MULTISCALE STRUCTURE DETECTION AND ITS APPLICATION TO IMAGE SEGMENTATION AND MOTION ANALYSIS

BY

MARK D. TABB

B.S., Cornell University, 1991
M.S., University of Illinois at Urbana-Champaign, 1993

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1996

Urbana, Illinois

UMI
300 North Zeeb Road
Ann Arbor, MI 48103

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

THE GRADUATE COLLEGE

FEBRUARY 1996

WE HEREBY RECOMMEND THAT THE THESIS BY

MARK D. TABB

ENTITLED MULTISCALE STRUCTURE DETECTION AND ITS APPLICATION TO IMAGE SEGMENTATION AND MOTION ANALYSIS

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Narendra Ahuja
Director of Thesis Research

Edward Mast
for Head of Department

Committee on Final Examination†

Narendra Ahuja
Chairperson

W. O'Brien

Zhi-pei Liang

M. Orchard

† Required for doctor's degree but not for master's.

ABSTRACT

The problem of structure detection in images involves the identification of local groups of pixels that are both homogeneous and dissimilar to all nearby areas. Homogeneity can be measured with respect to any criteria of interest, such as color, texture, motion, or depth. No prior knowledge is assumed regarding the number of structures, their size or shape, or the degree of homogeneity that they must possess. Only the homogeneity criteria of interest have to be known. Structures may be either connected (pixels form contiguous areas) or disconnected, but the former case is treated in detail by this thesis. Structure identification is inherently a multiscale problem. For example, a texture contains subtexture, which itself contains subtexture, etc. In the absence of prior information, an algorithm must identify all such structures present, regardless of the scale. A formulation of scale is given that is able to describe image structures at different scales. A nonlinear transform is presented that has the property that it makes structure information at a given scale explicit in the transformed domain. This property allows the processes of automatic scale selection and structure identification to be integrated and performed simultaneously. Structures that are stable (locally invariant) to changes in scale are identified as being perceptually relevant. The transform can be viewed as collecting spatially distributed evidence for edges and regions and making it available at contour locations, thereby facilitating integrated detection of edges and regions without restrictive models of geometry or homogeneity variation. An application of this structure identification to the problem of estimating 2–D motion fields from video sequences is given. This approach has advantages in being able to compute accurate motion near occlusion boundaries and in areas with little variation in intensity.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Narendra Ahuja, for his support and guidance throughout the course of this research. His enthusiasm for the subject made this work truly enjoyable. I also would like to thank the committee members, Professors Zhi-Pei Liang, William O'Brien, and Michael Orchard, for their suggestions to improve the thesis as well as my presentation of it. In addition, I would like to thank the students in the Computer Vision Laboratory at the Beckman Institute, particularly Arun Krishnan, Tanuja Joshi, and Tod Courtney, who spent a great deal of their time keeping the machines up and running.

TABLE OF CONTENTS

Page

1. INTRODUCTION

The problem of structure detection involves the identification of local areas within some set of data that are both homogeneous and dissimilar to all nearby areas. Homogeneity can be measured with respect to any criteria of interest. In the fields of image processing and computer vision, some such criteria include color, texture, motion, and depth. This problem is approached from the perspective of a total lack of prior information. No knowledge is assumed regarding the number of structures, their size or shape, or the degree of homogeneity that they must possess. Only the homogeneity criteria of interest have to be known. Further, the structure detection problem is treated from a low-level perspective in this thesis. No high-level reasoning is employed, and no human supervision of the process is allowed.

Because of the lack of *a priori* information, issues related to scale become significant. It is well-known that the real world contains structure at multiple scales. Consider, for example, structures defined by similarity of texture. A forest is a texture composed of trees. However, each tree is also a texture defined by leaves, and each leaf a texture defined by veins. None of these textures is more relevant than any other; they just exist at different scales. Thus, structure forms a natural pyramidal hierarchy in which a structure contains substructures, which themselves contain substructures, etc. If a particular image contains structures at multiple scales (such as if a forest, individual trees, and individual leaves are all visible), then the best a structure detection algorithm can hope to do is to identify all of the structures in the image, regardless of the scale at which they are present.

A general framework is introduced by this thesis within which this kind of problem can be solved. It is then shown in detail how to apply this framework to solve the specific structure detection problem known as *image segmentation*. This problem involves the identification of homogeneous areas that are connected (defined by contiguous pixels), and for which homogeneity is defined in terms of variation on the actual values of the data, and not some derived attribute such as texture. A method is then described that uses the structures identified by the image segmentation process to derive 2–D motion fields from a video sequence.

1.1. Motivation and Approach

Identifying structures at different scales requires a formal description of scale and its relationship to structure. This thesis argues that there are three distinct types of scale that are necessary to capture image structure. The first, integration scale, describes the amount of spatial support necessary at a given pixel in order to extract the homogeneity criteria of interest (e.g., texture). The second, homogeneity scale, expresses the amount of homogeneity present within a structure, and the last, spatial scale, determines the degree of proximity among the elements of a structure. Together, these scales fully characterize a structure.

For any given structure to be identifiable by a structure detection algorithm, it is necessary that the algorithm be tuned to the scale at which the structure exists. In other words, structure identification requires that the scale of the structure be known. Unfortunately, the scale of a structure cannot be computed unless the structure is itself known. Thus, the processes of scale selection and structure detection must be integrated and performed simultaneously. This is accomplished by a transform that utilizes this formulation of scale. The transform can be thought of as a form of force-based clustering that groups together similar pixels. It does this in a manner that makes explicit the structures present within an image at a given scale. One might imagine that there are many scales at which a particular grouping of pixels is reasonable but not perceptually relevant. It is argued that the importance of a structure is directly related to its extent in scale. Because the transform makes the structure at a particular scale explicitly available, it is easy to examine the change in structure resulting from a change in scale. Those structures that are stable (invariant) to local changes in scale then can be identified as meaningful. In this way, scale selection and structure identification are integrated together.

A framework is presented for using the transform in this manner in order to solve the structure detection problem. However, a full solution to the general case is not given. Instead, the thesis focuses on applying the framework to the problem of image segmentation. The structure connectedness requirement of this problem results in structures forming regions that are separated by closed edge contours. For this case, the transform can be viewed equally as either a region detector or an edge detector. This has some interesting ramifications because previous image segmentation algorithms are either solely region-based or edge-based. In addition, because the process of edge detection is typically formulated in terms of the local maximum of the gradient, some comparisons are made between the transform and the notion of a multiscale gradient operator.

Finally, the problem of estimating the 2–D motion field from a video sequence is addressed. An algorithm is presented that utilizes the region structures produced by the image segmentation algorithm to produce 2–D motion fields, identify areas that exit (become occluded) and enter (become disoccluded) the field of view, identify areas having similar motion, and determine the relative distance from the camera of these motion regions. A region-based approach has advantages over other motion estimation methods in terms of its ability to produce accurate motion estimates in the presence of noise and changes in illumination, and within areas having little intensity variation. In addition, the close relationship between occluding contours and region structure boundaries defined by intensity simplifies the problem of producing accurate motion estimates near the occluding contours. The algorithm considers a video sequence two frames at a time. Each image is segmented, and the identified regions in the two frames are matched at multiple scales. The motion of the pixels within a pair of matched regions is modelled by an affine transformation, which is computed independently for each matched pair at all scales. Because a pixel may belong to several different regions, it may have multiple motion vectors associated with it. An integration module then attempts to select the most correct motion vector at each pixel in order to yield a single motion field. Motion segmentation and occlusion and layer identification algorithms are then applied to this motion field.

1.2. Previous Work

No general framework for approaching the problem of structure detection has been given previously. In terms of image segmentation, many algorithms exist for computing an image segmentation at a single scale. These include thresholding techniques [1, 2], region growing [2–4], split-and-merge [5], watersheds [6], rule-based systems [7], and MRF-based models [8]. Such methods are generally able to identify structure that exists at scales to which they are tuned, however, they cannot identify structures at other scales. In addition, in areas where no structure exists at the tuned scale, unintuitive regions will be identified. Prior attempts at multiscale image segmentation [9–11], represent an image at different scales and apply one of the single scale segmentation algorithms to the representation of the image at each scale. This type of scale is shown to be an integration scale, and, as such, is only weakly related to structure because the homogeneity and spatial aspects of scale are not modelled.

Previous approaches to the problem of 2–D motion estimation can be classified as either pixel-based (intensity-based) or feature-based. The pixel-based approaches include algorithms that utilize constraints based upon local spatial and temporal derivatives [12, 13], as well as the popular block-based correlation algorithm (BCA). These algorithms generally perform well in textured areas of the scene; however, they suffer from the problematic assumptions that intensity changes are caused solely by motion and that motion causes variations in intensity. As a result, noise and lighting changes induce motion, and motion estimates are difficult to obtain in areas having little variation of intensity. In addition, motion discontinuities at occlusion boundaries also may cause problems. The feature-based methods extract features from images and then match them across frames, thereby obtaining a displacement field. Such features include points defined by local intensity extrema [14], edges [15–17], corners [17, 18], and regions [19–24]. These algorithms generally provide accurate but sparse motion fields. Because each pixel in an image belongs to at least one region, the use of region features can provide dense motion estimates. The previous region-based algorithms follow the same general approach as the algorithm described in this thesis, but are all monoscale, use simple methods to match regions, and are error prone near occlusion boundaries. The region matching algorithm used in this thesis is similar to the region adjacency graph matching method of [25], but has some advantages with regard to reduced computational complexity, coarse-to-fine matching, and ability to properly match regions lying entirely inside another region.

1.3. Thesis Organization

The relationship between structure and scale is discussed in detail in Chapter 2. This includes a discussion of the three distinct kinds of scale, as well as the number of scale parameters required to fully characterize structures of arbitrary size, shape, and degree of homogeneity. The transform is introduced in Chapter 3. Its ability to make both connected and disconnected structures explicit is explained, and some pointers are given toward using it to solve the general structure detection problem. Chapter 4 considers the problem of image segmentation. A method for automatically selecting scale parameters and detecting region structures is given. A comparison is also made between the transform and the concept of a gradient. Chapter 5 motivates and describes the use of region structures in estimating and segmenting 2–D motion fields. Finally, some conclusions are made in Chapter 6.

2. STRUCTURE AND SCALE

This thesis is concerned with the problem of automatically identifying low-level structures present in data. Image and video data are used exclusively in this thesis, but the approach is truly general and can be applied to any type of dataset. A structure consists of a local group of datapoints (pixels) within a dataset that are relatively homogeneous with respect to some homogeneity criteria and that are dissimilar to other nearby points outside of the structure. No prior information concerning the size, shape, number, or degree of homogeneity of the structures is assumed. Homogeneity can be measured in terms of the variation of the values taken by the pixels within a structure, or by using some attribute of the data. In the latter case, a feature vector that characterizes the homogeneity criteria of interest has to be computed at each pixel . Structure homogeneity is then measured in terms of the amount of variation in the values of these feature vectors. Consider, for example, a color image. Because color information is available directly, one could identify structures having similar color by using the distance between colors in some color space (e.g., CIE L*a*b* [26]) as a similarity measure. However, one may be interested in identifying textural structures. Textures are comprised of texture elements (texels) that have similar properties. Texels cannot be sensed directly, and the properties by which they are similar are unknown *a priori*. In this case, a similarity measure is not directly available and has to be estimated. An example of low-level structure in a video sequence is moving objects, which are defined by areas of similar motion, a criterion that, once again, is not directly available and must be estimated.

Humans are quite adept at discriminating colors, textures, and moving objects given these constraints, but one might wonder whether the problem of detecting such structures is truly low-level, or whether higher-level processes such as reasoning have to be involved. Psychophysical experiments on humans, such as those performed by Julesz [27], indicate that a wide variety of textures are discriminated preattentively, that is, using only low-level processing in the visual cortex, and not any cognitive reasoning. Thus, the problem does seem to be well-defined at the low level.

2.1. The Relationship Between Structure, Scale, and Resolution

Before one can address the issue of developing an algorithm capable of automatically performing structure identification, it is first necessary to understand the complex relationship between structure and scale. Structure is inherently recursive, i.e., a structure contains substructure, which itself contains substructure, etc. Which of these structures is "relevant" in some sense is determined by the scale at which one is looking. Scale is not the same thing as resolution, however, even though the terms sometimes are used interchangeably. Resolution describes the distance from which data are viewed, thereby determining which structures are visible in the data. For example, at a particular resolution, one may be able to see a forest of trees. Viewed closer, individual trees become visible, and, closer still, individual leaves. At any given resolution, only a finite number of structures are visible because the number of levels of embedded structure is limited (typically 2–4). For example, if the forest is viewed at a resolution where it encompasses the entirety of one's field of view, then the forest is the coarsest scale structure visible (call this scale the outer scale). Individual trees are also visible, and these can be considered structures at some finer scale. One may imagine that individual leaves are visible on some trees, representing structure at an even finer scale. It is not likely, however, that any substructure within a leaf would be discernible, and, hence, the scale of the leaf represents the inner scale. As a result, for the given resolution, there are three levels of structure visible in this example. It is clear that resolution is extrinsic to the data in that it affects which structures are visible (can be resolved), but has no relationship to the structures themselves. Scale, on the other hand, actually defines and characterizes the structures, and, hence, is an intrinsic property of the data.

2.2. Comparison with the Fractal Model of Structure

Self-similar fractals are sometimes touted as being good models of real-world structure. Fractal models were popularized by Mandelbrot [28] fairly recently, but have a long history dating back at least as far as Cantor [29] and Fatou [30]. Such models are recursive in that a structure at some scale is composed of finer scale structures that are similar to it. Many examples are given in [28] of visually realistic depictions of objects such as ferns, trees, and mountains using these models. Real-world structure, however, is not well-represented by fractal models. For example, a brick wall is a texture composed of individual brick texels. At a finer scale, each brick is a texture, formed not by smaller bricks, but instead by rock and clay grains. In general, the statistics of a texture are unrelated across scales. Self-similar fractals represent a special instance of multiscale structure in which the homogeneity characteristics are unchanged across scales. They produce realistic looking objects because they capture the multiscale aspect of structure. This makes them quite useful for applications in computer graphics involving the synthesis of natural scenes, but not especially useful for applications involving image analysis, such as texture segmentation or image compression.

2.3. On Scale

Recall that a structure consists of a local area that is homogeneous with respect to some similarity measure. The degree of similarity and proximity that must be present among a set of pixels in order for it to be considered a structure has to be specified. In addition, if feature vectors have to be computed, then the extent of the surrounding area used at each pixel in this computation also has to be specified. This information is provided by three distinct kinds of

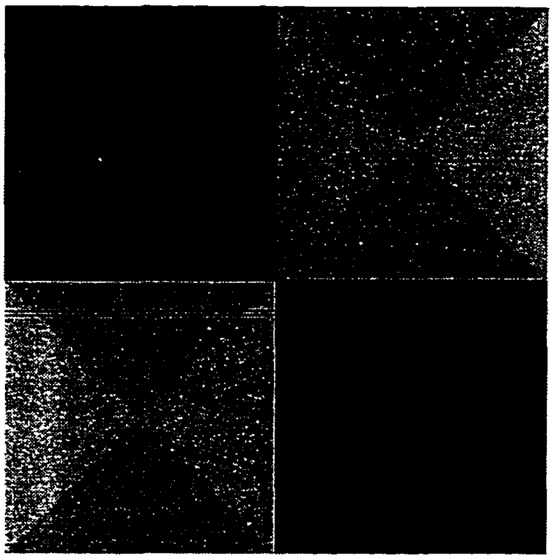


Figure 2.1. Synthetic graylevel image. At a scale allowing for little homogeneity variation, the image consists of triangular structures. However, at a coarser homogeneity scale, the image consists of square structures.

scale: *homogeneity* scale, *spatial* scale, and *integration* scale, respectively. Each structure within the data is characterized by some combination of these scales. Scale parameters are assumed to take values in the range $[0, \infty]$ on the real line. A larger value (coarser scale) indicates more variation in the structure characteristic described by the scale parameter, and a smaller value (finer scale) indicates correspondingly less variation.

Some examples may be useful to clarify these concepts. Consider Fig. 2.1, a synthetic image where each pixel takes on one of four different values. At a fine homogeneity scale (i.e., relatively little variation allowed within the structures), the pixels group into triangular structures. Similarly, at a coarser homogeneity scale, structures have more variation and the squares become relevant. The interaction between both homogeneity and spatial scale is demonstrated in Fig. 2.2, which contains discrete 2–D data of 1–D, integer-valued feature vectors. Each data point is shown labelled by its integer value. The group of feature vectors comprising each structure is circled

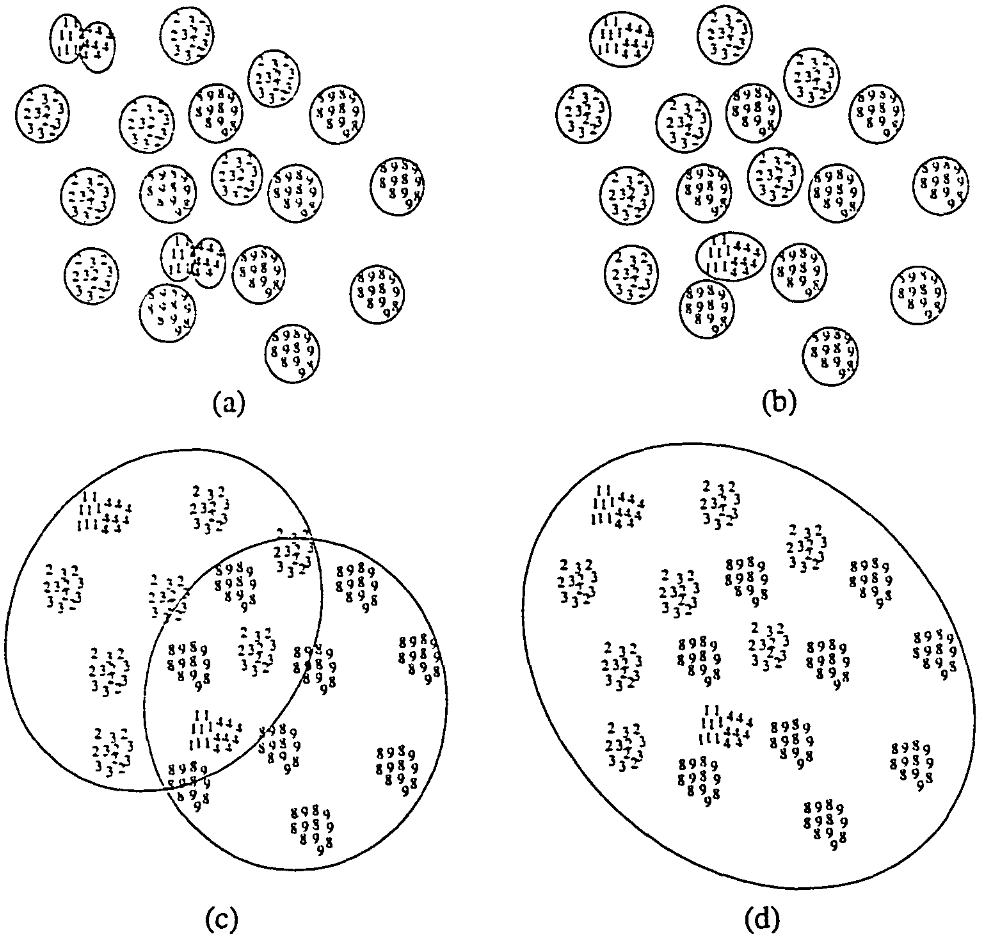


Figure 2.2. A 2–D set of discrete, integer-valued data. (a) Some structures with a strong degree of homogeneity and spatial locality. (b) Some structures with less homogeneity than in (a), but the same degree of spatial locality. (c) Some structures with less spatial locality than in (b), but the same degree of homogeneity. (d) A structure with less homogeneity and spatial locality than in (c).

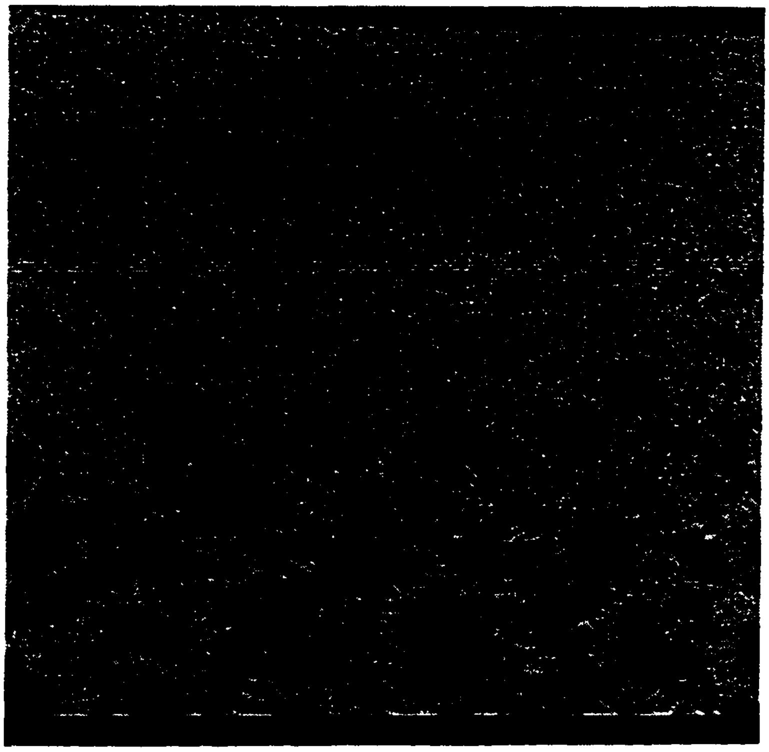


Figure 2.3. Example demonstrating integration scale. The sunflower field is imaged nonfrontally, resulting in the nearer sunflower texels appearing much larger in the image plane than the farther sunflowers. The spatial support of an operator that computes feature vectors capturing the texel characteristics should have spatial support similar to the texel size. Hence, the texture field is characterized by a coarser integration scale at the bottom of the image than at the top.

in Fig. 2.2(a)-(d). At some fine homogeneity and spatial scale, the structures shown in (a) are present, and if the homogeneity scale is made more coarse, the structures in (b) become present. If the spatial scale is then made more coarse, the larger structures in (c) become relevant, and if both scales are then made coarser still, the structure in (d) becomes relevant. None of the structures present in (a-d) is any more valid or important than any other; they merely exist at different scales. An example of integration scale is given in Fig. 2.3. This figure is an image of a sunflower field comprised of individual sunflower texels. The field has been imaged nonfrontally, so that the texels are at varying distances relative to the camera, causing the nearer

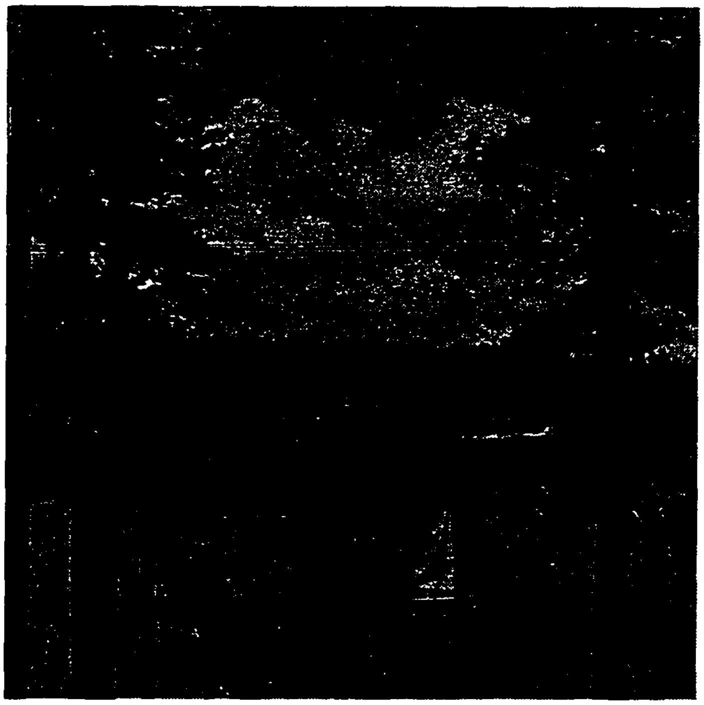


Figure 2.4. Image of a sailboat on a lake. The image contains considerable multiscale structure defined by areas of homogeneous intensity. For example, at a scale where regions have significant intensity variation, the cloud mass and water each can be considered as single regions. However, at a finer scale with increased sensitivity to intensity variation, individual clouds and the streaks within the water should be identified as regions.

texels to be much larger than the farther ones. The spatial support of an operator that computes feature vectors capturing the texel characteristics should have spatial support similar to the texel size. Hence, the texture field is characterized by a coarser integration scale at the bottom of the image than at the top.

Figures 2.4–2.5 give more examples of multiscale structure present in real images. Figure 2.4 is a grayscale image of a sailboat on a lake. Let homogeneity be measured in terms of graylevel differences for this image. Both the cloud mass and the water are relevant structures at some scale, and the individual clouds within the mass and the streaks within the water correspond

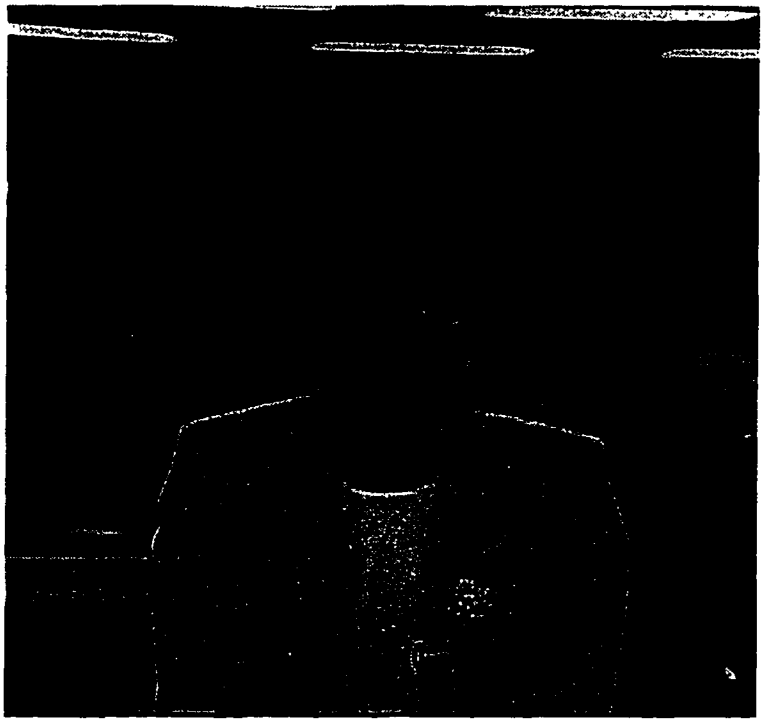


Figure 2.5. Image of a television news announcer. The woman's hair and sportcoat each form textures at some scale. The primary orientation of her individual hairs is different on either side of her part; thus, the part divides the hair into two different subtextures at a finer scale. Similarly, the orientation of the texture on the sportcoat is horizontal over the main body of the coat, but vertical on the arms. Thus, the coat subdivides into three different textures (main body plus each sleeve) at a finer scale.

to relevant structure at some finer scale. Figure 2.5 is also a grayscale image, but now let homogeneity be measured in terms of textural characteristics. The woman's hair and sportcoat are both textures at some scale. The primary orientation of her individual hairs is different on either side of her part; thus, the part divides the hair into two different subtextures. Similarly, the orientation of the texture on the sportcoat is horizontal over the main body of the coat, but vertical on the arms. Thus, the coat subdivides into three different textures (main body plus each sleeve) at a finer scale.

Although three kinds of scale are sufficient to describe low-level structure, many more than three scale parameters are required. An image cannot be described with a single homogeneity scale because it may contain very homogeneous structures in one part of the image and much less homogeneous structures in another part. Hence, at least one homogeneity scale parameter is required for each structure. Further, the lack of any constraints on structure size or shape requires the use of a potentially different spatial scale at each pixel within the structure. For instance, an octopus-shaped structure requires coarse spatial scales to represent the main body and much finer spatial scales to represent long, narrow tentacles. In addition, a potentially different integration scale parameter also may be required at each pixel within a structure. Consider the sunflower field texture in Fig. 2.3. An integration scale parameter large enough to capture the characteristics of one of the closer texels also would capture characteristics of an amalgam of the farther texels, thereby giving unintuitive results. Similarly, a finer integration scale tuned to the farther texels would compute the characteristics of the subtexture present on the closer texels. Thus, in general, to describe any structure, it is necessary to utilize three scale parameters (one of each kind of scale) for each pixel.

For a given dataset, each perceptually relevant structure has a scale at which the grouping makes sense. However, many other pixel groupings will exist in the data which also are reasonable at some scale, but which are not perceived as being valid. It is assumed that perceptually relevant structures are distinguished from these nonrelevant groupings by being stable with scale (i.e., they make sense over some continuous range of scale). The motivation for this assumption is the following: The degree to which a structure is relevant is directly proportional to the ratio between the homogeneity difference between a structure and adjacent structures (interstructure homogeneity) and the intrastructure homogeneity variation. This same relationship holds for

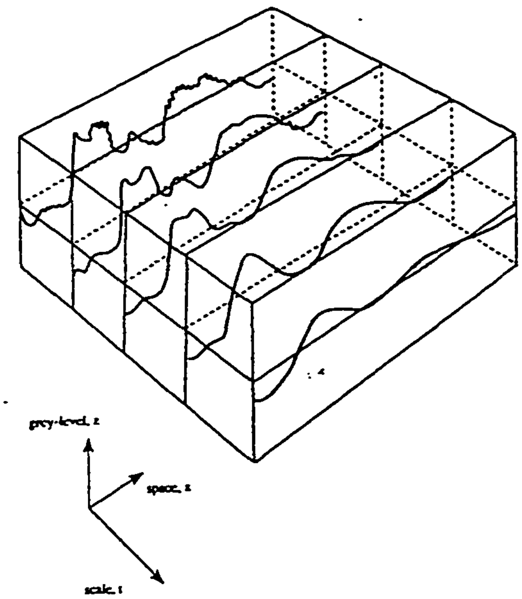


Figure 2.6. A 1–D continuous signal and its representation at different integration scales computed by convolving the signal with a Gaussian kernel (isotropic diffusion).

spatial and integration scales as well. These ratios are reflected in the extent of a structure in scale, where higher ratios correspond to greater extent in scale.

2.4. Comparison with Other Formulations of Scale

In the literature, the term *scale* generally refers to a one-parameter index into a hierarchical decomposition of a signal. Some examples of such decompositions include isotropic diffusion [9, 31], anisotropic diffusion [32], wavelets [33], and morphology [34]. The original signal is represented with decreasing amounts of detail as the scale becomes coarser. The scale parameter may be continuous [31], or take a discrete set of values [33]. The continuous case is generally referred to as a *scale-space*. An example of a scale-space decomposition obtained using isotropic diffusion is given in Fig. 2.6.

The type of scale used by all of these methods is actually an integration scale. The scale parameter defines the size of a spatial neighborhood over which a filter is applied. These filters are inherently low-pass, thereby emphasizing the signal mean over the given neighborhood. The decompositions obtained with this class of methods are not structural. There is no simple relationship between the information removed by these filters and the structure present within the signal. In this thesis, such methods are referred to as representing a *signal* at different scales, and should not be confused with the general formulation of scale given in this chapter that is directly related to structure.

3. A TRANSFORM FOR STRUCTURE DETECTION

This chapter addresses the issue of how to utilize the ideas presented in the previous chapter on the relationship between scale and structure, to perform structure detection. Because structure is inherently characterized by scale, structure detection cannot be performed unless the scale is known. Similarly, the scale of a structure cannot be computed unless the structure already has been identified. Because neither the structures nor their scale is known *a priori*, scale selection and structure detection must be integrated. This problem is solved through the use of a new transform [35], which makes the structure present at a given scale explicit in the transformed domain. This allows easy examination of the change in structure resulting from a change in scale. A search across all scales for the set of structures that are invariant to local changes in scale results in the simultaneous identification of the perceptually relevant structures and the scale at which they exist.

