we advocate a novel method of performing multiscale segmentation. We segment the original image, obtaining the most detailed segmentation possible. At each stage of our multiscale segmentation, we analyze the segmented regions globally, merging the regions that are most similar to each other, in the context of all the segmented regions and the scale at which the segmentation is being done.

A comparison of the method of segmentation proposed in this dissertation and the two traditional approaches (image pyramids and quad-tree decomposition) follows:

- The image pyramid approach employs images at different scales. The original image is filtered at different scales and these filtered images are segmented independently. Thus, the original image itself is decomposed. The proposed method does not decompose the original image, only the regions identified in the original image.

- In the quad-tree approach, the regions at each stage are merged or divided in a systematic manner based on the geographical location of the regions, local characteristics and a hard threshold. In the proposed approach, the choice of which regions to merge is made globally although only promixate regions are merged.

- In the image pyramid approach, the result is a set of segmented regions for each of the filtered images. Associating the components identified at different scales is a hard problem. This problem is sidestepped in the proposed method, as the multiscale aspect operates purely on the segmented regions.

- In the image pyramid approaches (in [77] for example), the texture boundaries are not reliable at coarse scales, because the boundaries themselves have to be expressed at the coarse scale. Yet, at higher details, since the statistics are not reliable, the classification is not reliable. In our approach, boundaries remain reliable even at the most coarse segmentation, because the boundaries are expressed at the scale of the original image itself.

The scale in the image pyramid approach, especially, is tied to the image space. In the proposed method, the scale in question refers to the components themselves. We argue that in the segmentation problem, component scale is more appropriate – it matches our intuitive understanding of scale (buildings vs. windows) while scale in image space refers more to spatial extent. Thus, in the segmentation problem which is concerned with decomposition into components, it is more appealing to work in the context of these components themselves.

Our argument is buttressed by the needs of the techniques that often follow the segmentation stage. Since the input to these techniques is the set of components at the various scales, the proposed approach can provide a better quality input. For example, in the tracking problem (examined in the appendixes to this dissertation), it is necessary to match segmented components across frames. This matching is, by the nature of the problem, subject to association uncertainties. In a multiscale tracking problem, the matching of segmented components will have to match not only across frames but across scale as well. By ensuring that there is no leeway in the association of components across scale, we reduce the dimensionality of the association problem in multiscale tracking.

Our argument against scale-space decomposition was made recently in [78] with the author dismissing scale-space decomposition, as determined by a scale parameter, as a blurring of the image to different degrees, along with all the consequence that this entails in boundary detection at different scales. The author [78] states as a goal, one similar to ours:

> ... the objective is to derive multiscale segmentation of the image and represent it through a hierarchical, tree structure in which the different image segments, their parameters, and their spatial interrelationships are made explicit.

This interpretation of multiscale segmentation, in terms of explicit spatial interrelationships, has found wide applicability – in image enhancement [79], for robust segmentation in image sequences [80] and in image registration [81].

The methodology followed by [78], is, however, different from the one proposed in this dissertation.

In [78], the image is transformed using a transformation that partitions the image such that each cell of the partition has a characteristic property. This is done by computing an attraction-force field over the image with the force at each point denoting the affinity to the rest of the image. Thus, spatial extent is carried through the entire process, albeit in a different form. The paper starts out with a motivation similar to ours – of building, from the bottom up, a hierarchical tree with explicit relationships between regions at different scales without getting drawn into the linear space-scale relationships. The method of [78] is exclusively in the image space since the transformation described is purely in the image domain. In that sense, it is not a scale-space method. However, the transformation function decays with distance, incorporating scale in much the same manner that the scale-space methods do (by utilizing filters of finite support or filters whose impulse responses decay with distance). Scale is still closely connected with spatial extent in [78]. Moreover, the method of [78] specifies the texture vector to be used. In the method we propose, the choice of texture vector can be made depending on the types of images being segmented.

Another recent paper [82] is similar to the work described in this paper in that a lot of the segmentation work is carried out in the region domain rather than in the image domain. The motivation of the work is, however, different – it is to simply use a single set of constraints in both the segmentation and interpretation problems. The two normally independent stages of the typical computer vision problem cooperate to minimize errors. In fact, the interaction happens via region clusters [82], which are the atomic unit in our multiscale approach as well. However, [82] does not deal either with texture or with multiscale segmentation.

A considerable body of work in clustering [83, 84, 85, 86] deals with partitioning clustering techniques.

Partitioning techniques are dynamic, in that they allow pixels to move from cluster to another at different stages. While the use of partitioning techniques, especially in combination with fuzzy clustering [87], can improve cluster validity, partitioning clustering algorithms do not satisfy the requirement of nested partitions. Nested partitions are essential to the formation of a hierarchical tree representation of the segmented regions.

A recent paper [88] tries to reconcile the relative advantages of hierarchical and partitioning clustering by determining the "optimum" number of clusters via competitive agglomeration [89]. The method in this dissertation, like that of [88], uses agglomerative hierarchical partitioning rather than a divisive method. In the method proposed in this dissertation, global cluster information is incorporated via the K cluster means in the image, subject to Markovian constraints. In [88], global information is incorporated via an a priori assumption of shape, for example that all clusters are ellipsoidal. An a priori knowledge of cluster shape would be impossible in weather images, one important application of the segmentation method proposed in this dissertation.

The method of segmentation – K-means clustering of textural feature vectors – is not new. We drew on existing segmentation approaches that used K-means clustering on texture vectors. As in [58], a combination of K-means clustering and morphological operators was used here. As in [69], we iteratively estimate the local means of each region. As in [57], we use a contiguity-enhanced measure and as in [70], we model the cluster regions as random fields. We extend previously published work in this area in these ways:

1. Multiscale segmentation – we show how the use of inter-cluster distances leads naturally to a hierarchically arranged tree of identified regions.

2. We use K-means clustering not to do the segmentation, but just to requantize the gray-levels in the image. In other words, we follow the K-means clustering stage with morphological processing and a region growing step where connected pixels belonging to the same cluster are labeled the same. This ensures that we need not know the number of objects before hand, since K is only an intermediate step to finding the number of regions in the image. In our variation of K-means clustering, then, K is better thought of as similar to the number of gray-level values we wish to requantize the measurement space into. This idea itself is not novel – a similar approach was taken in [45] in the context of MRF-

based texture segmentation. However, a histogram-based technique, not K-means clustering, was used in [45].

3. Further, we follow the clustering and region growing stage with steps that use texture-vector distances between statistically unsound regions and their statistically sound neighbors to yield a robust segmentation at the finest level of detail.

4. COMPARISON OF SEGMENTATION ALGORITHMS

4.1 Real-World Images

4.1.1 Gröningen database

A picture of a building from database referred to in [1] has been segmented using the method of this dissertation and the results demonstrated in Figure 4.1.

Notice that in the coarser segmentations, the components in the detailed segmentation that corresponded to the windows are subsumed into the building itself, so that the two main regions are those corresponding to the building and to the sky.

For comparison, the same figure is shown segmented using watersheds in Figure 4.2. Watershed segmentation can not handle textured images and so, the poor quality of the results is no surprise.

The building picture was segmented using the iterative, Gabor filter texture segmentation approach introduced in [42] and the results are shown in Figure 4.3. The Gabor filter set used had a standard deviation for the Gaussian of two pixels and used eight orientations. The detailed segmentation here is impressive – the edges detected correspond to the salient edges in the image and even the tree's edges have been found correctly, something that is not accomplished by any of the other algorithms. However, the edges are not closed, and the regions are not identified. Another problem is the resources demanded by this technique (See Section 4.3).

The same building image was segmented using a Markov Random Field-based texture segmentation approach due to [52]. The results are shown in Figure 4.4. The segmentation approach does not do well near the image boundary where the texture vector can not be computed well. Otherwise, the detailed segmentation in Figure 4.4 is better than the detailed one introduced in this dissertation in that it can differentiate between the

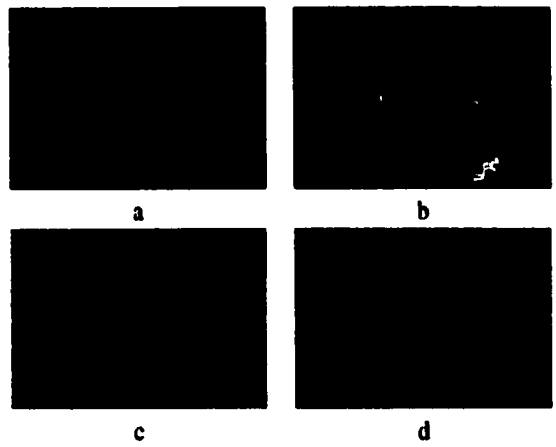


Fig. 4.1: Multiscale segmentation: (a) The photograph of a building that has been segmented using the multiscale segmentation algorithm described in this paper. The results at each scale, colored such that each component is a different color, are shown. (b) the most detailed segmentation. (c) more coarse segmentation. (d) most coarse segmentation.

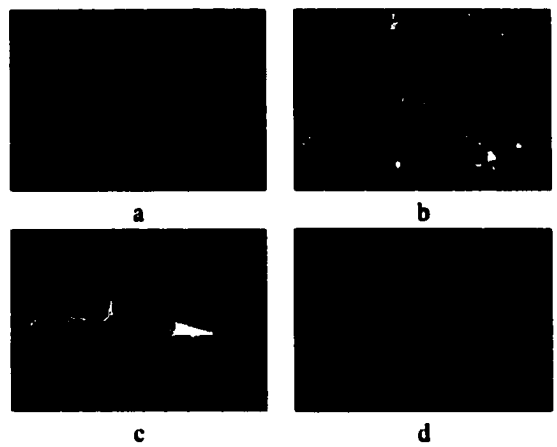


Fig. 4.2: Watershed segmentation: Left to right, top to bottom: (a) Photograph of a building (b) Segmented using the watershed segmentation of [40] after smoothing in a 5x5 neighborhood. (c) Segmented using the watershed segmentation of [40] after smoothing in a 25x25 neighborhood. (d) The salient watershed lines of (c) chosen by the method of [90]

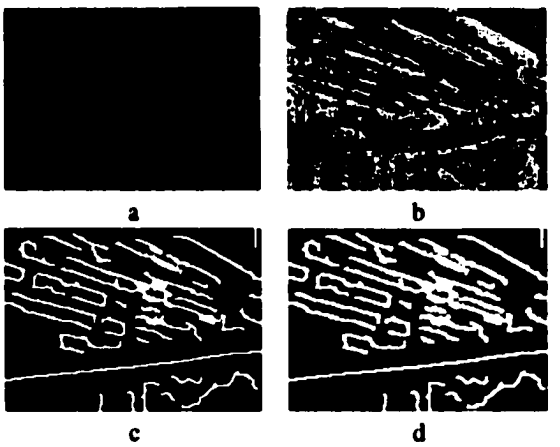


Fig. 4.3: Edgeflow (Gabor filter) segmentation: Left to right, top to bottom: (a) Photograph of a building (b) The direction of the flows – the phase is colored so that black and white are in different directions. (b) Edges found using the texture segmentation method of [42] (c) Edges laid over the original image.

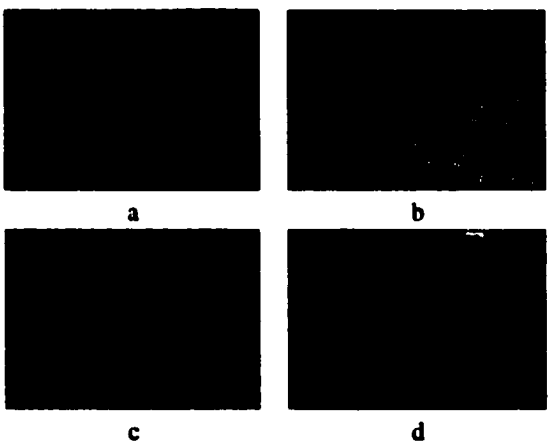


Fig. 4.4: MRF-based texture segmentation approach of [52]: Top to bottom, left to right: (a) Photograph of a building (b) Initialization using the modal interval [55]. (c) Intermediate step in the segmentation (d) Final segmented result.

various ground floor parts of the building. Overall, however, the method introduced in this dissertation performs better.

For the image of a building from the Groningen database, the Gabor-filter based method of [42] performs the best, followed by the method of this dissertation . Watershed segmentation and the MRF-based method perform poorly.

4.1.2 San Francisco

The aerial photograph of San Francisco [91] has been segmented and the results shown in Figure 4.5.

This is a particularly hard image to segment because there is very high variability within the regions of the image – note the presence of large black strips within the land portion of the image – and because there is very little difference between what are truly different parts of the image. For example, the glare on the water makes that parts of the sea resemble the land surface more than the darker water areas. Thus, it is not surprising that the results of segmentation using the Markov Random Field-based texture segmentation approach of [52] are not very good (See Figure 4.5e).

Statistical texture segmentation methods fare better in the comparison. In the original study [91], segmentation was performed by using statistical tests as a measure of homogeneity and formulating texture segmentation as a data clustering problem, with inter-cluster differences defined by a multi-scale Gabor filter image representation. Clustering was done by simulated annealing, with the number of clusters assumed to be known a priori. Unlike the study [91] from which this image was taken, we obtained these results without any a priori assumption of the number of regions in the image.

The edge-flow method of [42], like the original study, is Gabor-filter based. The result of segmentation using the edge flow method is shown in Figure 4.5f. Except for the water areas, the segmentation performs quite well.

It is noteworthy that we obtain a segmentation of quality comparable to the Gabor-filter methods without any a priori assumptions about the number of clusters in the image.

4.1.3 Leg ulcer

The gray level histogram of an ulcerated leg wound is shown in Figure 4.6a. Since there is a shallow valley between the peaks in the histogram corresponding to the wound (see Figure 2.2) and to the undamaged skin i.e. since the histogram is a bimodal distribution, it is possible to threshold the image so that pixels below a certain threshold are said to belong to a wound (See Figure 4.6b).

The detailed segmentation using the method of this dissertation(shown in Figure 4.6c) performs better than the amplitude segmentation used in the original study. The coarser segmentation, shown in Figure 4.6d, captures the salient, open part of the wound. These results are possible, even in the absence of any customization to the problem of interest, because the ulcer image satisfies the requirements of what we call "real-world

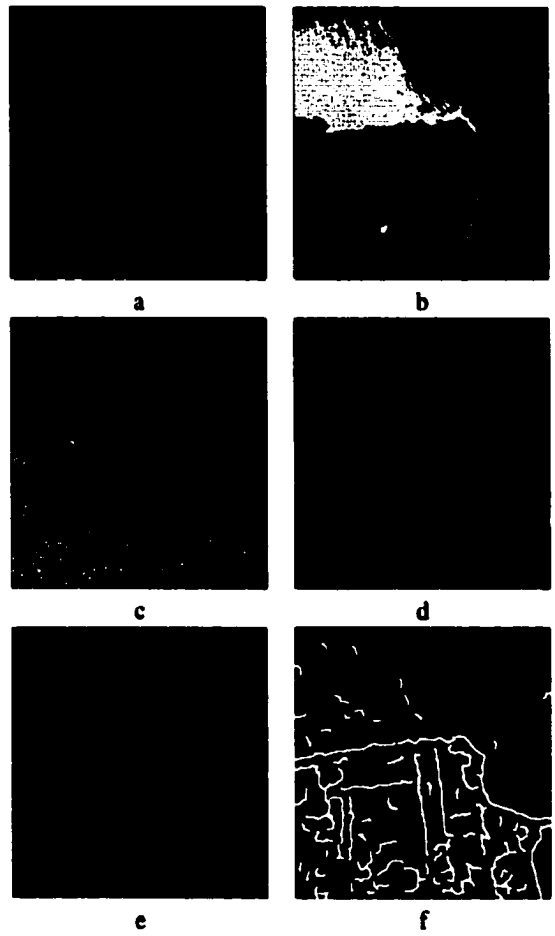


Fig. 4.5: Left to right, top to bottom: (a) An aerial image of San Francisco. (b) The most detailed segmentation of the image using the method of this dissertation. (c) Segmentation at a coarser level using the method of this dissertation. There is no a priori assumption of the number of regions in the image in the method described in this paper. (d) The result of segmentation using the method of [91] (a Gabor filtering and clustering based-method, from which this image was taken) assuming that there are four clusters. (e) Final segmented result, using the MRF-based approach of [52]. (f) Edges found using the Gabor filter texture segmentation method of [42]

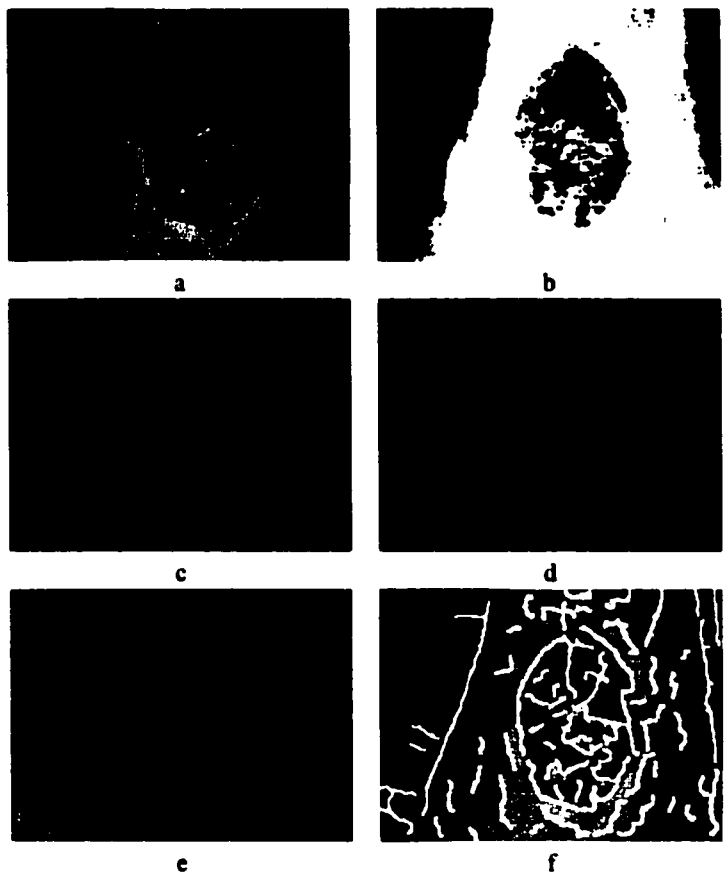


Fig. 4.6: Segmenting a photograph of a leg wound. (a) Photograph of a leg wound, from [35]. (b) The wound image amplitude thresholded. (c) The most detailed segmentation of the method of this dissertation. (d) Segmentation at a coarser level using the method of this dissertation. There is no a priori assumption of the number of regions in the image in the method described in this paper. (e) Final segmented result, using the MRF-based approach of [52]. (f) Edges found using the Gabor filter texture segmentation method of [42]

scenes": the texture is statistical, rather than regular and periodic, and the different objects in the scene (skin and lesion, in this case) differ significantly in texture.

The Markov-Random Field (MRF) and Gabor filter methods perform acceptably. This image illustrates the drawbacks of these methods. The MRF-segmented image does not hew to the actual boundaries of the wound. The Gabor filter captures all the edges, not just the salient ones.

The multiscale, hierarchical segmentation technique introduced in this dissertation outperforms the other segmentation methods.

4.2 Weather Images

A comprehensive literature survey of existing segmentation techniques to weather imagery is presented in Section B. In this section, some results of segmentation are presented.

4.2.1 Infrared satellite images

A single infrared satellite image was segmented using the various segmentation methods discussed in this chapter. The results are shown in Figure 4.7.

The results of segmentation using the other approaches are pretty bad. This is not surprising because the infrared satellite weather imagery has several characteristics that make it hard to segment: very low dynamic range (from about 225K to 240K) for the regions of interest, poor resolution as compared to the scale of the phenomena of interest, and high pixel value variance, even in the absence of edges. It is instructive to compare the poor performance of these algorithms on the satellite image (see Figure 4.7) with the performance of the same algorithms on radar reflectivity images (See Figure 4.9, discussed next.)

4.2.2 Radar Reflectivity Images

A single 0.5-degree elevation scan of radar reflectivity data collected by the Weather Service Doppler Radar (WSR-88D) at Fort Worth, TX on May 5, 1995 was used to compare the segmentation results of the various techniques. The results are shown in Figure 4.9.