3.1. The Transform

For a given scale, the transform maps a set of data into an attraction force field, within which the structure at that scale is explicitly encoded. Let $\vec{I}(\vec{x})$ represent a continuous k-dimensional dataset of l-dimensional vectors, i.e.,

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_k \end{pmatrix} , \quad \vec{I} = \begin{pmatrix} I_1 \\ I_2 \\ \cdot \\ \cdot \\ I_l \end{pmatrix} \tag{3.1}$$

For a given triplet of scale parameters $(\sigma_g, \sigma_s, \sigma_i)$ at each point, denoting, respectively, the homogeneity, spatial, and integration scale at that point, a force field, **F**, is computed as follows:

$$\begin{aligned}
\mathbf{F}\left(\vec{x}\ ;\ \sigma_g(\vec{x}), \sigma_s(\vec{x}), \sigma_i(\vec{x})\right) &= \int_R \!\!....\!\!\int F_{yx} \frac{\vec{r}_{yx}}{\|\vec{r}_{yx}\|} dy_1 ... dy_k \\
F_{yx} &= d_g(\Delta I, \sigma_g(\vec{x})) \cdot d_s(\vec{r}_{yx}, \sigma_s(\vec{x})) \\
\Delta I &= \left\|\vec{I}(\acute{\vec{x}}) - \vec{I}(\acute{\vec{y}})\right\| \\
\vec{I}(\acute{\vec{x}}) &= k\left(\vec{I}(\vec{x}), \sigma_i(\vec{x})\right) \\
R &= domain\left(\vec{I}\right) \setminus \{\vec{x}\} \\
\vec{r}_{yx} &= \vec{y} - \vec{x}
\end{aligned} \tag{3.2}$$

In the case that $\vec{I}(\vec{x})$ is discrete, Eq. (3.2) becomes

$$\mathbf{F}\left(\vec{x}\ ;\ \sigma_g(\vec{x}), \sigma_s(\vec{x}), \sigma_i(\vec{x})\right) = \sum_R F_{yx} \frac{\vec{r}_{yx}}{\|\vec{r}_{yx}\|} \tag{3.3}$$

The transform computes at each pixel a vector sum of pairwise affinities between the pixel and all other pixels. The resultant vector produced by the transform at each pixel defines both the direction and magnitude of attraction experienced by the pixel from the rest of the image. The affinity of a pixel $\vec{x}_0$ for another pixel $\vec{y}_0$ is given by the term $F_{y_0x_0}(\vec{r}_{y_0x_0}/\|\vec{r}_{y_0x_0}\|)$, where $\vec{r}_{y_0x_0}$ is the vector from $\vec{x}_0$ to $\vec{y}_0$, and $F_{y_0x_0}$ is the magnitude of the attraction for pixel $\vec{y}_0$ experienced by pixel $\vec{x}_0$. This magnitude is given by the product of a homogeneity distance function, $d_g(\cdot)$, which measures the degree to which the two pixels are similar, and a spatial distance function, $d_s(\cdot)$, which measures the proximity of the pixels. The homogeneity between two pixels, ΔI, is given by the Euclidean distance between their associated, m-dimensional feature vectors,

$$\vec{I'} = \begin{pmatrix} I'_1 \\ I'_2 \\ . \\ . \\ I'_m \end{pmatrix} \tag{3.4}$$

where the feature vector at each pixel is determined by an operator, $k(\cdot)$, which computes the desired homogeneity characteristics over an area surrounding the pixel with spatial support given by σ_i.

3.2. Properties of the Homogeneity and Spatial Distance Functions

With the definition of the force field **F** given in Section 3.1, pixels are grouped together into structures consisting of sets of pixels that are mutually attracted to one another. The force direction at each pixel points towards the interior of the structure to which it groups. Pixels may group together into any possible spatial configuration, allowing structures of any conceivable size or shape to be detected. As the scale parameters are varied, the force vectors align to form different structures. If the values of the homogeneity scales are increased, less homogeneous structures should be encoded in the field. Similarly, an increase in the spatial scales should cause larger structures to be encoded. To ensure such relationships between scale and structure, the distance functions, $d_s(\cdot)$ and $d_g(\cdot)$, should possess the following properties:

1. *Unit Range.* The transform measures the degree of attraction among pixels, not repulsions, so the functions should be nonnegative. For convenience, and without loss of any generality, the maximum value of these functions is set to unity. Hence, $0 \leq d_g(\cdot), d_s(\cdot) \leq 1$.

2. *Decreasing Attraction (image characteristic).* The degree of attraction between pixels should be directly proportional to their similarity, $d_g(\Delta I_1, \sigma_g) \geq d_g(\Delta I_2, \sigma_g)$ for $\Delta I_1 \leq \Delta I_2, \forall \sigma_g$, and proximity, $d_s(\vec{r}_1, \sigma_s) \geq d_s(\vec{r}_2, \sigma_s)$ for $\|\vec{r}_1\| \leq \|\vec{r}_2\|, \forall \sigma_s$.
3. *Increasing Attraction (scale characteristic).* Pixel similarity should be directly proportional to both homogeneity scale, $d_g(\Delta I, \sigma_g^1) \leq d_g(\Delta I, \sigma_g^2)$ for $\sigma_g^1 \leq \sigma_g^2$, and spatial scale, $d_s(\vec{r}, \sigma_s^1) \leq d_s(\vec{r}, \sigma_s^2)$ for $\sigma_s^1 \leq \sigma_s^2$.
4. *Isotropicity.* Structure should not be detected by the transform preferentially in any direction. Thus, $d_s(\vec{r}, \sigma_s) = f(\|\vec{r}\|, \sigma_s)$ is required.
5. *Locality.* The field at each pixel should depend only on a local (albeit adaptively determined) neighborhood around that pixel. Hence, let $d_s(\vec{r}, \sigma_s) = 0$ for $\|\vec{r}\| > c \cdot \sigma_s$, for some constant, c.

Two possible forms for the functions $d_g(\cdot)$ and $d_s(\cdot)$ satisfying these criteria are unnormalized Gaussian

$$d_g(\Delta I, \sigma_g) \sim \sqrt{2\pi\sigma_g^2} N_{\Delta I}(0, \sigma_g^2) \tag{3.5}$$

$$d_s(\vec{r}, \sigma_s) \sim \begin{cases} \sqrt{2\pi\sigma_s^2} N_{\|\vec{r}\|}(0, \sigma_s^2), & \|\vec{r}\| \leq 2\sigma_s \\ 0 & , \ \|\vec{r}\| > 2\sigma_s \end{cases} \tag{3.6}$$

and box-car window

$$d_g(\Delta I, \sigma_g) \sim B_{\Delta I}(\sigma_g) \tag{3.7}$$

$$d_s(\vec{r}, \sigma_s) \sim B_{\|\vec{r}\|}(\sigma_s) \tag{3.8}$$

where

$$B_x(c) = \begin{cases} 1, & |x| \leq c \\ 0, & else \end{cases} \tag{3.9}$$

Although **F** is not invariant to the exact form of $d_g(\cdot)$ and $d_s(\cdot)$, one would expect different forms to result in minimal change in the encoded structure information, so long as the forms satisfy the listed criteria, because relevant structure is locally invariant to changes in scale. This point is discussed in more detail in the context of the problem of image segmentation in the following chapter.

3.3. Structure in the Field Domain

If a structure is connected, then at a scale at which the structure exists, the structure is represented within the field as a region of contracting flow (inward force vectors) defined by contiguous pixels. Such a region is termed a *region of attraction*. Within this region, the flow sinks form a set of contours that characterize the skeleton of the structure. Similarly, if spatially adjacent structures are also connected, the region boundary is represented by the source of the flow. Consider a region whose boundary is given by a closed curve V, where ∇V is the outward normal of V. Denote by $\mathbf{F}^-$ the field immediately on the interior of V and by $\mathbf{F}^+$ the field on the immediate exterior. From the property of contracting flow, V satisfies the two relations

$$\nabla V \cdot \mathbf{F}^- \leq 0 \;, \; \nabla V \cdot \mathbf{F}^+ \geq 0 \tag{3.10}$$

because every point on a boundary curve separates at least two areas of contracting flow. Figure 3.1(a) shows $\mathbf{F}^-$ and $\mathbf{F}^+$ vectors for a 2–D scalar image containing a black region on a white background, with the scale parameters selected to yield regions of attraction. Other values of (σ_g, σ_s) scales may result in contracting flow over a set of disconnected regions (Fig. 3.1(b)). The term *zone of attraction* is used to denote the image space characterized by contracting flow, regardless of whether or not it consists of one or more regions. Representing both connected and disconnected structures within the field as areas of contracting flow is essential, because real-world structure may not always be connected. An example of both connected and disconnected

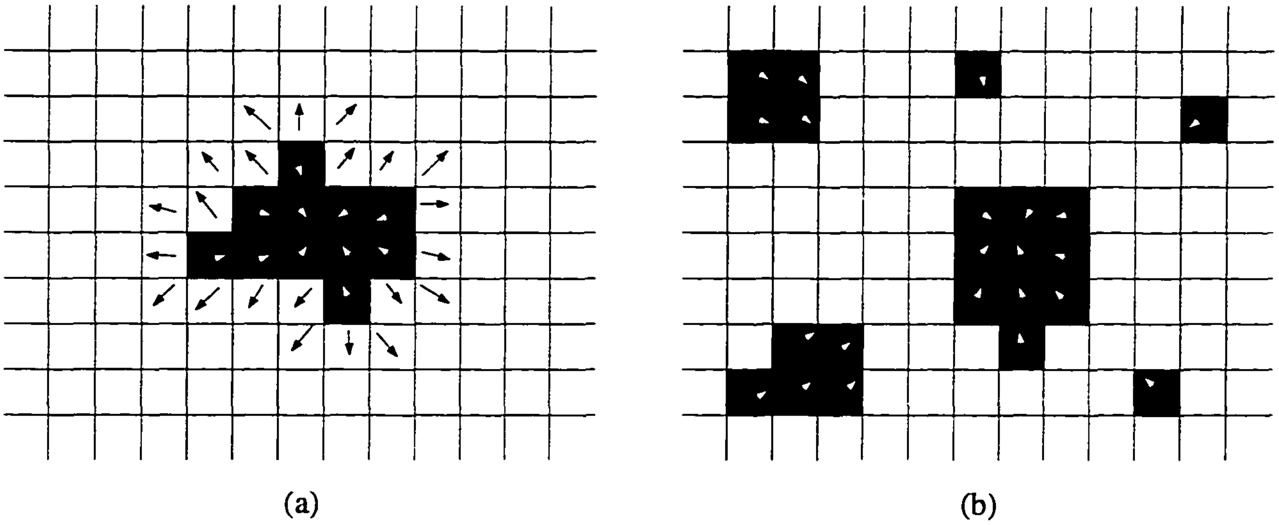


Figure 3.1. (a) Spatial and homogeneity scales selected so that the black region is transformed into a zone of attraction consisting of a single region of contracting flow. (b) Coarser spatial scales are used, resulting in a zone of attraction containing several regions of contracting flow.

structures in a real image is given in Fig. 3.2. Individual birds comprise connected structures of homogeneous graylevel, and the water forms a connected texture. The flock of birds, however, is a disconnected texture formed by individual bird texels. In addition, if one had a video sequence of this scene, the flock also would form a disconnected region defined by similarity of motion.

Structure identification in the field domain involves identifying zones of attraction that are stable with respect to local changes in scale. A general method for identifying zones of attraction is not presented in this thesis. Instead, we consider only the case where all structures are contiguous, i.e., all zones of attraction are also regions of attraction. A complete solution for this case is given in the following chapter.

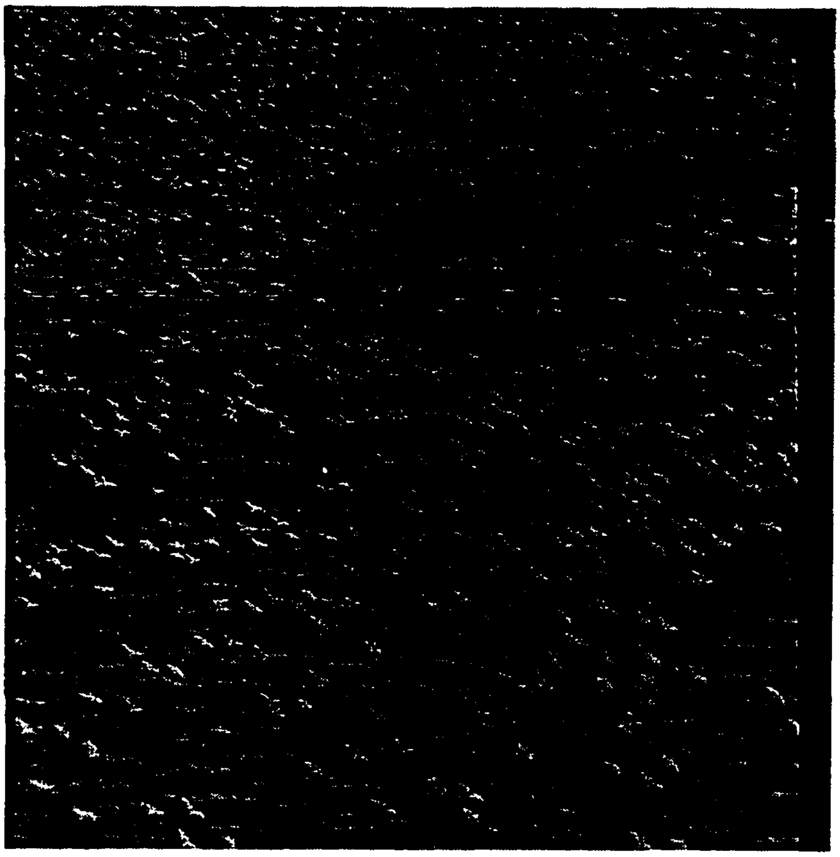


Figure 3.2. Image containing both connected and disconnected structure. Individual birds form connected regions of homogeneous intensity. Similarly, the water is a connected region of similar texture. The flock of birds, however, is a disconnected texture formed by individual bird texels.

3.4. Computing Feature Vectors

This section addresses some issues with regard to the operator $k(\cdot)$, which is used to compute feature vectors capturing some homogeneity criteria of interest. Homogeneity is measured in terms of distance between the actual values of the data for the case where $\sigma_i(\vec{x}) = 0$, because no information from any surrounding pixels is available at this integration scale. Thus, all possible operators should satisfy the relationship

$$k\left(\vec{I}(\vec{x}), 0\right) = \vec{I}(\vec{x}) \tag{3.11}$$

The exact form of $k(\cdot)$ is, of course, problem dependent. Because the application of structure detection considered in this thesis is the problem of image segmentation, for which $\sigma_i(\vec{x}) = 0$, no specific forms of $k(\cdot)$ are given in this thesis.

The problem of devising a specific form of $k(\cdot)$ for measuring local texture statistics has received considerable attention in the literature. Proposed methods utilize local geometric primitives [36–38], filter banks [39–43], local statistical features [44–46], and random field models [47–49]. Of these techniques, only that of Lindeberg [44] directly utilizes an integration scale, and this technique uses a fairly simple operator. Hence, more work still has to be done in this area in order to obtain an operator applicable to a wide variety of textures. In any case, all of these operators can be integrated directly into the transform.

4. MULTISCALE IMAGE SEGMENTATION

This chapter addresses a special case of the general structure detection problem known as image segmentation. This problem assumes that the structures are connected and that homogeneity is measured by differences among the data values (i.e., $\sigma_i(\vec{x}) = 0$). For greyscale images, this involves the identification of regions characterized by homogeneous intensity and high contrast with all spatially adjacent regions. The goal of image segmentation is to identify all such image structures without any smoothing of the structure boundaries, regardless of the scale at which the structures occur, and without incorrectly identifying any irrelevant areas as structures. Perceptual validity is used as a subjective measure of structure relevance. This chapter describes a method for identifying such structures using the transform, and it is shown how the processes of scale selection and scale-invariant structure identification can be integrated and performed simultaneously.

In the case of image segmentation, the transform can be viewed as collecting spatially distributed evidence for edges and regions and making it explicitly available at contour locations, and, in this sense, it performs Gestalt analysis. Further, it does this without using restrictive models of structure geometry or intensity variation, and without the need for any user-specified parameters. This allows the identification of precise structure boundaries that are completely unsmoothed, even at coarse scales. The transform incorporates into its definition the duality of region- and edge- based descriptions of image structure, namely, that the regions have a smooth variation of some property (e.g., intensity) in the interior and a sharp discontinuity across the boundary. Thus, the transform performs integrated edge and region detection, and can be viewed as a multiscale edge and blob detector at the same time.

The rest of this chapter is organized as follows: Section 4.1 reviews previous work on image segmentation. Section 4.2 describes the specific form of the transform used for image segmentation, and Section 4.3 gives an interpretation of the transform as a multiscale gradient. The extraction of image structure from the transformed image is discussed in Section 4.4, and experimental results for synthetic and real images are given in Section 4.5. Lastly, a final discussion of the method occurs in Section 4.6.

4.1. Previous Work

A large number of algorithms for image segmentation have been proposed over the years. However, almost all of these methods are inherently tuned to a particular scale. Any relevant structures that exist at that scale can be identified by such methods; however, structures at other scales cannot be identified. Worse, in any part of an image where no structure exists at that scale, or where structure exists but is not locally invariant to scale, such approaches identify unintuitive regions. Some examples of these approaches include: thresholding techniques [1, 2], region growing [2–4], split-and-merge [5], watersheds [6], rule-based systems [7], and MRF-based models [8]. Prior attempts at multiscale segmentation [9–11] all represent the image at different scales with one of the methods in Section 2.4, and then apply a single scale segmentation algorithm to the representation of the image at each scale. This does yield a region hierarchy, but the regions contained within it are not perceptually relevant for all but the most trivial images. This is because scale-spaces are not structural decompositions. Structure is contained implicitly within the scale-space, and the representation of the image at any given scale will contain, in general, multiscale structure. Hence, a multiscale segmentation algorithm is still required to extract the image structures from the scale-space decomposition. The use of these scale-spaces for image segmentation is inherently misguided because the scale parameter is simply an integration scale. Modeling image structure at different scales requires homogeneity and spatial scales to characterize the intensity variation and geometry (size and shape) of the structures. Further, scale-spaces inherently involve some smoothing of the image at all but the finest scale, so even if a true multiscale segmentation algorithm is applied to the representation of the image at each scale in the scale-space, the boundaries of any extracted structure will be smoothed as well.

4.2. Specific Form of the Transform Used for Image Segmentation

For a discrete, 2–D grayscale image, $I(x, y)$, the general transform (Eq. (3.2)) simplifies to

$$\begin{gathered}
\mathbf{F}\,(x, y\ ;\ \sigma_g(x,y), \sigma_s(x,y)) = \sum_{(u,v)!=(x,y)} d_g(\Delta I, \sigma_g(x,y)) \cdot d_s(\|\vec{r}\|, \sigma_s(x,y)) \frac{\vec{r}}{\|\vec{r}\|} \\
\vec{r} = (u-x)\vec{i} + (v-y)\vec{j} \\
\Delta I = |I(x,y) - I(u,v)|
\end{gathered} \tag{4.1}$$

For computational efficiency, the form of the homogeneity and spatial distance functions was selected as box-car. Although $\mathbf{F}$ is not invariant to the form of $d_g(\cdot)$ and $d_s(\cdot)$, different forms result in minimal change in the encoded structure information, so long as the forms satisfy the previously listed criteria. This is because structure is represented as converging and diverging vectors that are locally invariant to changes in scale. Experiments have been performed using different functions (box-car, Gaussian, linear, exponential) for both $d_g(\cdot)$ and $d_s(\cdot)$ on various real images. There was no change in the detected structures using different forms for $d_s(\cdot)$, and only marginal changes for different forms of $d_g(\cdot)$. A box-car function for $d_g(\cdot)$ gives the best results, a point that is explained in Section 4.2.4 after some necessary concepts have been introduced in the interceding sections.

4.2.1. Specific formulation of scale

Two independent scale parameters per pixel is overgeneral for characterizing image structure. In this thesis, a structure is modelled by a single σ_g that characterizes the degree of homogeneity of the structure, and a σ_s at each pixel that characterizes its size and shape. This number of spatial scales is necessary in order to identify a structure without any spatial smoothing. For example, identifying an octopus-like structure (large, main body with fine appendages) requires large values of σ_s within the body and small values within the appendages. In contrast, if retaining fine boundary detail is not necessary, then a (σ_g, σ_s) pair is sufficient to represent the scale of a structure. In Section 4.2.2, it is shown that if the value of σ_g for which a pixel belongs to a particular structure is known, then the appropriate value for σ_s at the pixel can be determined automatically. This allows Eq. (4.1) to be simplified to the following

$$\mathbf{F}\left(x, y\ ;\ \sigma_g(x,y), \sigma_s(x,y)\right) \ = \mathbf{F}_{\sigma_g}\left(x, y\right) \tag{4.2}$$

The homogeneity scale, σ_g, is made spatially invariant and is the sole independent scale parameter to the transform. This results in an image being mapped into a one-parameter family of attraction force fields, $\mathbf{F}_{\sigma_g}$, which contains all of the multiscale structure present in an image, thereby simplifying the problem of structure identification to a 1–D search.

In addition to regarding the set of spatial scales associated with a structure as describing its size and shape, an alternative (but equivalent) interpretation is that the spatial scales describe the appropriate amount of spatial information necessary to identify a structure having a given degree of homogeneity. For example, if σ_s at a pixel is too small, then there may not be enough spatial information available for the transform to determine to which region the pixel should group. Similarly, if σ_s is too large, then the transform may group the pixel into a region containing very distant and unrelated pixels. Figure 4.1 gives an example that demonstrates the necessity of selecting σ_s properly in order to determine whether or not a region boundary (edge) is present at a pixel. This example is independent of the method actually used to perform the region (edge) identification (the transform in this case).

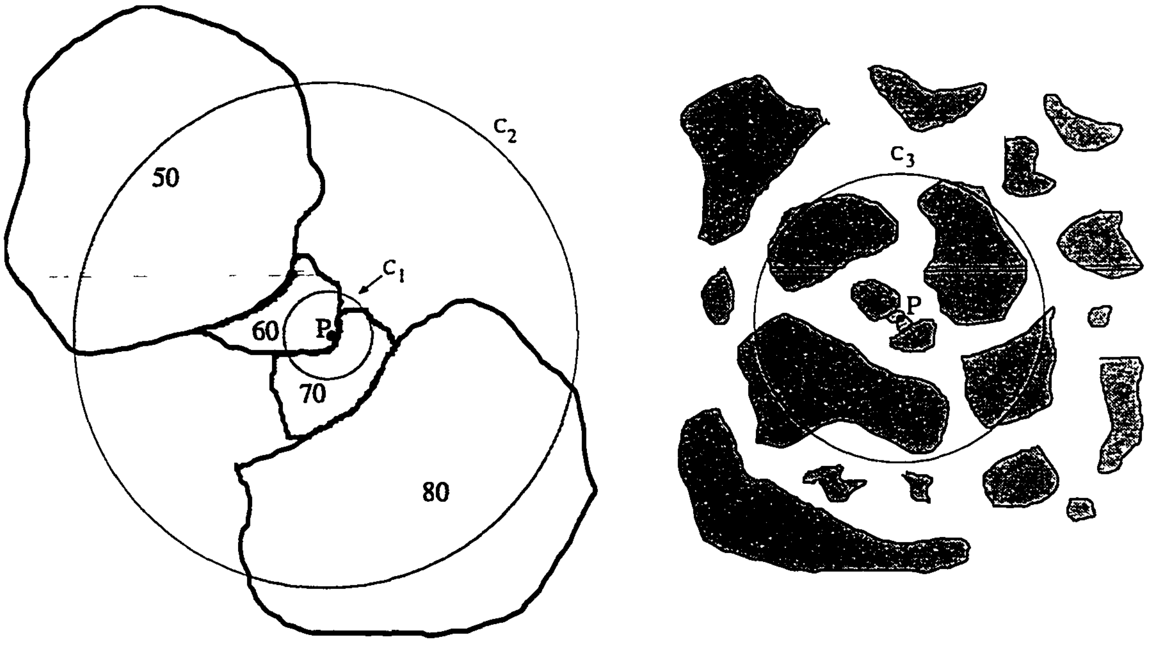


Figure 4.1. (a) For σ_g<10, the circle C_1 contains enough information to identify that an edge is present at Point P. For $10 \leq \sigma_g < 30$, the edge should still be present, but C_1 does not contain enough information to determine this. However, C_2 does. (b) This illustrates the problem of overscaling. There is too much information within circle C_3 to determine whether or not an edge is present at Point P. All local structure information has been lost.

There are four regions of constant intensity in Fig. 4.1(a), each labelled by its intensity. The most reasonable groupings with respect to σ_g for these regions are {50, 60, 70, 80} for $0 \leq \sigma_g < 10$, {50–60, 70–80} for $10 \leq \sigma_g < 30$, and {50–60–70–80} for $\sigma_g \geq 30$. Consider the point P that lies on the high-frequency edge separating regions 60 and 70, and the low-frequency edge between regions 50 and 80. Let the spatial information usable for determining whether or not an edge is present at P be contained within a circle of radius σ_s centered at P, and also let r_i denote the radius of circle C_i. For $\sigma_g < 10$, $\sigma_s = r_1$ provides enough information for determining that an edge should be present at P. For $10 \leq \sigma_g < 30$ an edge should still be present, but $\sigma_s = r_1$ does not provide enough information to support this conclusion, although

$\sigma_s = r_2$ does. Too much information is just as much of a problem as too little. For both these values of σ_g, choosing $\sigma_s = r_3$ as in Fig. 4.1(b) provides much more information than necessary. Effectively, the local structural information is lost. It should be apparent from Fig. 4.1 that, for each value of σ_g at each pixel, there is some appropriate range of values for σ_s that is necessary for structure identification to be possible.

4.2.2. Determining values for σ_s

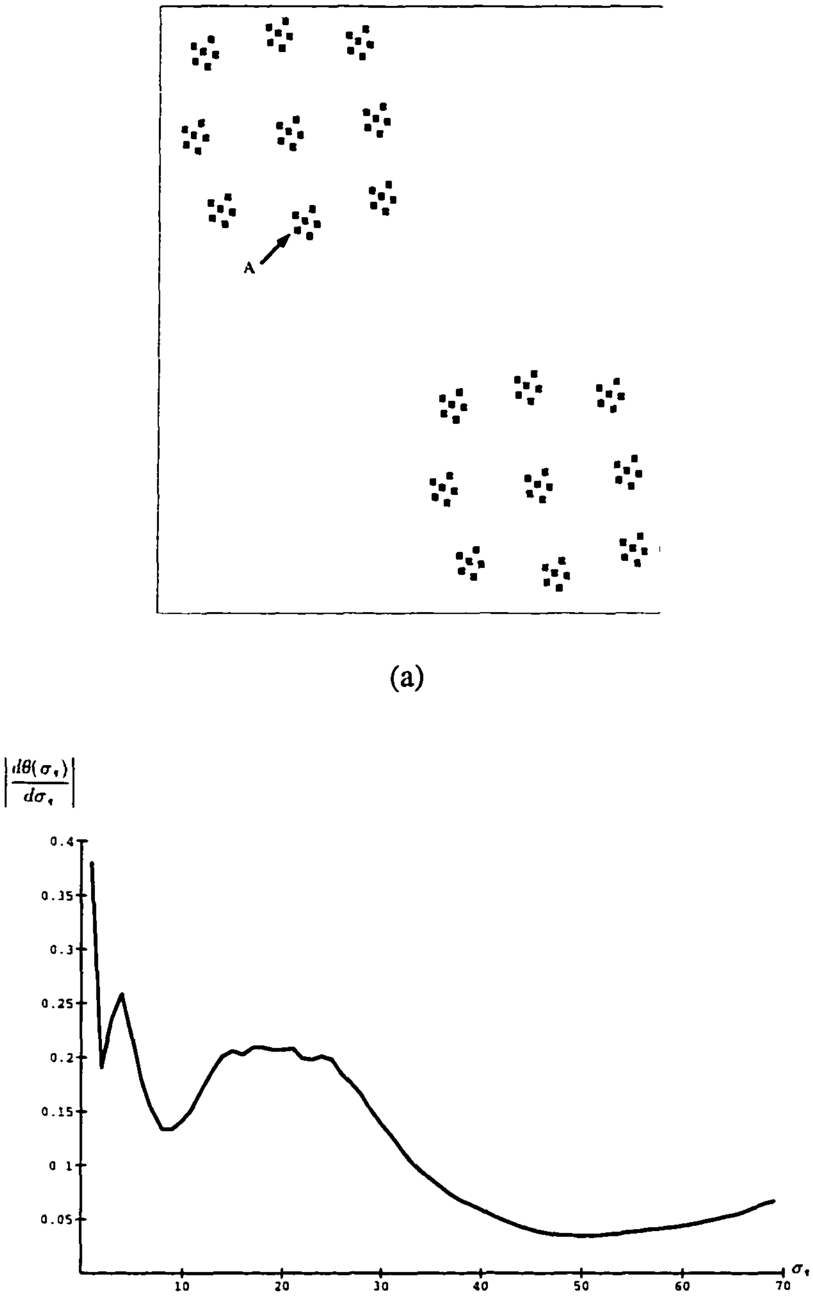
For a given σ_g, at each pixel (x_0, y_0) within some region *R*, $\sigma_s(x_0, y_0)$ needs to be chosen such that *R* corresponds to a region of contracting flow in $\mathbf{F}$. In general, a continuous range of values of $\sigma_s(x_0, y_0)$, denoted $[\sigma_s^-, \sigma_s^+]$, yields contracting flow. This range is determined by examining the behavior of $\mathbf{F}_{\sigma_g}(x_0, y_0)$ as $\sigma_s(x_0, y_0)$ is varied. $||\mathbf{F}_{\sigma_g}(x_0, y_0)||$ is small if the pixel intensities are uniformly distributed where $d_s(\cdot)$ is nonzero because the pairwise vector components in the vector sum computed at (x_0, y_0) by Eq. (4.1) tend to cancel out. As the intensity distribution becomes more asymmetric, $||\mathbf{F}_{\sigma_g}(x_0, y_0)||$ becomes larger since the pairwise vector components no longer cancel. For every σ_g, a pixel (x_0, y_0) belongs to some region of attraction. If $\sigma_s(x_0, y_0)$ is small enough so that the spatial support of $d_s(\cdot)$ does not extend beyond the region, then $||\mathbf{F}_{\sigma_g}(x_0, y_0)||$ is small and the vector direction very sensitive to noise. If $\sigma_s(x_0, y_0)$ is increased so that the spatial support of $d_s(\cdot)$ extends beyond the region, $||\mathbf{F}_{\sigma_g}(x_0, y_0)||$ becomes larger and the vector direction becomes stable approximately orthogonal to the nearest part of the region boundary. When this occurs, the pixel has been properly scaled.

Define

$$\sigma_s^- = \min(\sigma_s) \text{ such that } ||\mathbf{F}_{\sigma_g}(x_0, y_0)|| \geq T(\sigma_\mathrm{g}, \sigma_\mathrm{s}) \tag{4.3}$$

The value of *T* determines the smallest feature size that is captured in the field structure. Explicit relationships between structure scale and size can be created by varying *T* with scale. For example, making T an increasing function of scale increases the effective minimum structure size as scale becomes coarser. In general, in order to allow connected structure of arbitrary size and shape to be detected, one should select $T = \epsilon$, where $\epsilon \gtrapprox 0$. However, if T is too small, the transform is overly sensitive to heavily aliased structures consisting of only a couple of pixels. Experimentally, selecting $T \in [4, 10]$ eliminates the detection of such structures while still allowing all other structures to be identified. In general, T represents the minimum value for $\left\|\mathbf{F}_{\sigma_g}(x_0, y_0)\right\|$ that signifies the existence of an edge, and when $\left\|\mathbf{F}_{\sigma_g}(x_0, y_0)\right\| = T$, an edge exists roughly within a distance of σ_s^- from the pixel. If the pixel is near the region boundary, σ_s^- is small, and σ_s^- is approximately equal to the radius of the region if the pixel is near the region center.