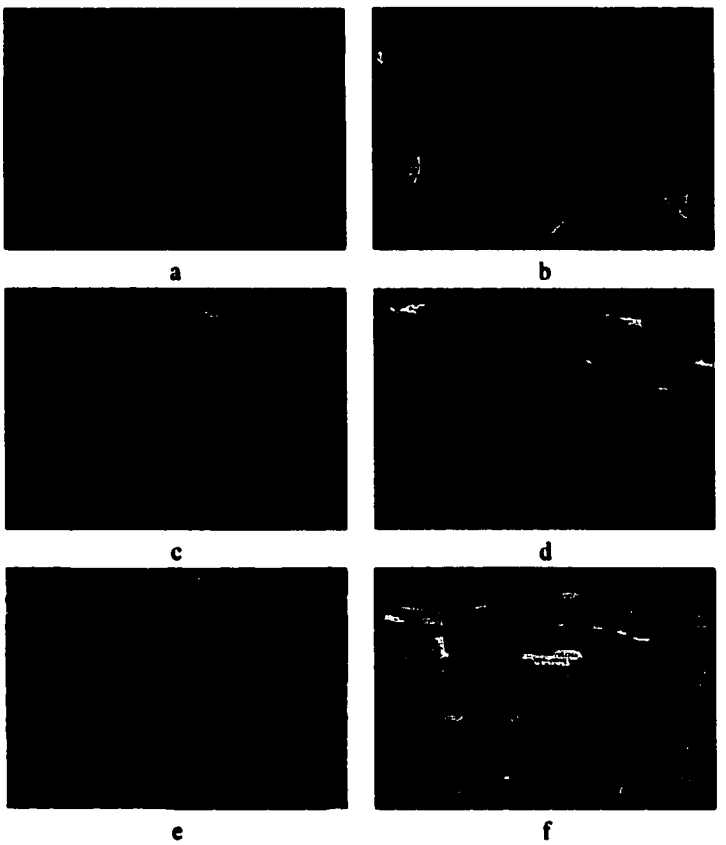


Fig. 4.7: Segmenting an infrared satellite weather image. (a) The infrared image being segmented. Notice the various storms at the top of the image. The darker areas in the bottom correspond to ground. (b) The result of segmenting the image using Markov Random Field (MRF) approach of [52]. There is no detail – it is effectively a binary segmentation. (c) The result of segmenting the image using the method of this dissertation (the most detailed scale). Notice the fine detail within the clouds. (d) The next higher scale of segmentation using the method of this dissertation. The strong storm cells being significantly colder are retained – the large cloud masses are merged. (e) Simply separating the image into contiguous bands of $1 Kelvin$. There is a lot of detail, but no organization. (f) Using the watershed segmentation approach of [90]. Because of the textural nature of the data, the watershed algorithm has very poor performance. See also Figure 4.8.

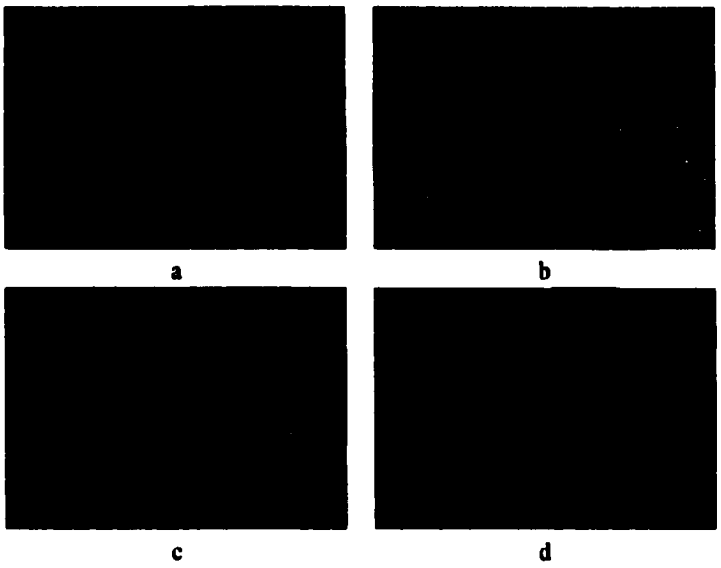


Fig. 4.8: A close-in look at the results shown in Figure 4.7.. (a) The infrared image being segmented (same as Figure 4.7a). (b) A close-in look at the input satellite infrared image of (a). (c) A close-in look at the result of segmenting the image using the method of this dissertation (the most detailed scale). Notice the fine detail within the clouds. (d) A close-in look at the next higher scale of segmentation using the method of this dissertation.

As can be seen from Figure 4.9, all the methods perform quite well on this image. This is not surprising because radar data has a good dynamic range (from $-7dBZ$ to $64dBZ$), tends to have well-demarcated storm cells and has good resolution when compared to the typical size of a storm cell. In all of these respects, it is different from satellite infrared weather images. The results of segmentation bear this out.

4.3 Segmentation speed

A comparison of the computer resources required by each of the segmentation algorithms on a Sun Ultra 10 running Solaris to segment the image of an ulcerated leg wound (See Section 4.1.3) is shown in Figure 4.1. The resources required to segment a single radar reflectivity image is also shown.

It should be noted that the method of this dissertation performs multiscale segmentation, so the reported resources correspond to segmenting all the scales, not just the single scale of the other methods. It is apparent that the method reported in this dissertation is the fastest to converge on a result, even though this has to be done in a multiscale fashion. It is also the least memory intensive. As can be seen in Figure 4.6, the results of segmentation are also the best.

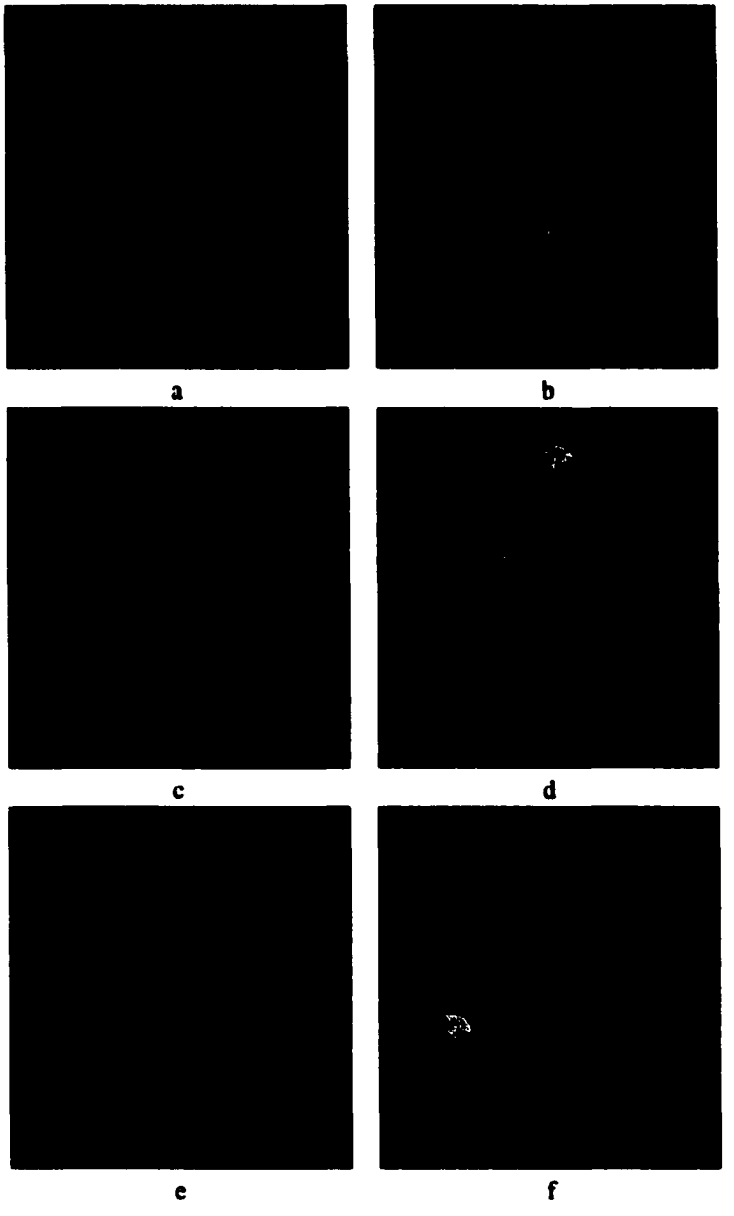


Fig. 4.9: Segmenting a radar reflectivity image. (a) A radar reflectivity image, from Fort Worth May 5, 1995. (b) The result of segmenting the radar reflectivity image using the Markov Random Field (MRF) approach of [52]. (c) The result of segmenting the image using the method of this dissertation, tweaked to process the reflectivity range of interest. The most detailed scale is shown. (d) The next higher scale of segmentation using the method of this dissertation. (e) Simply separating the image into contiguous bands of $10dBZ$. (f) Using the watershed approach of [90].

Method	CPU time (μsec)	Time sec	Memory kB
A 256x178 image of a leg ulcer wound			
Gabor [42]	1641.79	2005.44	30880
MRF [52]	68.02	1.45	80808
Method of this dissertation	3.46	1.13	24664
A 307x307 image of radar reflectivity			
Gabor [42]	2632.21	3604.80	45760
MRF [52]	115.24	3.05	144344
Method of this dissertation	3.73	0.94	24240

Tab. 4.1: The computer resources required by each of the texture segmentation methods compared in this section. Sections 4.1.3 and 4.2.2 have descriptions of the images and demonstrations of the results of segmentation.

4.4 Segmentation accuracy

Texture classification algorithms can be evaluated and compared quantitatively. Typically [92, 93], various texture representations – Markov Random Field, Gabor filters, fractal dimension measures, logical operators, coöccurence matrices or statistics – are used to compute a vector of measurements for every pixel. Using the computed textures over a number of textured images, a model of the input classes is formed. This model is then used in tandem with a texture classification technique such as a K-Nearest neighbor classifier or with a Gauss-Markov classifier to classify the texture vectors found in a set of test images. By varying either the texture representation or the texture classification method, the percentage of pixels that are correctly classified may be compared with other known known techniques.

Thus, frameworks such as [94] may be used to evaluate texture representations or texture classification algorithms. However, they can not be used to evaluate texture segmentation algorithms. In the literature, for example [93], this short-coming is avoided by casting the segmentation problem into a classification one by requiring that a suitably trained classifier be used for segmentation. Since we are proposing a new technique of doing image segmentation, a metric geared toward only segmentation needs to be utilized.

We propose a metric for image segmentation that is computed similar to the way metrics for texture classification are computed. A test image, comprised of different textural swathes, is presented to the segmentation algorithm. At the time of presentation, a labeled image is created with labels $1, 2..., K$ depending on which texture was used for that point in the test image. The segmentation algorithm then performs the segmentation and delivers its output labeled $1, 2, ..M$. If the segmentation algorithm requires prior knowledge of the

number of regions, then the value of K is passed into the algorithm (in which case, the resulting M would be equal to K). At this point, we have the correctly labeled image (the "true" segmentation) and the segmented image. We can then measure the performance of the segmentation algorithm.

The segmentation accuracy of the i^{th} object in the scene can be computed as:

$$a_i = \frac{S_{i1} \vee S_{i2} \vee \ldots \vee S_{iM}}{N_i} \tag{4.1}$$

where S_{ij} is the number of pixels in the region of the i^{th} object that is occupied by the j^{th} segmented region and $\vee$ is the max operator:

$$a \vee b = \max(a, b) \tag{4.2}$$

Therefore, $\bigvee_j S_{ij}$ gives the segmented region that covers the largest fraction of the i^{th} object. The fraction of the i^{th} object occupied by this segmented region is also a measure of the accuracy with which the segmentation method has segmented that object.

We could measure the overall segmentation accuracy as a fuzzy measure given by the minimum accuracy with which individual objects have been identified, i.e. by:

$$a = \bigwedge_i^K \frac{\bigvee_j S_{ij}}{N_i} \tag{4.3}$$

where $\wedge$ is the min operator:

$$a \wedge b = \min(a, b) \tag{4.4}$$

However, this measure has a shortcoming in that it does not enforce the separate identification of different regions. This can be enforced by computing the accuracy as:

$$a = \frac{\bigvee_j S_{i_1 j}}{N_{i_1}} \wedge \frac{\bigvee_{j \in E_1} S_{i_2 j}}{N_{i_2}} \wedge \ldots \frac{\bigvee_{j \in E_{k-1}} S_{i_k j}}{N_{i_k}} \tag{4.5}$$

where the regions from the true segmentation (the i's) are selected such that $N_{i_p} > N_{i_{p+1}}$, i.e. in decreasing order of size. The choice of the regions from the result of the segmentation technique (the j's) is restricted to

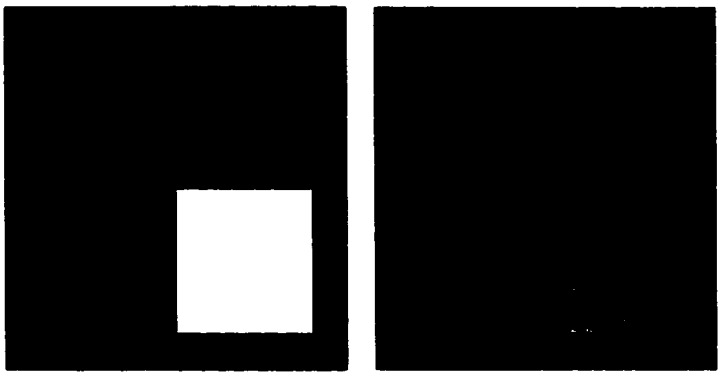


Fig. 4.10: Measuring the accuracy of a segmentation algorithm involves comparing the result of segmentation (right) with the true object locations (left). The segmentation accuracy for the small square at the top left of the image is 0.79 and that of the background pixels is 0.83 since they are essentially captured by regions that cover 79% and 83% of the surface of the original objects. The third region, at the bottom left is poorly segmented, with the best possible association yielding an accuracy of 0.27. The overall segmentation performance is the minimum of the three objects' accuracy measures, so 0.27.

the set E_p, the set of $j's$ that have not been selected as matching any of the previous p regions.

The effect of this accuracy measure is illustrated here using the segmentation result shown in Figure 4.10. The accuracy measure of Equation 4.5 requires that the regions in the true segmentation be ordered in decreasing order of size. Hence, the regions should be considered in this order: the background, the object at the bottom right and finally, the object at the top left.

The following discussion pertains to the segmented result in Figure 4.10b. For the background, consisting of 8300 pixels, the maximum coverage is provided in the segmented image by a region of 6894 pixels, giving $a_1 = 0.83$ (See Equation 4.1). The next object to be considered is the object at the bottom right. This object, of 1600 pixels, is covered by several objects, but since Equation 4.1 requires only the maximum coverage, we can use at best a region of 431 pixels, giving $a_2 = 0.27$. The third object is covered by a region of 79 pixels, giving an $a_3 = 0.79$. The accuracy measure, given by the minimum of the three objects' measures, is therefore 0.27.

4.4.1 Natural Images

Following [92, 93], natural textures were chosen from the Brodatz [29] database and arranged within an image in a known manner. Some of the images created from the Brodatz textures and the best resulting

Test/Technique (Parameter)	KS Test [52, 55]	K-Means [57] K=2or3	K-Means [57] K=4	Hierarchical K-Means K=2	Hierarchical K-Means K=4	Hierarchical K-Means K=8	Hierarchical K-Means K=16
D12 and D15	0.22	0.21	0.03	0.03	**0.37**	**0.42**	0.37
D24, D38, D68	0.05	0.18	0.08	0.06	0.02	**0.84**	**0.57**
D84, D94, D29	**0.32**	0.02	0.04	0.00	0.02	0.04	**0.13**
D112 and D19	0.05	0.03	0.33	0.02	**0.58**	**0.72**	0.53
D15, D112,D38	0.14	**0.39**	0.20	0.03	0.02	0.24	**0.28**
D38, D64	0.06	0.73	0.69	0.76	**0.89**	**0.88**	0.88
μ	0.14	0.26	0.228	0.149	0.316	**0.523**	**0.459**
σ/μ	0.722	0.938	1.013	1.822	1.051	**0.599**	**0.519**

Tab. 4.2: Images comprised of Brodatz [29] swathes (See Figure 4.11) were segmented using various unsupervised, untrained algorithms and the accuracy computed as in Equation 4.5. As can be seen, the method of this dissertation with K=8 or 16 is always in the top two. Overall, our method is the best performing (maximum mean) and the most consistent (least sigma/mean).

segmentations of those images are shown in Figure 4.11. The hierarchical, agglomerative, K-Means technique described in this dissertation was compared against two other texture segmentation algorithms reported in the literature – a contiguity-enhanced K-Means method [57] and with a Kolmogorov-Smirnov test-based method [52]. These techniques were chosen because they are completely unsupervised and because they, like the method introduced in this dissertation, require no training on the textures in the image. To make meaningful comparisons possible, the same set of statistical measurements (mean, variance, homogeneity, kurtosis and contrast) were computed in the same neighborhood (7x7) about a pixel for all of the segmentation algorithms. For the hierarchical method of this dissertation, the second most detailed scale of segmentation was chosen for all the measurements – this yields the benefits of both the K-Means requantization and the inter-cluster distance merging steps.

Since the traditional K-Means technique requires prior knowledge of the actual number of regions in the test image, this was input to the algorithm. We also show the result of over-estimating the number of regions. We demonstrate the effect of K, the number of significant quantization levels in our technique by showing the results when K is chosen in a range from 2 to 16. The method of [52] was implemented with reasonable defaults and the best performing initialization [55] and the same set of parameters used for the entire data set. The accuracy of segmentation is shown in Table 4.2. For easy reference, the two best performances are shown in bold.

As expected, textures that are insufficiently captured by the texture representation (e.g: middle of Figure 4.11e) are segmented poorly. When the texture is well represented, as in Figure 4.11c, the resulting

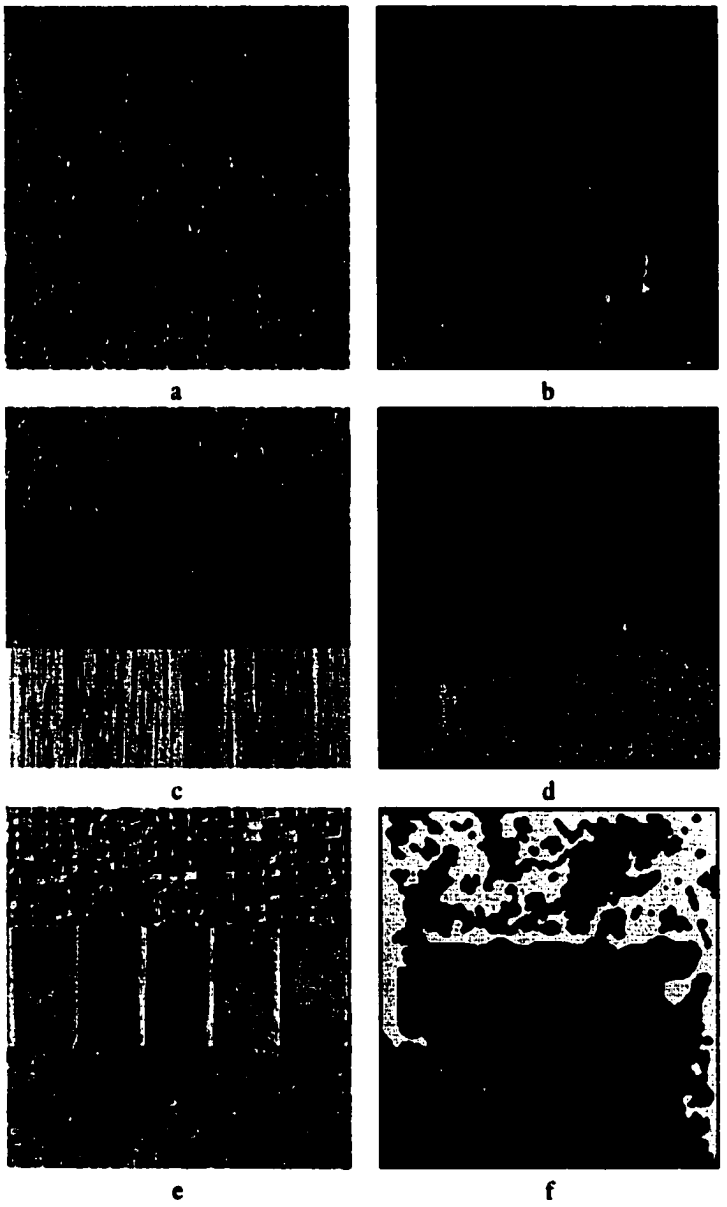


Fig. 4.11: Images comprised of Brodatz [29] swathes were segmented using various unsupervised and untrained texture segmentation algorithms and the accuracy computed as in Equation 4.5. The images on the left are the images that were segmented. The images on the right show the result of the best segmentation. See also Figure 4.12 (a) D38, D24. (b) The result of segmentation by the method of this dissertation with K=4. Note that K is not the number of regions – there are actually 83 regions in the segmentation result shown. Most of these regions are agglomerated in later stages of the multiscale segmentation algorithm (See, for example, Figure 4.14). (c) D24, D38 and D68. (d) The result of segmentation by the method of this dissertation with K=8. The texture vector captures all three of these textures well, hence the excellent performance. (e) D84, D94 and D29. (f) The result of segmentation by the method of [52, 55].