The value of σ_s^+ corresponds to the largest $\sigma_s(x_0, y_0)$ that does not result in overscaling, i.e., the pixel becoming attracted to a disconnected zone of attraction. This situation is detected by examining the behavior of $\mathbf{F}_{\sigma_g}(x_0, y_0)$ as $\sigma_s(x_0, y_0)$ is increased beyond σ_s^-. As this occurs, the pixel initially belongs to the connected zone of attraction. The vector direction tends to change very slowly as long as this is true. However, when $\sigma_s(x_0, y_0)$ becomes large enough, a transition occurs and the pixel now belongs to a different zone of attraction. The vector direction changes much more rapidly during such a transition. As $\sigma_s(x_0, y_0)$ is further increased, the same cycle of behavior may recur: an interval of little change in the vector direction followed by rapid change during transition. We term the state of little change as characterized by *spatial stability*. For each zone of attraction to which the pixel (x_0, y_0) belongs, the value of $\sigma_s(x_0, y_0)$ for which the vector direction changes the least is called a *spatially stable point*. Figure 4.2(a) shows a binary image containing squares composed of nine pixels. A plot of $\left|\frac{d}{d\sigma_s}\theta(\sigma_s)\right|$ is given in Fig. 4.2(b) for a pixel belonging to the square pointed to by A, where $\theta(\sigma_s)$ is defined as the angle in



(a)

(b)

Figure 4.2. (a) Binary image containing square black regions. (b) A plot of the change in vector direction with σ_S is given for a pixel belonging to the square pointed to by A. The three main minima correspond to the spatially stable points. The pixel belongs to a different zone of attraction at each of these points, corresponding to the three perceptual dot clusters to which the pixel belongs.

radians of $\mathbf{F}_{\sigma_g}$ (x_0, y_0) from some reference direction. The stability points of the graph occur at $\sigma_s = \{2, 9, 48\}$. These are the locations of the three largest local minima in the graph. Each minimum corresponds to a different zone of attraction for this pixel. These zones of attraction correspond to the three clusters in the image to which one perceives the pixel *A* belonging. The cluster containing all pixels in the image also has a stability point, but this point occurs for a larger σ_s than shown in Fig. 4.2(b). To prevent overscaling, the upper bound, σ_s^+, is defined as the value of the first spatially stable point, i.e.,

$$\sigma_s^+ = \min(\sigma_s) \text{ such that } \frac{d}{d\sigma_s}\left|\frac{d}{d\sigma_s}\theta(\sigma_s)\right| \geq 0 \tag{4.4}$$

for $\sigma_s \geq \sigma_s^-$. Selecting $\sigma_s(x_0, y_0) \in \left[\sigma_s^-, \sigma_s^+\right]$ for each pixel yields an $\mathbf{F}$ with every force vector belonging to a region of attraction.

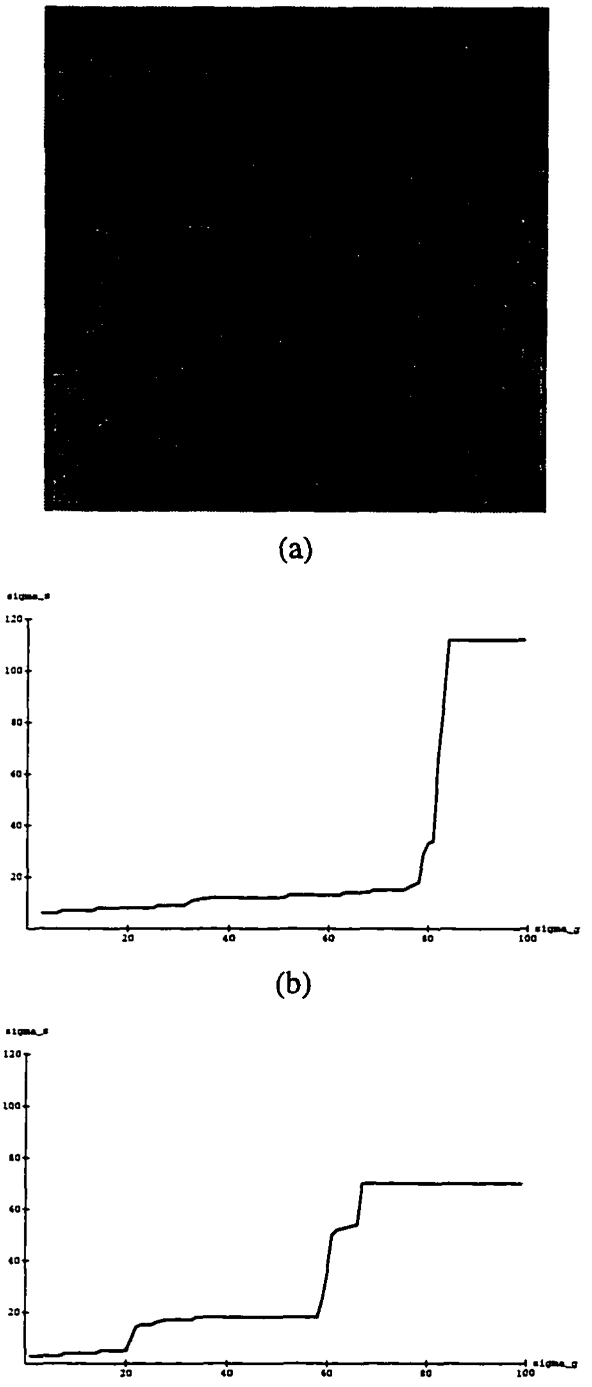
4.2.3. Behavior of F with σ_g

In this section, the effect of varying σ_g on $\mathbf{F}$ is examined first at the level of individual pixels, and then for entire structures. For a given value of σ_g, each $\sigma_s(x_0, y_0)$ is chosen properly as detailed in the preceding section.

As σ_g varies, a pixel initially belongs to the same region of attraction for some range of values, but at some point it makes a transition and becomes a part of another region of attraction. These transitions are readily discernible by examining the $\sigma_g - \sigma_s$ plot of a pixel. As σ_g increases, the region of attraction to which a pixel belongs increases in size as pixels previously outside the region (because they were too dissimilar), gradually merge into the region. Because σ_s^- represents the approximate distance between the pixel and the nearest boundary of its region of attraction, σ_s^- is nondecreasing as σ_g increases. While the pixel belongs to the same region of attraction, σ_s^- increases slowly as σ_g increases. The nearest boundary point of the next (coarser) region of attraction of the pixel is farther away than the nearest boundary point of the present region of attraction, resulting in a discontinuity in the value of σ_s at the σ_g where the transition occurs. Examples of two typical $\sigma_g - \sigma_s$ plots are given in Fig. 4.3.

Figure 4.3(a) shows an image of people at a 3–D movie. Plots of σ_s vs. σ_g are shown in Fig. 4.3(b) and (c) for the pixels located at the centers of the leftmost and rightmost crosshairs (+) overlaid onto the image, respectively. The pixel for (b) belongs to two relevant regions of attraction. The first consists of the woman's face, and the second occurs as the face merges into the background region. The pixel for (c) belongs to three relevant regions of attraction. The first consists of the man's shirt, the second occurs when the shirt merges with the rest of the man, and the third occurs when the man merges into the background. For both (b) and (c), transitions between regions of attraction are clearly indicated by discontinuities in the value of σ_s.

From examination of the σ_g–σ_s plot of a pixel, both the number of structures to which the pixel belongs as well as the range of σ_g for which the pixel belongs to each structure can be determined. However, in and of itself, this does not provide enough information to do structure identification. An understanding of the behavior of entire structures with respect to σ_g is also required. As σ_g varies, some structures coalesce while others disappear. Because image structures generally have nonzero variance, regions of attraction form and disappear over some range of σ_g. Hence, for any given structure, as σ_g is increased, the structure gradually forms, then remains relatively stable for some interval, and then gradually disappears as its constituent pixels form into other, coarser structures. The identification of these structures from **F** is addressed in Section 4.4.



(a)

(b)

(c)

Figure 4.3. (a) Image of people at a 3–D movie. (b) and (c) show plots of σ_s vs. σ_g for the two pixels located under the leftmost and rightmost crosshairs (+) overlaid onto the image, respectively. For (b), the pixel belongs to two regions of attraction: the woman's face before and after it merges into the background. For (c), the pixel belongs to three regions of attraction: the man's shirt, the shirt merged into the rest of the man, and the man merged into the background. Transitions between regions of attraction are clearly indicated by discontinuities in the value of σ_s.

4.2.4. Homogeneity distance function selection

The purpose of this section is to motivate briefly the use of a box-car function for $d_g(\cdot)$. Consider Fig. 4.4, which contains a uniform triangular region having contrast C with the background. The transform is computed at points P_1 and P_2 within the region for the same value of σ_s, represented by the radius of the drawn circles. If a box-car is used for $d_g(\cdot)$, then both points merge into the background region at $\sigma_g = C$. However, if $d_g(\cdot)$ is Gaussian, P_1 merges into the background earlier than P_2 because the proportion of area within the circle centered at P_1 occupied by the region is less than the proportion within the circle at P_2. The net effect is that the range of σ_g for which the region merges into the background is increased. This result also holds under the addition of zero-mean noise. It is shown below that, for a 1–D signal, the form of $d_g(\cdot)$ that yields the shortest transition interval is the box-car function. The same result also holds for higher dimensional signals.

Consider the 1–D signal

$$I(x) = \begin{cases} i_1, & 0 \leq x \leq L \\ i_2, & else \end{cases} \tag{4.5}$$

where $|i_1 - i_2| = C$. For the discontinuity at $x=0$ we seek the $d_g(\cdot)$ that jointly maximizes

$$\{d_g(0,\sigma_g) - d_g(C,\sigma_g)\} \int_0^L d_s(x,\sigma_s)\, dx \tag{4.6}$$

for $\sigma_g \leq C_1$ and minimizes

$$C_2 - C_1 \tag{4.7}$$

subject to the constraints

$$||\mathbf{F}_{\sigma_g}(0)|| \geq T, \quad \sigma_g \leq C_1 \tag{4.8}$$

$$||\mathbf{F}_{\sigma_g}(0)|| < T, \quad \sigma_g > C_2 \tag{4.9}$$

$$C_2 \geq C_1 \tag{4.10}$$

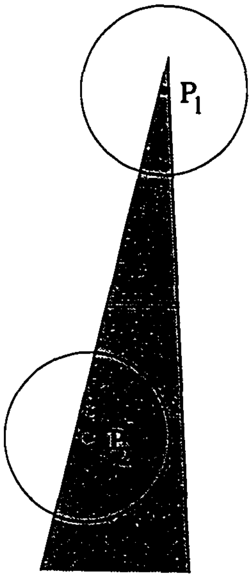


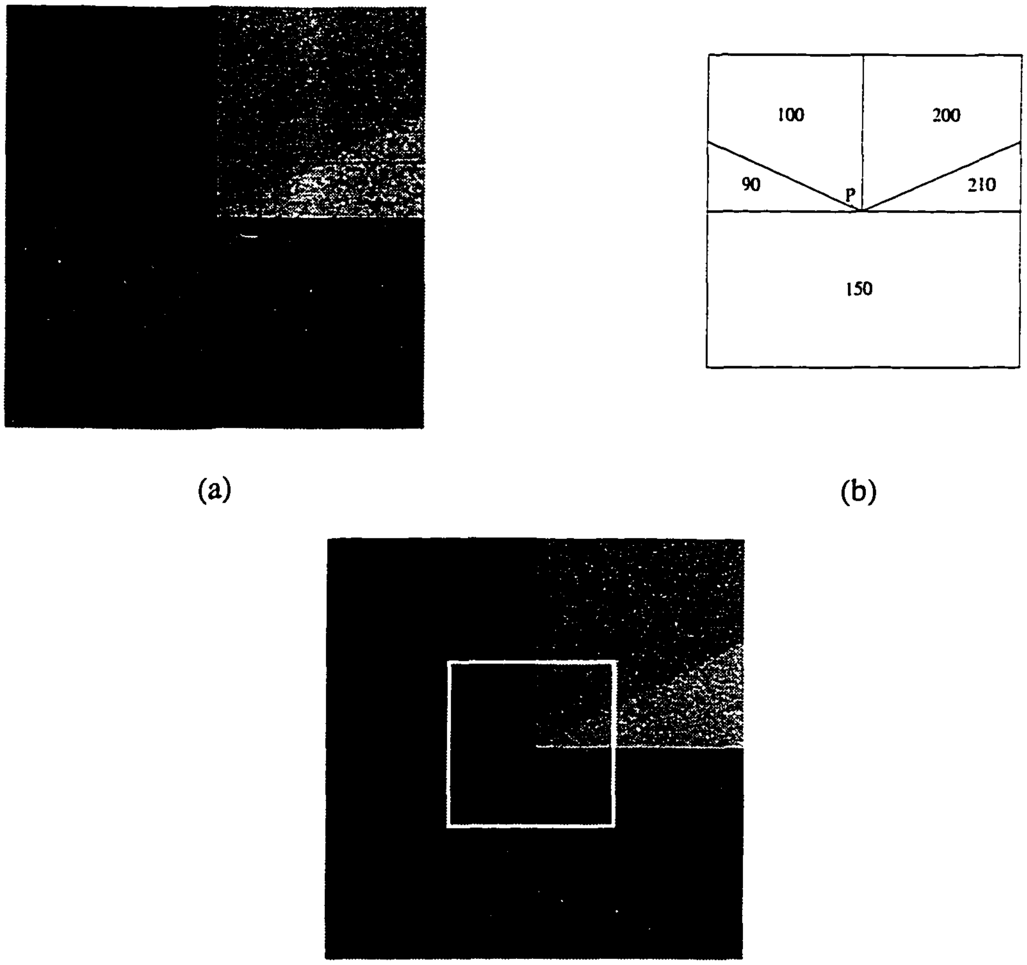
Figure 4.4. Demonstration of the increase in the interval of σ_g for which structures transition because of the use of functions for $d_g(\cdot)$ other than a box-car . Consider the transform computed at points P_1 and P_2 within a uniform triangular region, and using the same value of σ_s, represented by the radius of the drawn circles. If $d_g(\cdot)$ is box-car, both points transition to the background at the same value of σ_g, whereas if $d_g(\cdot)$ is Gaussian, P_1 transitions before P_2, resulting in a finite transition interval for the region.

Eq. (4.6) represents the smallest structures that can be detected by the transform, while Eq. (4.7) minimizes the transition interval. The constraints (Eqs. (4.8)–(4.10)) require the transition to be well-behaved. Given these criteria, it is trivially shown that the optimal form of $d_g(\cdot)$ is the box-car window. For this form, the transition occurs at $\sigma_g = C = C_1 = C_2$. Also, it is readily shown that this form is also optimal for field vectors at all other values of x.

Other forms for $d_g(\cdot)$ result in a longer transition interval, thereby reducing the interval of σ_g for which a structure is stable, possibly even eliminating the interval entirely. Only structures that are marginally stable (using the stability measure presented in Section 4.4) are negatively affected by this, which explains why the use of a box-car for $d_g(\cdot)$ *vis-a-vis* other functions results in only a slight improvement in segmentation quality.

4.3. The Transform as Multiscale Gradient Operator

Because for connected structures the field domain contains both a region and an edge description of the image structure, it is interesting to think of the transform as an edge detector because an edge has a precise definition, namely, that an edge is equivalent to a local maximum of the gradient. At the inner scale, a gradient is a linear operator and the standard definition applies, i.e., $\nabla I = I_x\vec{i} + I_y\vec{j}$, where I_x and I_y are computed by nearest neighbor differences because I is discrete. Consider the difficulty, however, of defining a gradient at any other scale. A gradient has to capture the local rate of change in intensity over an area defined by the spatial scale, while ignoring intensity variations that are irrelevant at the present homogeneity scale. Virtually all previous work on edge detection (see, for example, [50–57]) defines a gradient as utilizing operations based upon local differences in intensity populations. Such approaches fail in areas with complicated geometry, such as in Fig. 4.5(a). This image consists of the five regions of constant intensity shown in (b), where the horizontal and vertical edges are step transitions, and the two diagonal edges have linear ramp profiles six pixels in length, and to which white, Gaussian noise having $\sigma = 5$ was added. At some scale, each of these regions is relevant. Consider what sort of operator is necessary to compute the gradient at point P, which lies within region 100 and along a diffuse edge of low contrast and low signal-to-noise ratio, and which is in close proximity to much sharper and contrasted edges. An operator that directly looks for local differences in intensity populations is virtually guaranteed of being unduly influenced by the discontinuities between region 100 and regions 200, 210, and 150. As a result, both the gradient magnitude and direction will not reflect the edge between regions 100 and 90.

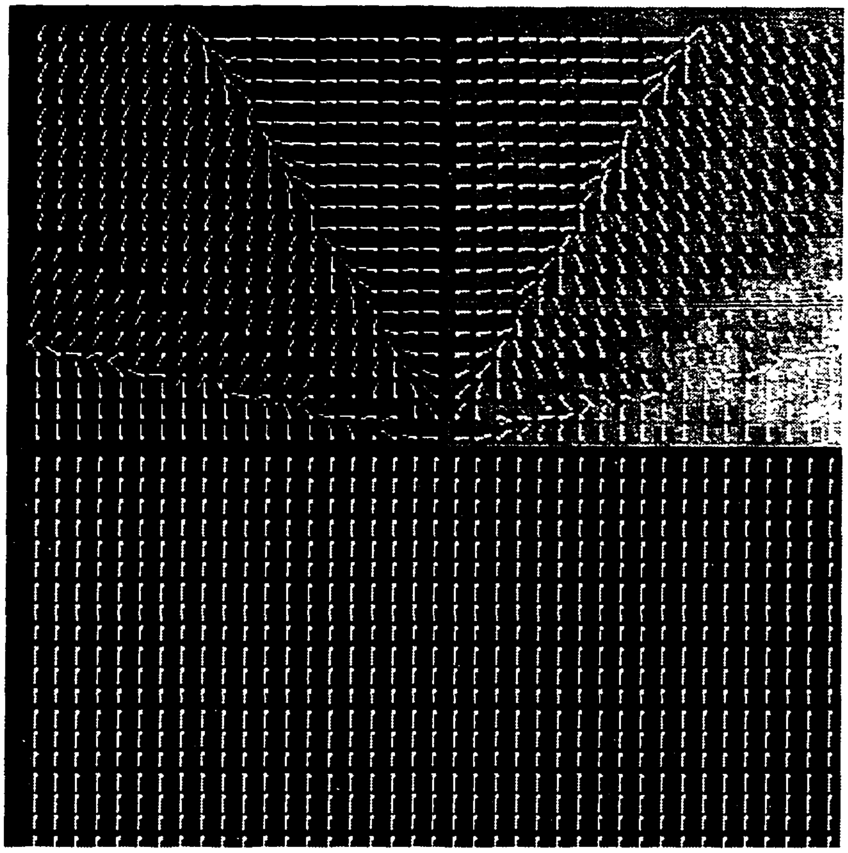


(a)

(b)

(c)

Figure 4.5. Demonstration of the transform as a multiscale gradient operator. (a) Synthetic image consisting of the five regions of constant intensity shown in (b), where the horizontal and vertical edges are step transitions, and the two diagonal edges have linear ramp profiles six pixels in length, and to which white, Gaussian noise having $\sigma = 5$ was added. (c) Regions of attraction present within $\mathbf{F}_7$ displayed by their average intensity.



(d)

Figure 4.5 (cont.). (d) F_7 vectors for each pixel within the box in (c). The vector directions align very closely with the perceived gradient direction at the given scale. This occurs even at complex geometries, such as at point *P*, which lies within region 100 and along a diffuse edge of low contrast and low signal-to-noise ratio, and which is in close proximity to much sharper and contrasted edges.

The transform, however, is a fundamentally different sort of operator than those of previous methods because it is based upon pixels grouping together on the basis of similarity. Information about discontinuities in local pixel populations arises purely as a by-product of this grouping process. For example, at a scale where the edge between regions 90 and 100 exists, point *P* experiences relatively little attraction for any pixels outside of region 100. Hence, both the number and geometry of the structures outside of region 100 is largely irrelevant to the magnitude and direction of the computed force vector. As a result, the force vector directions align very closely to the direction of the gradient. One can verify this in Fig. 4.5(d), which displays $\mathbf{F}_7$ as a needle diagram for each pixel within the box in (c). At this value of σ_g, all five structures are present, as can be seen from (c), which displays the average intensity of each of the regions of attraction present within $\mathbf{F}_7$. Because of the close alignment at any scale between the direction of a force vector and the gradient direction, the transform can be thought of as being very similar to a multiscale gradient operator, but with the bonus that local gradient maxima (edges) and minima (skeletons) are represented within the field by π phase discontinuities.

4.4. Identifying Regions of Attraction within F

This section presents a method for identifying the regions of attraction within $\mathbf{F}$ that correspond to relevant image structure. In order to illustrate the integration of edge- and region- based representations within $\mathbf{F}$, Section 4.4.1 describes a purely edge-based structure extraction method that gives closed contours, and, hence, regions. In Section 4.4.2, a stability criterion is used to identify which of these structures is perceptually valid. Finally, the issue of a compact and efficient form for representing the segmentation is addressed in Section 4.4.3.

4.4.1. Identifying region boundaries

Because Eq. (3.10) holds along each boundary curve, the vectors along each boundary curve diverge from one another. For the interval of σ_g for which a structure is fully-formed within $\mathbf{F}$, these boundaries form closed contours. Thus, structure identification consists of searching $\mathbf{F}$ for all sets of boundaries that form closed contours. Toward this end, a local measure for a region boundary has to be defined. In a continuous space, one can show that the only place within $\mathbf{F}$ where vectors diverge from one another is at region boundaries, and that this divergence is always π, regardless of the boundary geometry [35]. However, discretization may reduce this divergence somewhat, and, for regions containing concavities, causes vcctors to diverge from one another across the line segment formed by joining the center of the concavity on the region boundary to the nearest point on the medial axis of the region, as illustrated in Fig. 4.6. In a continuous space, the angle between two vectors across segment $\overline{AB}$ approaches zero in the limit as the vectors are chosen arbitrarily close to $\overline{AB}$. For a digital image, the finite spatial resolution causes a nonzero angle of at most ϵ to exist between the vectors across the line segment. The value of ϵ decreases as σ_s, and, hence, *T*, increases. For *T = 1*, it is easily shown that $\epsilon = 2\arctan\big(1/(\sqrt{2}+1)\big) \approx \pi/4$, and for *T = 4*, $\epsilon = 2\arctan\big((1/\sqrt{2}+1/\sqrt{5}+1/\sqrt{10})/(4+1/\sqrt{2}+2/\sqrt{5}+3/\sqrt{10})\big) \approx \pi/7$. The angular distributions of the divergence present at region boundaries and concavities never overlap, so ϵ can be used as a threshold for classifying these two cases. Region boundaries are identified by comparing each vector with its eight nearest neighbors. For example, to determine whether or not a region boundary exists between (x_0, y_0) and (x_0+1, y_0) at $\sigma_g = \sigma_{g_0}$, the following test is used

$$\begin{array}{l} \text{IF } \mathbf{F}_{\sigma_{g_0}}(x_0,y_0)_x \le 0 \ \ \& \ \ \mathbf{F}_{\sigma_{g_0}}(x_0+1,y_0)_x \ge 0 \\ \ \& \ \dfrac{\mathbf{F}_{\sigma_{g_0}}(x_0,y_0) \cdot \mathbf{F}_{\sigma_{g_0}}(x_0+1,y_0)}{\left\|\mathbf{F}_{\sigma_{g_0}}(x_0,y_0)\right\| \left\|\mathbf{F}_{\sigma_{g_0}}(x_0+1,y_0)\right\|} < \cos\epsilon \\ \text{THEN} \quad (x_0,y_0) \ \& \ (x_0+1,y_0) \text{ are boundary points} \end{array} \tag{4.11}$$

Analogous tests are used for the other neighboring vector pairs.

4.4.2. Structure stability

Not all of the structures identified in the previous section are perceptually valid. This is because σ_g reflects the relative homogeneity within a structure, but not the relative contrast of the structure with neighboring structures. The latter is reflected in the extent of a structure in $\mathbf{F}$, or *σ_g–lifetime*. This is measured from the point at which a structure is half-formed to when it has half-disappeared. This results in a pixel being assigned to a structure for every value of σ_g. Exact computation of the σ_g–lifetime of a structure requires examination of the $\sigma_g - \sigma_s$ plot of each pixel within the structure for the discontinuities in σ_s that indicate transitions of the pixel

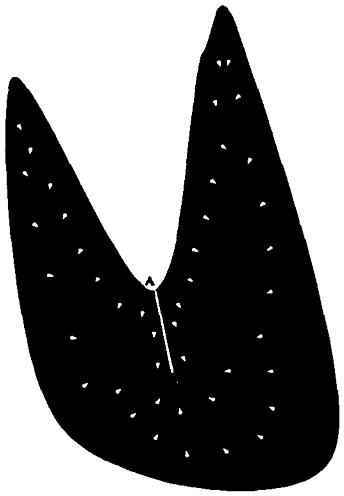


Figure 4.6. The presence of a concavity causes vectors to diverge across the line segment formed by joining the concavity center on the region boundary (Point A) to the nearest point on the medial axis of the region (Point B).

into and out of the structure. This could be done but is unnecessarily expensive computationally. A simpler approach is to approximate the lifetime using only the structure boundary pixels, because a transition at a boundary pixel is reflected in an edge appearing or disappearing, and this edge information is already computed in Section 4.4.1.

A structure is considered stable with σ_g, and, hence, perceptually relevant, if it obeys the following "octave" rule:

A structure with σ_g–lifetime $[g_1, g_2]$, is stable iff $g_2 > 2g_1$.

The octave rule essentially requires the homogeneity variation within a structure to be less than the relative contrast between the structure and neighboring structures. This results in structures that not only become less homogeneous as scale becomes coarser, but also have increasing contrast with neighboring regions. The octave rule yields good results in practice, but is not based on psychophysical criteria. Experiments are planned in which people subjectively evaluate

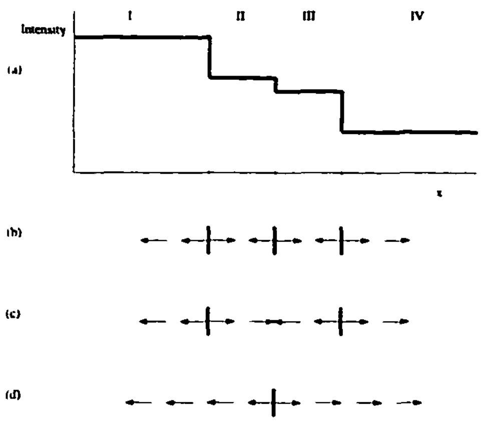


Figure 4.7. (a) 1–D signal with four levels. (b)-(d) Edges and some field vectors are shown for three different scales. (b) At fine scale, four regions are detected. (c) At a coarser scale, regions II and III merge together. (d) At an even coarser scale, the edge separating regions II and III has reappeared, resulting in one region consisting of regions I and II and another consisting of regions III and IV.

segmentations produced by different rules in order to determine a perceptually valid criterion for identifying the regions of attraction that correspond to relevant structures.

4.4.3. Segmentation representation

This section addresses the issue of how best to represent the segmentation obtained in Sections 4.4.1–4.4.2. The segmentation consists of a list of relevant structures along with their σ_g–lifetimes and constituent pixels. Because this segmentation retains the geometric fidelity of structure boundaries at all scales, the segmentation often can be represented as a merging pyramid, i.e., every structure consists of a set union of structures present at any finer scale. In certain isolated situations, however, this does not hold true, as is demonstrated in Fig. 4.7. A 1–D signal having four levels is shown in (a). At some initial scale, the resulting field is shown in (b) along with the three edges detected. These edges separate the signal into four regions, numbered as shown. At some coarser scale, (c), the edge separating regions II and III disappears and these regions merge. However, at a yet coarser scale, (d), this edge reappears and the other two edges disappear.

Because a merging pyramid is such a convenient representation, the segmentation is post-processed to explicitly force it into this form. This is accomplished by projecting structure boundaries downward to all finer scales. This prevents certain regions, such as II and III in Fig. 4.7, from merging together. The segmentation is now representable as a merging pyramid, such as the one in Fig. 4.8. Each node in the pyramid corresponds to a segmented structure. Information about a structure that can be stored at its associated node includes average intensity, texture statistics, σ_g–lifetime, boundary chain-code, moments, and area. The base of the pyramid contains the finest scale structures, and incrementally coarser structures are represented as the pyramid is traversed upwards. The pyramid representation has many useful aspects, including efficient coarse-fine access to the image structures as well as compact storage of the segmentation.

4.5. Implementation Details

This section describes some implementation details of the image segmentation algorithm. The use of box-car functions for the homogeneity and spatial distance functions eliminates the three multiplies required to compute each pairwise pixel attraction vector. Hence, Eq. (4.1) can be evaluated using only look-up tables and integer additions. To further minimize the computation time it is desirable to select σ_s as small as possible, i.e., $\sigma_s = \sigma_s^-$. The transform vectors that take the longest to compute are those of pixels distant from any region boundary becauseσ_s^- is slightly larger than the distance from a pixel to the nearest boundary. Fortunately, the structure extraction algorithm only requires that the force vectors be computed along each region boundary. Thus, σ_s may be capped at the maximum value necessary to compute a force vector at a boundary pixel. This value is given by $\sigma_s^{max} = L/2$, where L is the length of the edge profile of the most

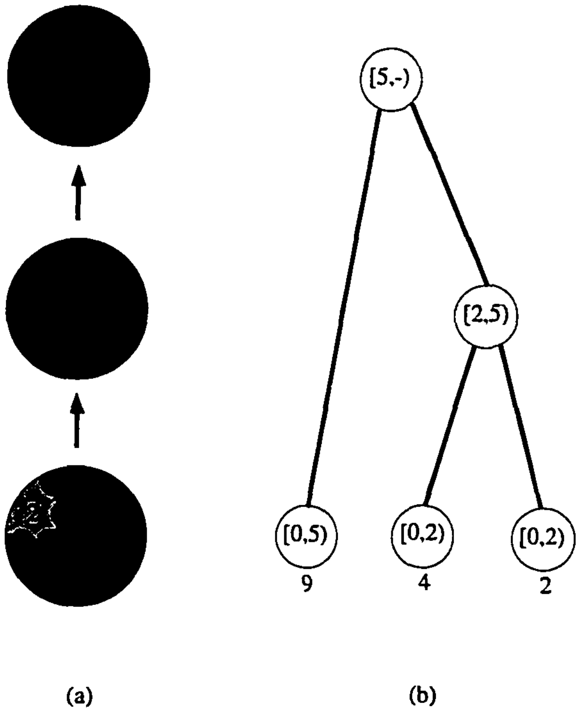


Figure 4.8. (a) Segmented regions at different scales, with the intensities of the finest scale regions labelled. (b) The pyramid that corresponds to this segmentation. Each node corresponds to a different region, and the range of σ_g for which each region is present is shown within each node.

diffuse edge in the image, because at a pixel on the edge boundary (i.e., at the center of the edge profile) the spatial support of the transform should extend into the regions on either side of the edge. Experimentally, $\sigma_s^{max} = 10$ has been found to work well over a wide variety of images. Thus, in the evaluation of Eq. (4.3), if σ_s reaches σ_s^{max}, then the computation is stopped and the pixel is considered not to border a region boundary at the given value of σ_g. In addition, it is not necessary to compute $\mathbf{F}$ over the entire range of $I(x, y)$ (assume 8–bit data, i.e., range of 0–255). Edges of contrast less than 3 are not visible, and regions with a homogeneity variation greater than 50 typically correspond to structures that are coarser than desired because most images have a graylevel histogram with standard deviation of about 20. Thus, σ_g is restricted to the range $\sigma_g \in [3, 50]$. Furthermore, σ_g is subsampled within this range. We have found that 10 samples suffice to identify all structure present within this range. Thus, the scale at each pixel is selected from among 100 possible points within the $\sigma_g - \sigma_s$ plane of that pixel.

4.6. Experimental Results

This section presents the results of the described image segmentation method for a variety of images. Figures 4.9–4.15 use synthetic images to illustrate the performance of the transform in the presence of multiscale structure, complex boundary topology, different forms of the functions $d_g(\cdot)$ and $d_s(\cdot)$, shading, and noise. In all cases, the difference between the actual structure and the detected structure was zero. Next, the performance of the segmentation algorithm on several real images is demonstrated in Figs. 4.16–4.24.