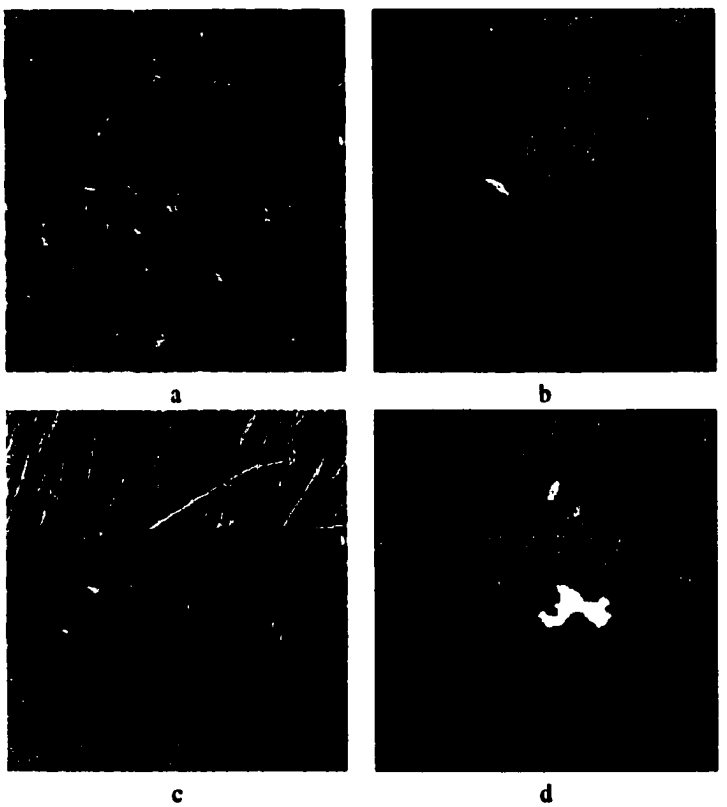


Fig. 4.12: Images comprised of Brodatz [29] swathes were segmented using various algorithms and the accuracy computed as in Equation 4.5. The images on the left are the images that were segmented. The images on the right show the result of the best segmentation. See also Figure 4.11 (a) D112, D19 (b) The result of segmentation by the method of this dissertation with K=8. (c) D15, D112 and D38 (d) The result of segmentation by the contiguity-enhanced K-Means technique of [57] with the number of regions, K=3.

Test No.	μ_{bg}	σ_{bg}	μ_1	σ_1	μ_2	σ_2
1	5	1	35	1	60	1
2	20	8	45	10	60	10
3	20	3	45	5	60	5

Tab. 4.3: The normal distributions (mean, μ, and standard deviation, σ, from which pixel values for the various objects in the scene were selected. The figure generated from Test 2 is shown in Figure 4.15.

segmentation is very good. It should be noted that the performance of all these segmentation algorithms is without the benefit of texture training – these are unsupervised texture segmentation algorithms that have not seen these objects before.

As can be seen from Table 4.2, the method introduced in this dissertation consistently outperforms the other techniques except on the the third test pattern, where our choice of textural representation does not capture the middle component (see Figure 4.11e) well. In that case, the method of [52], with its use of global statistics, outperforms.

The result of segmenting the second test pattern using each of these techniques is shown in Figure 4.13.

The method introduced here is a hierarchical method, and using only one of the resulting scales actually underestimates the performance of the technique. The different scale segmentations that result when segmenting the fourth test case (D112 and D19) is shown in Figure 4.14.

4.4.2 Synthetic Images

Synthetic images were created with an eye to approximate weather data. The synthetic images were created by selecting pixel values from normal distributions (a different normal distribution for each object in the scene). The synthetic image shown in Figure 4.15b, for example, is organized in this manner: the background pixels are normally distributed with a mean of 20.0 and a standard deviation of 8.0, the two square objects are composed of pixel values normally distributed about 45 and 60 with standard deviations of 10.0. Three sets of synthetic images were created, with the pixel values for each of the objects in the scene chosen from different normal distributions, as shown in Table 4.3.

The most accurate segmentation algorithm would be one that segments the image into three objects and assigns every pixel in the squares to the corresponding objects.

It was not possible to obtain a region-based segmentation from the edge-flow method of [42], since the

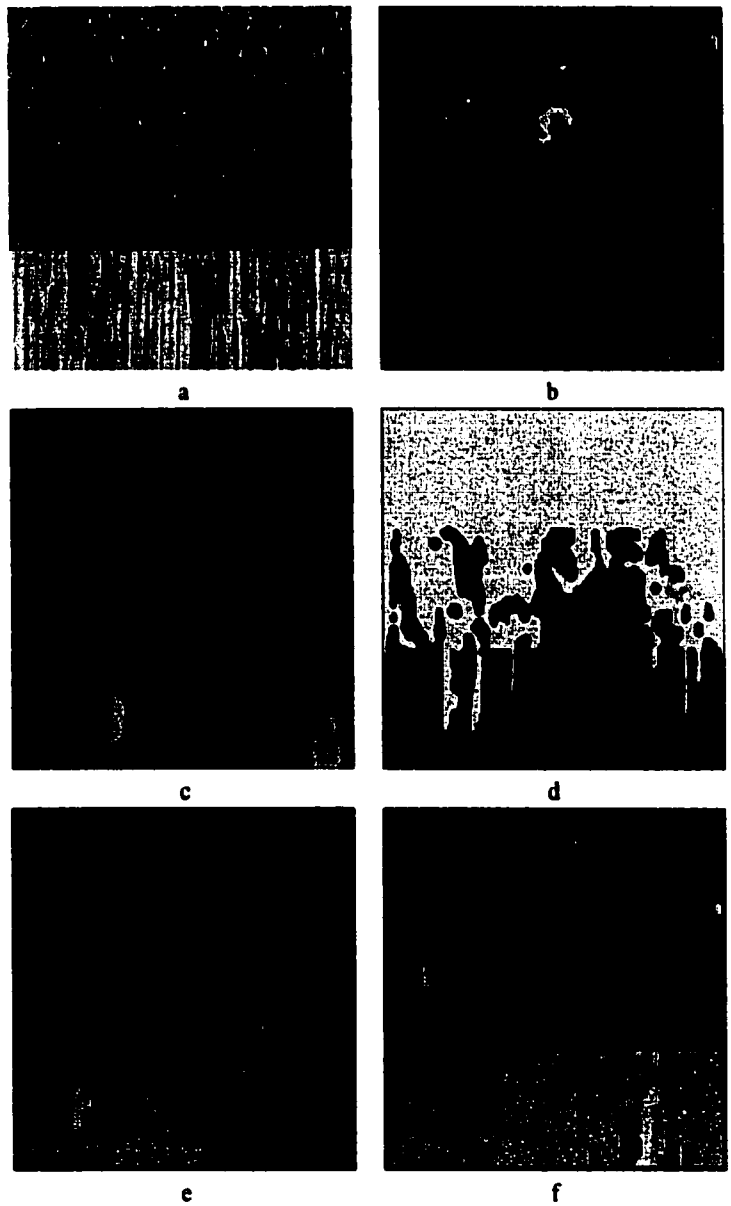


Fig. 4.13: (a) Image comprised of Brodatz [29] textures D24, D38 and D68. The result of segmentation by (b) the method of [52]. (c) the method of [57] with K=3 (there are 3 regions). (d) the method of this dissertation with K=2. (e) the method of this dissertation with K=8 (the best performance). (f) the method of this dissertation with K=16. This is the second test referred to in Table 4.2.

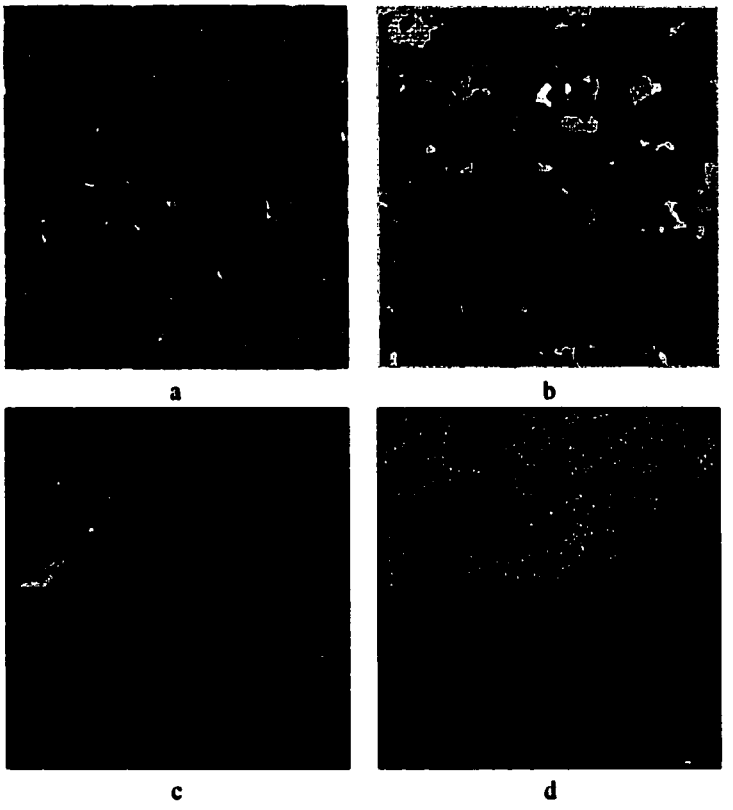


Fig. 4.14: (a) Image comprised of Brodatz [29] textures D112 and D19. The result of segmentation using the method of this dissertation with K=16 at various scales. (b) most detailed (accuracy=0.04) (c) second most detailed – this is what is used in the accuracy computation in Table 4.2. accuracy=0.53. (d) coarse – this is actually the most accurate (accuracy=0.76). This is the fourth test referred to in Table 4.2.

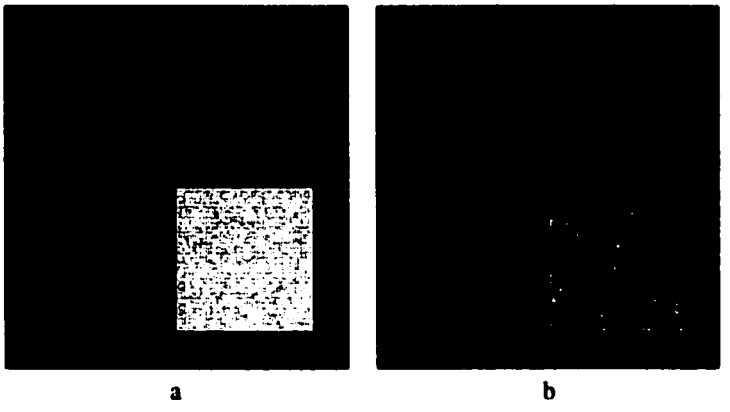


Fig. 4.15: Synthetic textured images created to compare the accuracy of the segmentation produced by different techniques. The pixel values are selected from three different normal distributions, giving rise to two objects and a background. The image on the right is selected from a normal distribution with higher variance.

Technique	Test 1	Test 2	Test 3
Watershed [90]	0.11	0.04	0.06
MRF [52]	0.42	0.28	0.26
Method of this dissertation (most detail)	0.90	0.27	0.90
Method of this dissertation (second scale)	0.90	0	0

Tab. 4.4: The accuracy of different segmentation techniques on the scene with three synthetic, but statistically generated, objects. An example scene from Test 2 is shown in Figure 4.15 and the description of the normal distributions for each of the tests is given in Table 4.3.

result of the segmentation in that technique is a set of edges, rather than a set of closed regions. Three segmentation methods from the literature were compared on the set of three images described in Table 4.15 and the results are shown in Table 4.4. The resulting segmentation results are shown in Figures 4.16-4.18.

At higher scales, the method of this dissertation attempts to reduce the number of regions from 3 to 1 and hence, merges the smaller region with the background. Since the smaller region is then missed completely, the accuracy ends up being zero.

It is noticed from Table 4.4, as well as from Figures 4.16-4.18, that the method of this dissertation outperforms the other segmentation techniques except on Test 2, the test where the images were generated using normal distributions of high variance. In Test 2, the results are comparable to the performance of the Markov Random Field approach of [52], but not better than it.

4.5 MPEG methods

The MPEG-4 [95] standard specifies a bit stream syntax for multimedia as well as a set of interfaces for the builders of multimedia applications. MPEG-4 specifications allow the development of such applications as content-based storage and retrieval to enable searching through image archives based on such attributes as shape and motion. To achieve this, MPEG-4 requires that image sequences be segmented before coding.

The MPEG-4 standard allows for different implementations of segmentation, but requires that video contents be represented by image analysis techniques [96]. Techniques reported in the literature usually utilize one of the major segmentation approaches described in Chapter 2 and compared previously in this chapter. For example, the MPEG-4 methods of [96] and [97] both use clustering. The method of [96] uses statistical texture vectors while the method of [97] uses density gradients.

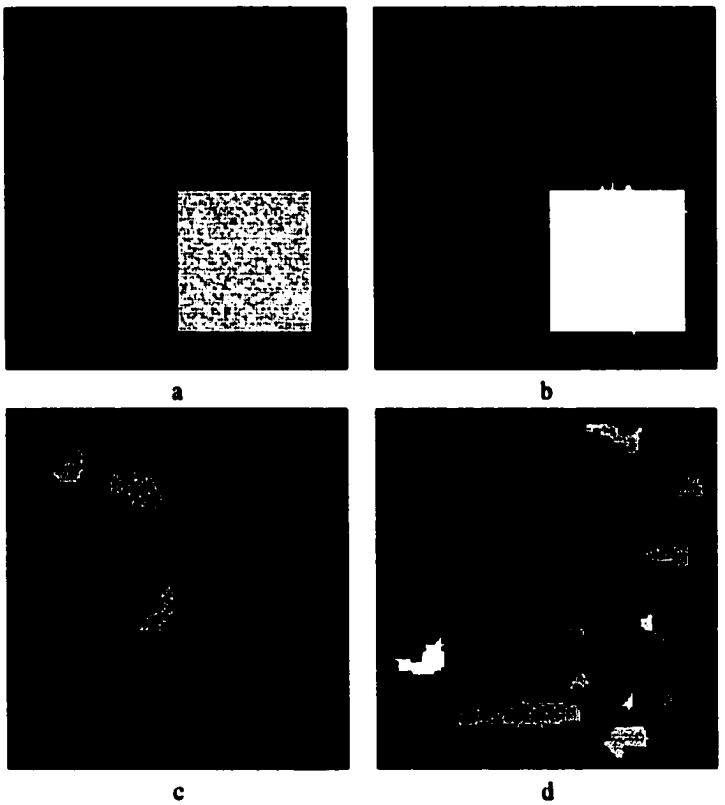


Fig. 4.16: The synthetic texture image in (a) was segmented using various segmentation techniques to compare the accuracy of segmentation. (b) When segmented using the method of this dissertation(c) When segmented using the MRF method of [52] (d) When segmented using watershed segmentation. This is Test 1 referenced in Table 4.4.

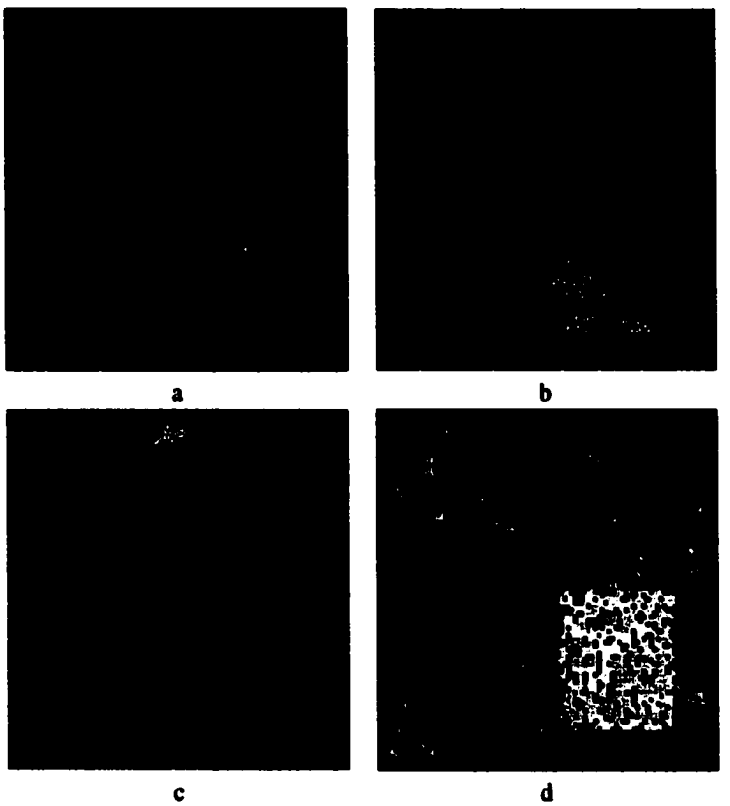


Fig. 4.17: The synthetic texture image in (a) was segmented using various segmentation techniques to compare the accuracy of segmentation. (b) When segmented using the method of this dissertation(c) When segmented using the MRF method of [52] (d) When segmented using watershed segmentation. This is Test 2 referenced in Table 4.4.

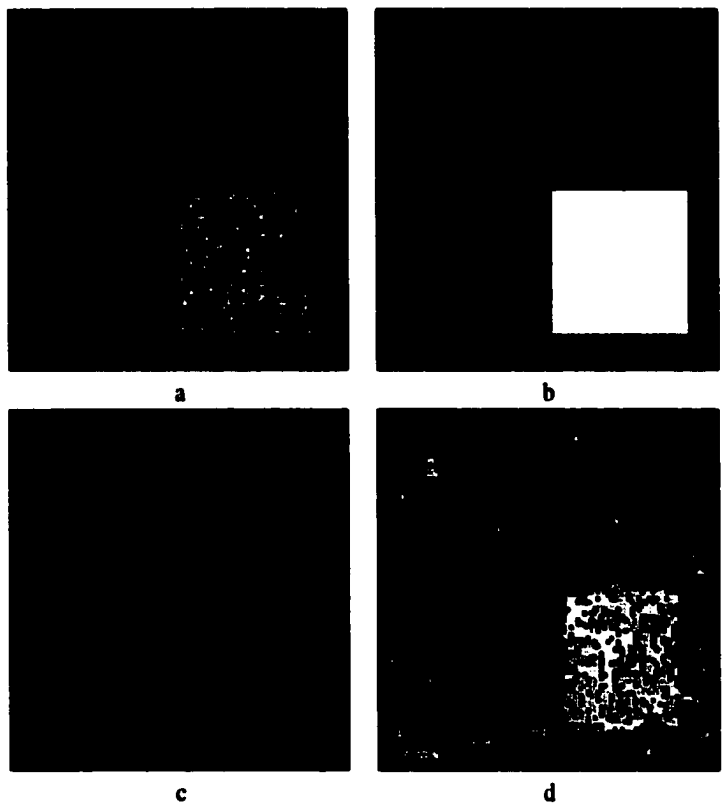


Fig. 4.18: The synthetic texture image in (a) was segmented using various segmentation techniques to compare the accuracy of segmentation. (b) When segmented using the method of this dissertation(c) When segmented using the MRF method of [52] (d) When segmented using watershed segmentation. This is Test 3 referenced in Table 4.4.

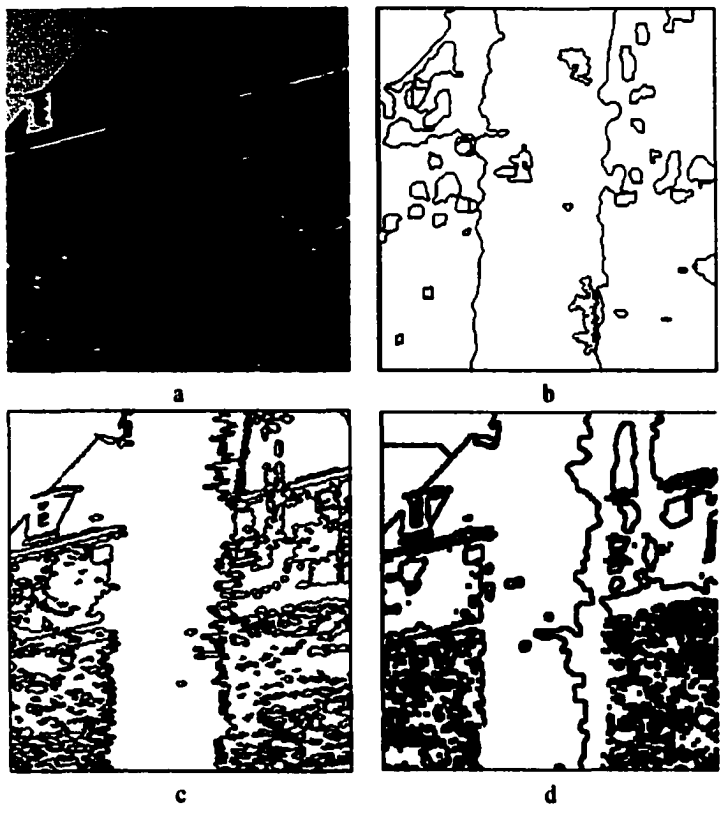


Fig. 4.19: (a) The image from [96] was segmented using MPEG-4 segmentation techniques and compared against the method of this thesis. (b) When segmented using the MPEG-4 method of [96] (c) When segmented using the MPEG-4 method of [97] (d) When segmented using the method of this dissertation

We did not develop our segmentation technique to fit into the MPEG-4 framework, or to follow the MPEG specifications. However, it is possible to compare the results of segmenting images using the method described in this dissertation with MPEG-4 methods that provide access to this intermediate output, as reported in the literature. MPEG-4 methods deal with color images, but the method reported in this dissertation currently uses texture vectors computed on gray-level values only. Hence, the images in Figures 4.20 and 4.19 were converted to gray-scale before being segmented using the hierarchical K-Means clustering technique reported here. The images in Figures 4.19 and 4.22 were gray-level images originally, and were used as-is.

As can be seen, on these images, the method introduced in this dissertation provides performance comparable to the studies from which the images were drawn. The fact that the method of [97] uses density gradients, rather than textures, explains the rather poor performance of this technique on the ping-pong picture (Figure 4.22).