Figure 4.9 demonstrates the ability of the transform to make structure explicitly available. Figure 4.9(a) consists of two rectangles having a constant intensity of 1 on a background of intensity 255. The σ_s values that correspond to $\mathbf{F}_{20}$ are shown intensity coded in (b) with image brightness directly proportional to the σ_s value. The vector directions of $\mathbf{F}_{20}$ are shown in (c) intensity coded so that the brightness is proportional to the clockwise angle the force vector makes from the positive x-axis. Because structure boundaries and skeletons are represented within $\mathbf{F}$ by diverging and converging field vectors, respectively, with phase differences of approximately π, they appear as sharp intensity discontinuities. Some intensity discontinuities, however, are artifacts resulting from the branch cut present along the positive x-axis, and should be ignored. In (d), the vector directions are displayed after first being rotated clockwise by $\pi/2$. This shifts the bright-dark intensity coding artifacts, allowing easier verification of the image structure in areas in (c) where such artifacts occur. In (e), the vectors in (c) are shown as a needle diagram after 5:1 spatial subsampling to accommodate display on the page. Note from (c)-(e) that the structure boundaries are all present, have closed contours, and retain geometric fidelity.

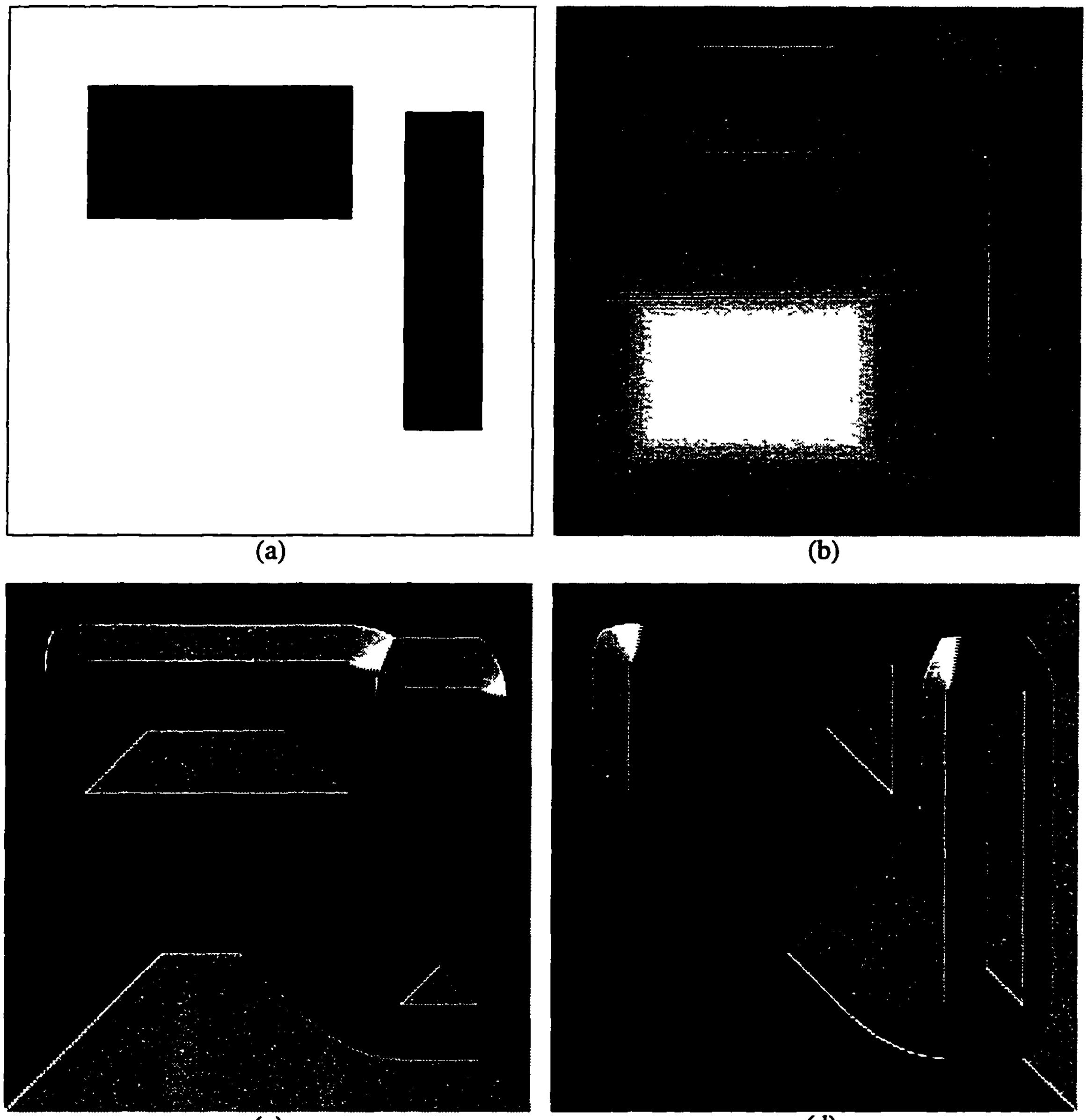
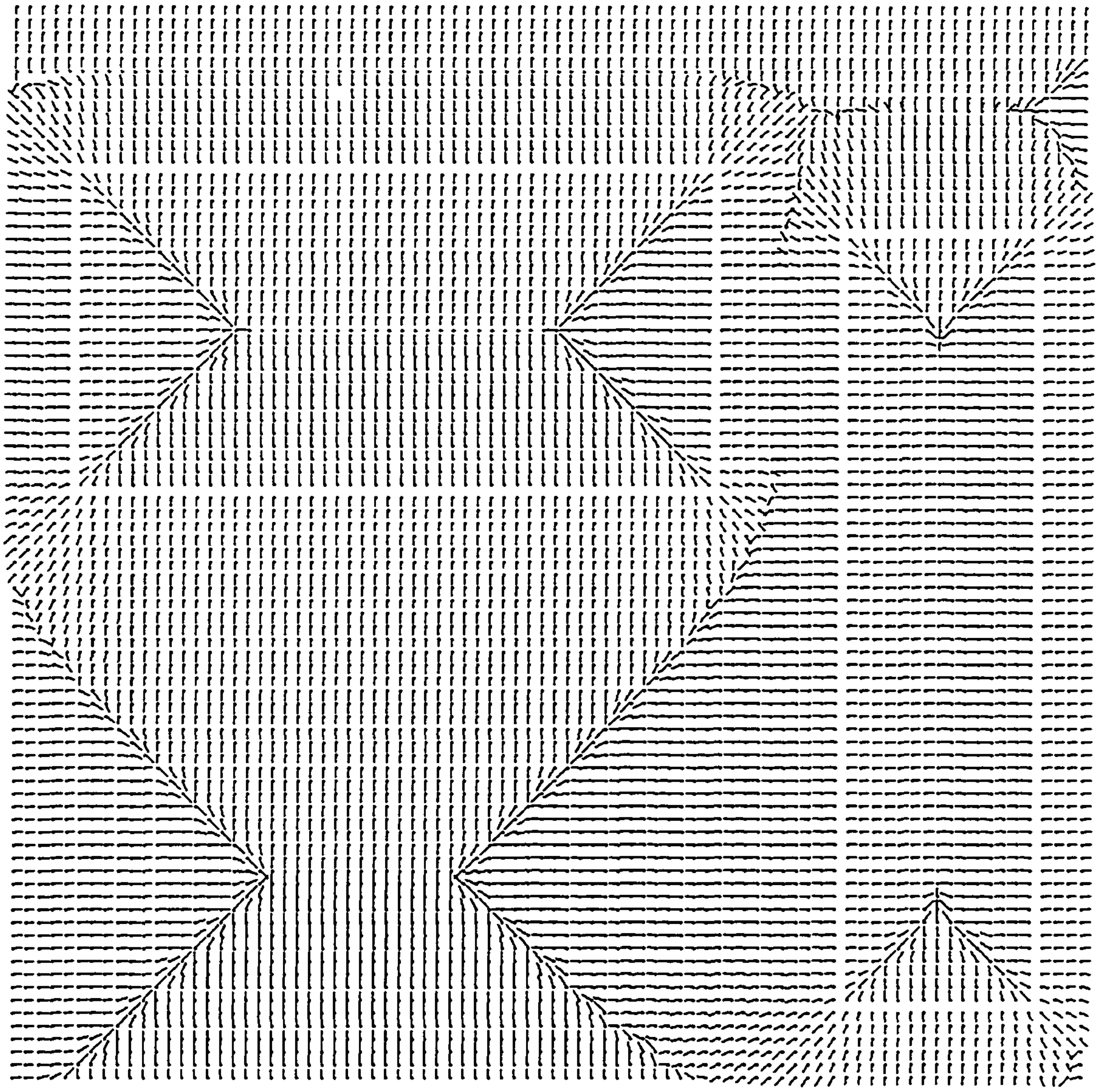
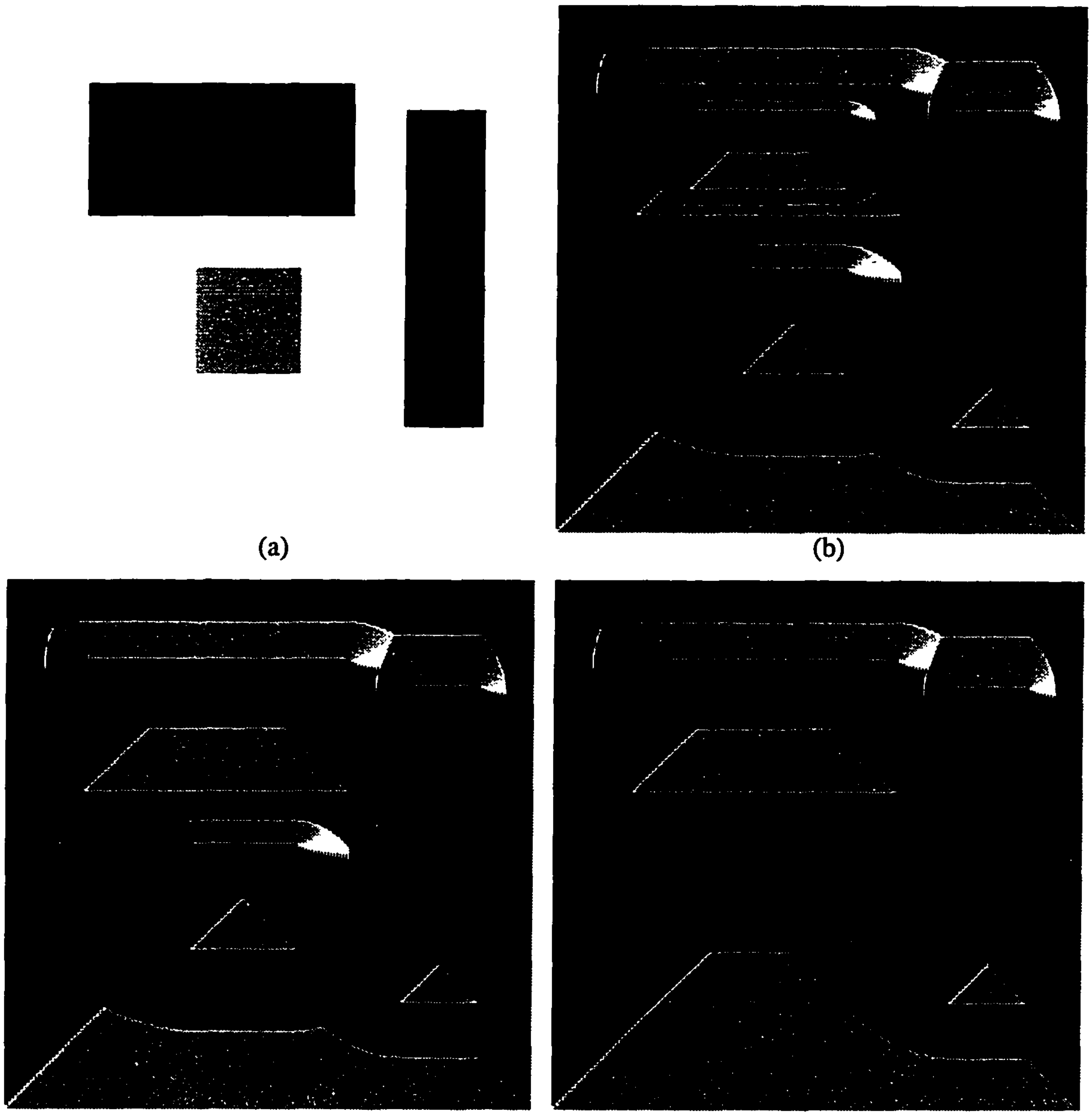


Figure 4.9. Demonstration of the properties of the transform and the region signatures through a simple synthetic image. (a) Two rectangles having a constant intensity of 1, on a background of intensity 255. (b) Intensity coded image of σ_s values chosen to correspond to the rectangular regions. Image brightness is proportional to σ_s value. (c) Force vectors obtained after the transform has been applied using the σ_s values shown in (b), and $\sigma_g = 20$. The vector directions are intensity coded so that the brightness is proportional to the clockwise angle of the force vector from the positive x-axis. Some intensity discontinuities are artifacts of intensity based (linear) coding of the cyclic direction values. (d) Same as (c), but with vector directions rotated 90° clockwise and then intensity coded. This shifts the bright-dark artifacts resulting from the intensity coding, allowing one to more easily verify the field in (c) near areas where such artifacts occur.



(e)

Figure 4.9 (cont.). (e) The vectors in (c)-(d) shown as line segments. The length of the line segment represents the vector magnitude, and the tail of the vector is indicated by a small square.



(a)

(b)

(c)

(d)

Figure 4.10. Demonstration of the multiscale capability of the transform. (a) Same as Fig. 4.9, except that two new, lower contrast regions have been added. The new small rectangle has a graylevel of 30, and the new square has a graylevel of 200. The three contrasts present are 29, 55, and 254. (b) Intensity coded vector direction image using $\sigma_g = 10$. Because 10 is less than all contrasts, all four regions are detected. (c) Analogous to (b) using $\sigma_g = 40$. Because 40 > 29, the new rectangle is lost. (d) Analogous to (b) using $\sigma_g = 60$. Because 60 > 29,55, both new regions are lost, leaving a result identical to Fig. 4.9(c).

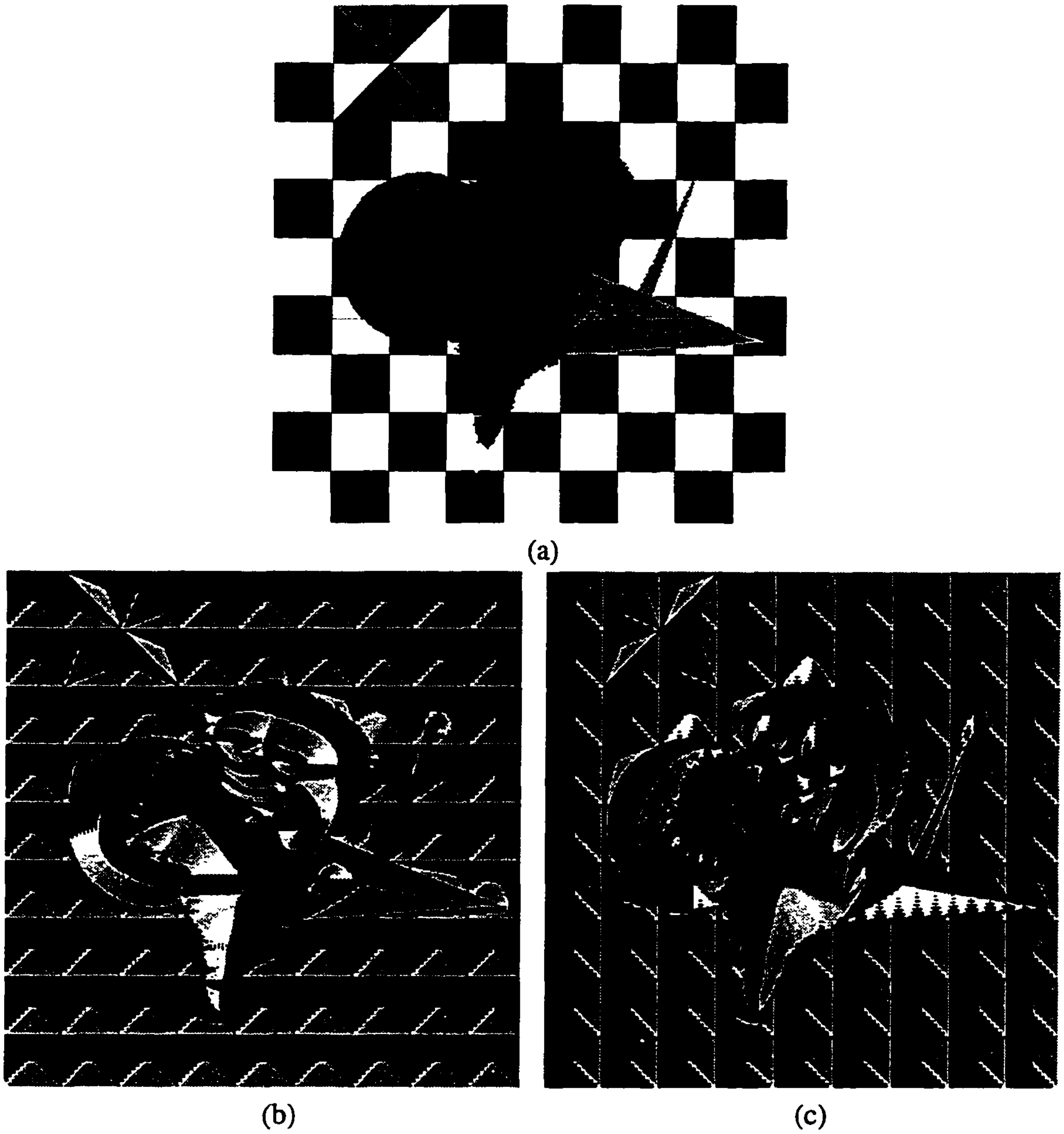


Figure 4.11. Demonstration of the insensitivity of the transform to boundary curvature and topology. (a) Constant value regions having complex boundary structure. The boundaries contain smooth as well as high-curvature segments including corners. Some of the boundary points are vertices where more than two regions meet. All region boundaries have a contrast of at least 15 greylevels. (b) Intensity coded directions computed by the transform using σ_s values chosen corresponding to the region structure for $\sigma_g = 10$. (c) Same as (b), but with vector directions rotated 90° clockwise and then intensity coded. The regions are represented in (b)-(c) by their force field signature without distortion.

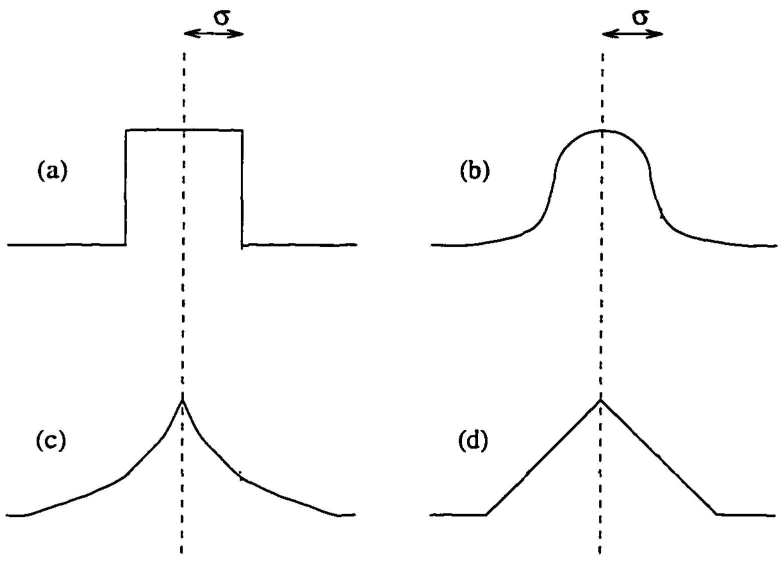
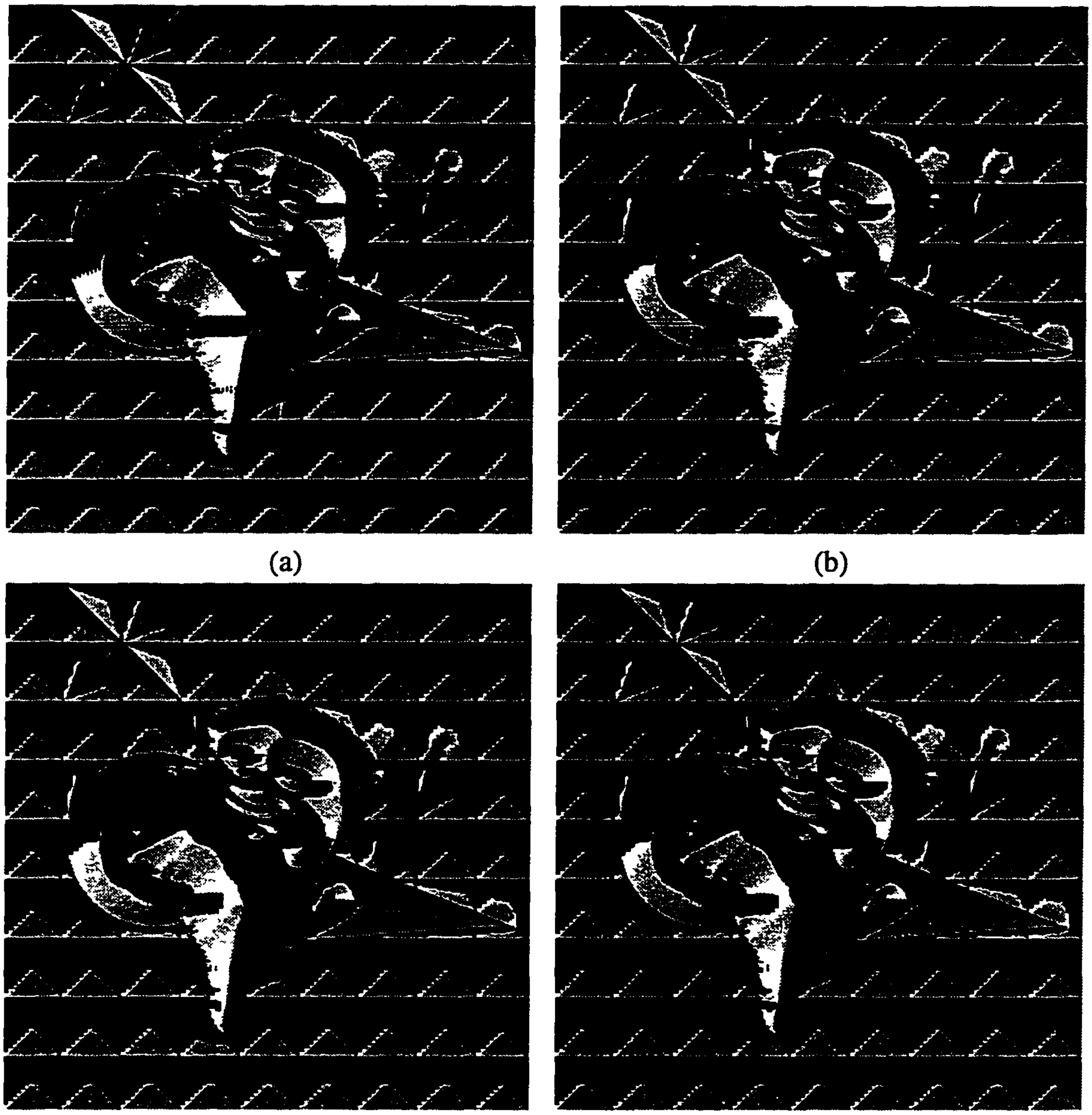


Figure 4.12. Examples of four functions which are used for both $d_g(\cdot)$ and $d_\varsigma(\cdot)$ in Fig. 4.13. (a) Box-car, (b) Gaussian, (c) Exponential, and (d) Linear. The σ shown indicates the σ_g and σ_ς values used by $d_g(\cdot)$ and $d_\varsigma(\cdot)$, respectively.



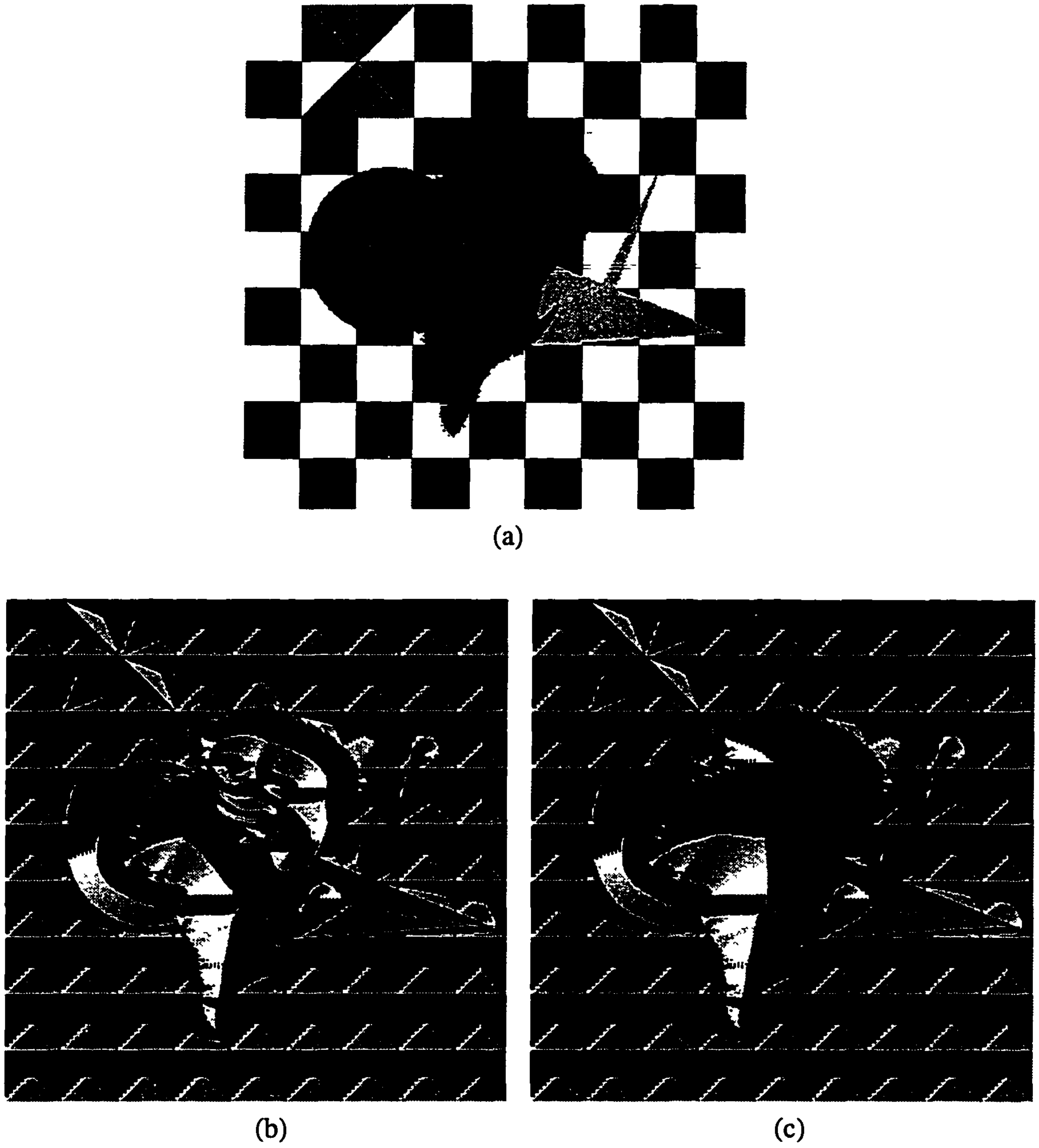
(a)

(b)

(c)

(d)

Figure 4.13. Demonstration of the insensitivity of the transform to the choice of the functions $d_g(\cdot)$ and $d_s(\cdot)$. (a)-(d) Intensity coded directions of Fig. 4.11(a) computed by the transform using σ_s values chosen corresponding to the region structure for $\sigma_g = 10$. The forms used for $d_g(\cdot)$ and $d_s(\cdot)$ are box-car, Gaussian, exponential, and linear, respectively. The region structure is represented within the field without distortion in all cases.

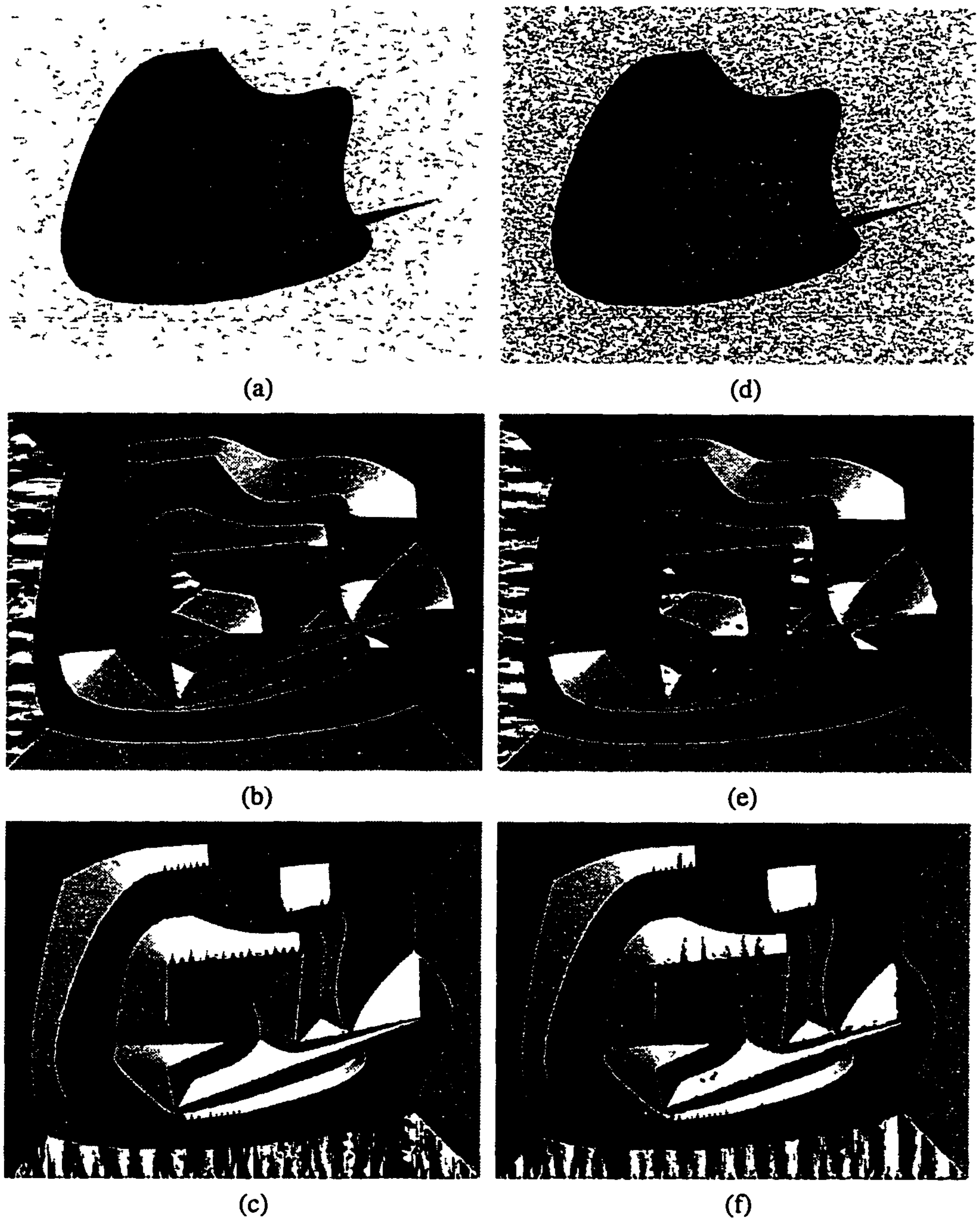


(a)

(b)

(c)

Figure 4.14. Demonstration of the insensitivity of the transform to shading. (a) Same image as in Fig. 4.11(a), but different regions now have spatially varying greylevels. Here, the disk contains linearly increasing greylevels from left to right, whereas the polygonal region has quadratically increasing greylevels from right to left. The irregularly shaped center region contains quadratically varying greylevels which increase from bottom to top. (b) Intensity coded force directions after the transform has been applied using $\sigma_g = 10$. (c) Same as (b) but with $\sigma_g = 25$. At this value of σ_g, the microstructure within the irregular center region has disappeared.