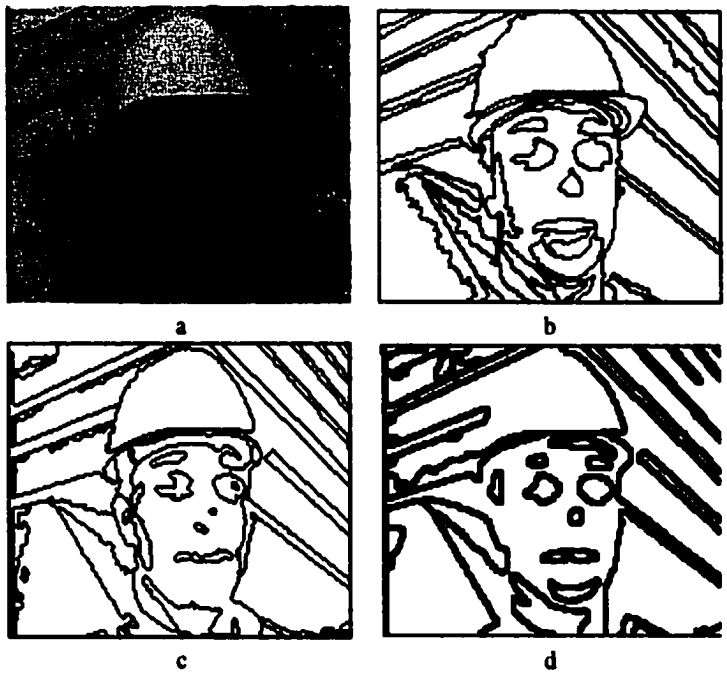


Fig. 4.20: (a) The image from [96] was segmented using MPEG-4 segmentation techniques and compared against the method of this thesis. (b) When segmented using the MPEG-4 method of [96] (c) When segmented using the MPEG-4 method of [97] (d) When segmented using the method of this dissertation

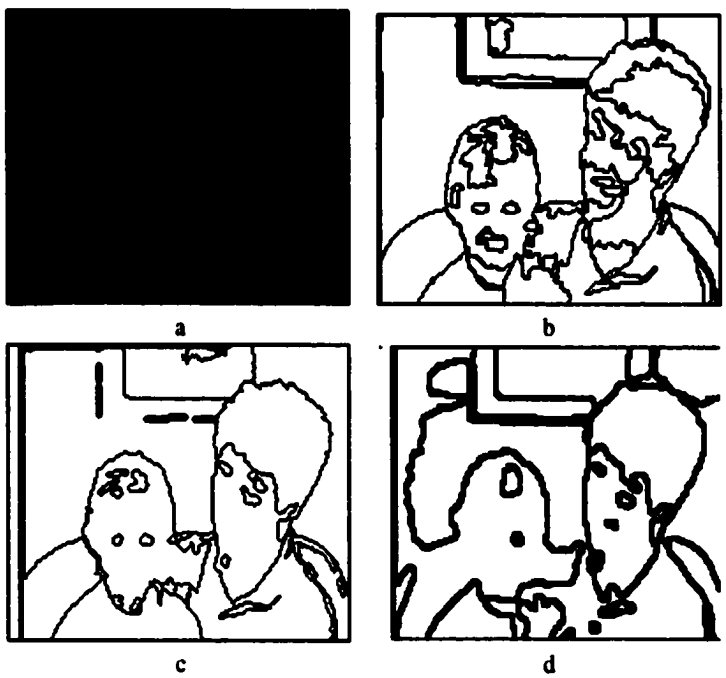


Fig. 4.21: (a) The image from [96] was segmented using MPEG-4 segmentation techniques and compared against the method of this thesis. (b) When segmented using the MPEG-4 method of [96] (c) When segmented using the MPEG-4 method of [97] (d) When segmented using the method of this dissertation

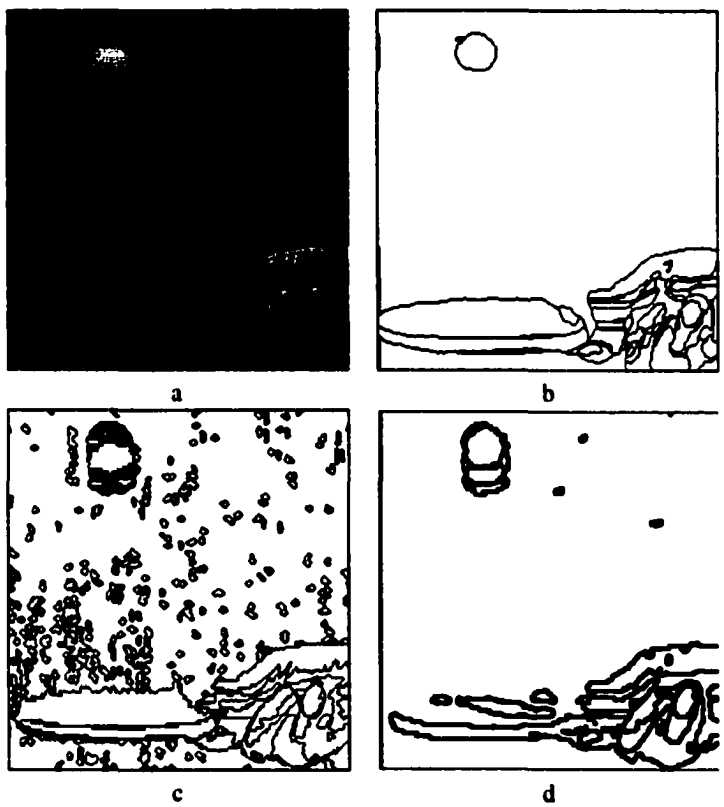


Fig. 4.22: (a) The image from [96] was segmented using MPEG-4 segmentation techniques and compared against the method of this thesis. (b) When segmented using the MPEG-4 method of [96] (c) When segmented using the MPEG-4 method of [97] (d) When segmented using the method of this dissertation

5. CHOICES OF PARAMETERS

In the algorithm as described in Chapter 3, several choices were made. We will identify these choices and explore the reasons, if any, behind those decisions. We will also explore the effects of changing any *ad hoc* choices on synthetic and real-world images. Since it is not possible to verify the effects of all possible combinations of all these parameters, interesting combinations of these choices will be studied.

5.1 The number K

We use K-Means clustering to requantize the input image into K levels, which are then segmented using an iterative, hierarchical segmentation technique. We used an uniform K=4 for all the image segmentations in Chapter 4, but will examine the effects of changing K here.

The number, K, is the number of quantization levels into which the image's gray levels are initially partitioned. As such, varying K will vary the number of regions into which the most detailed segmentation takes place. The actual number of regions depends on the image and on the distribution of pixels belonging to the same cluster, since the regions are identified by region growing, a technique that relies on pixels belonging to a region being proximate to one another.

The effect of varying K when segmenting the building image from [1] is shown in Figure 5.1.

As the number K increases, the clusters cover a smaller range in the texture space. Therefore, parts of the image, such as the trees in the foreground, that did not get identified as separate objects at low values of K are identified as new objects at K=8 (see Figure 5.1).

The value of K affects the result of segmentation. However, ours is a hierarchical technique, and examining the effects at a single level does not show the complete picture. What is the effect of larger values of K at higher levels? Using the windows image, the total number of regions in the image was graphed at

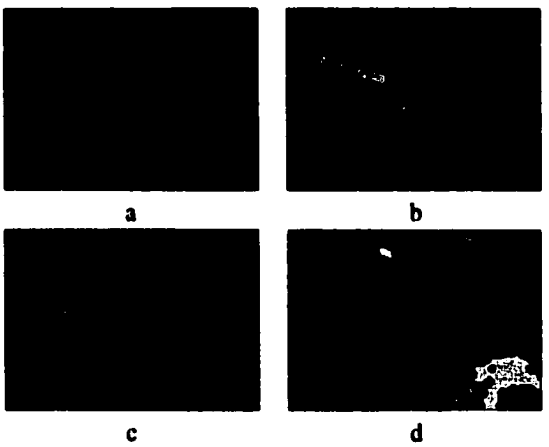


Fig. 5.1: The effect of varying K on the most detailed segmentation. (a) Original image, from [1]. Segmentation result (most detail) with (b) K=2 (c) K=4 (d) K=8

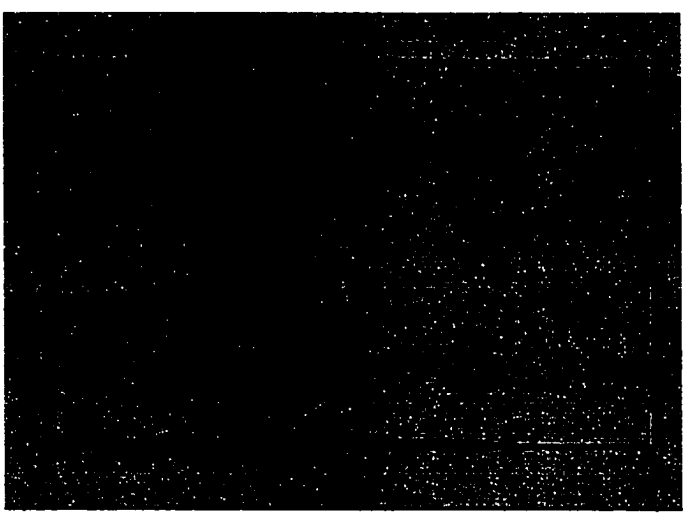


Fig. 5.2: The effect of varying K on the number of regions at various scales. The image being segmented is the building image shown in Figure 5.1.

successive levels. The result is shown in Figure 5.2. We see that although K does not affect the number of regions directly, it does so indirectly. At successive iterations, we attempt to halve the number of regions in the image, and so the original choice of K does play a part in the first two iterations.

Although we don't need to know the exact number of regions in the image, the characteristics of the image should dictate the selection of the value of K. We claim that the choice of the number of significant gray level quantization levels is an easier one to make than that of the exact number of objects in the image and thus, the method proposed is an improvement over traditional K-means clustering.

In case the number of regions is not known a priori, a very high value of K (for example, 16, in Figure 5.2) may be chosen. The most detailed segmentation may have too many regions, but coarser levels might yield the desired result. This is one advantage of using a hierarchical technique.

5.2 The texture weight λ

The choice of the label for the pixel (x, y) in the $(n+1)^{th}$ iteration, S_{xy}^{n+1}, is given by the label $k \epsilon S_{N_{xy}}^{n}$ that minimizes the energy, $E(k)$. given by Equation 3.13 and repeated here for convenience:

$$E(k) = \lambda d_m(k) + (1 - \lambda) d_c(k) \tag{5.1}$$

The weight, λ is the weight given to the texture component of this energy, $d_m(k)$, relative to the contiguity measure, $d_c(k)$. We used a fixed value of $\lambda = 0.6$ for all the images, but wish to study the effect of varying this measure.

The texture component captures the distance, in texture space, between the texture vector at a pixel and the mean texture vector of a cluster. The contiguity energy captures the smoothness of the segmentation – that a pixel belongs to the same cluster as the majority of its neighbors.

Using $\lambda = 0.0$, we disregard the texture component and use only the discontiguity measure. In other words, we want pixels to belong to the same cluster regardless of how different their individual texture vector is from that cluster mean. We should expect to get smooth region boundaries. Using $\lambda = 1.0$, we disregard the contiguity component and choose as the label for a pixel the cluster whose mean it is closest to. We should

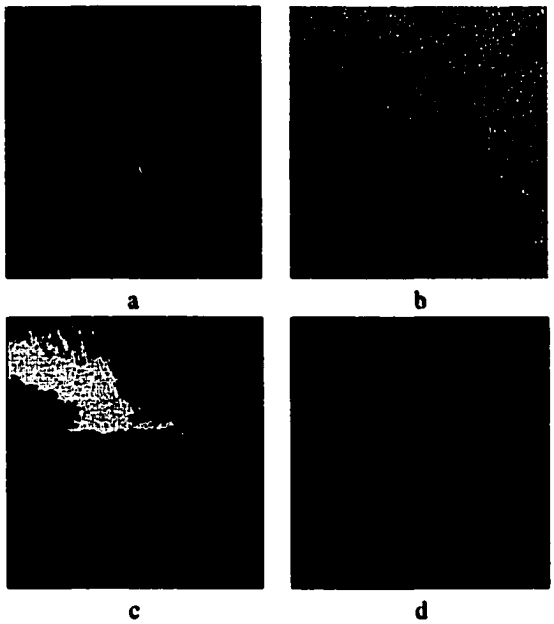


Fig. 5.3: Extreme values of λ: The aerial photograph of San Francisco from [91] shown in Figure 4.5 has been segmented using the method of this dissertation. (a) $\lambda = 0$, most detailed. (b) $\lambda = 0$, next coarser level. (c) $\lambda = 1$, most detailed. (d) $\lambda = 1$, next coarser level. With $\lambda = 0$, the discontiguity is all that matters, while with $\lambda = 1$, the cluster whose mean is closest in texture space gets chosen.

expect to see a very noisy segmentation. To some extent, as seen in Figure 5.3, our expectations are met.

However, the segmentation results are not that different. Disregarding contiguity considerations by setting $\lambda = 1.0$ did not cause an overly noisy segmentation. The reason can be gleaned by looking at the higher level of segmentation (Figure 5.3d). We notice that much of the noisiness of the detailed level is gone. Since the K-Means step is followed by a region growing and merging step akin to what happens at each level of the segmentation, the effect of using only the texture energy is muted – the region growing and merging step handles extremely small regions based on contiguity anyway.

We do notice the effect of not using any texture consideration in Figures 5.3a and b. Since the cluster means are not part of the equation, the glare on the water which does have different texture properties from the darker water or land is missed completely.

In Figure 5.4, the effects of segmentation using intermediate values of λ are shown. As can be seen, any one of the mid-range λ's does just as well as any of the others. We can control the smoothness of the resulting segmentation using the value of λ, but not by much.

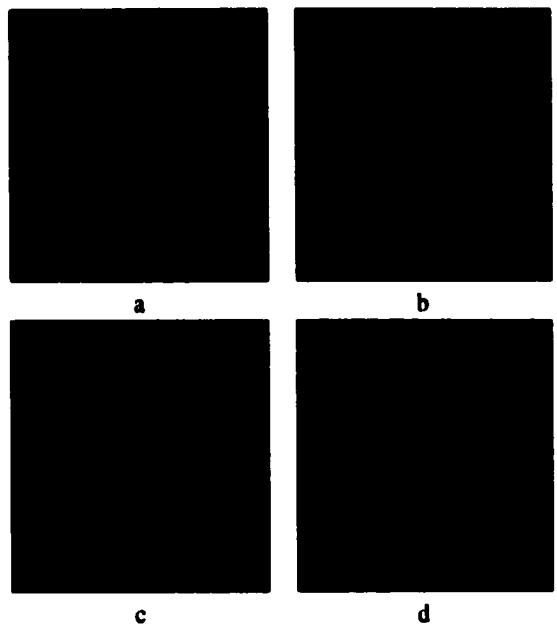


Fig. 5.4: The value of λ doesn't matter much. The most detailed segmentation for these values of λ is shown: (a) $\lambda = 0.2$ (b) $\lambda = 0.4$ (c) $\lambda = 0.6$ (d) $\lambda = 0.8$

5.3 Choice of Texture Vectors

The starting point for the hierarchical texture segmentation algorithm discussed in this dissertation is a vector of measurements taken in the neighborhood of a pixel. This vector of measurements is associated with that pixel. In the literature, there is little consensus on which set of measurements are optimal, or what the standards for the choice of measurements should be. The suggested approach, as discussed in Section 3.2, is to include as many characteristics as possible and to utilize a dimensionality reduction algorithm to choose the characteristics that will actually be used in the segmentation.

In this section, we will demonstrate the results of segmenting an image taken by the Multi-Angle Imaging Spectro-Radiometer (MISR) instrument on NASA's Terra satellite of a dust storm in the Canary Islands. The image is shown in Figure 5.5a. Notice that the image possesses natural texture – the Saharan winds at the bottom of the image and the ocean ripples on the right hand side form a textured backdrop to the islands themselves.

The image is segmented using:

1. The textural measurements listed in Section 3.2 over a 7x7 neighborhood. See Figure 5.5.

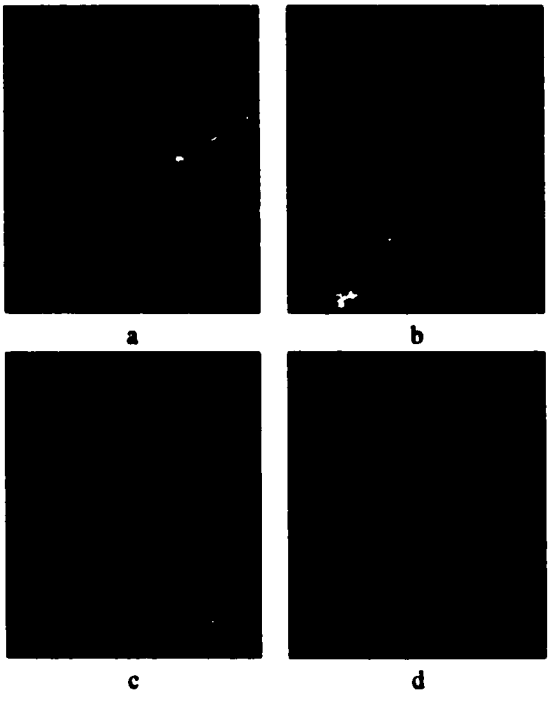


Fig. 5.5: Segmenting a satellite image using the texture vector described in Section 3.2. (a) An image of a dust storm over the Canary Islands, taken June 7, 2000. (b) Most detailed segmentation. (c) Less detailed segmentation. (d) Coarse segmentation.

2. The Daubechies p=2 wavelet coefficients [98, 99] used as the textural measurements. See Figure 5.6.

3. The Daubechies p=4 wavelet coefficients [98, 99] used as the textural measurements. See Figure 5.7.

4. Using only the mean and variance within a 7x7 neighborhood of the pixel. See Figure 5.8.

5.3.1 Wavelets

Although the components of the texture vectors used in this dissertation were statistics computed in the neighborhood of the pixels, the method of hierarchical K-Means segmentation would be equally applicable to any vector that describes the local texture at a pixel. The wavelet transform of an image (or the Gabor-filtered image) can be used to capture local texture [13, 100]. So, the images can be wavelet filtered and the wavelet coefficients used as the components of the texture vectors. These texture vectors may then be used to segment the given images.

There is an advantage to using wavelet coefficients – the scale of the most detailed segmentation could be

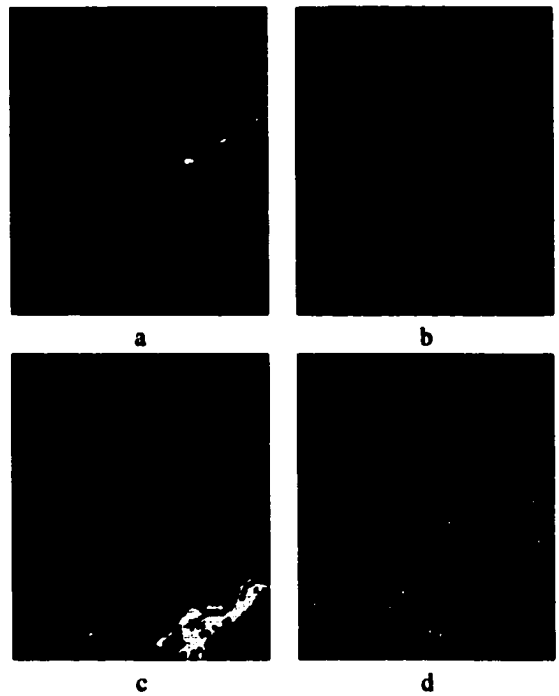


Fig. 5.6: Segmenting a satellite image using the Daubechies p=2 wavelet coefficients as the texture vector. (a) An image of a dust storm over the Canary Islands, taken June 7, 2000. (b) Wavelet transform of the image. (c) Most detailed segmentation. (d) Less detailed segmentation.

controlled by the appropriate choice of the coefficients. In combination with the scale-based measures introduced in Section 6.2.2, such a segmentation scheme can be used to fine-tune the hierarchical segmentation process.

As can be seen, there are differences in the segmented outputs, with small advantages and disadvantages accruing to one method or the other. The number of regions at each iteration using each of these methods to compute texture vectors is shown in Figure 5.9. On the whole, any of these textural measurements could be used to satisfactory results. This might just be a characteristic of the type of image – in Figure 5.10, the result of segmenting the image using a scalar value – the mean pixel value in a 7x7 neighborhood – is shown. Hence, we caution that this is an observation made solely on the MISR image segmented. In general, experimenting with various choices of texture vectors is suggested.

What effect does the neighborhood size play? We would expect that with a tight neighborhood, we would get a large number of small regions, and that with a large neighborhood, we would get a fewer number of larger regions. That expectation is borne out if we segment the MISR image using the mean and variance

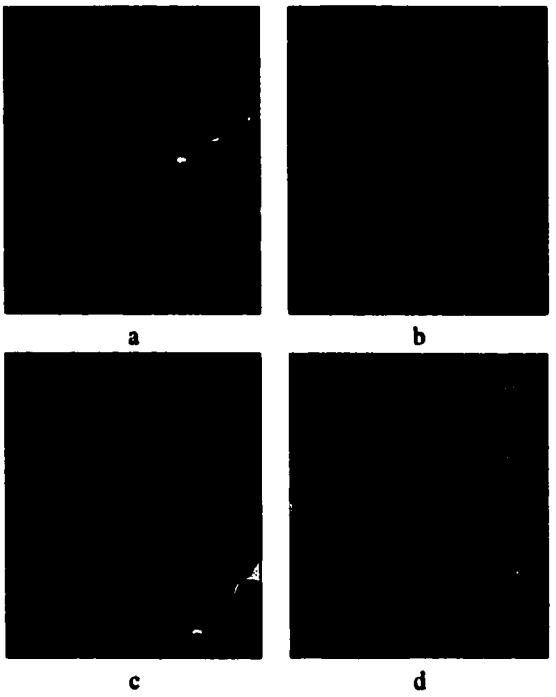


Fig. 5.7: Segmenting a satellite image using the Daubechies p=4 wavelet coefficients as the texture vector. (a) An image of a dust storm over the Canary Islands, taken June 7, 2000. (b) Wavelet transform of the image. (c) Most detailed segmentation. (d) Less detailed segmentation.

computed in varying neighborhood sizes. See Figures 5.11 and compare to Figure 5.8. Notice, from Figure 5.12, that increasing the neighborhood size beyond about 7x7 brings no further decrease – the scale of the regions in the image has probably been reached.

5.4 Minimum Cluster size

We do not accept clusters of pixels that are too few to yield reliable statistics, requiring at least 10 pixels. We would expect that if we allow smaller clusters, we would get a larger number of smaller regions at the most detailed scale. As shown in Figures 5.14 and 5.13, that is exactly the case. Since the method is hierarchical, however, a cluster size that is too small does not unduly affect the segmentation.

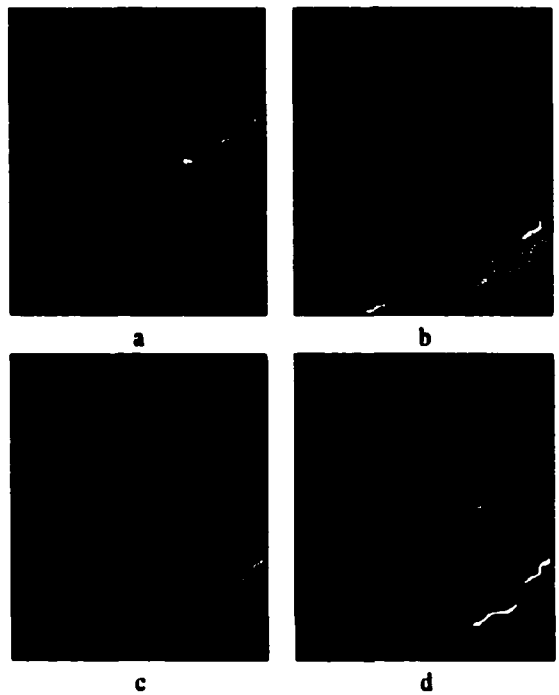


Fig. 5.8: Segmenting a satellite image using a texture vector comprised only of the first three elements of the texture vector described in Section 3.2. (a) An image of a dust storm over the Canary Islands, taken June 7, 2000. (b) Most detailed segmentation. (c) Less detailed segmentation. (d) Coarse segmentation.

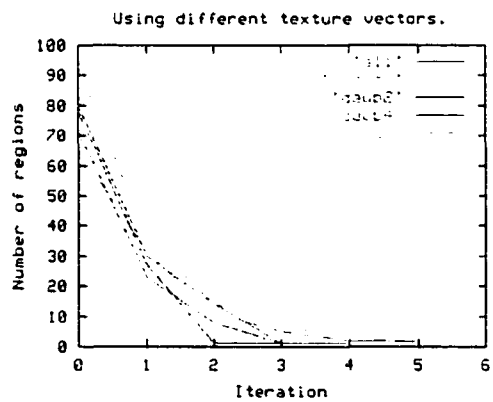


Fig. 5.9: The number of regions at each iteration using the various texture vectors discussed.

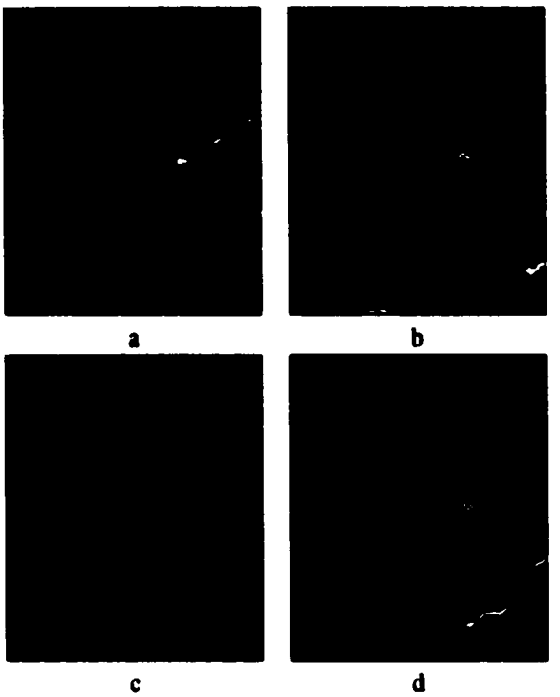


Fig. 5.10: Segmenting a satellite image using just the mean value within a 7x7 neighborhood. (a) An image of a dust storm over the Canary Islands, taken June 7, 2000. (b) Most detailed segmentation. (c) Less detailed segmentation. (d) Coarse segmentation.

5.5 Interpolated images

We do not accept clusters of pixels that are too few to yield reliable statistics, requiring at least 10 pixels. One way to segment even smaller regions (without identifying noisy pixels as separate regions) is to use a pseudo-high resolution form of the original images. Since interpolation increases the size of the image, 10 pixels in the interpolated image covers a smaller geographical area than 10 pixels in the image before interpolation. It is possible, therefore, that we can segment smaller regions that the 10 pixel limit would suggest.

However, interpolation methods smooth the image. Hence, the effect of segmenting interpolated images may be contrary to our goal – due to smoothing, small regions may become indistinguishable from their surroundings. It all depends on the image under consideration.

Regardless of which interpolation method is used (See Figures 5.15 and 5.16), we find that the result of segmenting an interpolated image is to find smaller regions. Notice for example, that the shore-line considered too small to segment in Figure 4.5 is segmented as a separate region at the detailed segmentation levels.

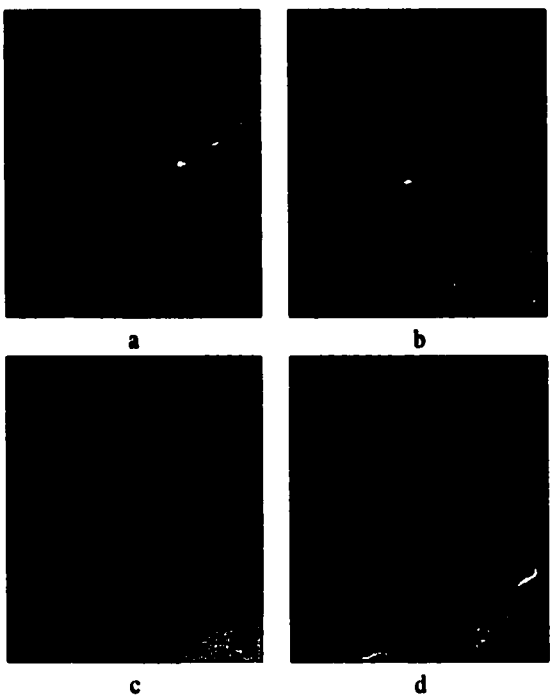


Fig. 5.11: Segmenting a satellite image using statistics computed in neighborhoods of varying sizes. (a) An image of a dust storm over the Canary Islands, taken June 7, 2000. (b) Most detailed segmentation with a neighborhood size of 3. (c) Most detailed segmentation with a neighborhood size of 5. (d) Most detailed segmentation with a neighborhood size of 7.

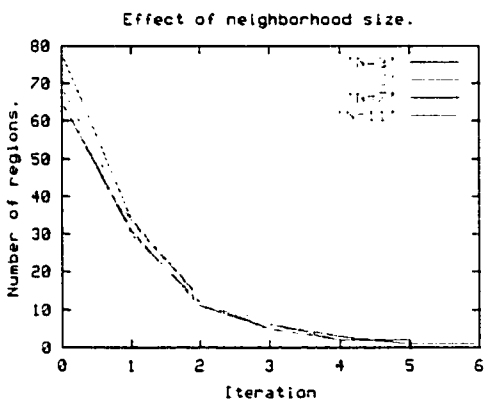


Fig. 5.12: The number of regions at each iteration using texture vectors computed at different neighborhood sizes.

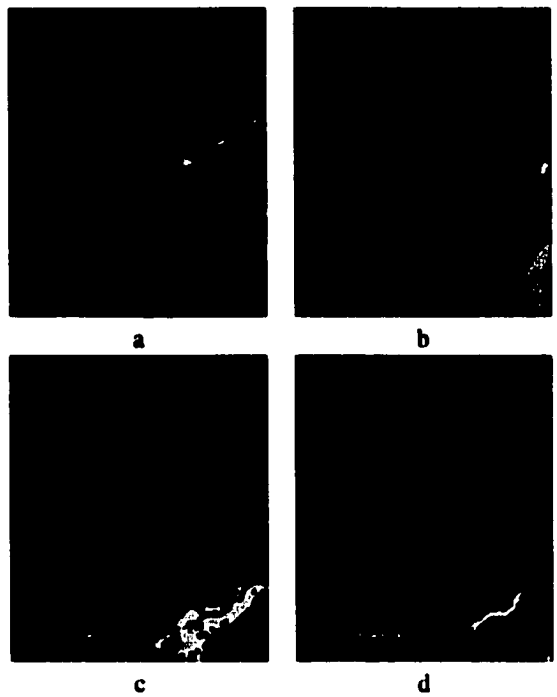


Fig. 5.13: Segmenting a satellite image rejecting regions smaller than a certain number of regions. The Daubechies p=2 wavelets were used. (a) An image of a dust storm over the Canary Islands, taken June 7, 2000. (b) Rejecting regions smaller than 5 pixels. (c) Rejecting regions smaller than 10 pixels. (d) Rejecting regions smaller than 20 pixels.

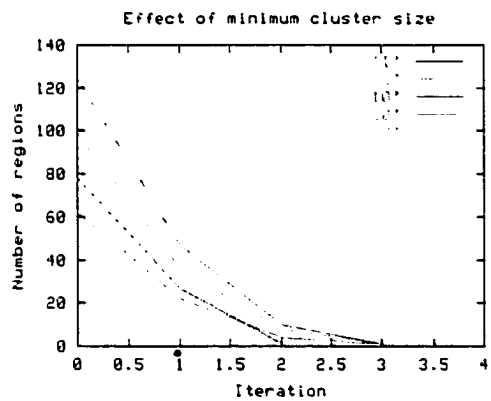


Fig. 5.14: The number of regions at each iteration when rejecting regions smaller than a certain size.

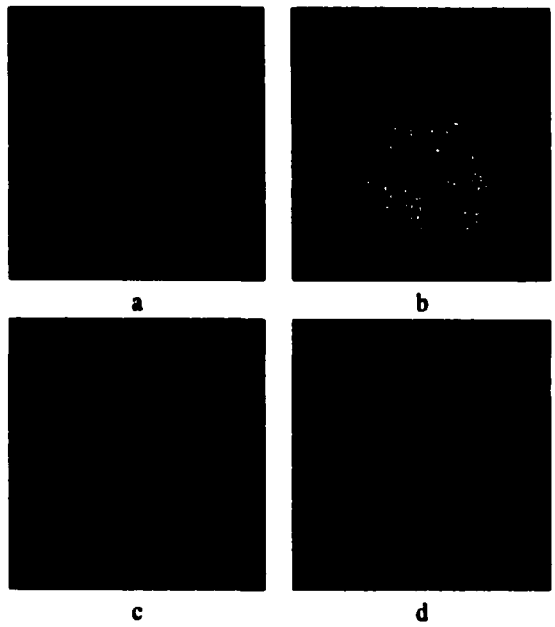


Fig. 5.15: Segmenting a spline-interpolated image. Compare with Figure 4.5 and Figure 5.16. (a) Aerial image of San Francisco after interpolation. (b) Most detailed segmentation. (c) Coarser segmentation. (d) Very coarse segmentation.

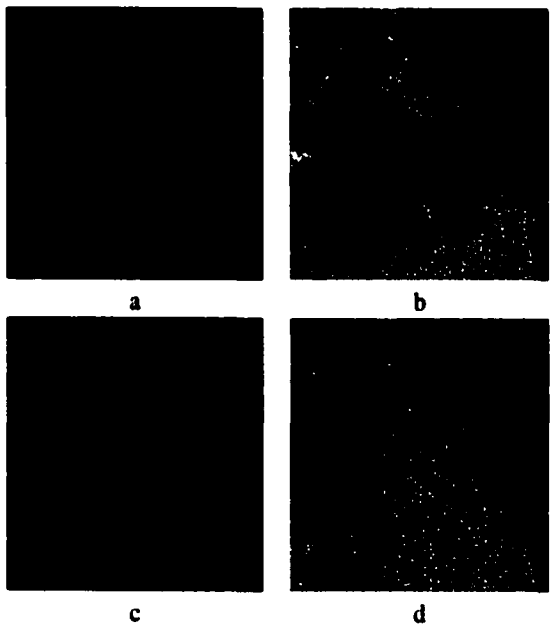


Fig. 5.16: Segmenting a pseudo-high resolution image using the method of [101]. Compare with Figure 4.5 and Figure 5.15. (a) Aerial image of San Francisco after interpolation. (b) Most detailed segmentation. (c) Coarser segmentation. (d) Very coarse segmentation.

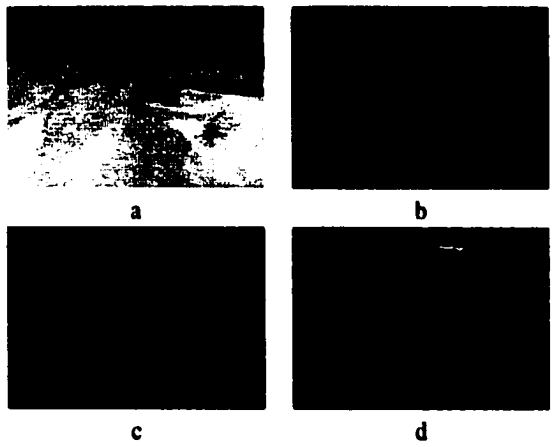


Fig. 5.17: Segmenting a interpolated image. (a) A satellite weather image after interpolation. (b) Most detailed segmentation. (c) Coarser segmentation. (d) Most detailed segmentation of original image.

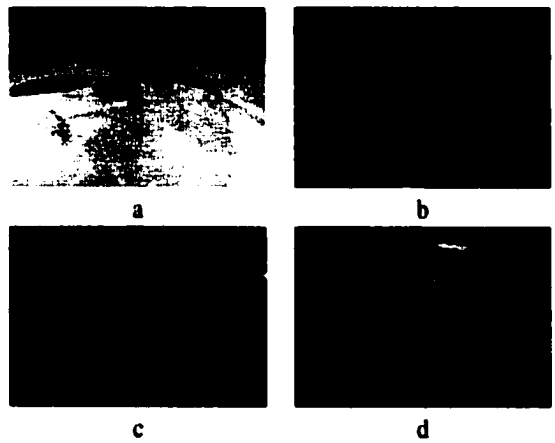


Fig. 5.18: Segmenting a pseudo-high resolution image using the method of [101]. (a) A satellite weather image after interpolation. (b) Most detailed segmentation. (c) Coarser segmentation. (d) Most detailed segmentation of original image.

For satellite weather images, where the textural difference between regions is very small, using interpolated regions has a detrimental effect. As shown in Figures 5.17 and 5.18, the effect of using interpolated images is to degrade the segmentation result.

6. RESULTS AND CONCLUSIONS

6.1 Summary of Results

The method introduced in this dissertation is faster, as shown in Table 4.1, than existing segmentation methods when segmenting real-world images. When measured over synthetic images constructed based on statistical texture or based on strips of Brodatz textures, as shown in Tables 4.4 and 4.2, the method of this dissertation yields a more accurate segmentation than the other techniques considered.

Qualitatively, the method of this dissertation produces segmentation results that are much better than existing techniques when applied to real-world images, whether they are building photographs, terrain shots or medical images. We propose that this is because these scenes are organized in terms of different objects whose pixels values belong to different normal distributions. Since we cluster hierarchically based both on the statistical distribution of pixel values and on spatial proximity, our technique produces good results.

For other types of texture, notably for structural texture images such as those of Brodatz [29], statistical texture vectors of the kind used in this dissertation are most likely not the best choice. It is conceivable that with a different choice of texture vector, the underlying idea of hierarchical K-Means segmentation may perform well. In any case, the statistical texture vectors used in this dissertation are not universal.

Even on real-world images of the kind our algorithm performs very well on, some limitations are noticeable. In Figure 6.1, the result of segmenting an image of an outdoor flowerbed is shown. Here, we notice one of the drawbacks of the technique – since we are not using contextual information in the segmentation (only the textural vectors), the region merging merges the ground which is closer texturally to the wall. A region merge that took context also into account might choose to merge the regions corresponding to the flowers and the flowerbed wall.

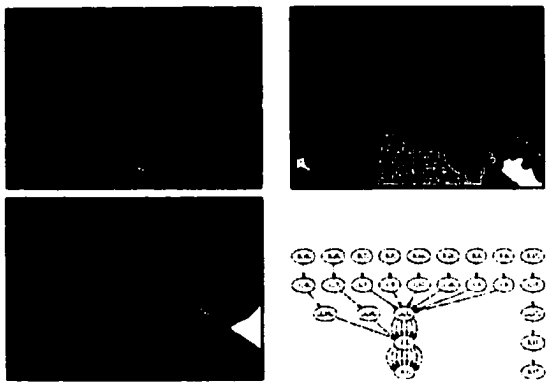


Fig. 6.1: Problem case: Left to right, top to bottom: (a) an image of an outdoor flower pot. (b) The most detailed segmentation of the image. (c) Segmentation at a coarser level. Because the texture features of the walls of the flowerbed are close to the texture features of the ground, the two regions are merged together in the second stage. (d) A hierarchical representation of the results of segmentation.

6.2 Topics for Further Research

The method described in this dissertation can be extended in several ways. Possible avenues for research are described in this section.

6.2.1 Continuous update

In the method described above, the most detailed segmentation is achieved through an iterative method of K-Means clustering. The cluster means are updated not when a new pixel is added or removed from a cluster but only after one complete pass through the image. It is possible to continuously update the cluster means, by storing the sum of the texture vectors and the number of vectors contributing to the cluster. Then, when a pixel is moved from one cluster to another, the sum of the first cluster is reduced by that vector and that of the second cluster is increased by the same vector.

Whether this would cause any improvement in the clustering result is unclear. Since the cluster means are more up-to-date, the clustering result might be more accurate. On the other hand, the order in which the pixels of an image are considered become very important since there are two parts to Equation 3.13 and the second part, $d_c(k)$, depends on the labels of the pixels in the neighborhood of a pixel. Thus, it is conceivable that a method that continuously updates the cluster means (and has to, therefore, also continuously update the labels of the pixels) will be extremely dependent on the order in which the pixels of the image are traversed.

6.2.2 Scale-based measures

In Section 3.4, the method of hierarchical segmentation involved the iterative use of inter-cluster distances between pairs of clusters to arrange the clusters formed at the most detailed segmentation into a hierarchical tree. The inter-cluster distances were computed by computing the Euclidean distance between the mean texture vectors of the two clusters.

At coarser segmentations, the texture vector computed on the original image is not a good representation of the local texture. A better technique is to compute a set of wavelet coefficients as in Section 5.3.1 and use different sets of coefficients at different levels in the hierarchical segmentation process. Then, the inter-cluster distance between the same two clusters when computed at different levels of the process would be different because different sets of texture vector components would have been used at the different levels.

A careful choice of the wavelet coefficients used at each of the scales could be used to yield application-specific segmentations.

6.2.3 Merging criterion

In this dissertation, the only criterion for merging two regions was based on the mean texture vectors of the pixels in the two regions, merging whenever the Euclidean distance between the vectors was less than some globally chosen threshold. We can justify this on the basis that the two regions are close to each other statistically, and can therefore be assumed to belong to the same region. A less justifiable approach, but one we have noticed works quite well, is to merge regions based purely on a globally chosen size criterion, and to increase this size threshold (instead of the inter-cluster distance threshold) with each iteration.

The result of segmenting the building image from [1] is shown in Figure 6.2.

The results are more appealing than those using the inter-cluster distance shown in Figure 4.1. What type of images could justifiably use the size-based criterion, and how these image types can be adaptively recognized by the algorithm remains a topic for research.