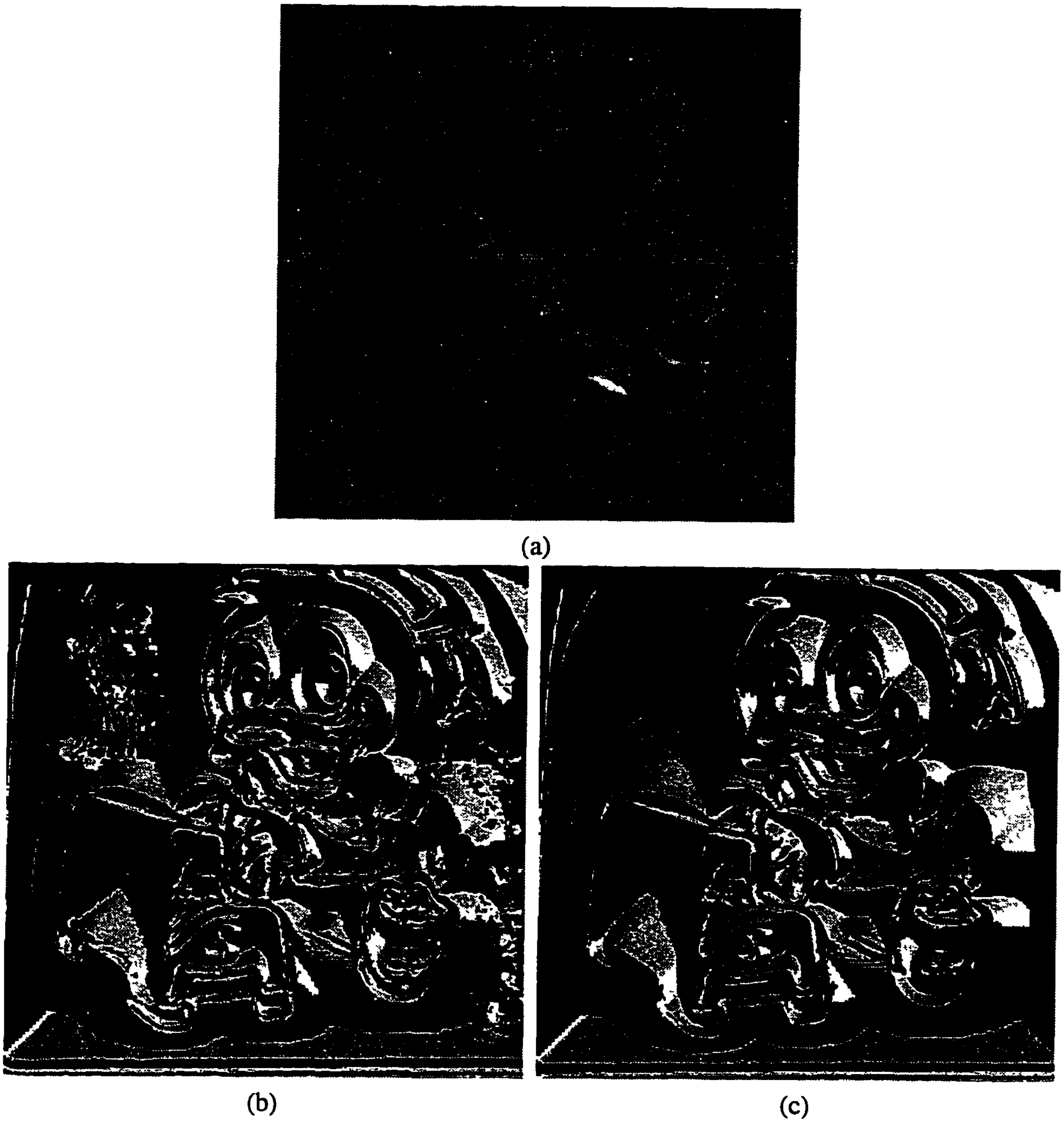


(a) (d)

(b) (e)

(c) (f)

Figure 4.15. Demonstration of the insensitivity of the transform to additive, white Gaussian noise. (a) Image containing two regions of intensity 0 and 100 on a background of intensity 200, and with zero-mean noise of standard deviation $\sigma = 50$. (b) Intensity coded directions computed using $\sigma_g = 50$. (c) Same as (b), but with vector directions rotated 90° counterclockwise and then intensity coded. (d)-(f) Same as (a)-(c), but with noise having $\sigma = 100$. Note that the structure boundaries are all present, have closed contours, and retain geometric fidelity (For example, the sharp corner of the intensity 100 region has been preserved without smoothing), except, of course, where the region structure has been changed because of the noise, in which case the new structure is reflected.



(a)

(b) (c)

Figure 4.16. Demonstration of structures at different homogeneity scales present within **F** for a real image. (a) Donald Duck toy. (b)-(c) Intensity coded vector directions of $\mathbf{F}_{10}$ and $\mathbf{F}_{30}$, respectively. The background substructure, such as the musical notes, is encoded within $\mathbf{F}_{10}$, whereas the entire background is present as a single region in $\mathbf{F}_{30}$.

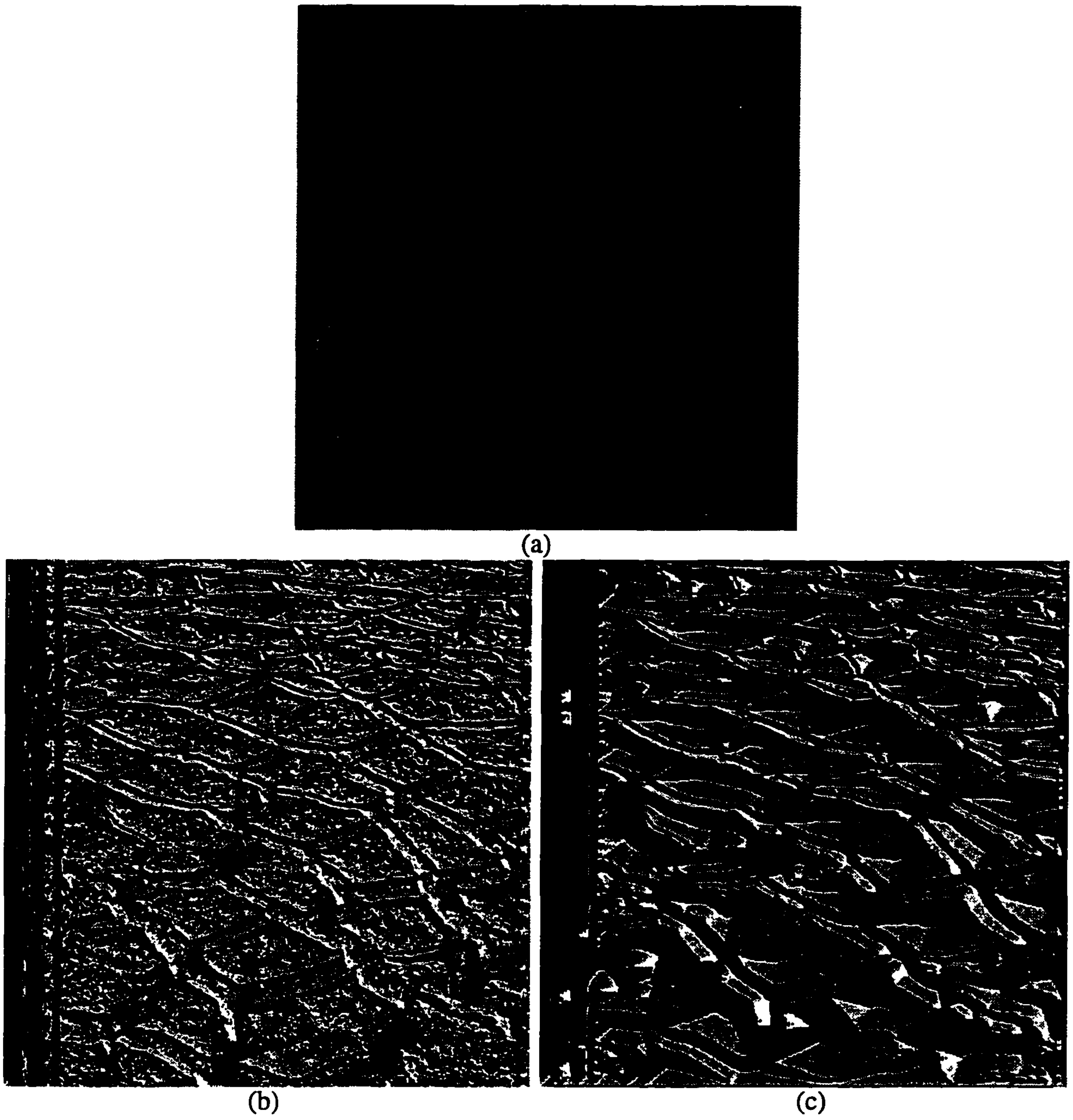


Figure 4.17. (a) An image containing structure at predominantly two scales, with jagged edges particularly visible inside the lighter, cellular regions. The structure is easy to follow visually, which makes it easy to test the transform performance with respect to boundary complexity. (b)-(c) Intensity coded vector directions of $\mathbf{F}_7$ and $\mathbf{F}_{26}$, which correspond to the two major scales.

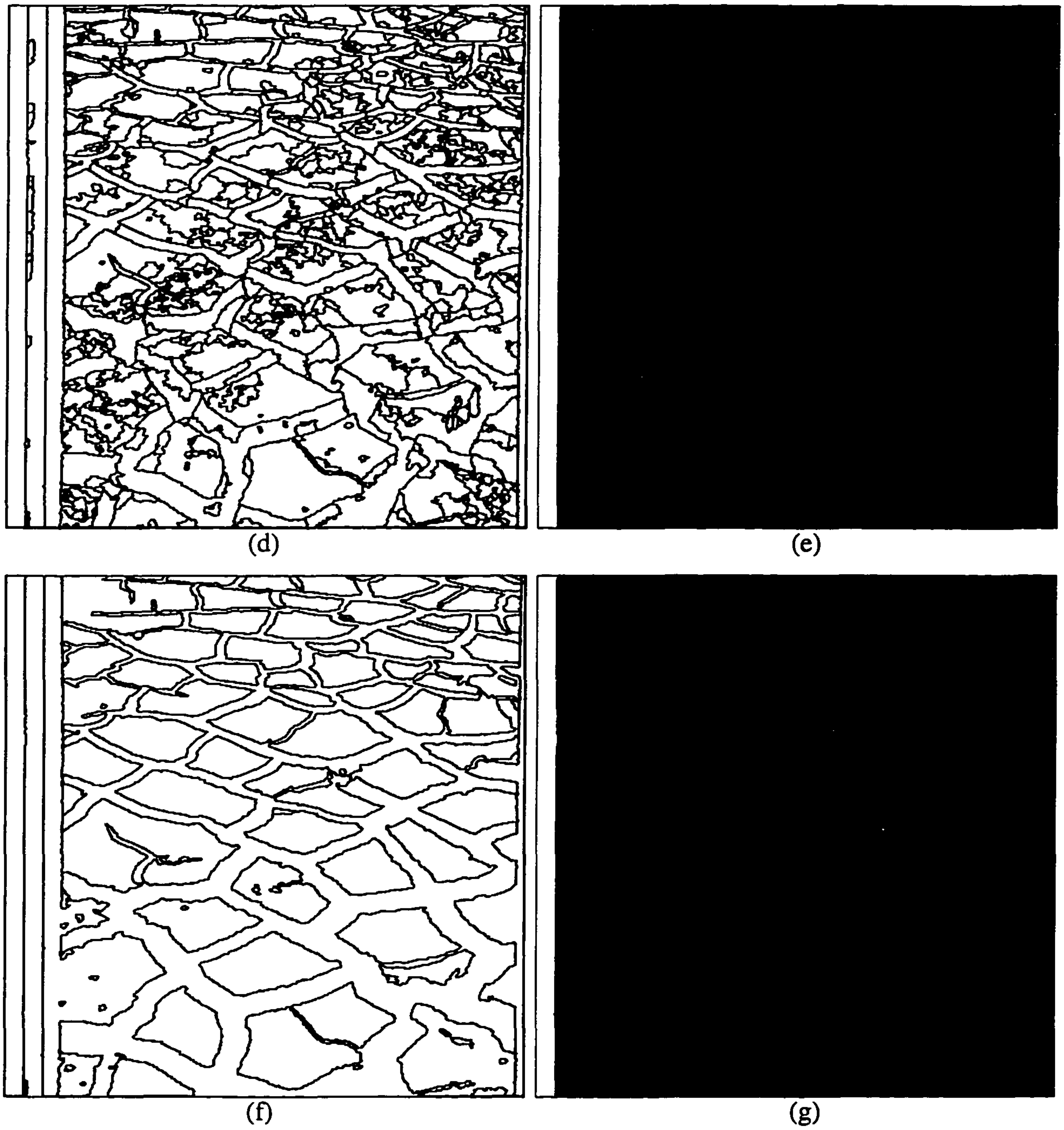
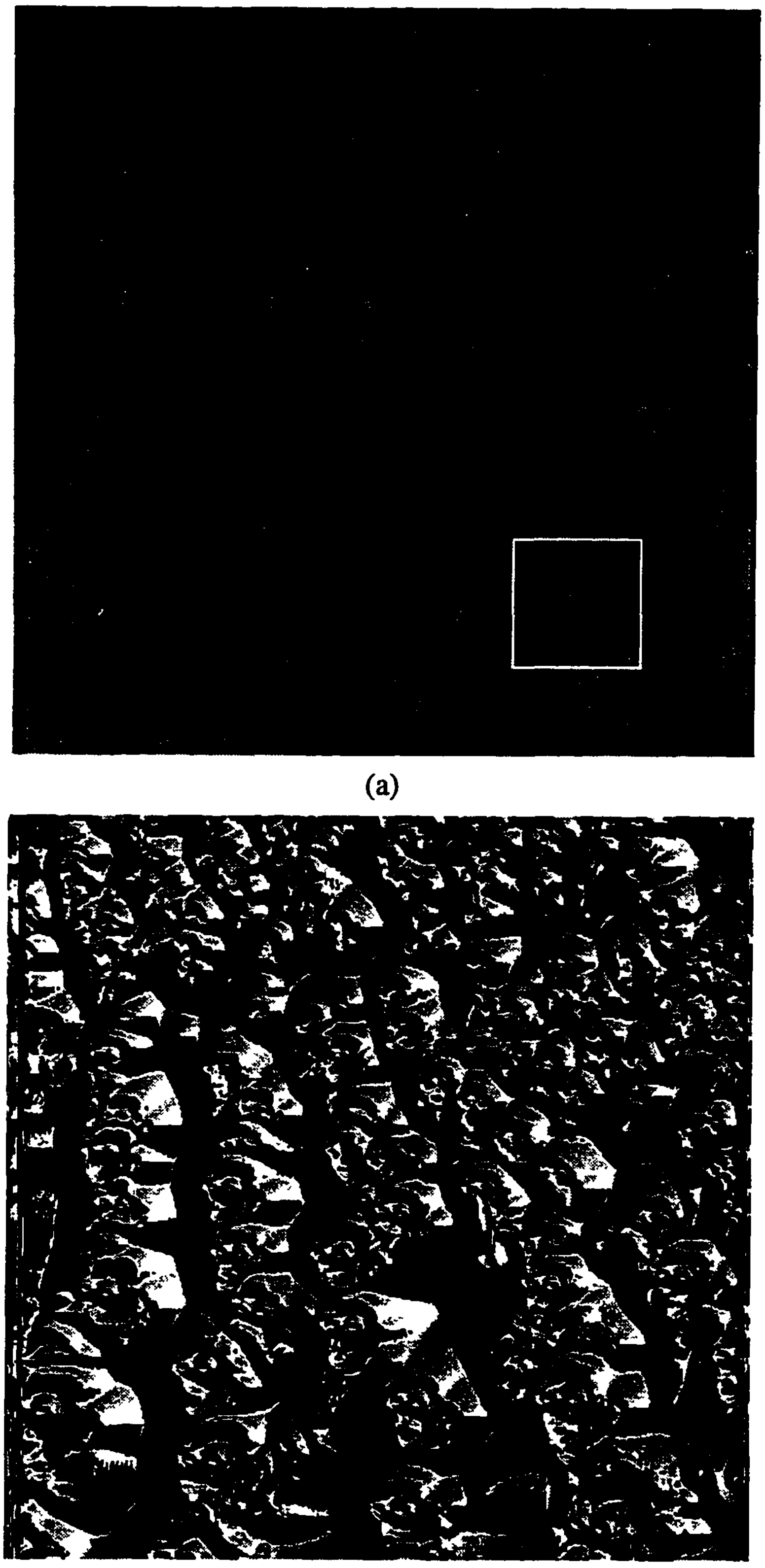


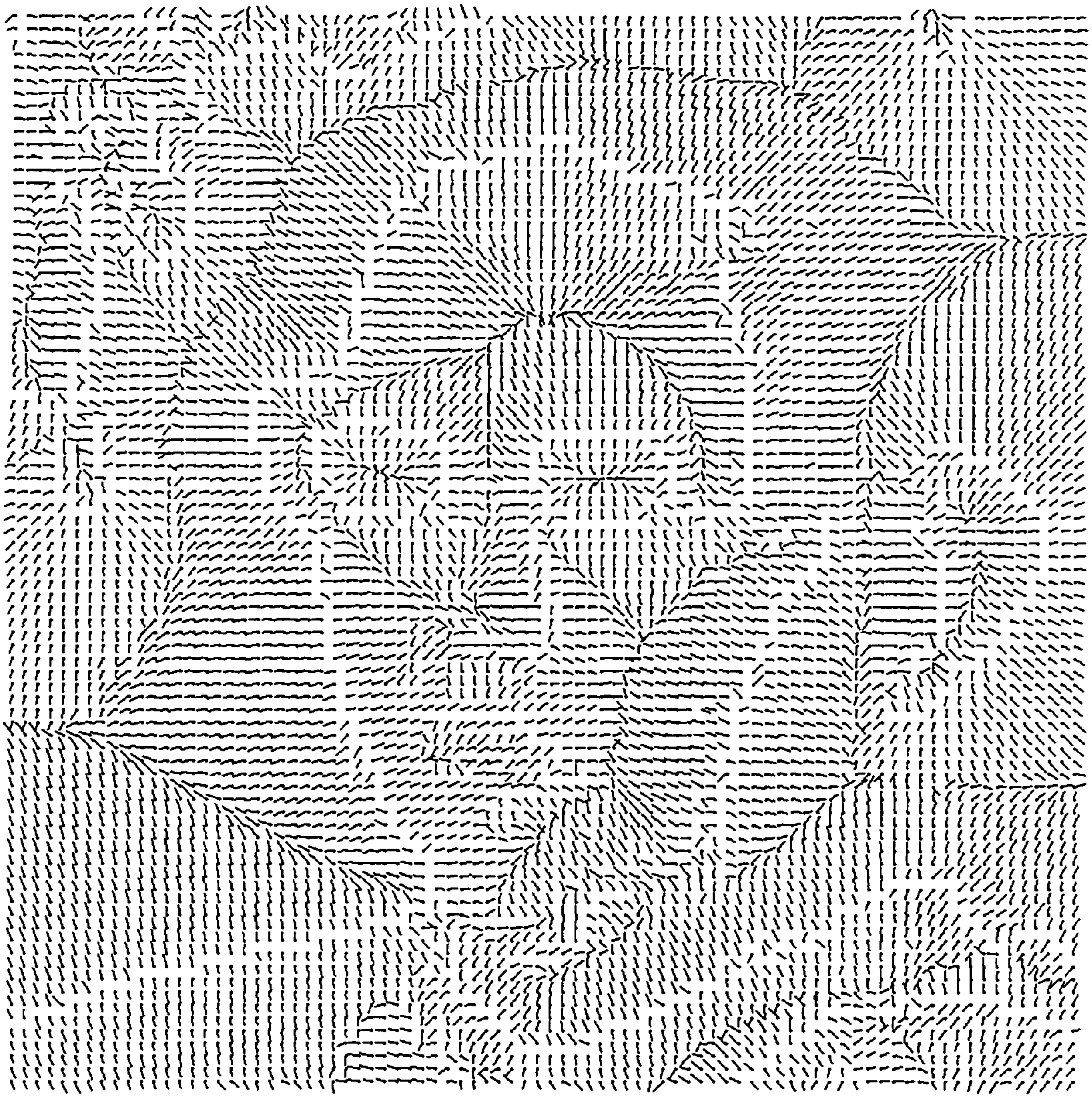
Figure 4.17 (cont.). (d)-(e) Structures present at $\sigma_g = 7$ displayed by both an edge map and by average graylevel. (f)-(g) Same as (d)-(e), but for $\sigma_g = 26$.



(a)

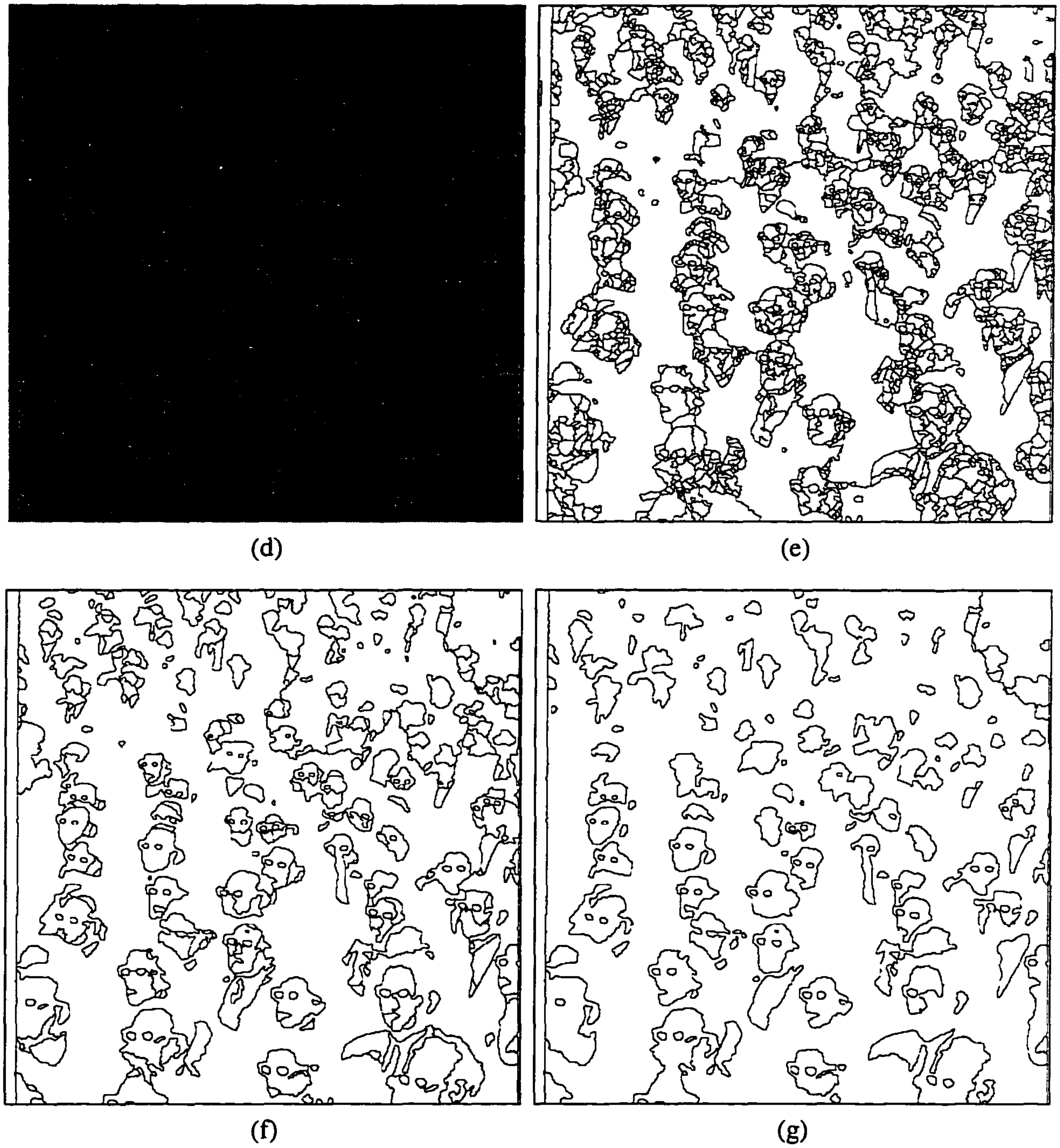
(b)

Figure 4.18. (a) People at a 3–D movie. (b) Intensity coded vector directions of $\mathbf{F}_{30}$.



(c)

Figure 4.18 (cont.). (c) $\mathbf{F}_{30}$ with the vector directions displayed by a needle diagram for the pixels within the window in (a).



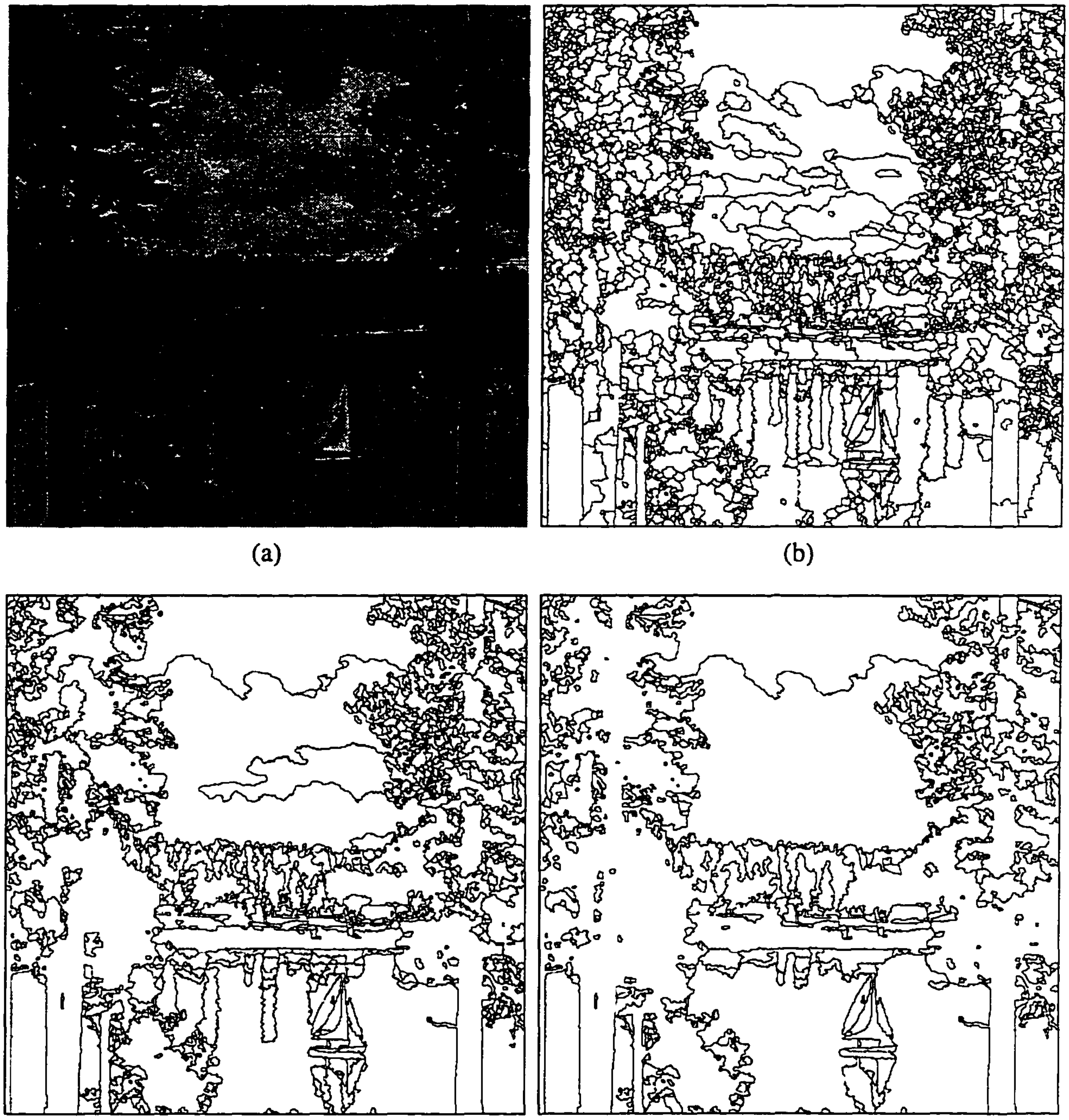
(d)

(e)

(f)

(g)

Figure 4.18 (cont.). (d) People at a 3–D movie. (e)-(g) Region boundaries present at, respectively, σ_g = 5, 21, 39.