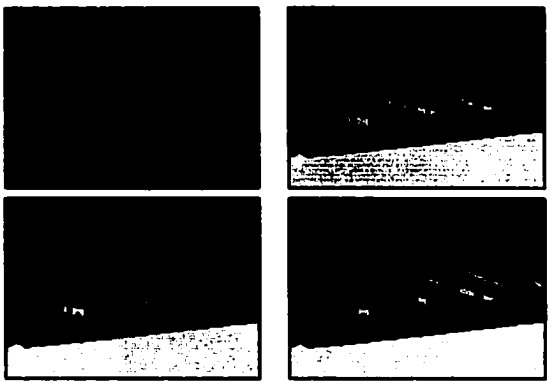


Fig. 6.2: Multiscale segmentation: The photograph of a building on the top left has been segmented using a modification of the multiscale segmentation algorithm described in this paper. A size-based criterion is used for doing the hierarchical merging, instead of the inter-cluster distance criterion proposed here.

6.2.4 Measure of segmentation accuracy

In Section 4.4, we introduced a way to compute the accuracy of a segmentation algorithm. To recap, an image comprised of various textures, $1, 2, ...N$, was presented to the segmentation algorithm. The segmentation algorithm then produces a labeled image $1, 2, ..M$, where each pixel is assigned one of the M labels. We can then form a confusion matrix defined on every pair of labels i and j as:

$$C_{ij} = \left\{ \begin{array}{c|cc} & j & \bar{j} \\ \hline i & a_{ij} & b_{ij} \\ \bar{i} & c_{ij} & d_{ij} \end{array} \right\} \tag{6.1}$$

In the confusion matrix, the number a_{ij} is the number of pixels belonging to the ith texture that have been segmented and identified as belonging to the jth region. The number b_{ij} captures the number of pixels belonging to the ith texture that have identified to belong to a region other than j. The number c_{ij} is the number of pixels not belonging to the ith texture that are identified as belonging to the jth region. The number d_{ij} captures the rest of the pixels.

Unlike a texture classification algorithm, the choice of the right j is not part of the metrics of a segmentation algorithm. We therefore developed a way of computing segmentation accuracy independent of the texture classification.

In Section 4.4, we followed image processing tradition in using percent-correct (PCC) or the classification rate. This overstates the performance of algorithms since the classification rate, defined by:

$$S = \frac{a+d}{a+b+c+d} = \frac{1}{1+\frac{b+c}{a+d}} \tag{6.2}$$

is close to 1 even for unskilled algorithms that classify every candidate as an non-event ($b = d = 0$) if the number of pixels belonging to a non-events ($a + b$) is much higher than the number of events ($c + d$) [102]. The Heidke Skill Score [103, 104] on the other hand compares the performance of the algorithm taking into account the a priori probability of the occurrence of an event. The comparison is against a random classifier that takes into account not the current textural measurement, but only the probability of occurrence of a texture. The choice of HSS rather than PCC is important because we consider every possible value of j, and for some of them the extreme situation will apply.

By using the $\max_{j=1,2,\ldots,M} HSS_{ij}$, we choose the Heidke Skill Score of the region that best matches the texture i. To reduce the vector of Heidke Skill Scores that we get, one for each texture in the input test image, to a scalar measure, we take the worst performance, the minimum. An acceptable alternative could be to take the average over the various textures.

The measure we propose, then, is computed from the "true" label-image and the segmentation output as:

$$m = \min_{i=1}^{K} \max_{j=1}^{M} HSS_{ij} \tag{6.3}$$

The Heidke Skill Score, HSS, is defined [103] as:

$$HSS_{ij} = \frac{2(d_{ij} * a_{ij} - c_{ij} * b_{ij})}{(d_{ij} + c_{ij})(c_{ij} + a_{ij}) + (d_{ij} + b_{ij})(b_{ij} + a_{ij})} \tag{6.4}$$

on the confusion matrix of the pair of labels ij.

6.3 Contributions of dissertation

A novel method of performing multiscale segmentation of images using texture properties has been introduced. Various methods of segmentation at a single scale, including texture segmentation, were described and compared with the K-Means clustering of texture vectors used in this dissertation.

An objective way to measure the accuracy of texture segmentation algorithms independent of texture classification was developed and used to compare different texture segmentation algorithms.

Multiscale segmentation in the context of the segmented regions themselves was introduced. This new approach to multiscale segmentation was incorporated into the K-Means clustering technique as a steady relaxation of inter-cluster distances.

Our multiscale segmentation approach was demonstrated on several families of real-world images. It was shown that quality of the segmented results at the different scales was significantly better, in terms both of speed and of accuracy, than existing approaches.

BIBLIOGRAPHY

[1] J. van Hateren and A. van der Schaar, "Independent component filters of natural images compared with simple cells in primary visual cortex," in *Proc. R. Soc. Lond.*, vol. B 265, pp. 359–366, 1998.

[2] R. Haralick, "Image segmentation survey," in *Fundamentals in Computer Vision* (O. E. Faugeras, ed.), vol. 117, pp. 209–223, Cambridge Univ. Press, 1983.

[3] R. Haralick and L. Shapiro, "Survey: Image segmentation techniques," *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 100–132, 1985.

[4] R. Haralick, "Statistical and structural approaches to texture," *Proc. IEEE*, vol. 67, pp. 786–804, May 1979.

[5] G. Smith, *Image Texture Analysis using Zero Crossings Information*. PhD thesis, U. of Queensland, Brisbane, 1998.

[6] A. Bovik, M. Clarke, and W. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Trans. on Pattern Anal. and Machine Intell.*, vol. 12, pp. 55–73, 1990.

[7] H. Greenspan, R. Goodman, R. Chellappa, and C. Anderson, "Learning texture discrimination rules in a multiresolution system," *IEEE Trans. on Pattern Anal. and Machine Intell.*, vol. 16, no. 9, pp. 894–901, 1994.

[8] A. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, New Jersey: Prentice Hall, 1989.

[9] X. Huang, Y. Akiri, and M. Jack, *Hidden Markov Models for Speech Recognition*. Edinburgh: Edinburgh Univ. Press, 1990.

[10] R. Chellappa, "Two dimensional discrete gaussian markov random field models for image processing," in *Progress in Pattern Recognition* (L. Kanal and A. Rosenfeld, eds.), pp. 79–112, Amsterdam: North Holland, 1985.

[11] C. Nikias, "Higher order spectral analysis," in *Advances in Spectrum Analysis and Array Processing* (S. Haykin, ed.), pp. 326–365, Englewood Cliffs, NJ: Prentice Hall, 1991.

[12] P. Chen and T. Pavlidis, "Segmentation by texture using correlation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 5, pp. 64–69, Jan. 1983.

[13] A. Jain and F. Farroknia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[14] D. Dunn and W. E. Higgins, "Optimal Gabor filters for texture segmentation," *IEEE Trans. Image Proc.*, vol. 4, pp. 947–964, July 1995.

[15] T. P. Weldon and W. E. Higgins, "An algorithm for designing multiple Gabor filters for segmenting multi-textured images," in *Proc. IEEE Int'l. Conf. Image Proc.*, (Chicago, IL), October 4-7 1998.

[16] J. Havlicek, "The evolution of modern texture processing," *Elektrik*, vol. 5, no. 1, pp. 1–28, 1997.

[17] B. Julesz, "Experiments in the visual perception of texture," *Sci. Amer.*, vol. 232, pp. 84–92, 1975.

[18] D. Gabor, "Theory of communication," *J. Inst. Elec. Eng. London*, vol. 93, no. III, pp. 429–457, 1946.

[19] S. Marcelja, "Mathematical description of the responses of simple cortical cells," *J. Opt. Soc. Am.*, vol. 70, no. 11, pp. 1297–1300, 1982.

[20] A. Jain and K. Karu, "Learning texture discrimination masks," *IEEE Trans. on Pattern Anal. and Machine Intell.*, vol. 18, pp. 195–205, 1996.

[21] A. Rao and B. Schunck, "Computing oriented texture fields," *Proc. IEEE Comput. Soc. Conf. Comput. Vision Patt. Recog.*, pp. 61–68, Jun 1989.

[22] M. Turner, "Texture discrimination by Gabor functions," *Biol. Cybern.*, vol. 55, no. 2, pp. 71–82, 1986.

[23] Y. Zeevi and M. Porat, "Combined frequency-position scheme of image representation in vision," *J. Opt. Soc. Am. A*, vol. 1, p. 1248, Dec 1984.

[24] M. Porat and Y. Zeevi, "Localized texture processing in vision: Analysis and synthesis in the Gaborian space," *IEEE Trans. Biomed. Engg.*, vol. 36, pp. 115–129, Jan. 1988.

[25] A. Bovik, "Analysis of multichannel narrow-band filters for image texture segmentation," *IEEE Trans. Signal Proc.*, vol. 39, pp. 2025–2043, Sep. 1991.

[26] M. Kass and A. Witkin, "Analyzing oriented patterns," *Comput. Vision, Graphics, Image Proc.*, vol. 37, pp. 362–385, 1987.

[27] J. Havlicek, D. Harding, and A. Bovik, "The multi-component AM-FM image representation," *IEEE Trans. Image Processing*, vol. 5, pp. 1094–1100, 06 1996.

[28] A. Bovik, N. Gopal, T. Emmoth, and A. Restrepo, "Localized measurement of emergent image frequencies by Gabor wavelets," *IEEE Trans. Info. Theory*, vol. 38, pp. 691–712, Mar. 1992.

[29] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*. Toronto: Dover Publishing Co., 1966.

[30] J. Johnson, P. Mackeen, A. Witt, E. Mitchell, G. Stumpf, M. Eilts, and K. Thomas, "The storm cell identification and tracking algorithm: An enhanced WSR-88D algorithm," *Weather and Forecasting*, vol. 13, pp. 263–276, June 1998.

[31] K. Browning, C. Collier, P. Larke, P. Menmuir, G. Monk, and R. Owens, "On the forecasting of frontal rain using a weather radar network," *Monthly Weather Review*, vol. 110, pp. 534–552, 1982.

[32] A. Witt, M. Eilts, G. Stumpf, J. Johnson, E. Mitchell, and K. Thomas, "An enhanced hail detection algorithm for the WSR-88D," *Weather and Forecasting*, vol. 13, pp. 286–303, 1998.

[33] M. Wolfson, B. Forman, R. Hallowell, and M. Moore, "The growth and decay storm tracker," in *8th Conference on Aviation*, (Dallas, TX), pp. 58–62, Amer. Meteor. Soc., 1999.

[34] C. Morel, F. Orain, and S. Senesi, "Automated detection and characterization of MCS using the meteosat infrared channel," in *Proc. Meteo. Satellite Data Users Conf.*, (Brussels), pp. 213–220, 1997.

[35] T. Jones, *Improving the Precision of Leg Ulcer Area Measurement with Active Contour Models*. PhD thesis, U. Glamorgan, May 1999.

[36] L. Liu and S. Sclaroff, "Deformable shape detection and description via model-based region grouping," in *IEEE Conf. on Comp. Vision and Patt. Recog.*, vol. 2, pp. 21–27, June 1999.

[37] C. Brice and C. Fennema, "Scene analysis using regions," in *Computer Methods in Image Analysis* (J. Aggarwal, R. Duda, and A. Rosenfeld, eds.), IEEE Computer Society, 1977.

[38] A. Marshall and R. Martin, *Computer Vision, Models and Inspection*. Singapore: World Scientific, 1992.

[39] S. Beucher, "Watersheds of functions and picture segmentation," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, (Paris), pp. 1928–1931, May 1982.

[40] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. on Patt. Anal. and Mach. Intell.*, vol. 13, pp. 583–598, June 1991.

[41] S. Beucher and C. Lantuejoul, "Use of watersheds in contour detection," in *Proc. Int. Workshop Image Proc., Real-Time Edge and Motion Detection/Estimation*, (Rennes, Fr.), Sep 1979.

[42] W. Ma and B. Manjunath, "Edge flow: a framework of boundary detection and image segmentation," in *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, (San Juan, Puerto Rico), pp. 744–749, June 1997.

[43] J. Mao and A. Jain, "Texture classification and segmentation using multiresolution simultaneous autoregressive models," *Pattern Recognition*, vol. 25, no. 2, pp. 173–178, 1992.

[44] C. Kervrann and F. Heitz, "A markov random field model-based approach to unsupervised texture segmentation using local and global spatial statistics," *IEEE Trans. on Img. Proc.*, vol. 4, pp. 856–862, Jun 1995.

[45] P. Andrey and P. Tarroux, "Unsupervised segmentation of Markov Random Field modeled textured images using selectionist relaxation," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 20, no. 3, pp. 252–262, 1998.

[46] G. Cross and A. Jain, "Markov random field texture models," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 5, no. 1, pp. 25–39, 1983.

[47] R. Chellappa, S. Chatterjee, and R. Bagdazian, "Texture synthesis and compression using gaussian markov random field models," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 15, no. 2, pp. 298–303, 1985.

[48] R. Dubes and A. Jain, "Random field models in image analysis," *J. Applied Statistics*, vol. 16, no. 2, pp. 131–164, 1989.

[49] M. Jiang, "Mathematical models in computer vision and image processing." http://vesuvius.radiology.uiowa.edu/ jiangm/courses/mm-cv-ip/vision.html, Nov. 2000.

[50] B. Manjunath and R. Chellappa, "Unsupervised texture segmentation using markov random field models," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 13, no. 5, pp. 478–482, 1991.

[51] D. Geman, C. Geman, C. Graffigne, and D. Pong, "Boundary detection by constrained optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, pp. 609–628, Jul 1990.

[52] J. Blum and J. Rosenblat, *Probability and Statistics*. W.B. Saunders Company, 1972.

[53] P. Chou and C. Brown, "The theory and practice of bayesian image modeling," *Int. J. Comp. Vis.*, vol. 4, pp. 185–210, Feb 1990.

[54] P. Bouthemy and E. Francois, "Motion segmentation and qualitative dynamic scene analysis from an image sequence," *Int. J. Comp. Vis.*, vol. 10, pp. 157–182, Feb 1993.

[55] V. Lakshmanan, V. DeBrunner, and R. Rabin, "Texture-based segmentation of satellite weather imagery," in *Int'l Conference on Image Processing*, (Vancouver), pp. 732–735, Sept. 2000.

[56] K. Gowda, "A feature reduction and unsupervised classification algorithm for multispectral data." *Pattern Recognition*, vol. 17, pp. 667–676, 1984.

[57] J. Theiler and G. Gisler, "A contiguity-enhanced K-Means clustering algorithm for unsupervised multispectral image segmentation," in *Proc. SPIE*, vol. 3159, pp. 108–118, 1997.

[58] C. Chen, J. Luo, and K. Parker, "Image segmentation via adaptive k-mean clustering and knowledge-based morphological operations with biomedical applications," *IEEE Trans. on Image Processing*, vol. 7, pp. 1673–1683, Dec 1998.

[59] A. Yullie, D. Cohen, and P. Hallinan, "Feature extraction from faces using deformable templates," in *Proc. of IEEE Conf. Computer Vision and Pattern Recog.*, pp. 104–109, June 1989.

[60] D. Cooper, "Maximum likelihood estimation of markov-process blob boundaries in noisy images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 4, pp. 372–384, 1979.

[61] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Computer Vis.*, vol. 1, pp. 321–331, 1988.

[62] K. Seidel, M. Schröder, H. Rehrauer, and M. Datcu, "Advanced query and retrieval techniques for remote sensing image databases." http://www.vision.ee.ethz.ch/ rsia/talks/RSL_talk/, Dec. 1999.

[63] C. Bouman and B. Liu, "Multiple resolution segmentation of textured images," *IEEE Trans. on Patt. Anal. and Mach. Intell.*, vol. PAMI-13, no. 2, pp. 99–113, 1991.

[64] P. Thevenaz and M. Unser, "Optimization of mutual information for multiresolution image registration," *IEEE Trans. on Image Proc.*, vol. 9, pp. 2083–2099, Dec 2000.

[65] J. Liu and Y. Yang, "Multiresolution color image segmentation," *IEEE Trans. on Patt. Analy. and Mach. Intell.*, vol. 16, pp. 689–700, July 1994.

[66] D. Tzovaras, S. Vachtsevanos, and M. Strintzis, "Optimization of quadtree segmentation and hybrid two-dimensional and three-dimensional motion estimation in a rate-distortion framework," *IEEE Journal on Selected Areas in Communications*, vol. 15, pp. 1726–1738, Dec. 1997.

[67] A. Jain and R. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, New Jersey: Prentice Hall, 1988.

[68] B. Everitt, *Cluster Analysis*. New York: John Wiley & Sons, 1974.

[69] T. Pappas, "An adaptive clustering algorithm for image segmentation," *IEEE Trans. Signal Processing*, vol. 40, pp. 901–914, 1992.

[70] H. Derin and H. Elliot, "Modeling and segmentation of noisy and textured images using gibbs random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, pp. 39–55, 1987.

[71] A. Solberg and A. Jain, "Texture fusion and feature selection applied to SAR imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 35, pp. 475–479, 3 1997.

[72] K. Etemad and R. Chellappa, "Signal and image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, pp. 1453–1465, Oct 1998.

[73] K. Kufunaga, *Introduction to Statistical Pattern Recognition*. New York: Academic, 2 ed., 1990.

[74] P. Burt, "Fast algorithms for estimating local image properties," *Computer Graph. Image Processing*, vol. 21, pp. 368–382, 1983.

[75] R. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, pp. 179–188, 1936.

[76] M. Unser and M. Eden, "Multiresolution feature extraction and selection for texture segmentation," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 11, no. 7, pp. 717–728, 1989.

[77] H. Choi and R. Baranuik, "Multiscale texture segmentation using wavelet-domain hidden markov models," in *Proc. 32nd Asilomar Conf.*, Nov. 1998.

[78] N. Ahuja, "A transform for multiscale image segmentation by integrated edge and region detection," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 18, no. 12, pp. 1211–1235, 1996.

[79] J. Eledath, J. Ghosh, and S. Chaktravarthy, "Image enhancement using scale-based clustering properties of the radial basis function network," Tech. Rep. UT-CVIS-TR-97-006, Center for Vision and Image Sciences, U. Texas, Austin, 1997.

[80] M. Yang and N. Ahuja, "Recognizing hand gestures using motion trajectories," in *IEEE Conf. on Comp. Vision and Patt. Recog.*, vol. 1, pp. 466–472, Jun 1999.

[81] Z. Liang, H. Pan, Y. Hui, R. Magin, N. Ahuja, and T. Huang, "Automated image registration by maximization of a region similarity metric," *Int. J. Imaging Sys. and Tech.*, no. 6, pp. 513–518, 1997.

[82] I. Kim and H. Yang, "An integrated scheme for image segmentation and labeling based on markov random field model," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 18, no. 1, pp. 69–73, 1996.

[83] L. Bobrowski and J. Bezdek, "C-means clustering with the l_1 and l_∞ norms," *IEEE Trans. Systems, Man and Cybernetics*, vol. 28, no. 3, pp. 545–554, 1991.

[84] G. Taubin, "Estimation of planar curves, surfaces, and nonplanar space defined by implicit equations with application to edge and range segmentation," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 13, pp. 1115–1138, Nov 1991.

[85] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. New York: Wiley Interscience, 1990.

[86] Y. Ohashi, "Fuzzy clustering and robust estimation," in *Ninth Meeting of SAS Users Group Int'l*, (Hollywood Beach, Fla.), 1984.

[87] R. Dave and K. Bhaswan, "Adaptive fuzzy c-shells clustering and detection of ellipses," *IEEE Trans. Neural Networks*, vol. 3, pp. 643–662, May 1992.

[88] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with applications in computer vision," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 21, pp. 450–465, May 1999.

[89] H. Frigui and R. Krishnapuram, "Clustering by competitive agglomeration," *Pattern Recognition*, vol. 30, no. 7, pp. 1223–1232, 1997.

[90] L. Najman and M. Schmitt, "Geodesic saliency of watershed contours and hierarchical segmentation," *IEEE Trans. Patt. Anal. and Mach. Intell.*, vol. 18, pp. 1163–1173, 1996.

[91] T. Hofmann, J. Puzicha, and J. Buhmann, "A deterministic annealing framework for unsupervised texture segmentation," Tech. Rep. IAI-TR-96-2, Institut fr Informatik III, U. Bonn, 1996. http://www-dbv.cs.uni-bonn.de/image/example4.html.

[92] P. Ohanian and R. Dubes, "Performance Evaluation for Four Classes of Textural Features," *Pattern Recognition*, vol. 25, pp. 819–833, 1992.

[93] V. Manian, R. Vasquez, and P. Katiyar, "Texture classification using logical operators," *IEEE Trans. on Image Proc.*, 2000.

[94] G. Smith, "Meastex image texture database and test suite." http://www.cssip.uq.edu.au/staff/meastex/meastex.html, May 1997.

[95] Moving Pictures Experts Group, "Coding of moving pictures and audio." Tech. Rep. ISO/IEC JTC1/SC29/WG11, International Organization for Standardisation, 1997.

[96] J. Kim and T. Chen, "A VLSI architecture for image sequence segmentation using edge fusion," in *International Workshop on Computer Architectures for Machine Perce*, (Padova, Italy), Sep. 2000.

[97] D. Comaniciu and P. Meer, "Robust analysis of feature spaces: Color image segmentation," in *IEEE Conf. on CVPR '97*, (San Juan, Puerto Rico), 1997.

[98] I. Daubechies, *Ten Lectures on Wavelets*. Philadelphia, PA: SIAM, 1 ed., 1992.

[99] S. Mallat, *A Wavelet Tour of Signal Processing*. San Diego, CA: Academic Press, 1 ed., 1998.

[100] C. Bouman and B. Liu, "Multi-resolution segmentation of textured images," *IEEE Trans. on Patt. Anal. and Mach. Intell.*, vol. 13, pp. 99–113, 1991.