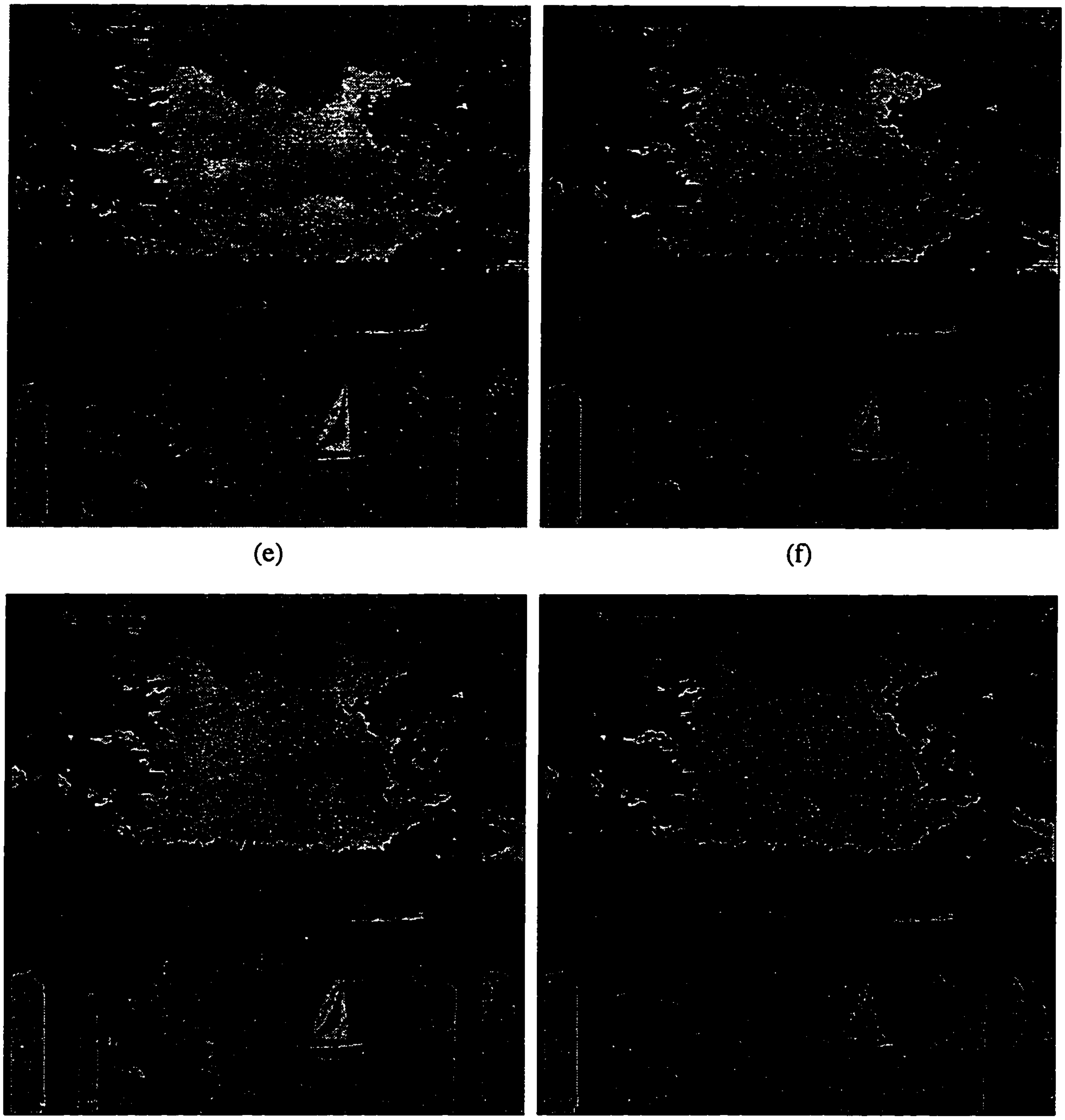
(a)

(b)

(c)

(d)

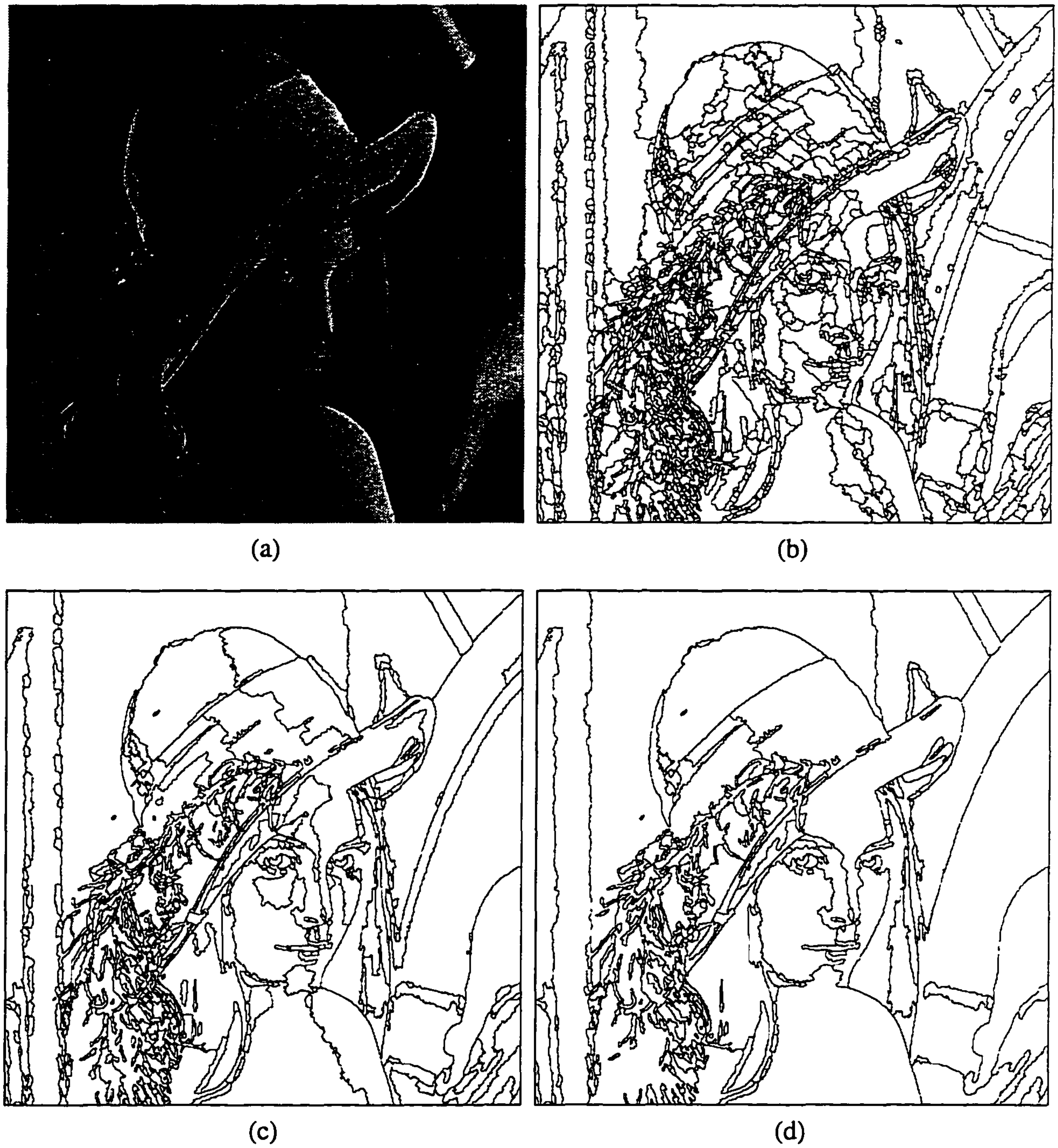
Figure 4.19. (a) A sailboat on a lake. (b)-(d) Region boundaries present at, respectively, $\sigma_g = 5, 10, 26$.



(e) (f)

(g) (h)

Figure 4.19 (cont.). (e) A sailboat on a lake. (f)-(h) Average grayscales of the regions present at, respectively, σ_g = 5, 10, 26.



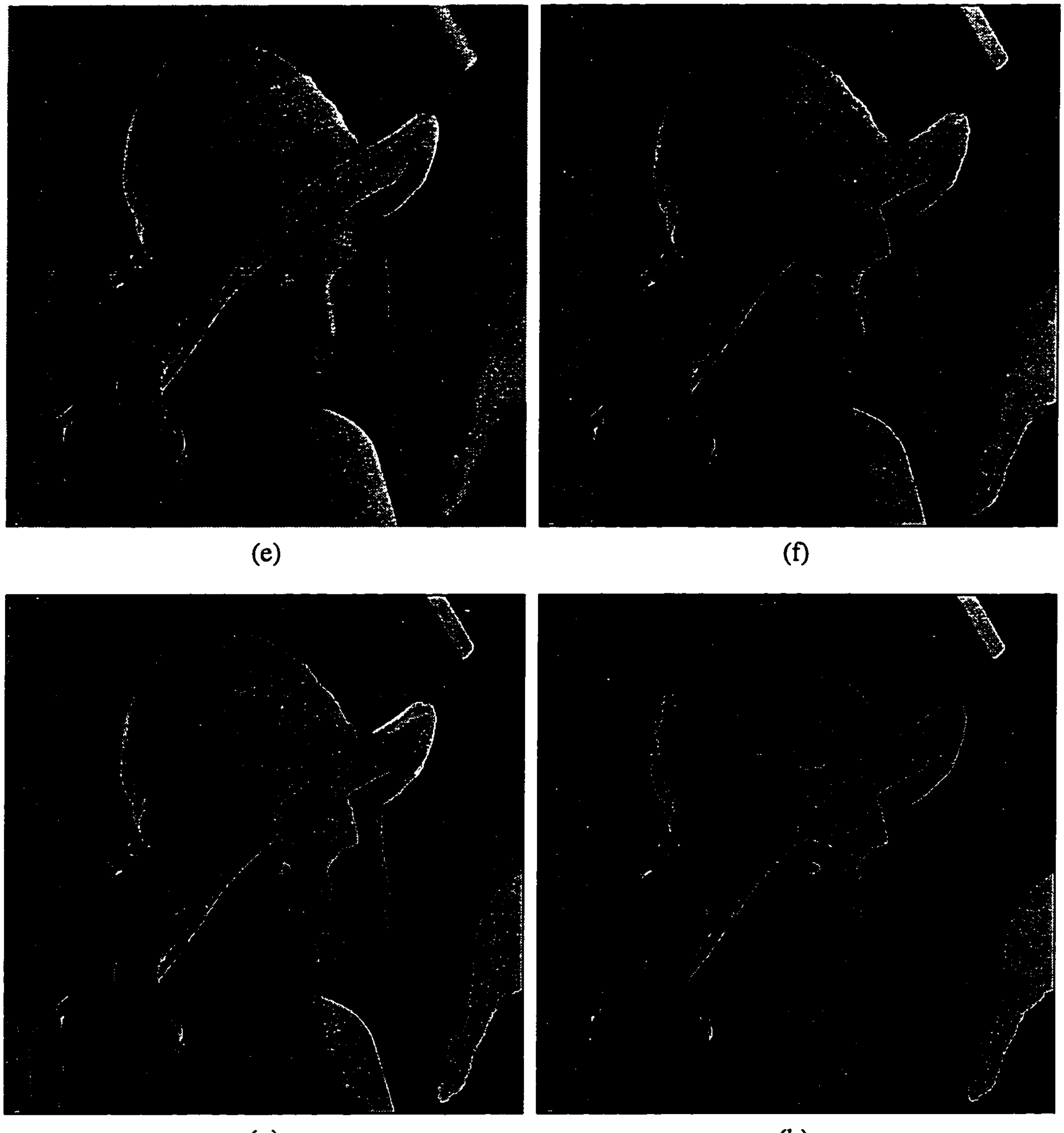
(a)

(b)

(c)

(d)

Figure 4.20. (a) Lena. (b)-(d) Region boundaries present at, respectively, σ_g = 7, 10, 21.



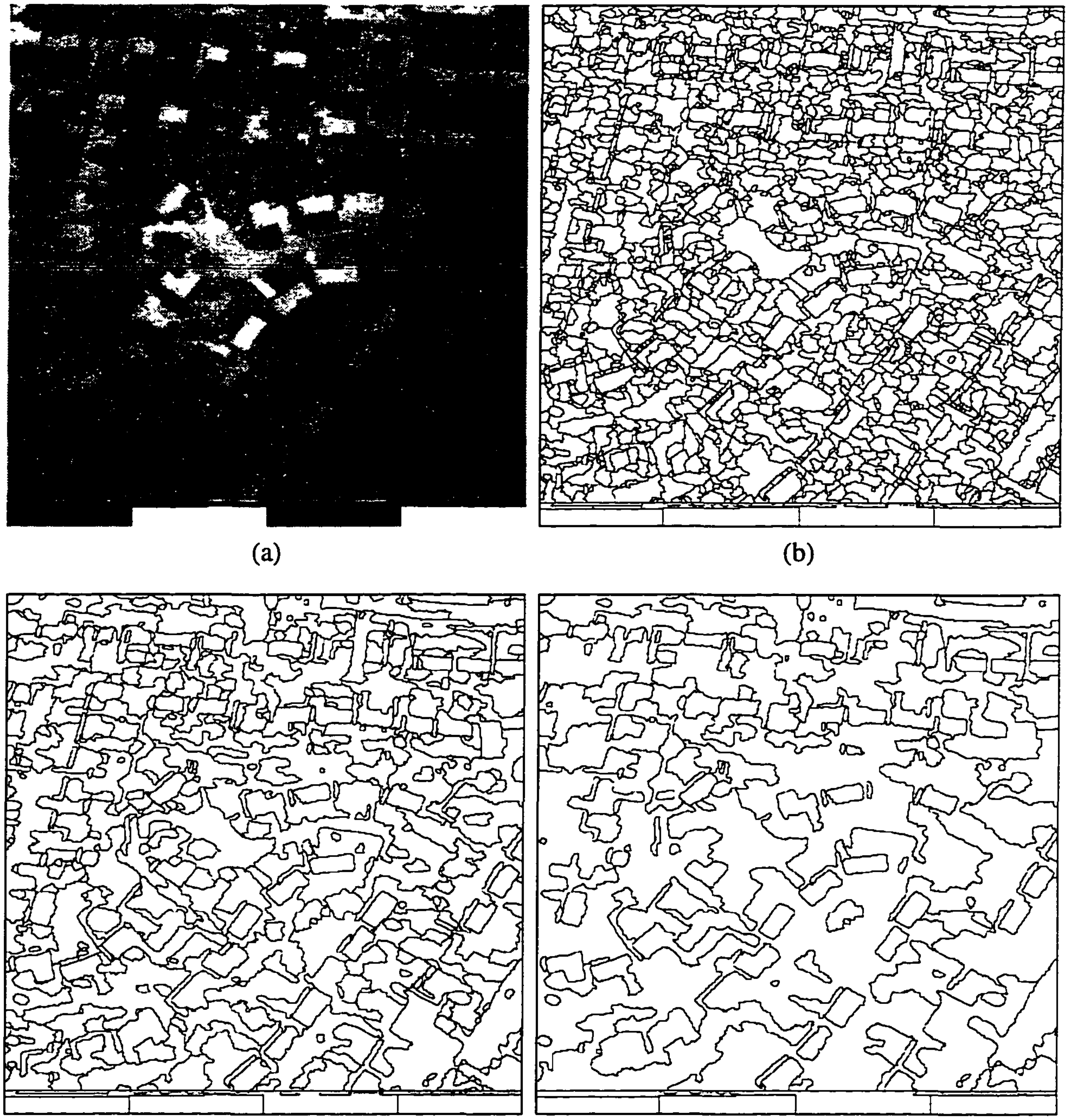
(e)

(f)

(g)

(h)

Figure 4.20 (cont.). (e) Lena. (f)-(h) Average grayscales of the regions present at, respectively, $\sigma_g = 7,\ 10,\ 21$.



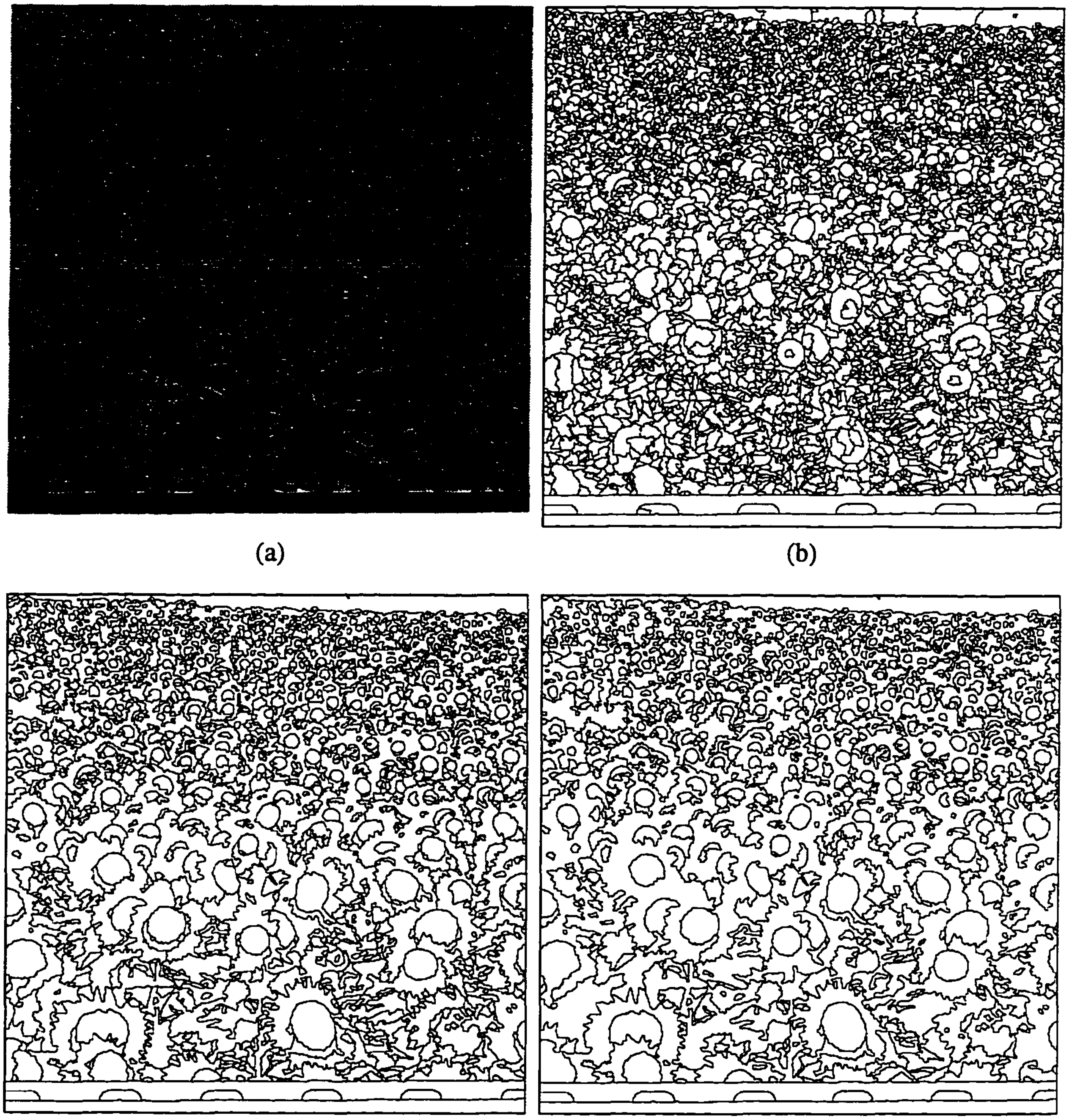
(a)

(b)

(c)

(d)

Figure 4.21. (a) Aerial scene. (b)-(d) Region boundaries present at, respectively, σ_g = 7, 13, 26.



(a)

(b)

(c)

(d)

Figure 4.22. (a) Field of sunflowers. (b-d) Region boundaries present at, respectively, $\sigma_g = 5, 13, 21$.

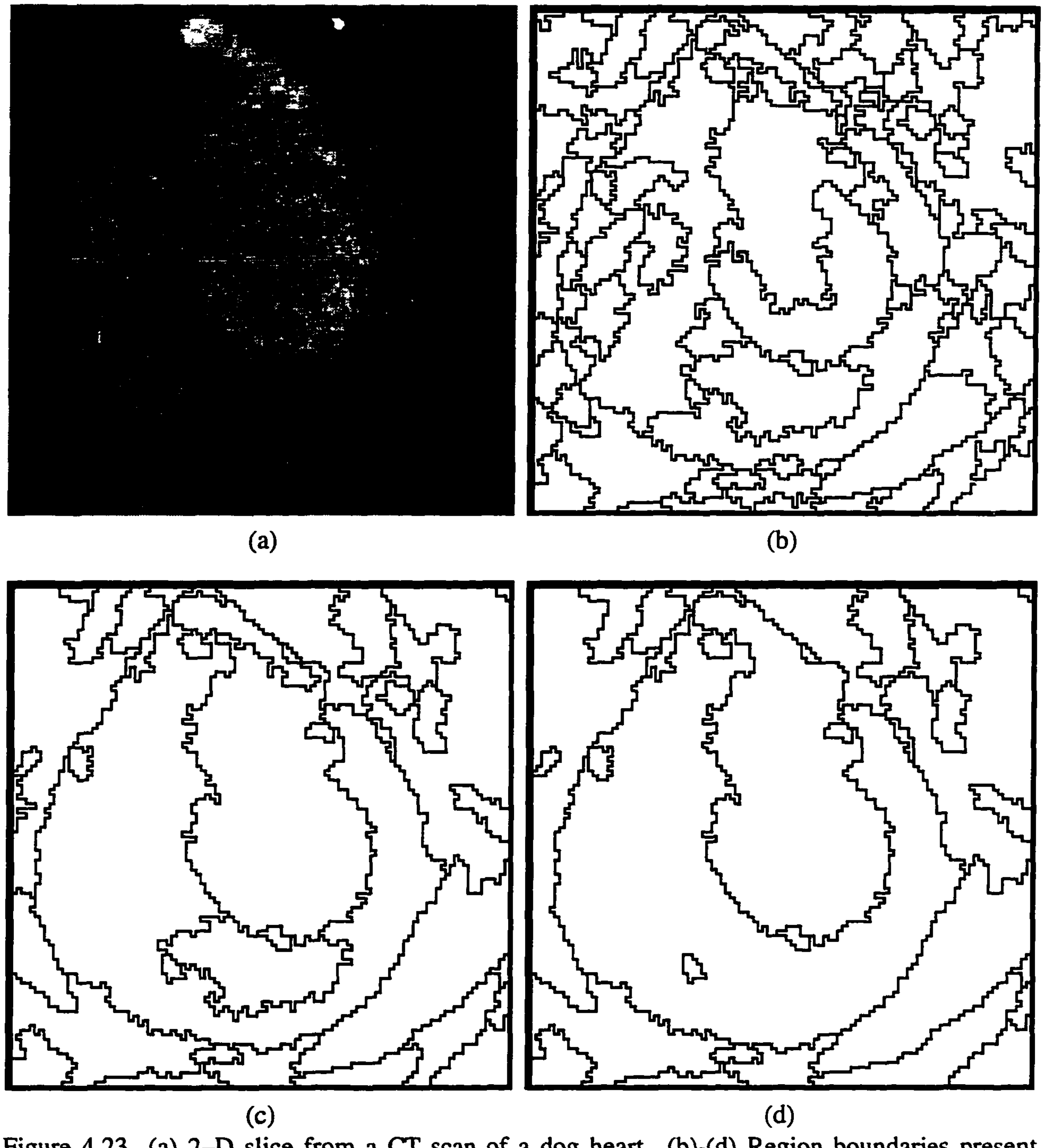
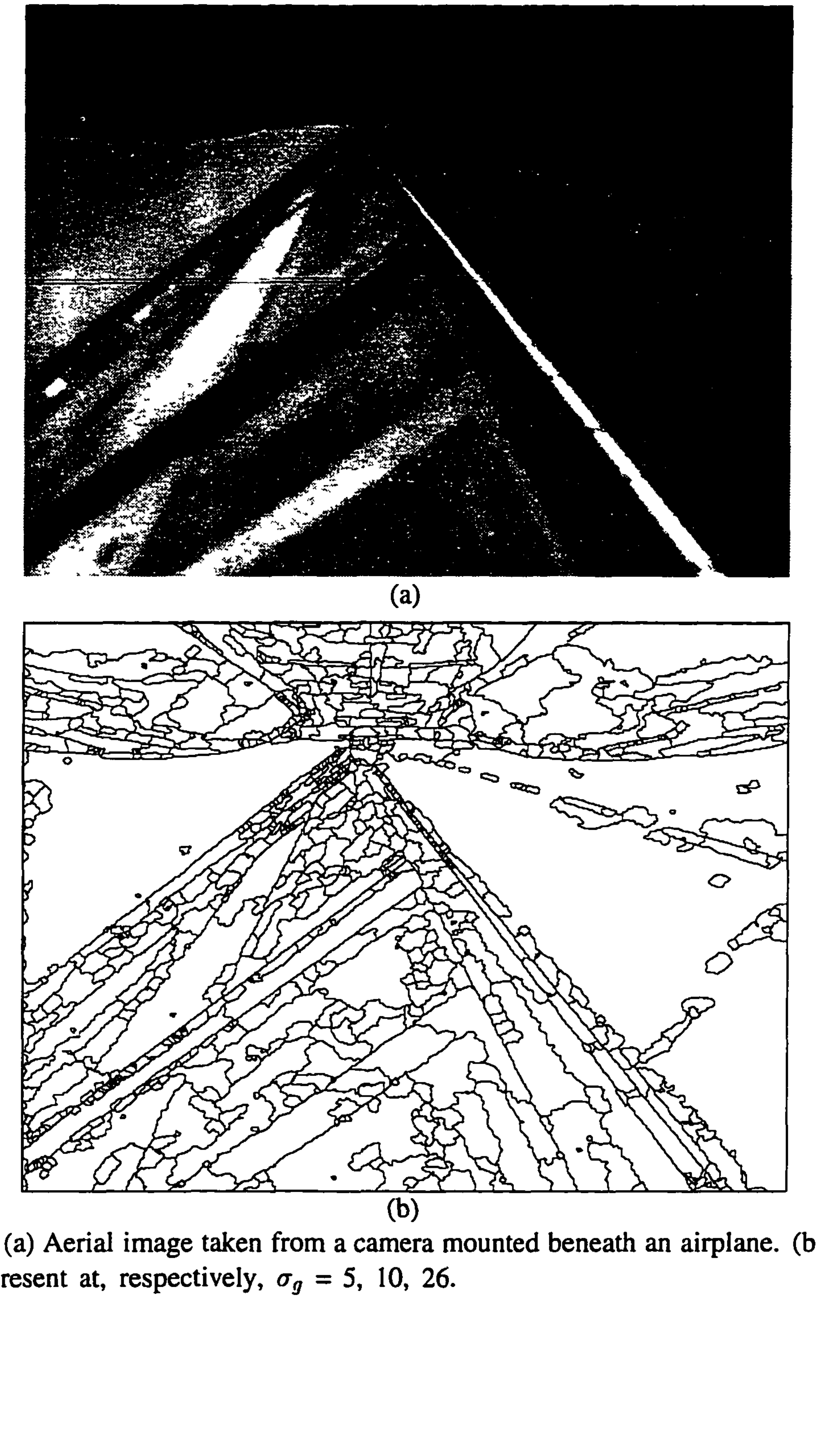


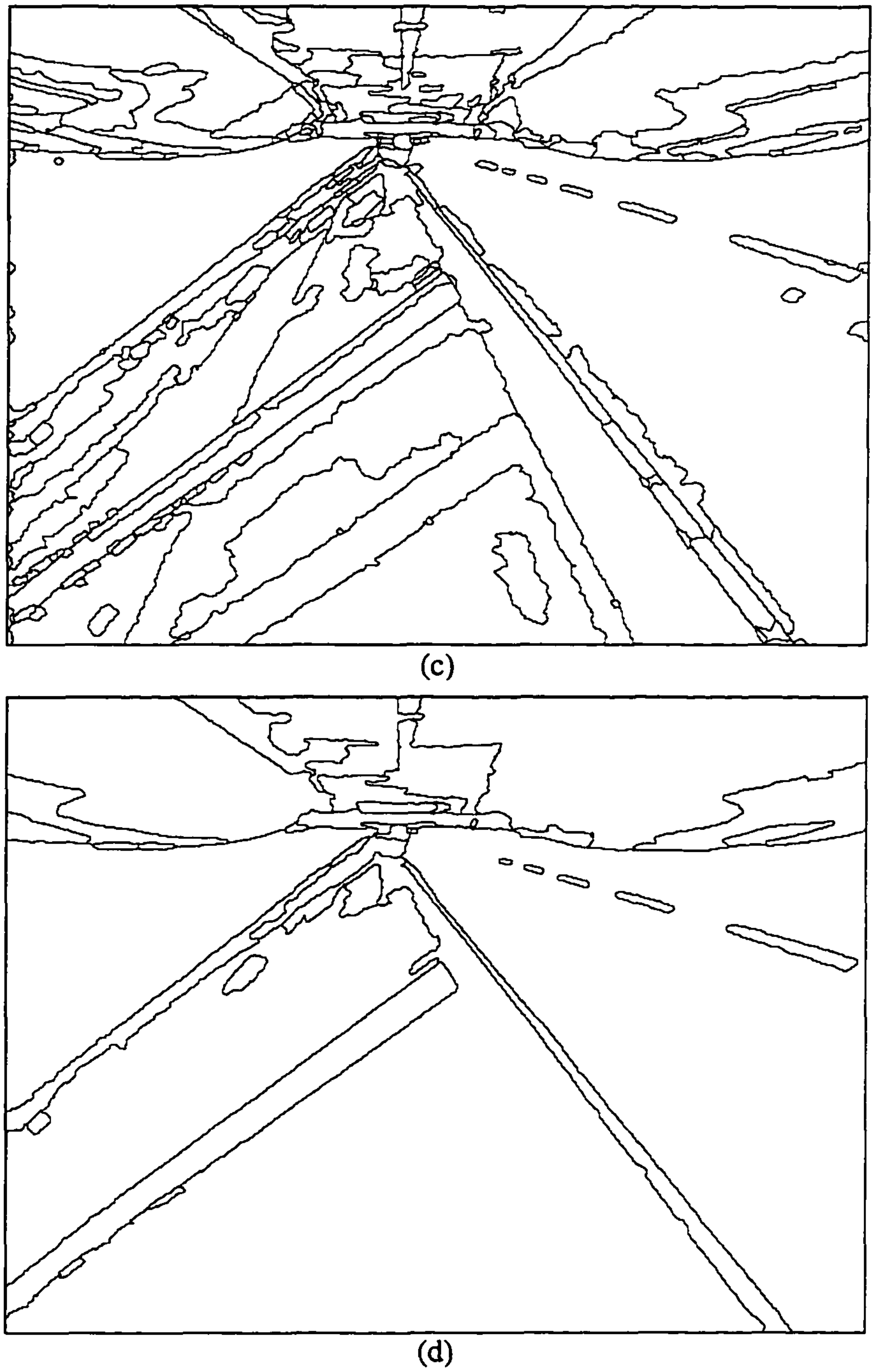
Figure 4.23. (a) 2–D slice from a CT scan of a dog heart. (b)-(d) Region boundaries present at, respectively, σ_g = 5, 13, 21.



(a)

(b)

Figure 4.24. (a) Aerial image taken from a camera mounted beneath an airplane. (b)-(d) Region boundaries present at, respectively, σ_g = 5, 10, 26.



(c)

(d)

Figure 4.24 (cont.).

The ability of the transform to encode information at multiple homogeneity scales is demonstrated by Fig. 4.10. The image in (a) is the same as that of Fig. 4.9(a), except that two new lower contrast regions have been added. The field vectors are displayed in (b)-(d) at correspondingly coarser homogeneity scale. In (b), all of the regions are present, whereas in (c) one of the new low contrast regions has disappeared, and in (d) the other new region has disappeared. Fig. 4.11 demonstrates the insensitivity of the transform to boundary curvature and topology. The image in (a) contains regions with smooth, piecewise linear, and rough boundaries, as well as a vertex where 8 regions meet and several sharp corners. Field vectors for $\mathbf{F}_{10}$ are shown in (b)-(c), from which one can see that all regions are represented within the force field without distortion.

Four different functions that could be used for $d_g(\cdot)$ and $d_s(\cdot)$ are displayed in Fig. 4.12. These functions include box-car, Gaussian, exponential, and linear distributions. In Fig. 4.13, the force field from Fig. 4.11(b) is shown computed using each of these four functions for both $d_g(\cdot)$ and $d_s(\cdot)$. The encoded structure remains unchanged for all four cases. The insensitivity of the transform to shading is demonstrated in Fig. 4.14. The image in (a) is identical to that of Fig. 4.11(a), except that both linear and quadratic shading are now present. $\mathbf{F}_{10}$ is displayed in Fig. 4.14(b), and is structurally equivalent to the $\mathbf{F}_{10}$ that is displayed in Fig. 4.11(b). In addition, $\mathbf{F}_{25}$ is displayed in Fig. 4.14(c). The microstructure within the irregular center region has now disappeared, and the remaining structure is unaffected by the shading.

Figure 4.15 demonstrates the insensitivity of the transform to Gaussian noise. Both (a) and (d) consist of a synthetic image containing two regions of intensity 0 and 100 on a background of intensity 200, and corrupted by additive, zero-mean, white Gaussian noise of standard deviation $\sigma = 50$ in (a) and $\sigma = 100$ in (d). Vector directions of $\mathbf{F}_{50}$ are shown in (b) and (e) intensity coded as in Fig. 4.9(c) and in (c) and (f) intensity coded as in Fig. 4.9(d). Note that the structure boundaries are all present and retain geometric fidelity (For example, the sharp corner of the intensity 100 region has been preserved without smoothing), except, of course, where the region structure has been changed because of the noise, in which case the new structure is reflected.

Figure 4.16 demonstrates structure of different homogeneity scales encoded within $\mathbf{F}$ for a real image. Intensity coded vector directions of $\mathbf{F}_{10}$ and $\mathbf{F}_{30}$ are shown for an image of a Donald Duck toy. The background substructure, such as the musical notes, is encoded within $\mathbf{F}_{10}$, whereas the entire background is present as a single region in $\mathbf{F}_{30}$.

Figures 4.17–4.24 demonstrate the segmentation performance on a variety of real images. For each image, the pyramid representing the multiscale segmentation is computed. The segmentation is visualized by displaying the structure boundaries, as well as the average grayscale of the structures for Figs. 19–20, for each structure present within the pyramid at a particular value of σ_g. For each image, three different values of σ_g were selected so that a wide variety of different structures can be seen. For Fig. 4.17, in addition to the segmentation results shown in (d)-(g), $\mathbf{F}_7$ and $\mathbf{F}_{26}$ vectors are shown in (b) and (c), respectively. Within $\mathbf{F}_7$, the substructure present within the lighter, cellular regions is encoded, and within $\mathbf{F}_{26}$, the cellular regions themselves are present. Although these regions exist at a coarse homogeneity scale, one can verify by examining the field that the region boundaries have been preserved. The format of Fig. 4.18 is similar to that of Fig. 4.17, except that (b) shows $\mathbf{F}_{30}$ with the vectors directions encoded as graylevels, and (c) shows the vectors within the white box in (a) displayed by a needle diagram. Fig. 4.18(c) allows one to inspect closely the structures encoded within $\mathbf{F}_{30}$.