[101] M. Yao, *Model-Based Methods for Image Interpolation and Enhancement That Retain Edge Information*. PhD thesis, U. Oklahoma, 1999.

[102] V. Lakshmanan and A. Witt, "Detecting rare signatures," in *Artificial Neural Networks in Engineering ANNIE '97*, (St. Louis, MO), pp. 521–526, ASME Press, 1997.

[103] P. Heidke, "Berechnung des erfolges und der gute der windstarkvorhersagen im sturmwarnungsdienst," *Geogr. Ann.*, vol. 8, pp. 301–349, 1926.

[104] C. Marzban, "Scalar measures of performance in rare-event situations," *Weather and Forecasting*, vol. 13, pp. 753–763, 1998.

[105] J. Wiklund and G. Granlund, "Tracking of multiple moving objects," in *Time-Varying Image Processing and Moving Object Recognition* (V. Cappellini, ed.), pp. 241–250, Elsevier Science Publishers B.V., 1987.

[106] M. Dixon, *Automated Storm Identification, Tracking and Forecasting – A Radar-Based Method*. PhD thesis, University of Colorado and National Center for Atmospheric Research, 1994.

[107] E. Nering and A. Tucker, *Linear Programs and Related Problems*. Boston: Academic Press, Inc., 1993.

[108] V. Lakshmanan, R. Rabin, and V. DeBrunner, "Identifying and tracking storms in satellite images," in *Second Artificial Intelligence Conference*, (Long Beach, CA), pp. 90–95, American Meteorological Society, 2000.

[109] E. Lai, P. Li, C. Chan, M. Chu, and W. Wong, "Pattern recognition of radar echoes for short-range rainfall forecast," in *15th Intl. Conf. on Pattern Recog.*, vol. 4, pp. 299–302, IEEE, 2000.

[110] J. Lee, R. Weger, S. Sengupta, and R. Welch, "A neural network approach to cloud classification," *IEEE Trans. on Geoscience and Remote Sensing*, vol. 28, pp. 846–855, Sep. 1990.

[111] T. Markus and D. Cavalieri, "An enhancement of the NASA team sea ice algorithm," *IEEE Trans. on Geoscience and Remote Sensing*, vol. 38, pp. 1387–1398, May 2000.

[112] S. Narasimhan and S. Nayar, "Chromatic framework for vision in bad weather," in *IEEE Conf. on Comp. Vision and Patt. Recog.*, vol. 1, pp. 598–605, IEEE, 2000.

[113] J. Peak and P. Tag, "Segmentation of satellite weather imagery using hierarchical thresholding and neural networks," *Journal of Applied Meteorology*, vol. 33, no. 5, pp. 605–616, 1994.

[114] H. Shi, "Cloud movement detection for satellite images," in *4th Intl Conf. on Signal Proc.*, vol. 2, pp. 982–985, IEEE, 1998.

[115] L. Zhou, C. Kambhamettu, and D. Goldgof, "Extracting nonrigid motion and 3d structure of hurricanes from satellite image sequences without correspondence," in *IEEE Computer Society Conf. on Computer Vision and Patt. Recog.*, vol. 2, June 1999.

[116] Y. Dong, A. Milne, and B. Forster, "Segmentation and classification of vegetated areas using polarimetric sar image data," *IEEE Trans. Geosci. Remote Sensing*, vol. 39, pp. 321–329, Feb. 2001.

[117] M. Schroder, H. Rehrauer, K. Seidel, and M. Dutcu, "Spatial information retrieval from remote-sensing images – part ii: Gibbs-markov random fields," *IEEE Trans. Geosci. Remote Sensing*, vol. 36, pp. 1446–1455, Sep 1998.

[118] P. Smits and S. Dellepine, "Discontinuity-adaptive markov random field model for the segmetnation of intensity sar images," *IEEE Trans. Geosci. Remote Sensing*, vol. 37, pp. 627–631, Jan. 1999.

[119] T. Crum and R. Alberty, "The WSR-88D and the WSR-88D operational support facility," *Bulletin of the American Meteorological Society*, vol. 74, pp. 1669–1687, 1993.

[120] T. Smith, "Visualization of WSR-88D data in 3d using application visualization software," in *14th Conf. on Weather Forecasting*, 1995.

[121] V. Lakshmanan, R. Rabin, and V. DeBrunner, "Segmenting radar reflectivity data using texture," in *30th International Conference on Radar Meteorology*, (Munich), pp. 53–55, American Meteorological Society, Jul 2001.

[122] K. Browning, "The FRONTIERS plan: A strategy for using radar and satellite imagery for very-short-range precipitation forecasting," *The Meteorological Magazine*, vol. 108, pp. 161–184, 1979.

[123] A. Bellon and I. Zawadzki, "Forecasting of hourly accumulations of precipitation by optimal extrapolation of radar maps," *Journal of Hydrology*, vol. 157, pp. 211–233, 1994.

[124] V. Lakshmanan, *A Heirarchical, Multiscale Texture Segmentation Algorithm for Real-World Scenes*. PhD thesis, U. Oklahoma, Norman, OK, 2001.

APPENDIX

A. TRACKING OF HIERARCHICALLY SEGMENTED IMAGES

In this dissertation, we looked at a method of segmenting natural scenes in such a manner that the result of segmentation is a hierarchical tree of segmented regions. One major advantage of producing such a hierarchical segmentation is the additional constraints that the tree structure can provide downstream processing. We will look here at a tracking method that utilizes the hierarchical tree structure of the segmented results.

A.1 Introduction

Tracking of objects in scenes is useful in a variety of applications [105]. Typically, the images are segmented to find the objects and these objects are then matched across frames. Tracking using segmented regions is, for the most part, a global assignment problem [106, 105]. We seek to find the regions in one frame that match particular regions in an adjacent frame. Then, by noting the location and change in properties of the regions across frames, we may be able to predict the location and properties of the region in the frame(s) to follow.

This assignment may be done in a number of ways. In [106], tracking storms identified on radar imagery is treated as an optimal assignment problem that reduces to a linear programming problem with an optimal solution that may be obtained by the Hungarian method [107]. However, in this method, splits and merges are factored in after the linear programming problem has been solved. Doing so defeats the global optimization since treating two already tagged storms as having merged leaves at least one (sometimes both) the storms in the previous frame untracked. If we incorporate splits and merges, then this becomes a non-linear optimization problem.

Based on the method of [105], a good, fast converging, though not optimal solution can be found by computing the distances (defined in [108] as an aggregation of several fuzzy sets that capture the closeness of the regions spatially, in size, and in value) between each pair of regions and aging the distance leeway across

frames. Each region is then initially associated with the closest object and a penalty assigned for repeated associations. Consecutively iterating over changes to the assignments that reduce the global cost function, and not reusing discarded assignments, yields a quickly convergent, though not optimal, process that is at least better than the initial assignment of "closest" regions.

Either of the above schemes should, ideally, identify splits and merges as completely new cells since the cost function which incorporates size should not allow any other cells to be identified. In practice, however, that is not the case – there are usually several cells of approximately equal size close by The presence of these cells can throw the tracking off. In a global optimization process, especially the second where there is no way to recover from bad choices early on, mistakes cascade and result in very large errors.

We can, however, improve our process if we can do two things: a) use the hierarchical structure to constrain solutions b) recover from bad choices made initially. In addition, we also realize that optimal solutions for parts of the hierarchical structure are very likely to be part of the optimal solution for the entire frame. (This is a relaxation of the topological rule that the shortest path from A to B via P involves the shortest paths from A to P and P to B, except that in this case, the optimal paths AP and PB are not independent and can not, therefore, be evaluated in isolation).

A.1.1 Advantages of Going Hierarchical

The hierarchical structure of the segmented regions in each frame can then be arranged so that the "children" of a region immediately follow it. Then, if a part of this structure has been matched optimally with the hierarchical structure from the previous frame, it is likely that the part of the structure should be retained in the final optimal solution.

The constraining works like this. Suppose that two regions in the segmented results at the lowest level of detail ("top-level regions") are matched across adjacent frames. We can constrain all the regions that are child nodes of one of these regions to match regions that child nodes of the other top-level region in the adjacent frame.

One way that tracking can be done with this constraining is to devise a cost function that rewards regions maintaining their position in the hierarchy [108]. Thus, small features in one frame will tend to be assigned

to small features in another frame, and both are contained within larger features that have also been assigned to each other. Then, an optimization scheme can be used to minimize this cost function (or "energy") on the pair of images.

However, the hierarchical arrangement gives us an additional advantage. In addition to constraining the matching of child regions, we can utilize the movement and change in the textural features of the top-level regions to advect (forecast the location, shape and features of) the child regions in the next frame(s). This can allow us to utilize more robust assignment methods, such as checking overlaps [34], rather than less robust methods such as finding the nearest centroids.

A.2 Multiscale Overlap Tracking

In [34], it was shown that matching regions between frames using the amount of overlap between large scale features works very well. In [30], it was shown that matching small regions between frames worked well only if a closest-centroid approach is used. We theorize that the reason overlap matching is not robust for smaller features is that there is little to no overlap of small features, since the number of pixels they move between frames is greater than their extents.

We will use overlap tracking on the lowest level of detail (the larger features) and then use the resulting motion to advect the hierarchical tree, such that the inter-frame movement due to the larger features is factored out before matching is done on the smaller features. This, in addition to the constraints placed on the assignment of smaller regions by the assignments made with larger regions should make it possible to track smaller regions more robustly.

The steps then are these:

1. Trim the hierarchical trees.
2. At the coarsest segmentation level, do overlap assignment.
3. At higher segmentation levels, two levels – the current segmentation level as well as the one coarser than this one is used.

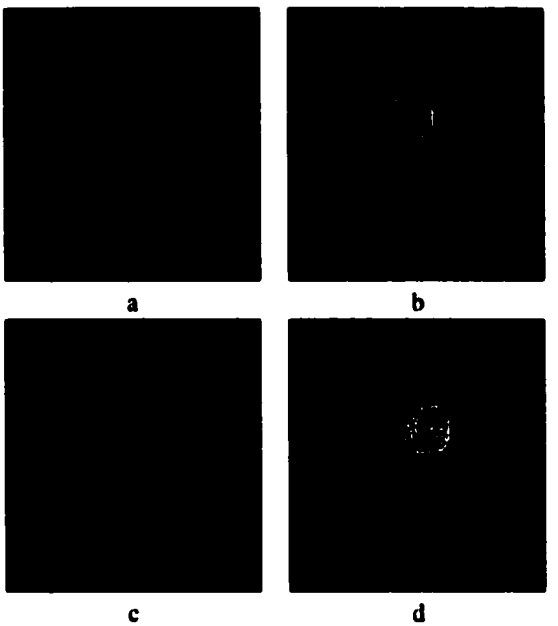


Fig. A.1: Synthetic sequence: Objects whose pixel values were drawn from Gaussian distributions are moved in known directions. The pixel values of overlapping objects are additive.

(a) Advect the previous frame's two levels using the assignments of the coarser level.

(b) Perform a constrained assignment using the advected previous two levels and the current frame's two levels.

We will look at each of these processes in more detail, demonstrating on the synthetic sequence shown in Figure A.1.

The synthetic image in Figure A.1a was drawn as follows. Four objects of known sizes were created. Their pixel values were drawn from known normal distributions (specified by a mean and standard deviation). Then, in each frame of the sequence, these objects were added to the images centered at known locations. If two objects overlapped, the pixel value was the sum of their contributions. The centers of the objects were moved from frame to frame and the process of adding to the image repeated. Thus, the simulated sequence shown in Figure A.1 was created.

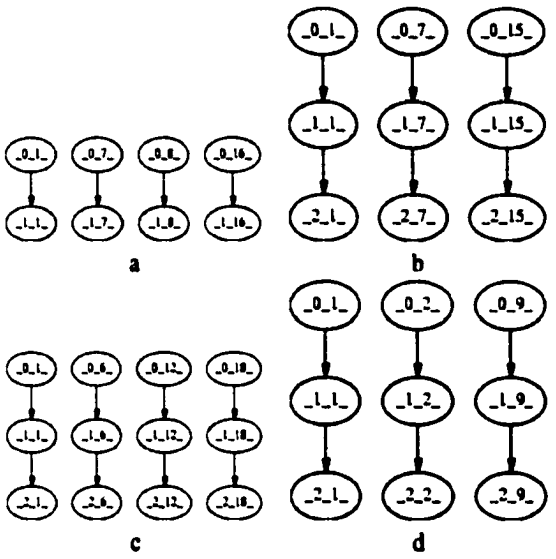


Fig. A.2: Hierarchical trees from synthetic sequence: Note that the number of segmentation levels can differ from frame to frame.

A.2.1 Tree Trimming

Since the hierarchical segmentation method stops only when there are no more regions to merge, adjacent frames could produce segmentation results with different depths. Hence, the deeper tree is trimmed so that the two trees are the same depth. The trimming is carried out from the coarser segmentation end, i.e. the most detailed segmentation is retained but several coarse segmentations may be discarded.

For example, the first two frames of the synthetic sequence are segmented differently (see Figure A.2). Thus, when computing the movement in the second frame of the sequence, the hierarchical tree is retained only up to two levels. When computing the movement in the third frame, however, both the trees have three levels each and thus, no trimming is done.

A.2.2 Overlap Assignment

For each pixel in the current frame, we find the region to which the pixel belongs at the current segmentation level. We also find the region to which the pixel at the identical location in the previous frame and same level belongs. This pair of regions, one in the current frame and one in the previous frame, represents a candidate assignment. For each region in the current frame, a number of candidate assignments are possible.

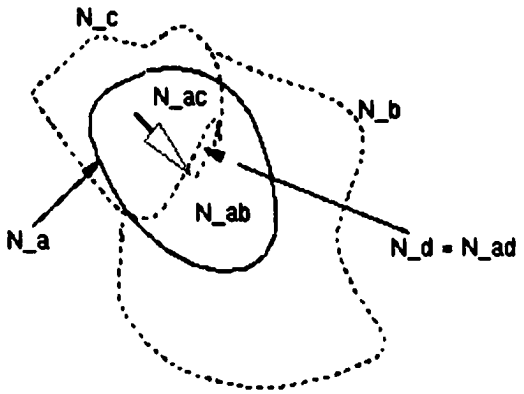


Fig. A.3: Overlap assignment: Dotted lines show regions in the previous frame, while a solid outline of a region in the current frame is drawn. There are three possible candidates for assigning to region a – regions b, c and d. Computing the matches for the assignments using Equation A.1, and choosing the assignment with the maximum match, region a in the current frame is assigned to region c in the previous. The resulting centroid movement is shown with the outline of an arrow.

The assignment that received the maximum match is selected as the best candidate. Note that since we do not differentiate between already selected regions, this method is stable (does not depend on the order in which regions are selected) and can handle splits and merges.

The match, m_{ab}, of a candidate assignment between region b in the previous frame and region a in the current frame is computed as follows:

$$m_{ab} = min(\frac{N_{ab}}{N_a}, \frac{N_{ab}}{N_b}) \tag{A.1}$$

where N_{ab} is the number of pixels that in the current frame belong to a, but in the previous frame belonged to b, while N_a is the number of pixels that belong to a. Similarly, N_b is the number of pixels that belonged to b. The use of this equation is illustrated in Figure A.3.

A.2.3 Advection of previous frame

Two levels in the previous frame are advected using the assignments of the coarser level. For each pixel in the coarse image (of the previous frame – advection happens on images in the previous frame), the corresponding region number in the current frame at the coarse level (since the assignments are at the coarse level only) is

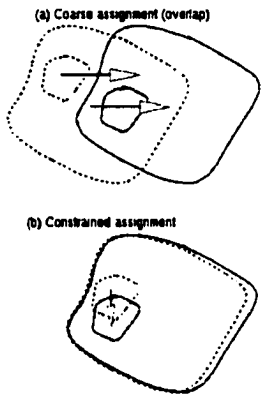


Fig. A.4: Advecting the detailed segmentation result makes constrained overlap assignment possible for smaller, fast-moving regions.

found. Using that region, if it is valid, the movement of this pixel is determined and the location where this pixel would have moved is updated. The pixel is updated with the region number obtained from the detailed segmentation of the previous frame.

The process is illustrated in Figure A.4. Notice that the child nodes (the detailed segmentation result) are moved based on the movement of the parent node (the coarse segmentation result).

The images in Figure A.5, show the result of advection that happens in the third frame of the sequence.

A.2.4 Constrained Assignment

The constrained assignment process assigns region matches in the detailed segmentations subject to the hierarchical position in the coarse segmentation. Thus, the only allowed candidates for a a region are those regions that are contained within the matching region in the other frame. The matching region in this context is the one selected at the coarser segmentation level. The best match among the candidates is selected for the detailed level using Equation A.1.

The determined movements for the third frame of the synthetic sequence are shown pictorially in Figure A.6. The quantitative movements computed match the predetermined amounts to which the different objects were moved.

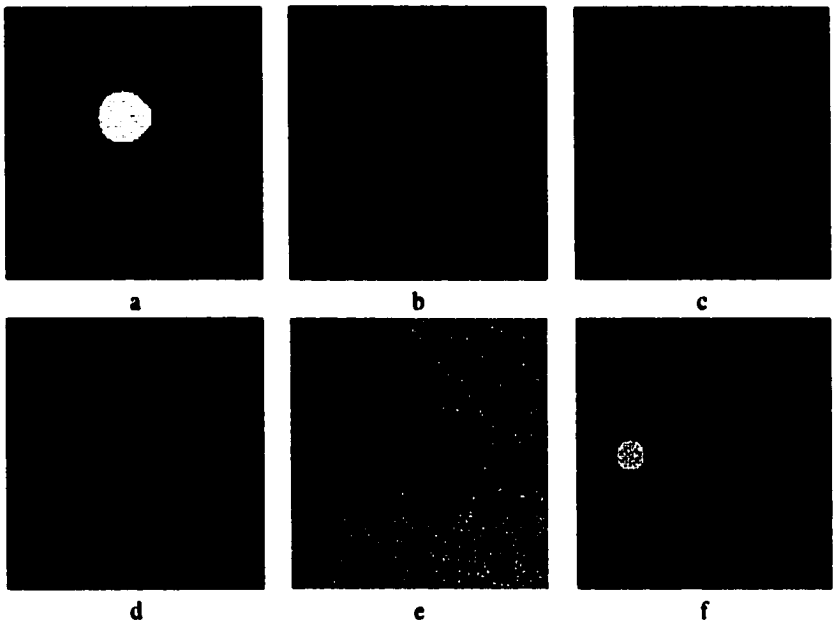


Fig. A.5: Advecting based on coarse movement. (a) Segmented result at coarse level, the previous frame. (b) Segmented result at coarse level, the current frame. (c) The advection of the segmented result in (a) to match (b) based on movement at a coarser level, which is zero. (d) Segmented result at detailed level, the previous frame. (e) Segmented result at detailed level, the current frame. (f) The advection of the segmented result in (d) using the coarse assignments found from the coarser level, i.e. from the process at (c). Notice the improvement in advection quality.

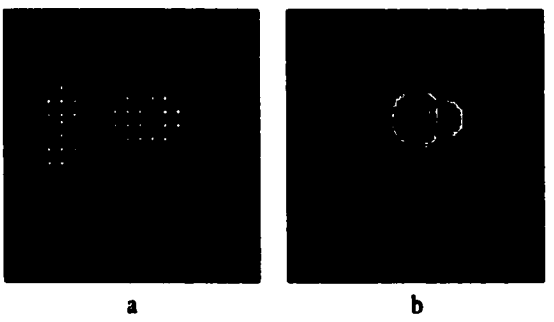


Fig. A.6: The movement of various regions. (a) At the level of the most detailed segmentation, shown such that the heads of the arrows pointing in the direction of movement. (b) At the next coarser level of segmentation, shown such that the outlines of the regions move in the direction of movement.

B. SEGMENTATION AND TRACKING OF WEATHER IMAGES

There are numerous pattern recognition algorithms that have been developed on weather images, such as for rainfall estimates [109] and cloud classification [110]. More commonly, weather in pattern recognition studies is something to be avoided, because the patterns of interest are on the ground [111, 112].

As pointed out in [113, 108, 30], segmentation of weather imagery is a fundamental problem to automated weather analysis. In the meteorological community, the importance of multiscale segmentation has been often noted [30, 33, 108]. However, an electronic search of American Meteorological Society (AMS) journals from 1988 to the time of writing identified only one paper dealing with segmentation. In that paper [113], the authors detail the difficulties that traditional segmentation algorithms have with satellite weather images because of the textural nature of clouds. As a result, a complex technique consisting of a sequence of fixed thresholds, followed by a neural network that decides how and when to prune or merge the resulting regions is proposed [113]. We show here that using a hierarchical technique in combination with a texture segmentation algorithm makes segmentation of satellite weather images possible such that even small cloud features can be identified.

The textural nature of weather imagery makes robust segmentation for storm tracking purposes very difficult. For storm tracking to be useful, the identification and tracking algorithm should be completely automated. The identification algorithm should not require training, i.e. the algorithm should not expect to see examples of all the "objects" it must identify. Storm "cells" (small scale features) should be capable of being identified. Because the notion of scale is natural in the storm tracking context, we would like to add the requirement that storms at various scales be identified, with their hierarchical structure intact. A multiscale tracking algorithm would be a significant improvement over current tracking schemes which concentrate either on small scales [30] or on large scales [33].