Figures 4.17–4.24 were selected to demonstrate the performance of this algorithm in a variety of different situations, for example, noise (Fig. 4.23), shading (Fig. 4.20), significant multiscale structure (Figs. 4.17, 4.19, 4.20, 4.23, 4.24), sharp and diffuse edges in close proximity

(Figs. 4.21, 4.23, 4.24), and high–curvature structure boundaries (Figs. 4.17, 4.18, 4.22). In all cases, the identified structures closely correspond to human perception of relevant structure, and the structure boundaries closely align with the actual boundaries in the image. Finally, the amount of computation required by this algorithm is reasonable. About 90 sec. is currently required to segment a 512 x 512 image on a Sun SPARC20, and a VLSI chip has been designed to compute the transform at frame rates (30 Hz).

4.7. Discussion

The application of a new nonlinear transform to the problem of image segmentation has been explained. The identified regions correspond to perceptually valid image structure, and the identified region boundaries align closely with the actual boundaries of the structures, regardless of the scale of the structures. Automatic selection of both the homogeneity and spatial scales avoids the need to make restrictive *a priori* assumptions about either the geometric or homogeneity characteristics of the structure. A pyramid is constructed that contains all of the structure in a given image, as well as the range of homogeneity scale for which each structure is present. This approach to structure detection is distinguished from previous methods in several respects. First, scale is formulated in a manner that naturally represents image structure. Second, the processes of scale selection and structure detection are integrated and performed automatically. In addition, a unification between region and edge detection is achieved in the transformed domain. Finally, structure of arbitrary geometry can be detected without any smoothing of the structure boundaries, even at coarse scales.

5. MULTISCALE REGION-BASED 2–D MOTION ESTIMATION AND SEGMENTATION

This chapter applies the image segmentation algorithm described in the previous chapter to the problems of estimating the 2–D motion field from a time sequence of images and identifying regions characterized by similar motion (motion segmentation). The goal of the algorithm is to determine the 2–D motion field, the moving objects, the areas of occlusion and disocclusion, the background region, and the relative depth of the moving objects. No restrictive assumptions are made with regard to the type of motion (i.e., slow, temporally smooth, rigid) or the number of moving objects.

The algorithm operates on the images in a video sequence a pair at a time. A multiscale image segmentation is computed independently on each image in the frame pair. Correspondences are then computed between the identified structures in each of the two frames. The motion of the pixels within each set of matched structures is modelled by an affine transformation. Because structure detection is multiscale, a pixel that belongs to multiple structures may have several different affine transformations associated with it. An integration step then is used to combine the multiple motions together, resulting in a motion field for the frame pair. Motion segmentation, occlusion identification, and relative depth are then computed from the motion field.

The rest of this chapter is organized as follows: Section 5.1 reviews previous work on 2–D motion estimation, and Section 5.2 motivates the approach of our algorithm. Section 5.3 then gives an overview of the major steps in the algorithm. The next several sections describe each of these steps in more detail. Section 5.4 describes the current method for matching the segmented structures, Section 5.5 describes the process by which the affine transformations are computed for each set of matched structures, and Section 5.6 describes the process by which the multiple motion estimates are integrated into a single motion field. This field is then segmented by the method given in Section 5.7. Next, Section 5.8 describes the occlusion and relative depth estimation steps, and Section 5.9 details the last two steps of the algorithm, namely, motion and occlusion prediction. Finally, Section 5.10 gives implementation details of the algorithm, Section 5.11 gives experimental results, and a final discussion of the method is given in Section 5.12.

5.1. Previous Work

Previous approaches to the problem of 2–D motion estimation can be classified as either pixel-based (intensity-based) or feature-based. Some reviews can be found in [58–60]. The pixel-based approaches (commonly referred to as *optical flow* methods) assume a direct relationship between object motion and intensity changes within a video sequence, i.e., they assume that intensity changes are caused by motion and that motion causes variations in intensity. As a result, motion estimation is formulated as an optimization problem where the motion field corresponds to the operator that best accounts for the intensity variations given various restrictions. Such methods include algorithms that utilize constraints based upon local spatial and temporal derivatives [12, 13], as well as the popular block-based correlation algorithm (BCA). These algorithms yield dense motion estimates.

Feature-based methods extract features from images and then match them across frames, thereby obtaining a displacement field. Such features include points defined by local intensity extrema [14], edges [15–17], corners [17, 18], and regions [19–24]. Most of these algorithms result in sparse motion fields.

5.2. Motivation for the Approach

Optical flow techniques often suffer difficulties near occlusion boundaries, where the motion is coarse, and in areas where the *aperture problem* dominates (This refers to the difficulty in computing the component of the motion that is tangential to an isobrightness contour using a local operator). The more sophisticated algorithms in this class, however, can handle these problems with some degree of success [61, 62]. The more fundamental problem with optical flow algorithms is the assumption of an equivalence between intensity changes and motion [63]. Intensity changes between image frames may be caused by changes in illumination, noise, and spatial sampling, in addition to the changes caused by motion. Also, it is not always correct that motion causes changes in intensity, a primary example being areas with little variation in intensity. The use of structures defined by homogeneity of intensity can help in both of these situations, however. The motion information in areas with little intensity variation is contained in the contours of the structures associated with such areas. In addition, structure geometry is fairly robust to noise and changes in illumination, so differences in the shape and position of the structure contours across time are generally caused by motion. Further, occluding contours are almost always defined by an intensity discontinuity at some scale, so the problems of identifying the occluding contours and obtaining good motion estimates near such contours are simplified with the use of structural information.

Non-region features (intensity extrema, corners, edges) provide some of the same benefits as image structure, and methods such as that of [17] integrate such information into the motion estimation process. These features have several disadvantages compared to region primitives, however. Regions are more stable with regard to noise and lighting changes, and the probability of an incorrect match is much lower because regions have a larger variety and complexity of attributes than the other types of features. Also, non-region features provide information at only a sparse set of points. Because regions are 2–D primitives, they are much more comprehensive in that every pixel in the image belongs to at least one region.

A few other region-based motion algorithms exist besides the algorithm described in this chapter; however, they are all fairly simplistic. First, these methods all use one of the standard, single-scale segmentation algorithms. Segmentation errors, such as unintuitive regions and inexact region boundaries, resulting from the use of such algorithms increase the difficulty of finding region correspondences across frames. In addition, the region hierarchy provided by a multiscale algorithm provides a much richer description of regions available for matching. Both structural changes and noise within a certain area of an image may cause an absence of matches for regions within that area at a particular scale. However, it is often the case that matches can be found within that area at other scales. As a result, a multiscale method is able to find region correspondences over a larger fraction of the image than a method that extracts regions at only a single scale. Second, these other methods use quite simple approaches to obtain region correspondences. In [24], matching is done by 3–D segmentation (x,y,t) under the assumption that regions overlap from frame to frame, and [20, 22, 64] all perform matching using heuristics on region attributes such as average intensity, area, and moments. In addition, these three methods compute a sparse motion field by assigning to the matched region centroids motion vectors corresponding to the displacement of the centroids of the matched regions. The motion of the pixels in a matched region pair are also modelled by an affine transformation in [21] and [24]; however, neither of these two methods attempt to account for the change in region shape resulting from occlusion.

All types of motion estimation algorithms make errors because of occlusion, but region-based methods are particularly sensitive to this problem. In pixel-based methods, occlusion can cause motion vectors to be estimated incorrectly in the neighborhood of the occlusion. For feature-based methods, occlusion may cause matches not to exist for features near the occlusion, and, as a result, the motion cannot be estimated in these areas. However, if the features are regions, it is usually the case that the occlusion only partially obscures a region. Hence, a region unoccluded in one frame and partially occluded in the following frame will still be matched, but the partial occlusion can change the shape of the region enough to cause the estimated motion parameters to be incorrect. If this region is large, motion vectors may be estimated incorrectly far away from the occlusion. Thus, occlusion can cause global errors in a region-based method. The method presented in this chapter reduces this problem by estimating and compensating for the effects of occlusion.

5.3. General Overview of the Algorithm

A general overview of the approach is given in Fig. 5.1. Let an image at frame t of a video sequence be given by $I_t(x, y)$. The algorithm processes a video sequence two frames at a time. For a pair of frames, (I_t, I_{t+1}), the algorithm computes a 2–D motion field, $\vec{M}_{t,t+1}(x, y)$, which describes the motion of each pixel in the image from frame t to frame $t + 1$. The motion field is then segmented into connected areas characterized by similarity of motion over the pair of frames. This motion segmentation is denoted as $MS_{t,t+1}(x, y)$, where the pixels in each region take on a distinct value that corresponds to that region. Because an image is a 2–D projection of a 3–D world, objects at different depths from the camera occlude one another. As objects move, certain areas of the image disappear (become occluded) while other areas appear (become disoccluded). Thus, the algorithm also computes an occlusion image, $O_{t,t+1}(x, y)$, which contains the areas that

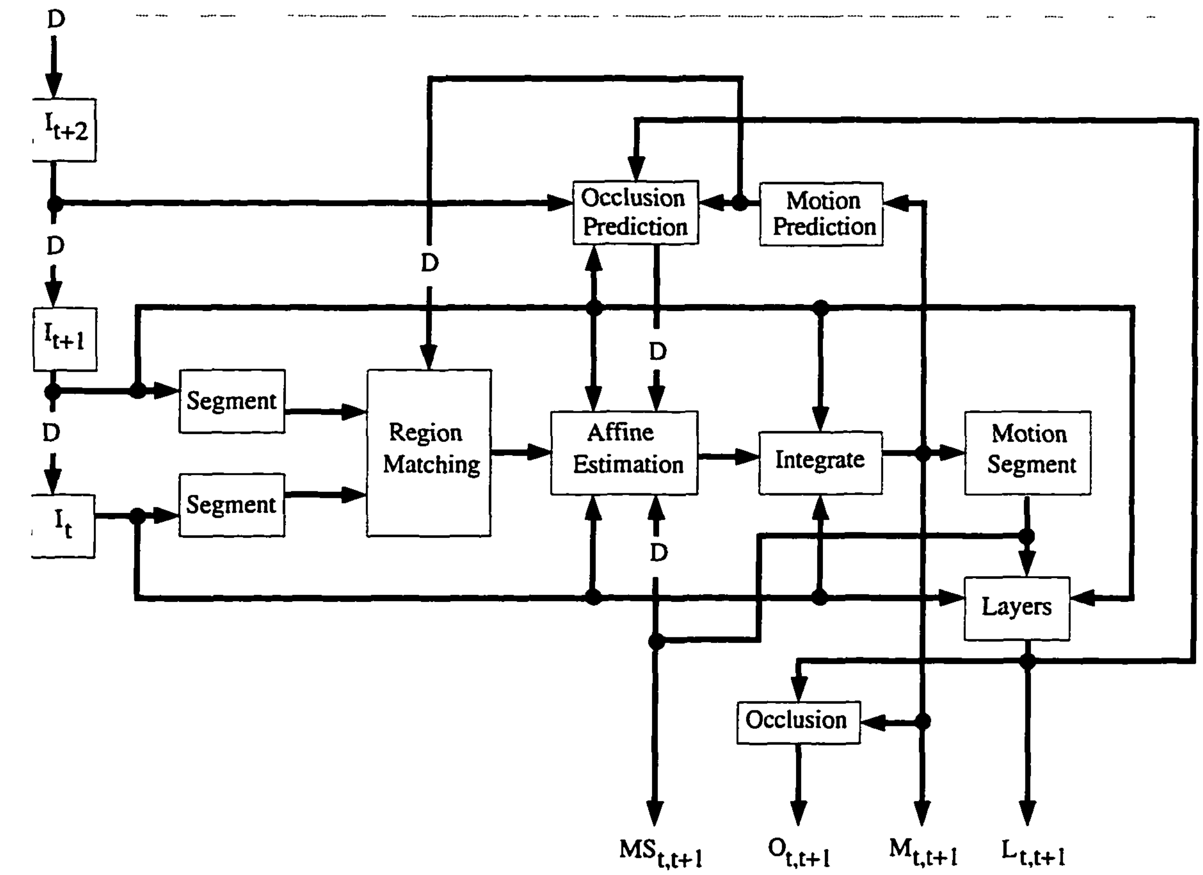


Figure 5.1. Block diagram of the motion estimation and segmentation algorithm. The letter D indicates a one timestep delay element.

become occluded and disoccluded between the pair of frames. Finally, the algorithm computes a layer image, $L_{t.t+1}(x, y)$, which reflects the algorithm's current understanding of the relative depth among the objects in the scene. A value is assigned to each motion region in $MS_{t,t+1}(x, y)$ that is inversely related to the distance of the region from the camera. Thus, the background region is given the smallest value, and the nearest object to the camera is given the largest value.

The region matching process is formulated as a graph matching problem. Three preselected values of homogeneity scale are used as indexes into the segmentation pyramid of each image to produce three different image partitions. Each pair of partitions at the same scale are matched from coarse to fine, with coarser scale matches guiding the finer scale matching. Each partition is represented as a region adjacency graph (RAG), within which each region is represented as a node and region adjacencies are represented as edges. Region matching at each scale then consists of finding the set of graph transformation operations (edge deletion, edge and node matching, and node merging) of least cost that create an isomorphism between the current graph pair. The cost of matching a pair of regions takes into account their similarity with regard to area, average intensity, expected position as estimated from each region's motion in previous frames, and the spatial relationship of each region with its neighboring regions. Once the image partitions at the three different homogeneity scales have been matched, matchings are then obtained for the regions in the first frame of the frame pair that were identified by the motion segmentation module at the previous timestep. The match in the second frame for each of these motion regions is given as the union of the matches of the set of finest scale regions that comprise the motion region. This gives a fourth matched pair of image partitions, and is considered to be the coarsest scale set of matches.

The affine estimation step computes an affine transformation for each set of matched regions, at all four scales. The change in shape of the regions resulting from occlusion is estimated by predicting the motion for the current frame pair on the basis of the motion from the previous frame pair, and then utilizing the computed layer information from the previous frame pair to determine areas of occlusion and disocclusion. The predicted occlusion information is used to compensate for the effects of occlusion on the affine transformation parameter estimation.

The computed affine parameters give a motion field at each of the four scales. These motion fields are then integrated into a single motion field by taking the coarsest motion field and then comparing, at each matched region at the next finer scale, the image prediction error generated by the current motion field and the motion field at the next finer scale. At any region where the prediction error using the finer scale motion improves by a specified percentage, the current motion is replaced by the finer scale motion. This process is repeated recursively downward in scale. This biases the final motion field toward the coarser scale motion fields. Because the algorithm only processes a video sequence two frames at a time, situations arise in which the prediction error generated by the true motion is somewhat larger than the error generated by some other motion. Since motion is typically temporally consistent, especially over adjacent pairs of frames, it is desirable to slightly bias the final motion field in favor of temporally consistent trajectories. For the other three scales, it is also desirable to favor the motion of the coarser scale regions. Coarser regions are larger than finer scale regions, which reduces the probability of these regions being incorrectly matched, and they also contain more intensity variation and have more complex boundaries, both of which make the estimated affine transformation more likely to represent the true motion of such regions. The coarser scale region bias is only slight, however, because the affine motion model is less likely to be valid for such regions, and there is an increased likelihood that these regions cross an occlusion boundary.

The resulting motion field is then segmented into regions of similar motion using a similarity threshold. By examining the image prediction error in areas where two motion regions overlap, occlusion and layer maps are generated. Next, the motion for the next frame pair is predicted by assuming that the affine transformation parameters describing the motion in the next frame pair will be identical to those computed for the present frame pair. This motion prediction plus the current layer information allow the occlusion maps to be estimated for the next frame pair. The motion prediction is used at the next frame pair to guide the region matching process, and the occlusion prediction is used to compensate for the effects of occlusion on the affine parameter estimation process.

Each of the modules in the algorithm is now discussed in more detail.

5.4. Region Matching

Each pair of partitions at the same scale are matched from coarse to fine, with coarser scale matches guiding the finer scale matching. Each partition is represented as a region adjacency graph (RAG), within which each region is represented as a node and region adjacencies are represented as edges. Figure 5.2 gives an example of an image partition and its associated RAG. The image frame is represented by the *0* node. Region matching at each scale then consists of finding the set of graph transformation operations (edge deletion, edge and node matching, and node merging) of least cost that create an isomorphism between the current graph pair. Our region matching algorithm is based on that of [25], but has some advantages with regard to reduced computational complexity, coarse to fine matching, and ability to match properly regions lying entirely inside another region.

Some notation and planar graph concepts are now introduced. Let the two RAGs to be matched be denoted by G and G'. Because each graph represents a 2–D image, the graphs are planar. Denote the ith node in G by N_i, the jth node in G' by N_j', and a matched pair of nodes by $\left(N_i, N_j'\right)$. Define the terms articulation node, biconnected graph, and biconnected component as follows:

> *Articulation node.* If there is a triple of distinct nodes N_1, N_2, N_3 in graph G such that N_3 lies on every path that connects N_1 and N_2, then N_3 is an articulation node of G.

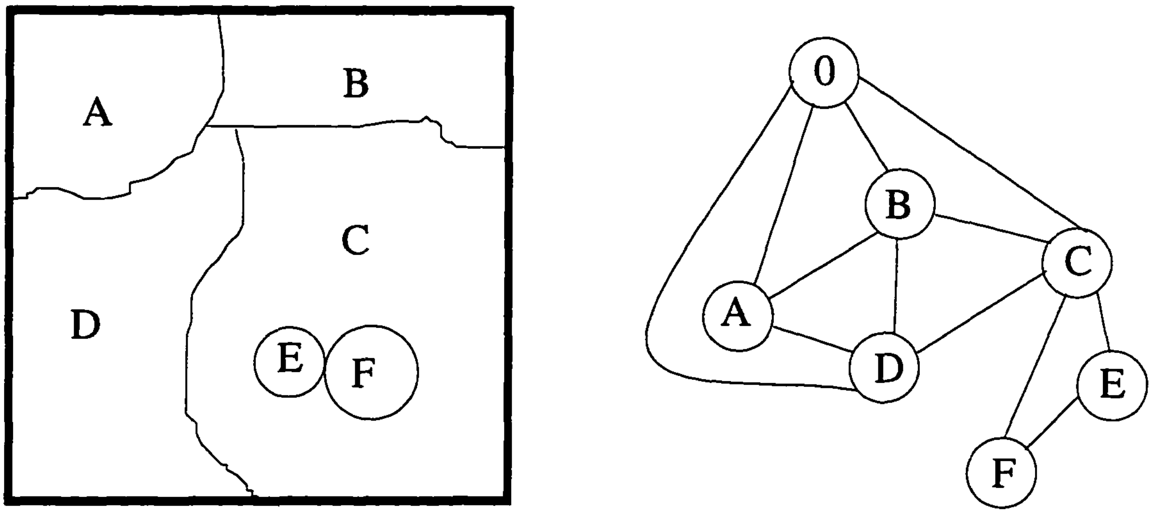


Figure 5.2. An image partition containing six regions, and its associated region adjacency graph (RAG). The image frame is denoted by a *0*.

Biconnected graph. A graph G is biconnected if for each triple of distinct nodes N_1, N_2, N_3 in G, there exists a path between N_1 and N_2 such that N_3 is not on this path.

Biconnected component. A maximal subgraph that is biconnected.

Articulation nodes correspond to regions, such as C in Fig. 5.2, which fully enclose other regions. Each RAG is partitioned into subgraphs consisting of each biconnected component in the graph. The subgraph that contains the image frame node is considered to be the main subgraph. The edges emanating from each node are represented in the order they occur as a region boundary is traced clockwise (counterclockwise if the node is an articulation node and the contour is one of the node's inner contours). Figure 5.3 shows the appropriate partitioning of the RAG in Fig. 5.2. The matching algorithm proceeds by matching the two main subgraphs, and, whenever an articulation node is matched, incorporating the subgraphs corresponding to the articulation node into the main subgraph of the node. Edges are represented cyclically because

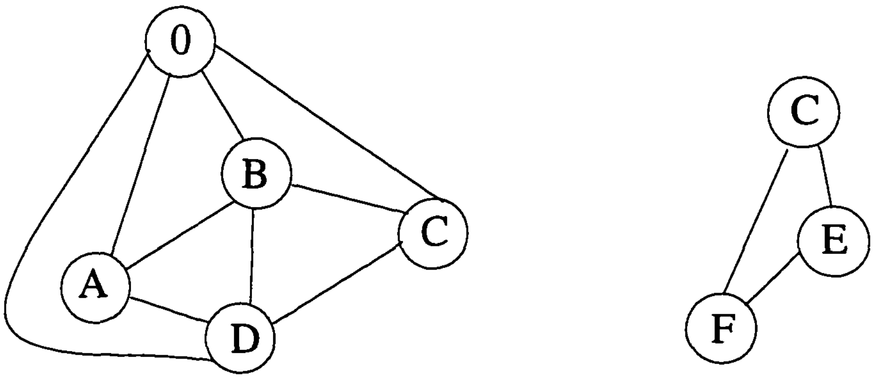


Figure 5.3. The RAG in Fig. 5.2 partitioned into subgraphs consisting of biconnected components.

of the *order reserving* property of planar graphs:

> If a node in a biconnected component of one graph is matched to a node in a biconnected component of another graph, then the edges between the corresponding nodes must be matched in a way that is cyclically order reserving.

Denote the jth edge at node N_i by N_{i_j}, with the edges subscripted by traversing a node clockwise. Let $\left(N_{i_j}\right)$ represent the same edge at the corresponding adjacent node, which itself is denoted as $\left[N_{i_j}\right]$. Further, let $\left(N_{i_j}\right)_{\pm n}$ be the nth edge from $\left(N_{i_j}\right)$ measured clockwise (+) or counterclockwise (-) around $\left[N_{i_j}\right]$. Thus, in Fig. 5.2, if N_{i_j} is the edge at node A and adjacent to node B, then $\left[\left(N_{i_{j+1}}\right)_{-2}\right] = C$ and $\left(\left(N_{i_{j+1}}\right)_{-2}\right)$ is the edge at C and adjacent to D. Also, by simple region adjacency, note that the following relation must hold

$$\forall i,j,\quad \left(\left(\left(N_{i_j}\right)_{+1}\right)_{+1}\right) = N_{i_{j-1}} \tag{5.1}$$

The isomorphism between G and G', is constructed from four basic graph operations:

1. Node matching. A match $\left(N_i, N'_j\right)$ is created.

2. Edge matching. When two node pairs $\left(N_i, N'_j\right)$ and (N_k, N'_l) are matched, the edge between N_i and N_k is matched to the edge between N'_j and N'_l, if both edges exist.

3. Edge deletion. Deletion of an edge N_{i_j}, causes the insertion of another edge in order to maintain the relationship of Eq. (5.1).

4. Node merging. A node N_i adjacent to node N_k may be merged into N_k, resulting in the deletion of their common edges, and a relabelling of the edges N_{i_j} as being attached to node N_k.

Edge deletion and node merging operations may occur on either graph. Each operation has an associated cost. The goal of the region matching algorithm is to find the set of operations of minimal cost that create an isomorphism between graphs *G* and *G'*. The algorithm is simple and has linear time complexity, but is not guaranteed to find the optimal matching.

5.4.1. Region matching algorithm overview

The region matching is performed by iteratively building isomorphic basic RAGs (BRAGs) of minimal cost at each pair of matched nodes. A BRAG consists of a matched node N_i, its associated edges N_{i_j}, and its neighboring nodes $\left[N_{i_j}\right]$. The algorithm utilizes two FIFO queues, Q and $Q2$, which contain the current set of matched nodes for which isomorphic BRAGs have not been computed. Q is initialized to the frame pair $(0, 0')$, and $Q2$ is initially empty. The algorithm proceeds as follows:

```
While Q not empty {
        get matched pair
        if conflict and pair not previously on Q2 then put pair on Q2
        else if articulation node then incorporate associated subgraphs
        else match BRAGs and put new matches on Q
        if Q empty and Q2 not empty then move matched pair from Q2 onto Q
        }
```

Each pair of BRAGs is matched independently of any previously obtained matches. New pairs of matched nodes created by this matching are placed onto Q. If a pair of nodes are matched that causes a conflict with a previously obtained matching, for example a current match $\left(N_i, N'_k\right)$ and a previous match $\left(N_i, N'_j\right)$, then the previous match is marked to indicate that a conflict has occurred. If the previous match has not yet been processed, then all unconflicted matches are first processed before the previous match. The main reason for this is that most conflicts occur as a result of the two conflicting regions not yet having merged. In the example, the correct match may be $\left(N_i, N'_j \cup N'_k\right)$, but the conflict occurs because N'_j and N'_k have not yet merged. Matching the BRAGs at $\left(N_i, N'_j\right)$ rather than $\left(N_i, N'_j \cup N'_k\right)$ generally results in edges being incorrectly deleted, which may cause some nodes not to be matched because all of their edges were deleted. Thus, deferring the processing of $\left(N_i, N'_j\right)$ as long as possible increases the likelihood of N'_j and N'_k first merging. In addition, if the conflict does indicate that the previous match is incorrect, then deferring the processing of this match minimizes the influence of the mismatch on the entire graph matching process.

Despite this deferred merging, some conflicted matched pairs may be processed before the conflicted regions have merged. As a result, the algorithm is run a second time, but with Q initially containing all unconflicted pairs of matched nodes from the previous pass for which matching of their associated BRAGs resulted in regions being merged, as well as the frame pair $(0,0')$. As a result, almost all conflicts occurring during the second pass are due to incorrect matches. Conflicts are resolved by selecting the matched pair (with the union case included) whose associated BRAG matching cost is the least. For the previous example, this amounts to selecting the minimum BRAG matching cost among the three matched pairs: $\left(N_i, N'_k\right)$, $\left(N_i, N'_j\right)$, and $\left(N_i, N'_j \cup N'_k\right)$.

The procedures for matching BRAGs and incorporating subgraphs at articulation nodes into the main subgraph are now described.

5.4.2. Matching of BRAGs

Assume the BRAGs at the matched pair $\left(N_i, N'_j\right)$ are to be matched, and let N_i have edges $\{N_{i_1}, N_{i_2}, ..., N_{i_k}\}$ and neighbors $\{[N_{i_1}], [N_{i_2}], ..., [N_{i_k}]\}$ and N'_j have edges $\left\{N'_{j_1}, N'_{j_2}, ..., N'_{j_l}\right\}$ and neighbors $\left\{\left[N'_{j_1}\right], \left[N'_{j_2}\right], ..., \left[N'_{j_l}\right]\right\}$. Because edges must be matched in an order-reserving fashion, the problem of determining the optimal isomorphism of two BRAGs is essentially a circular string matching problem. The minimal cost, $C(k,l)$, of this isomorphism is computed recursively as follows:

$$\begin{aligned} C(k,l) = \min\{ & C(k-1,l-1) + C_{mt}\left([N_{i_k}], [N'_{j_l}]\right), \\ & C(k,l-1) + C_{de}\left(N'_{j_l}\right), \\ & C(k-1,l) + C_{de}(N_{i_k}), \\ & C(k-1,l-1) + C_{dm}\left(N'_{j_l}\right), \\ & C(k-1,l-1) + C_{dm}(N_{i_k}), \\ & C(k,l-1) + C_{mg}\left(N'_{j_l}\right), \\ & C(k-1,l) + C_{mg}(N_{i_k}) \ \} \end{aligned} \tag{5.2}$$

subject to the constraint

$$C(x,y) = \begin{cases} 0 & ,x = y = 0 \\ \infty, & x < 0 \text{ or } y < 0 \end{cases} \tag{5.3}$$

The cost of matching nodes $[N_{i_k}]$ and $\left[N'_{j_l}\right]$ is defined as

$$C_{mt}([N_{i_k}],[N'_{j_l}]) = \begin{cases} \Delta I/\sigma_g, & \text{if } \Delta I \leq \sigma_g \text{ and } \lambda_{CS}{=}\lambda_D{=}\ 1 \\ \infty & ,\text{else} \end{cases} \tag{5.4}$$

where $\Delta I\Big([N_{i_k}],\left[N'_{j_l}\right]\Big)$ is the difference in average intensity of the two regions, σ_g is the homogeneity scale of the current partition, and $\lambda_{CS}\Big([N_{i_k}],\left[N'_{j_l}\right]\Big)$ and $\lambda_D\Big([N_{i_k}],\left[N'_{j_l}\right]\Big)$ are indicator functions that reflect whether the regions are spatially proximate enough to be treated as possible match candidates.

Matches previously obtained at the next coarser homogeneity scale partition (if it exists) are used to constrain the current scale matching process via λ_{CS}. Let a pair of nodes N_x and N'_y correspond to the nodes $\hat{N}_x$ and $\hat{N}'_y$, respectively, at the coarser scale. In lieu of the requirement that $\left(\hat{N}_x,\hat{N}'_y\right)$ be a matched pair in order for $\lambda_{CS}(N_x,N'_y) = 1$, the less strict criterion that either $\hat{N}_x$ or one of its neighbors be matched to $\hat{N}'_y$ is adopted.

The indicator function $\lambda_D(N_x,N'_y)$ determines whether or not the spatial positions of the regions corresponding to the nodes N_x and N'_y are close enough for these nodes to be possible matching candidates, and is defined as

$$\begin{aligned} \lambda_D(N_x,N'_y) = & \Big\{A\Big(B_e\big(\hat{N}_x\big) \cap B(N'_y)\Big) \geq a \cdot A(B(N'_y))\Big\} \cup \\ & \Big\{A\Big(B\big(\hat{N}_x\big) \cap B_e(N'_y)\Big) \geq a \cdot A\Big(B\big(\hat{N}_x\big)\Big)\Big\} \end{aligned} \tag{5.5}$$

$\hat{N}_x$ represents the shape and position of region N_x in the following frame as determined by the affine transformation predicted for this region by the motion prediction module. If no prediction exists, then $\hat{N_x} = N_x$. The bounding box of a region is given by $B(\cdot)$, whereas $B_e(\cdot)$ represents this bounding box extended outward by e_1 pixels. If no prediction exists for N_x, then $B_e(\cdot)$ extends all bounding boxes by e_2 pixels, where $e_2 \geq e_1$. Finally, the area of a box is given by $A(\cdot)$, and a is a constant in the range $0 \leq a \leq 1$.

$C_{de}(N_{i_k})$ is the cost of deleting edge N_{i_k}, and is defined as

$$C_{de}(N_{i_k}) = \frac{1}{1 + \exp\left(-T\left(l - p\bar{l}\right)\right)} \tag{5.6}$$

where l is the length of the region boundary corresponding to N_{i_k}, $\bar{l}$ is the average length the edges N_{i_m}, p a constant in the range $0 \leq p \leq 1$, and T a factor controlling the gradient of the function.