In trajectory and motion estimation problems on image sequences, the lack of robustness of the results of texture segmentation has often been noted [80]. Thus, robust segmentation of weather images is a problem that should be addressed by the electrical and computer engineering community. An extensive electronic search of IEEE journals and conference proceedings from 1988 to the time of writing found only two papers dealing with the segmentation of weather images. The first of those, [114], was content to identify pixels of the image that corresponded to precipitation, i.e. a binary segmentation. This would correspond to the coarsest level of segmentation in our hierarchical technique – a level of segmentation that is not very useful. The second paper [55] was a report of the performance of a Markov Random Field-based texture segmentation approach [52] to the problem of segmenting satellite images. The segmentation was found to be not very robust across frames and extremely sensitive to initial conditions [55].

Segmentation has sometimes been part of a larger technique. If some sort of region segmentation is required by the technique in question, such segmentation has been simplistic. In [114, 108], segmentation is performed by degrading the original image by smoothing and downsampling. In [115], hurricane structure is modeled on a "segmentation" where large square areas are assumed to possess the same structure and motion. The reason why weather image segmentation is poorly addressed is that the problems are significant – current segmentation algorithms are not robust enough to handle sequences of weather images [55]. Robust segmentation should be able to provide small changes in the segmented output when the weather pattern in the images changes only slightly. However, image segmentation algorithms, especially texture segmentation ones, tend to be very sensitive to inputs, causing a lot of problems in association between frames. For example, the authors of [115] had to develop a motion estimation method in the absence of correspondence between frames of a sequence. In this dissertation, we introduce a method of texture segmentation that is robust for sequences of weather data.

B.1 Weather Radar Images

Texture segmentation using Markov Random Field (MRF) models has been utilized to segment synthetic aperture radar (SAR) images, mainly because SAR images are characterized by a lot of speckle [116, 117,

118], a problem that the use of neighborhood statistics helps resolve. Another reason for using texture segmentation on SAR imagery is that the same MRF model used for segmentation can also be used for classifying the identified segments [116].

In this dissertation, we propose that texture segmentation can be applied to weather radar data, an application where it has not been used. In weather radar data, especially in cases where there is significant precipitation, the problem of speckle does not arise except in the immediate vicinity of the radar. Hence, traditional texture segmentation provides no significant advantage. In fact, as shown in Figure 4.9e, even a scalar segmentation approach works quite well. What neither the scalar segmentation approaches, for example [30], nor standard texture segmentation approaches [52, 91, 42] can provide is a nested partition of identified segments. The watershed segmentation approach of [90] can provide a nested partition, but does not segment weather data well (See Figure 4.9f). As shown in this dissertation, a hierarchical multiscale segmentation can be achieved by agglomerative K-Means clustering of texture vectors and slow relaxation of the allowed inter-cluster distance.

We wish to segment the reflectivity moment of radar elevation scans. The data have been mapped from polar coördinates into a Cartesian grid tangential to the earth's surface where each pixel is a square area of one kilometer on each side. The pixel values, in dBZ, range from about $-7dBZ$ to about $64dBZ$, with the reflectivity values for some pixels missing. Missing values and all reflectivity values less than $0dBZ$ were thresholded to be $0dBZ$ before the segmentation process.

The radar elevations scans in this study were collected every 5-6 minutes. As with the satellite images, segmentation of the radar sequence should be able to consistently identify the thunderstorms in the images. Ideally, when storms split or merge, the corresponding segmented regions should do the same. A very important requirement is that small changes in the storm structure should be reflected as small changes in the segmented region corresponding to the storm.

The hierarchical segmentation is less important in the case of radar images because with a single storm cell expected to stay more than seven or eight images, it should be possible to track even small storm cells. Still, multiscale hierarchical segmentation would be advantageous and produce more reliable tracking.

We segment the reflectivity moment of radar elevation scans obtained from Doppler Weather Service

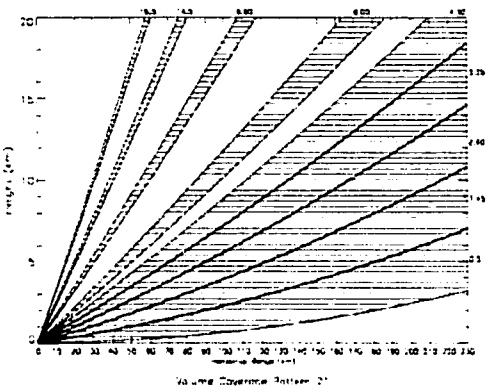


Fig. B.1: Volume Coverage Pattern (VCP) 21 of the WSR-88D, a weather surveillance radar used by the National Weather Service. The volume coverage is shown. The beamwidth is 0.95 degrees and there are 9 elevation scans in this VCP. Figure from [120].

Radar (WSR-88D). The weather surveillance radars used by the National Weather Service scan through thunderstorms starting at a low elevation angle, 0.5° for Volume Coverage Pattern (VCP) 21, and after completing a full 360° azimuthal sweep, progressively increase the elevation angle until an upper limit is reached (19.5° in VCP 21). See Figure B.1 [119, 120].

Depending on the VCP, a new radar volume is collected every 5-12 minutes. The volumes of data in this study were collected every 5-6 minutes. Currently, we segment only the lowest elevation scan of the volume shown in Figure B.1. In future work, we will extend the clustering technique discussed in this dissertation to three dimensions to better segment the data. In a real-time test, several weather algorithms, including our segmentation and tracking algorithm, are run on a single workstation. We found that with some optimization of the processing, it is possible to run the segmentation and tracking algorithm in real-time. The algorithm takes about 30 seconds to process an elevation scan of data.

The radar elevation scan data are polar in nature. We map the data from polar coördinates into a Cartesian grid tangential to the earth's surface where each pixel is a square area of one kilometer on each side. The pixel values, in dBZ, range from about $-7dBZ$ to about $64dBZ$, with the reflectivity values for some pixels missing. Missing values and all reflectivity values less than $0dBZ$ were thresholded to be $0dBZ$ before the segmentation process. Remapping polar data to Cartesian involves trigonometric computations and can therefore be time consuming. One of the optimizations was to precompute the mapping and to apply the

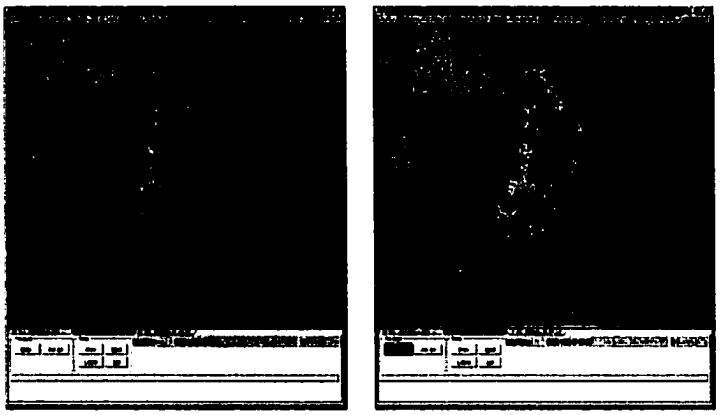


Fig. B.2: Segmentation of a reflectivity elevation scan from the Weather Service Radar in Fort Worth, TX on May 6, 1995. The segmented regions are approximated by their bounding octagons and displayed with thin lines. (a) Most detailed segmentation. (b) Coarser segmentation result.

precomputed mapping to the data for every elevation scan.

The remapped data were then segmented using the K-Means clustering technique described in [121] and in this dissertation. The result of segmentation, at two different scales, is shown overlaid on the original reflectivity data in Figure B.2.

Then, the segmented regions, at all the scales, were tracked using the constrained overlap method described in Appendix A. The resulting regions' boundaries were then approximated using the bounding octagon of those regions, i.e. by the smallest irregular convex octagon whose sides make 0, 45, 90 ...., 315 degrees with the horizontal axis. This is a good representation for small convex regions and is easy to compute (just compute the minimum and maximum values for x, y, x+y and x-y for each pixel in the segment). The result of tracking is, at each scale, the segmented region as well as a forecast region position. In Figure B.3, the forecast position is shown as thin red lines. The forecast position was found to match up with the location of the storm in the next frame of the sequence.

The tracking is shown over a smaller area in Figure B.4. The images on the left (a,c,e) are coarse segmentations. In the images on the right, detailed segmentations (thin lines) as well as the coarse results (thick lines) are shown. The solid lines represent the current objects while the dotted lines denote the forecast locations. Each row represents a frame of the sequence. There are several things of note here – between the first and second frames shown, reflectivity of the northern storm weakened. In the coarse segmentation output, this is

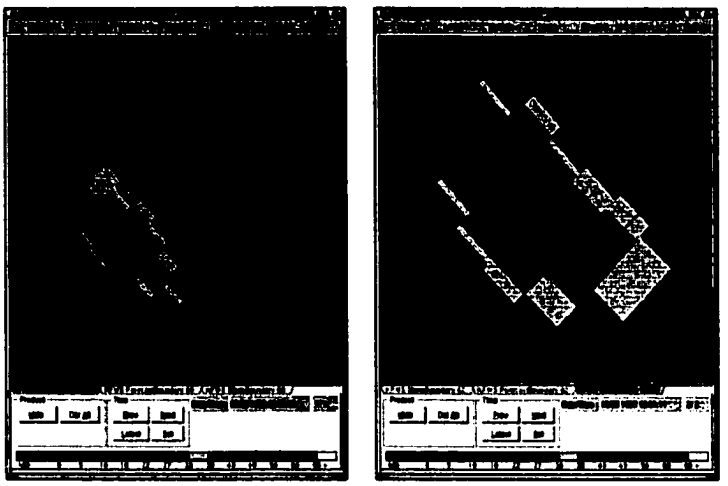


Fig. B.3: Tracking regions through constrained overlap assignment of the regions found by segmentation of a reflectivity elevation scan from the Weather Service Radar in Fort Worth, TX on May 6, 1995. The segmented regions are approximated by their bounding octagons and displayed with thin yellow lines. The forecast locations are shown by thin red lines. (a) Most detailed segmentation. (b) Coarser segmentation result.

noted as a merger of the two storms – they do not differ that much in reflectivity anymore. The storms are, however, tracked independently in the detailed segmentation (See Figure B.4d). The hierarchical segmentation is what makes identification of the larger entity possible. That this hierarchical identification is useful is seen in the next frame of the sequence, when the forecast locations (forecast as a result of considering the two objects together in the coarse segmentation) match up with the true locations seen in Figure B.4e.

B.2 Satellite Infrared Images

We wish to segment the infrared window channel (11μ) of GOES satellite imager data. The images are 200x300 with each pixel representing a $4km \times 4km$. The images are projected onto a plane tangential to the surface of the earth. The satellite data were collected over the continental United States using GOES-10 on March 29, 1998. The pixel values were also mapped from radiance values to temperature in degrees Kelvin before the segmentation. Images of the sequence are available at eight minute intervals.

The sequence of satellite images captures a day of significant thunderstorm activity. Several thunderstorms grow and decay during the day and the temperatures of the cloud tops in the images show corresponding changes both in storm geographical extent and in storm severity. Segmentation of this sequence should

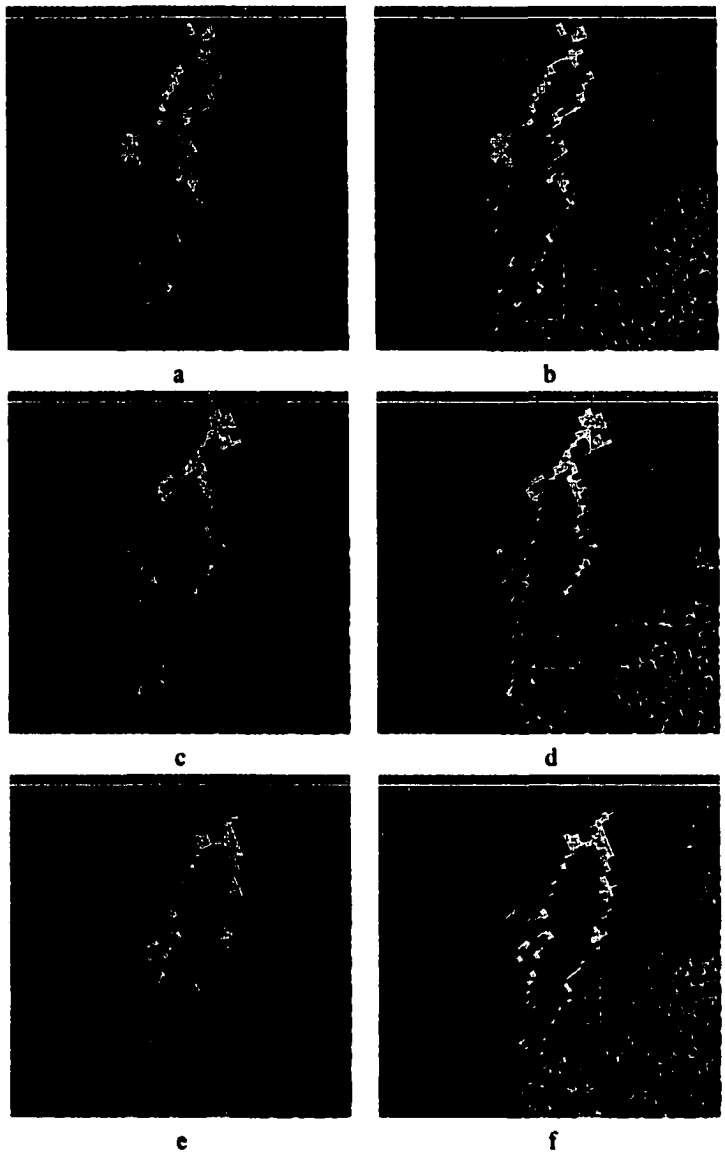


Fig. B.4: Tracking of a portion of a radar reflectivity image. The images on the left (a,c,e) are coarse segmentations and those on the right include detailed segmentations (thin lines) as well as the coarse results (thick lines). The solid lines are the current objects while the dotted lines denote the forecast locations. Each row represents a frame of the sequence. (a) and (b) The first frame of the sequence. There is not much difference between the coarse and detailed segmentation. (c) The second frame of the sequence. The two objects in (a) have merged into a single one because the reflectivity within the northern storm has weakened. (d) The same frame as (c), but a detailed segmentation. Notice the noisiness of the segmentation. In the absence of the coarse segmentation, i.e. (c), the merger would not have been noticed at all. (e), (f) The third frame of the sequence. Notice that the forecast locations of (c) and (d) have been borne out.

be able to consistently identify the thunderstorms in the images. Ideally, when storms split or merge, the corresponding segmented regions should do the same. A very important requirement is that small changes in the storm structure should be reflected as small changes in the segmented region corresponding to the storm.

Studies [122, 123] have shown that a single storm cell grows and decays in under an hour. Therefore, a storm cell can be expected to stay for no more than seven frames of the satellite sequence. However, a line of thunderstorms within which these cells crop up can be expected [122] to persist for up to six hours. Therefore, the segmentation should lend itself to segmenting regions corresponding to larger scale features while identifying small scale features that are contained within the large scale feature.

A single infrared image was segmented using various segmentation methods in the literature. The results are shown in Figures 4.7 and 4.8.

The results of segmentation using the other approaches are pretty bad. This is not surprising because the infrared satellite weather imagery has several characteristics that make it hard to segment: very low dynamic range (from about 225K to 240K) for the regions of interest, poor resolution as compared to the scale of the phenomena of interest, and high pixel value variance, even in the absence of edges. It is instructive to compare the poor performance of these algorithms on the satellite image (see Figure 4.7) with the performance of the same algorithms on radar reflectivity images.

Two successive frames of the satellite data were segmented using the K-Means segmentation algorithm described in this dissertation. The results are shown overlaid on the original infrared temperature data in Figure B.5. Regions are represented by a bounding polygon – either a rectangle or an octagon. Regions whose mean temperatures are less than 210K are represented by octagons and warmer regions are represented by their bounding rectangles. Using different polygon representations serves to draw attention to the colder regions, which are usually storm cells. Thicker lines represent coarser segmentations, while thin lines show the result of detailed segmentation.

As seen from Figure B.5, the segmentation approach is very robust even with all the difficulties that satellite infrared imagery poses. The poor spatial resolution of the satellite image does affects our algorithm in the scale of features that we can detect. Although we can detect features as small as 10 pixels in the image, this translates to about 40 km^2, a mid-size storm cell (although significantly more detailed than what

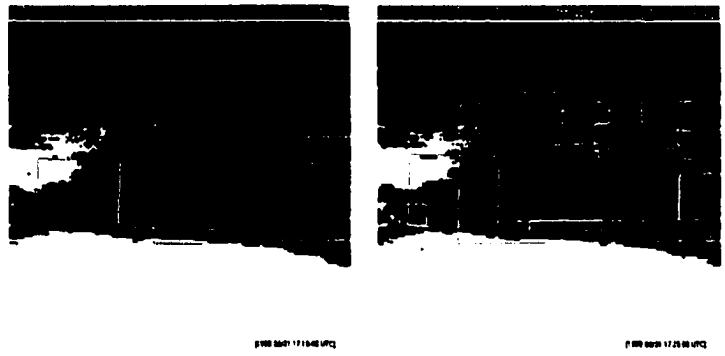


Fig. B.5: Segmentation of two successive frames of a satellite infrared sequence. Regions with mean temperature less than 210K are represented by their bounding octagons while other regions are represented by their bounding rectangles. Thicker lines represent regions at coarser levels of segmentation. The robustness of the segmentation, even on infrared satellite imagery, should be noted.

could be obtained using earlier approaches). The pruning threshold of 10 pixels was set in the algorithm so that any statistics collected are somewhat reliable. One possible way to relax this threshold is by creating a pseudo-high resolution form of the original image, thus getting less square kilometers in the 10-pixel threshold. Unfortunately, on satellite weather images, even a pseudo-high resolution technique [101] introduces unacceptable smoothing [124], resulting in worse performance. A second possibility, one that we have not yet looked into because of the prohibitive cost for a continuously running system, is to obtain weather satellite data that has higher spatial resolution.

A third possibility is to use the multi-channel nature of satellite weather information to form the pixel representation (instead of using a texture vector based on neighborhood statistics). Since we will not be using neighborhood statistics, the threshold of 10 pixels could be dropped.

Instead of using only texture measurements from only the infrared channel, we used texture measurements (mean and variance) computed on four channels corresponding to 3.9, 6.7, 11 and 12 microns (near infrared, water vapor, window and "dirty window" respectively). Since every pixel of the segmented output actually corresponds to four independent measurements (rather than just one), the minimum pruning size in the algorithm can be reduced from about 10 pixels to about 3. The result of using multi-channel information and a lower size threshold is shown in Figure B.6 where it is compared to the segmented result if only the 11 micron image had been used.

Notice that the result of segmenting using all four channels (Figure B.6f) has smaller regions than the

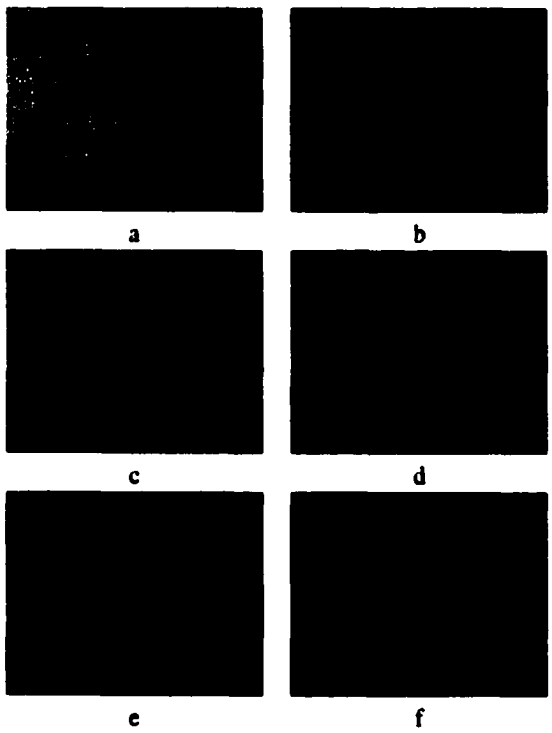


Fig. B.6: Using multi-channel satellite data for segmentation. (a) 3.9 micron infrared (b) 6.7 micron water vapor (c) 11 micron window (d) 12 micron dirty window channels of data. (e) Most detailed segmentation using only the 11 micron image (f) Most detailed segmentation using all four channels.

result that uses only the infrared window channel. It is not clear, however, how significant these smaller features are in the context of thunderstorms.

B.3 Conclusion

It is possible to use statistics computed in the neighborhood of the pixels of weather images, both radar and satellite, as texture measurements for the purposes of segmentation. A hierarchical texture segmentation technique, where regions identified in successive stages are agglomerated together provides a tree representation of the storm structures in the images. The method described here is an improvement over existing segmentation algorithms for weather data in the following ways. It would potentially allow reliable tracking of features from frame to frame. On satellite images, it allows for the identification of features in the storm scale, as small as 40 km^2. On radar images, features as small as 10 km^2 can be identified. It is possible to identify smaller features in satellite images, features of about 10 km^2, if one uses several channels of satellite imagery, but more study is required to ascertain the significance of these smaller features.