$C_{dm}(N_{i_k})$ considers the case where $[N_{i_k}]$ matches none of the $\left[N'_{j_m}\right]$, but $\left[N'_{j_l}\right]$ matches either of $\left[(N_{i_k})_{\pm 2}\right]$. Define

$$C_{dm}(N_{i_k}) = C_{de}(N_{i_k}) + \min\left\{C_{mt}\left(\left[(N_{i_k})_{+2}\right], \left[N'_{j_l}\right]\right), C_{mt}\left(\left[(N_{i_k})_{-2}\right], \left[N'_{j_l}\right]\right)\right\} \tag{5.7}$$

Finally, $C_{mg}(N_{i_k})$ is the cost of merging $[N_{i_k}]$ into $\left[N_{i_q}\right]$, where $q < k$ and $\left(\left[N_{i_q}\right], \left[N'_{j_r}\right]\right)$ is the last node pair matched at the present $C(k,l)$, and is defined as

$$C_{mg}(N_{i_k}) = \begin{cases} 0, & [N_{i_k}] = \left[N_{i_q}\right] \\ C_{mt}\left([N_{i_k}], \left[N'_{j_r}\right]\right) + C_a\left([N_{i_k}], \left[N'_{j_r}\right]\right), & \text{else} \end{cases} \tag{5.8}$$

where $C_a(\cdot)$ is the cost calculated for the case when the area of the merged nodes is greater than that of their corresponding match, and is defined as

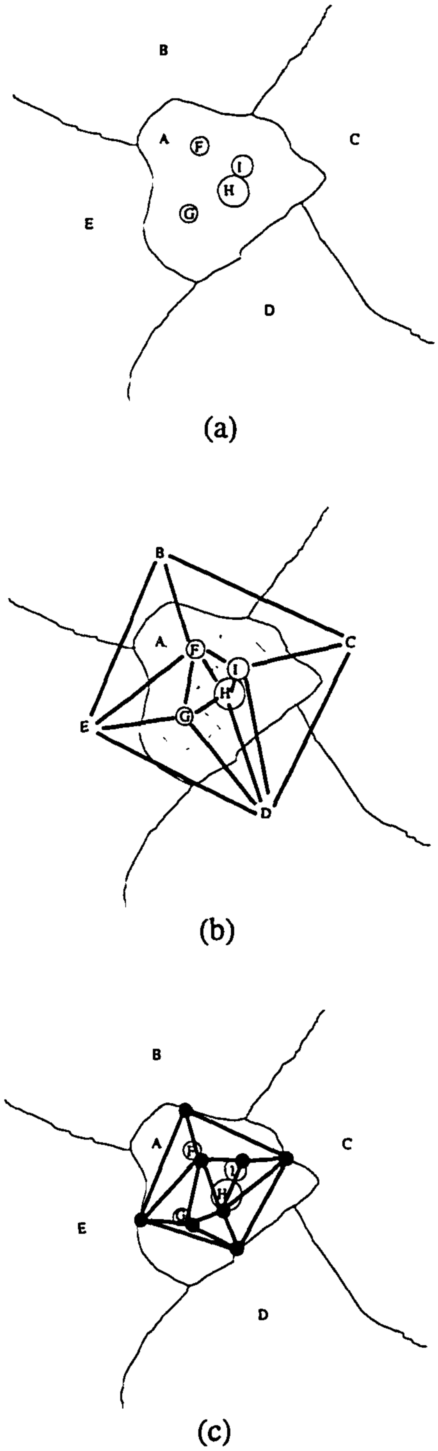
$$C_a\left([N_{i_k}], [N'_{j_r}]\right) = \begin{cases} 0, & A([N_{i_k}]) + A\left(\left[N_{i_q}\right]\right) \leq A\left(\left[N'_{j_r}\right]\right) \\ \left(\frac{A([N_{i_k}]) + A\left(\left[N_{i_q}\right]\right) - A\left(\left[N'_{j_r}\right]\right)}{A\left(\left[N'_{j_r}\right]\right)}\right)^2, & \text{else} \end{cases} \tag{5.9}$$

Because $\{N_{i_1}, N_{i_2}, ..., N_{i_k}\}$ and $\left\{N'_{j_1}, N'_{j_2}, ..., N'_{j_l}\right\}$ are circular sequences, $C(k,l)$ is computed k times with the sequence $\{N_{i_1}, N_{i_2}, ..., N_{i_k}\}$ shifted by one each time. The minimal of these $C(k,l)$ corresponds to the final minimal cost, and the associated graph transforming operations are found by tracing back the path of minimal cost from this $C(k,l)$ to $C(0,0)$.

5.4.3. Subgraph incorporation at articulation nodes

If either of the nodes in a matched pair is an articulation node, each of the nodes is deleted from its main subgraph, and any subgraphs associated with the node are then incorporated into the main subgraph. This process is described pictorally in Fig. 5.4. In (a), Region *A* is an articulation node with three associated subgraphs. The most reasonable procedure for deleting *A* is probably to assign each pixel in *A* to the region to which it is closest. Such an assignment, along with the resulting RAG, is depicted in (b). Note that all subgraphs have been incorporated into the main subgraph, and that the main subgraph remains biconnected. This procedure was judged to be too computationally intensive, however, and a procedure that yields an approximation of this result is used instead. This approximation first identifies the midpoint of every boundary between *A* and a neighboring region, and then computes the Delaunay neighbors of the set of midpoints. The resulting RAG using such an approach is shown in (c), and is quite similar to that of (b).

Because this procedure is only an approximation, it is important that no preexisting adjacency relations be altered. This is accomplished by adding only the following subset of the Delaunay edges to the subgraphs: Let the ith boundary adjacent to *A* be given by N_{A_i}. For each $[N_{A_i}]$ belonging to a graph of at least three nodes, consider the clockwise circular sequence formed from the identified Delaunay edges of this node. If a substring, $\left\{(N_{A_i})_{-1}, ..., (N_{A_i})_{+1}\right\}$, exists for which $(N_{A_i})_{-1}$ is the first edge and $(N_{A_i})_{+1}$ is the last edge, then the edges in this substring (besides the first and last, which are already present) are added to the subgraphs.



(a)

(b)

(c)

Figure 5.4. (a) Region *A* is an articulation node. It is to be deleted and its three associated subgraphs incorporated into the main subgraph. In (b), this is done by assigning each pixel in *A* to the region to which it is closest, and results in the shown RAG. In (c), a Delaunay graph is computed using the midpoints of each boundary between *A* and a neighboring region.

Although this preserves all of the previous adjacency relationships, it will not necessarily be the case that the resulting main subgraph satisfies Eq. (5.1). As a result, at every edge, N_{i_j}, adjacent to one of the $[N_{A_i}]$, and for which Eq. (5.1) does not hold, an edge is added from between $N_{i_{j-1}}$ and N_{i_j} to between $\left((N_{i_j})_{+1}\right)$ and $\left((N_{i_j})_{+1}\right)_{+1}$. Finally, all of the $[N_{A_i}]$ that have been matched previously and are not currently on either Q or $Q2$, are placed onto Q in order to establish correspondences for the new edges.

5.5. Affine Transformation Parameter Estimation

The region matching process yields matched regions at four different scales. Because the region matching algorithm allows for region merging, groups of regions generally are matched to one another as opposed to simple one-to-one region correspondences. For each matched region group, the best affine transformation between them is estimated iteratively. Denote by R_i^t the ith region group in frame t, and its matched group by R_i^{t+1}. Also, denote the coordinates of the pixels within R_i^t by $\left(x_{ij}^t, y_{ij}^t\right)$, with $j = 1..|R_i^t|$, where $|R_i^t|$ is the cardinality of R_i^t, and denote the pixel nearest the centroid of R_i^t by $(\bar{x}_i^t, \bar{y}_i^t)$. Each $\left(x_{ij}^t, y_{ij}^t\right)$ is mapped by an affine transformation to the point $\left(\hat{x}_{ij}^t, \hat{y}_{ij}^t\right)$ according to

$$\begin{aligned}(x_{ij}^t, y_{ij}^t)^T \rightarrow & \left[\mathbf{A_k}\begin{pmatrix} x_{ij}^t - \bar{x}_i^t \\ y_{ij}^t - \bar{y}_i^t \end{pmatrix} + \vec{T}_k + \begin{pmatrix} \bar{x}_i^{t+1} \\ \bar{y}_i^{t+1} \end{pmatrix}\right] \\ & = (\hat{x}_{ij}^t, \hat{y}_{ij}^t)_k^T \end{aligned} \tag{5.10}$$

A 2 x 2 deformation matrix, $\mathbf{A_k}$, and a translation vector, $\vec{T}_k$, comprise the affine transformation. The subscript k indicates the iteration number, and $[\cdot]$ indicates a vector operator that rounds each vector component to the nearest integer. Define the indicator functions

$$\lambda_i^t(x, y) = \begin{cases} 1, (x, y) \in R_i^t \\ 0, \ \ else \end{cases} \tag{5.11}$$

The amount of mismatch is measured as

$$\left(M_i^t\right)_k = \sum_{x,y} |I_t(x,y) - I_{t+1}(\hat{x},\hat{y})| \cdot \left[\lambda_i^t(x,y) + \lambda_i^{t+1}(\hat{x},\hat{y}) - \lambda_i^t(x,y) \cdot \lambda_i^{t+1}(\hat{x},\hat{y})\right] \quad (5.12)$$

The rounding performed in Eq. (5.10) causes M_i^t to be larger than would be the case if $(\hat{x},\hat{y})$ was a rational coordinate and interpolation kernels were used to estimate $I_{t+1}(\hat{x},\hat{y})$ from the four pixels nearest to $(\hat{x},\hat{y})$. However, the estimated affine transformations are very similar with and without rounding.

The affine transformation parameters that minimize M_i^t are estimated iteratively using a straightforward local descent criterion. As initial guesses, we set $\vec{T_0} = \vec{0}$ and $\mathbf{A}_0 = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. The initial guess is usually very close to the optimal transformation; hence, it is assumed that the first local minimum reached in the descent process corresponds to the global minimum. Typically, about ten iterations are required for convergence.

The region matching algorithm measures region similarity with respect to average intensity, area, position, and region adjacencies, but region shape is not taken into account. It is often the case that an incorrectly matched pair of regions have different shapes. This typically results in a computed affine transformation with a larger amount of deformation than will typically occur over adjacent frames for well-matched regions. Thus, letting a_{ij} denote the elements of $\mathbf{A}$, if

$$(|1 - a_{00}| > \epsilon) \cup (|a_{01}| > \epsilon) \cup (|a_{10}| > \epsilon) \cup (|1 - a_{11}| > \epsilon) \quad (5.13)$$

is true, a match is considered incorrect and is deleted.

If the shape of a region is altered between the two current frames because of occlusion or disocclusion, then the affine parameters estimated by Eq. (5.12) may have some error caused by the changed shape of the region. This problem can be reduced by compensating for this changed region shape.

The present motion field, $\vec{M}_{t,t+1}$, is predicted from the previous motion field, $\vec{M}_{t,t-1}$, if it exists, by the motion prediction module. If the motion field is reasonably smooth from frame to frame, then this predicted motion field will be fairly close to the actual motion field in the present frame. Motion vectors from two or more regions that map onto the same pixel indicate that the pixels are visible in the present frame but all but one of them are occluded in the next frame (VPON). Similarly, a pixel to which no motion vectors map is occluded in the present frame and visible in the next frame (OPVN). The occlusion prediction module identifies the VPON and OPVN areas in the predicted motion field, as well as the occluded regions to which they apply. Before the affine transformation parameters are computed between one of these identified occluded regions in the present frame and its match in the next frame, the shape of the region in the next frame is modified. First, connected sets of VPON pixels are identified. Any of these sets that are adjacent to the region are appended to it. Next, any OPVN contained within the region are subtracted from it. This process is demonstrated in Fig. 5.5. The degree to which the shape of the resulting region remains distorted by the occlusion depends upon the accuracy of the predicted motion field, but even if the predicted motion field is very inaccurate, the distortion will almost always be reduced by this process.

5.6. Motion Field Integration

The affine estimation module results in multiple estimated motion fields for the current frame pair. These fields are combined to yield a single motion field that is (hopefully) more accurate than any of the individual motion fields. Let N_{σ_g} denote the number of homogeneity scale image partitions, making for $N_{\sigma_g}+1$ motion fields. Further, let $\vec{M}_{t,t+1}$ be the final motion field, $\vec{M}^{j}_{t,t+1}$ be the motion field at scale *j*, and let $\vec{M}^{j}_{t,t+1}(x,y) = \left(m^j_x, m^j_y\right)^T$ be the motion vector at pixel (x,y). $\vec{M}_{t,t+1}$ is computed as follows:

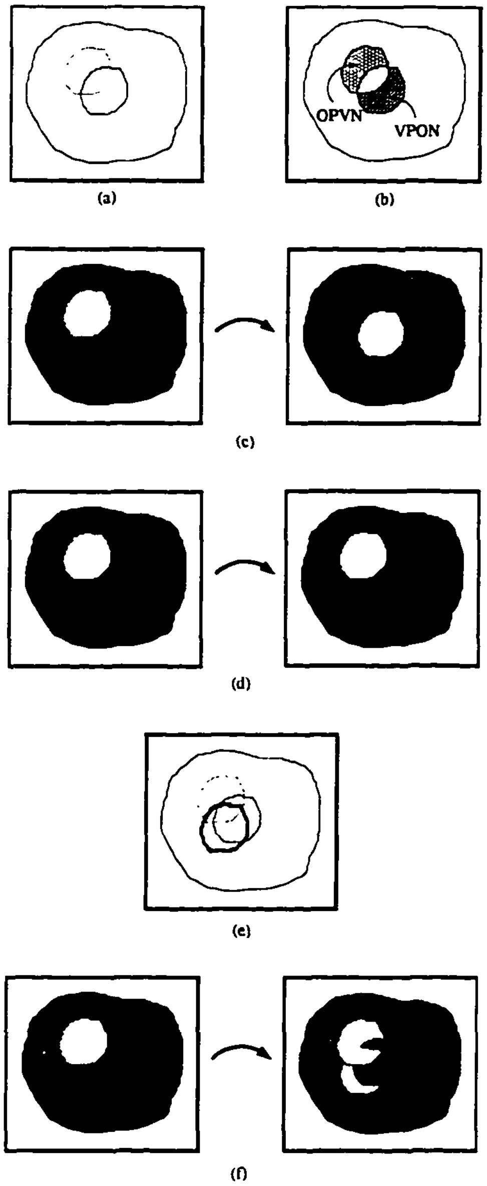


Figure 5.5. (a) A stationary region is occluded by a smaller region whose position (predicted and actual) in the present frame is shown with a dotted boundary and in the next frame with a solid boundary. (b) The corresponding VPON and OPVN pixels are shaded. (c)-(d) The shape of the larger region in both frames is shown before (c) and after (d) the occlusion compensation. In (d), the larger region in the next frame has had the VPON appended to it and the OPVN subtracted from it, which compensates for the shape distortion caused by the occluding region. If the predicted motion of the smaller region is inaccurate, as in (e) where the thicker solid boundary corresponds to its actual position, the occlusion compensation results in (f). Even though the motion prediction was somewhat inaccurate, the compensated shape of the stationary region is still an improvement over the uncompensated shape.

```
$\vec{M}_{t,t+1} = \vec{M}^{N_{\sigma_g}+1}_{t,t+1}$
$j = N_{\sigma_g}$
while ($j > 0$) {
        for each $R_i^t$ present at partition $j$
                if $\left(\delta \cdot \sum_{x,y} \left| I_t(x,y) - I_{t+1}\left(x + m_x^j, y + m_y^j\right) \right| \lambda_i^t(x,y) <\right.$
                        $\left. \sum_{x,y} \left| I_t(x,y) - I_{t+1}(x + m_x, y + m_y) \right| \lambda_i^t(x,y) \right)$
                        $\vec{M}_{t,t+1} = \left(1 - \lambda_i^t\right) \cdot \vec{M}_{t,t+1} + \lambda_i^t \cdot \vec{M}^j_{t,t+1}$
        $j \rightarrow j - 1$
        }
```

The parameter δ determines the extent to which the image prediction error must be reduced at a particular region by the present scale motion field, $\vec{M}^j_{t,t+1}$, over the error resulting from the current final motion field $\vec{M}_{t,t+1}$, and it takes the range $\delta \geq 1.0$. Finally, the subset of the regions $\{R_i^t\}$ whose associated affine transformations comprise $\vec{M}_{t,t+1}$ are considered to be the set of best matched regions.

5.7. Motion Segmentation

The motion field $\vec{M}_{t,t+1}$ is segmented into areas of similar motion. This could be done using the segmentation algorithm described in Chapter 4; however, $\vec{M}_{t,t+1}$ does not have the same degree of complexity as a typical image does. Because of the affine model, the motion is smooth everywhere except at occlusion boundaries where it is abruptly discontinuous. As a result, a simple region growing heuristic is used instead. This heuristic gives adequate results, and is much simpler computationally than using the transform to compute segmentation. The heuristic considers each pair of best matched regions, R_i^t and R_j^t, which share a common border, and merges them if the following relation is satisfied for all $\left(x_{ik}^t, y_{ik}^t\right)$ and $\left(x_{jl}^t, y_{jl}^t\right)$ that are spatially adjacent to one another:

$$\frac{\left\| \vec{M}_{t,t+1}(x_{ik}^t, y_{ik}^t) - \vec{M}_{t,t+1}(x_{jl}^t, y_{jl}^t) \right\|}{\max\left(\left\| \vec{M}_{t,t+1}(x_{ik}^t, y_{ik}^t) \right\| \cdot \left\| \vec{M}_{t,t+1}(x_{jl}^t, y_{jl}^t) \right\| \right)} < m_{\sigma_g} \tag{5.14}$$

where m_{σ_g} is a constant less than 1 that determines the degree of motion similarity necessary for the regions to merge.

The segmented motion regions are each represented in $MS_{t,t+1}$ by a different value. Because each of the best matched regions have matches, the matches in frame t + 1of the regions in $MS_{t,t+1}$ are known and comprise the coarsest scale regions that are used in the affine estimation module for the next frame pair.

It should be noted that because the motion segmentation is done over a single motion field, the segmentation will not necessarily correspond to the moving objects in the scene. Objects that are nonrigid, such as a person, will be segmented into multiple, piecewise rigid regions. In addition, slowly moving objects will not be identified. Handling both of these situations requires examining the motion field over multiple frames.

5.8. Layers and Occlusion

The computed motion field and motion segmentation are examined to determine the areas of occlusion and disocclusion over the current frame pair, as well as the relative depth of the regions in $MS_{t,t+1}$. Let the ith motion region in $MS_{t,t+1}$ be given by R_i^t, and its associated match by R_i^{t+1}. Further, let $\vec{M}_{t+1,t}$ denote the reverse motion field obtained by inverting the affine transformations of each of the best matched regions that comprise $\vec{M}_{t,t+1}$, and applying each inverted transformation to the region's associated match. For each pair of adjacent regions

R_i^t and R_j^t, the following score is computed:

$$L_{ij} = \sum_{(x_1,y_1),(x_2,y_2)} \left\{ \begin{array}{ll} d(I_t(x_1,y_1), I_t(x_2,y_2), I_{t+1}(x_3,y_3)), & \text{if } P \\ 0, & \text{else} \end{array} \right\} + \sum_{(x_4,y_4),(x_5,y_5)} \left\{ \begin{array}{ll} d(I_{t+1}(x_4,y_4), I_{t+1}(x_5,y_5), I_t(x_6,y_6)), & \text{if } Q \\ 0, & \text{else} \end{array} \right\}$$

$$\begin{aligned} &(x_1,y_1) = (x_{ik}^t, y_{ik}^t), \ \ (x_2,y_2) = (x_{jl}^t, y_{jl}^t) \\ &(x_3,y_3)^T = \vec{M}_{t,t+1}(x_1,y_1) + (x_1,y_1)^T \\ &P \ \ iff \ \ (x_3,y_3)^T = \vec{M}_{t,t+1}(x_2,y_2) + (x_2,y_2)^T \\ &(x_4,y_4) = (x_{ik}^{t+1}, y_{ik}^{t+1}), \ \ (x_5,y_5) = \left(x_{jl}^{t+1}, y_{jl}^{t+1}\right) \\ &(x_6,y_6)^T = \vec{M}_{t+1,t}(x_4,y_4) + (x_4,y_4)^T \\ &Q \ \ iff \ \ (x_6,y_6)^T = \vec{M}_{t+1,t}(x_5,y_5) + (x_5,y_5)^T \end{aligned} \tag{5.15}$$

The operator $d(\cdot)$ is given by

$$d(x,y,z) = \begin{cases} 1, & |x-z| < |y-z| \\ 0, & |x-z| = |y-z| \\ -1, & |x-z| > |y-z| \end{cases} \tag{5.16}$$

The sign of L_{ij} indicates whether R_i^t occludes (+) or is occluded by (-) R_j^t, and the magnitude of L_{ij} represents the degree of confidence in this result.

The L_{ij} are used in assigning a layer number to each region in $MS_{t,t+1}$, such that a larger number indicates that the region is closer to the camera. In order to make this assignment, any circular dependencies, such as $A<B<C<A$, must be eliminated. This is done by breaking each such chain at the constraint having the L_{ij} of smallest magnitude. Once this is done, layer number assignment is straightforward. The array $L_{t,t+1}$ is then formed, within which each pixel takes on the value of the layer number of its associated region in $MS_{t,t+1}$.

Areas of occlusion (VPON) are identified as the set of all (x_3, y_3) pixels in Eq. (5.15) for which the sign of the associated $d(\cdot)$ is in agreement with the sign of L_{ij}, and for which the layer constraint corresponding to L_{ij} was not deleted in resolving any circular constraint dependencies. Areas of disocclusion (OPVN) are identified in exactly the same manner, except that the (x_6, y_6) pixels are used. The VPON and OPVN areas are then stored in the occlusion array, $O_{t,t+1}$.

5.9. Motion and Occlusion Prediction

The process of predicting $\vec{M}_{t+1,t+2}$ is similar to the method for computing $\vec{M}_{t+1,t}$. The affine transformations of each of the best matched regions comprising $\vec{M}_{t,t+1}$, are applied to each region's associated match in frame $t + 1$, thereby yielding a prediction of $\vec{M}_{t+1,t+2}$. The accuracy of this predicted motion field depends upon the degree to which pixel trajectories are temporally consistent over adjacent frame pairs.

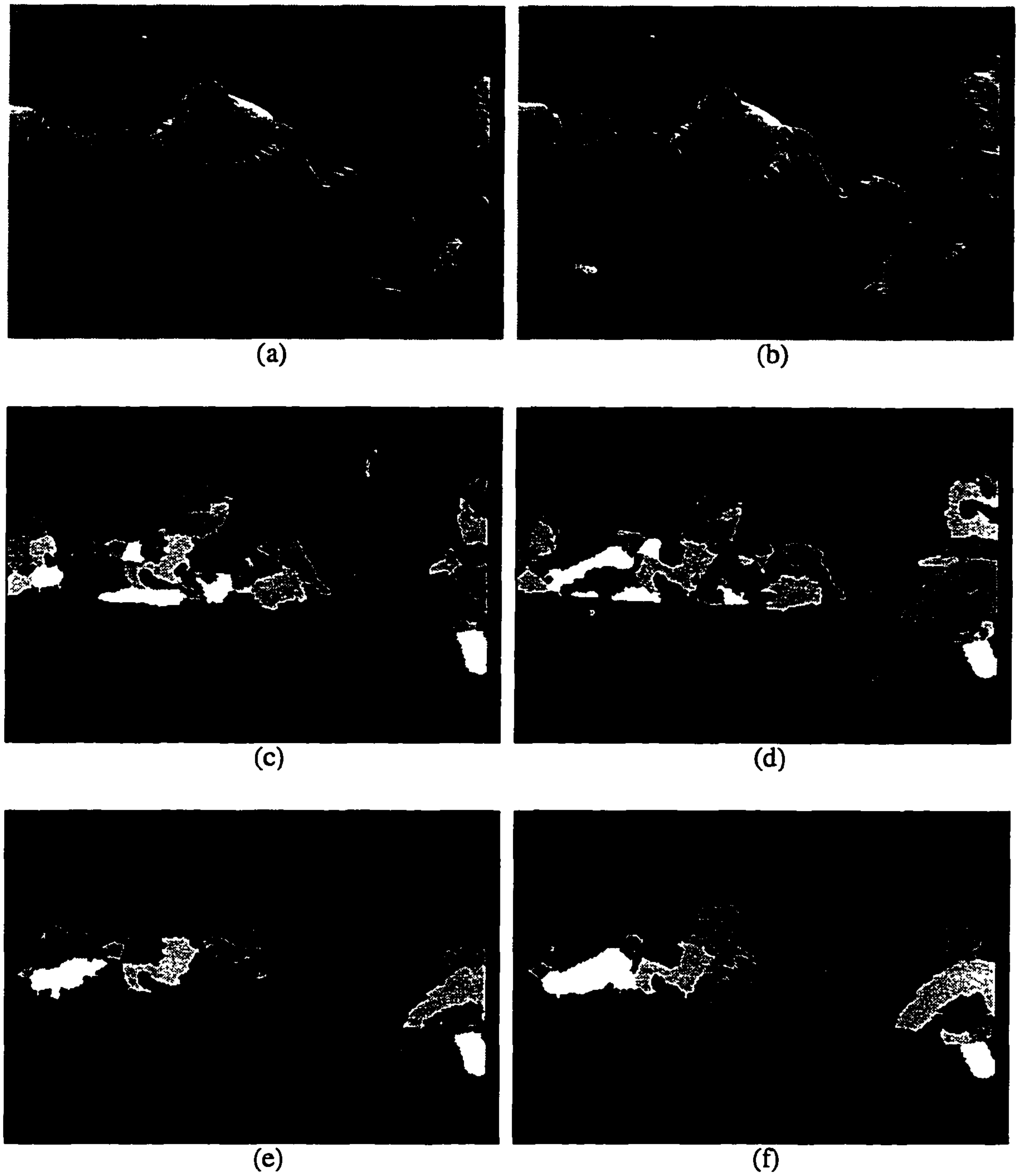
For occlusion prediction, the reverse motion field, $\vec{M}_{t+2,t+1}$ is predicted using the coarsest scale matches obtained from the region matching module for the frame pair $(t + 1, t + 2)$. Occlusion prediction is then performed in exactly the same manner as the algorithm for computing the occlusion described in Section 5.8.

5.10. Implementation Details

To form the region adjacency graphs required by the region matching module, the contours of the regions present at each of the image partitions are followed clockwise, and information regarding region adjacencies, articulation nodes, and the length and midpoint of the boundary between each pair of adjacent regions, is computed and stored. In a 2–D image, region adjacencies are measured using 4–nearest neighbor connectivity (i.e., N, S, E, W pixel neighbors). At vertices where four different regions meet, NW-SE connectivity is also used in order for Eq. (5.1) to remain true. Finally, the various parameters in the motion estimation and segmentation algorithm presently have the following values: $a = 0.9$, $e_1 = 20$, $e_2 = 35$, $p = 0.2$, $T = 20$, $\epsilon = 0.3$, $\delta = 1.15$, and $m_{\sigma_g} = 0.2$.

5.11. Experimental Results

The performance of the motion estimation algorithm is demonstrated in Figs. 5.6–5.13 for four different image sequences. Three consecutive frames were selected from each of these sequences. The three partitions of each image used by the region matching algorithm correspond to the sets of regions identified by the segmentation algorithm at $\sigma_g = 9, 15, 21$. Figures 5.6 and 5.7 contain results from a sequence of a football game. This sequence is characterized by very fast and nonrigid motion, as well as multiple layers of occlusion. The first two frames of this sequence are shown in Fig. 5.6(a)-(b). The result of the region matching process is shown in (c) and (d) for $\sigma_g = 9$, and in (e) and (f) for $\sigma_g = 21$. The regions in a given matched set are all displayed with the same intensity value, and all other regions spatially adjacent to this set are guaranteed to be assigned a different intensity value. In addition, unmatched areas are displayed as black. The final estimated motion field for this frame pair is shown in (g) with the motion sampled once every ten pixels in the horizontal and vertical directions. The segmentation of this motion field is given in (h) with each pair of adjacent regions labelled with different intensity values, and the layer image associated with the motion segmentation is given in (i). The regions are labelled so that brighter regions are closer to the camera than darker regions. Unmatched areas in (h) and (i) are displayed as black. Areas that become occluded are shown in (j) overlaid onto the first frame. Similarly, areas that become disoccluded are given in (k) overlaid onto the second frame. The next frame pair in this sequence (the second and third frames) is shown in Fig. 5.7(a)-(b). The results displayed in (c)-(f) correspond to the information in Fig. 5.6(c)-(f). In addition, the motion field predicted for this frame pair from the motion computed for the previous frame pair



(a)

(b)

(c)

(d)

(e)

(f)

Figure 5.6. (a)-(b) The first two frames of a football sequence. (c)-(d) Matched regions present at $\sigma_g = 9$. Matched regions are displayed with the same intensity value, and adjacent regions are assigned different intensity values. (e)-(f) Same as (c)-(d), but with $\sigma_g = 21$.

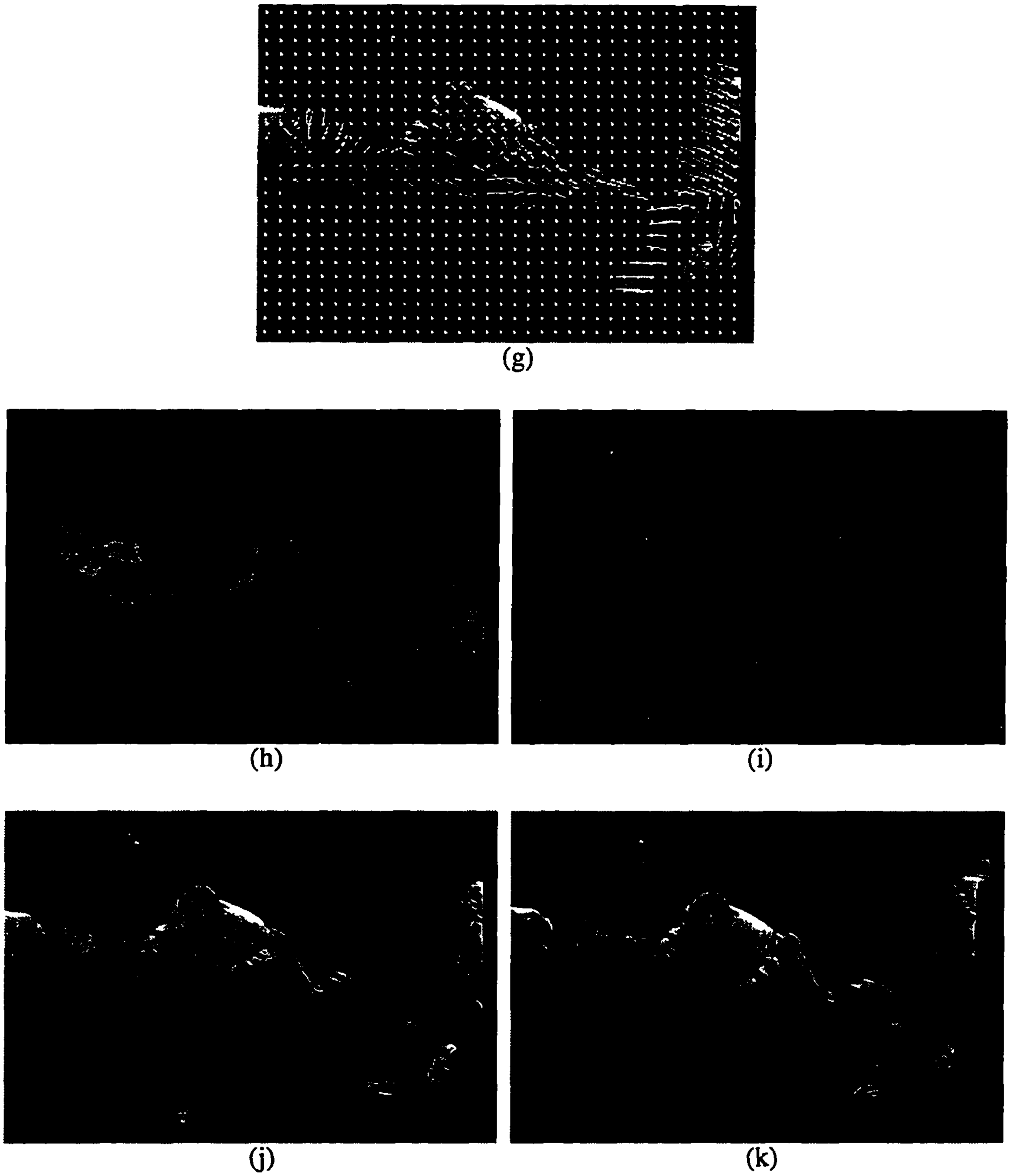


Figure 5.6 (cont.). (g) The calculated motion field shown subsampled. (h) Segmentation of the motion field in (g) displayed by assigning different intensity values to adjacent regions. (i) Layer image corresponding to the motion segmentation. The brighter the object, the closer it is to the camera. (j) Identified occlusion areas. (k) Identified disocclusion areas.

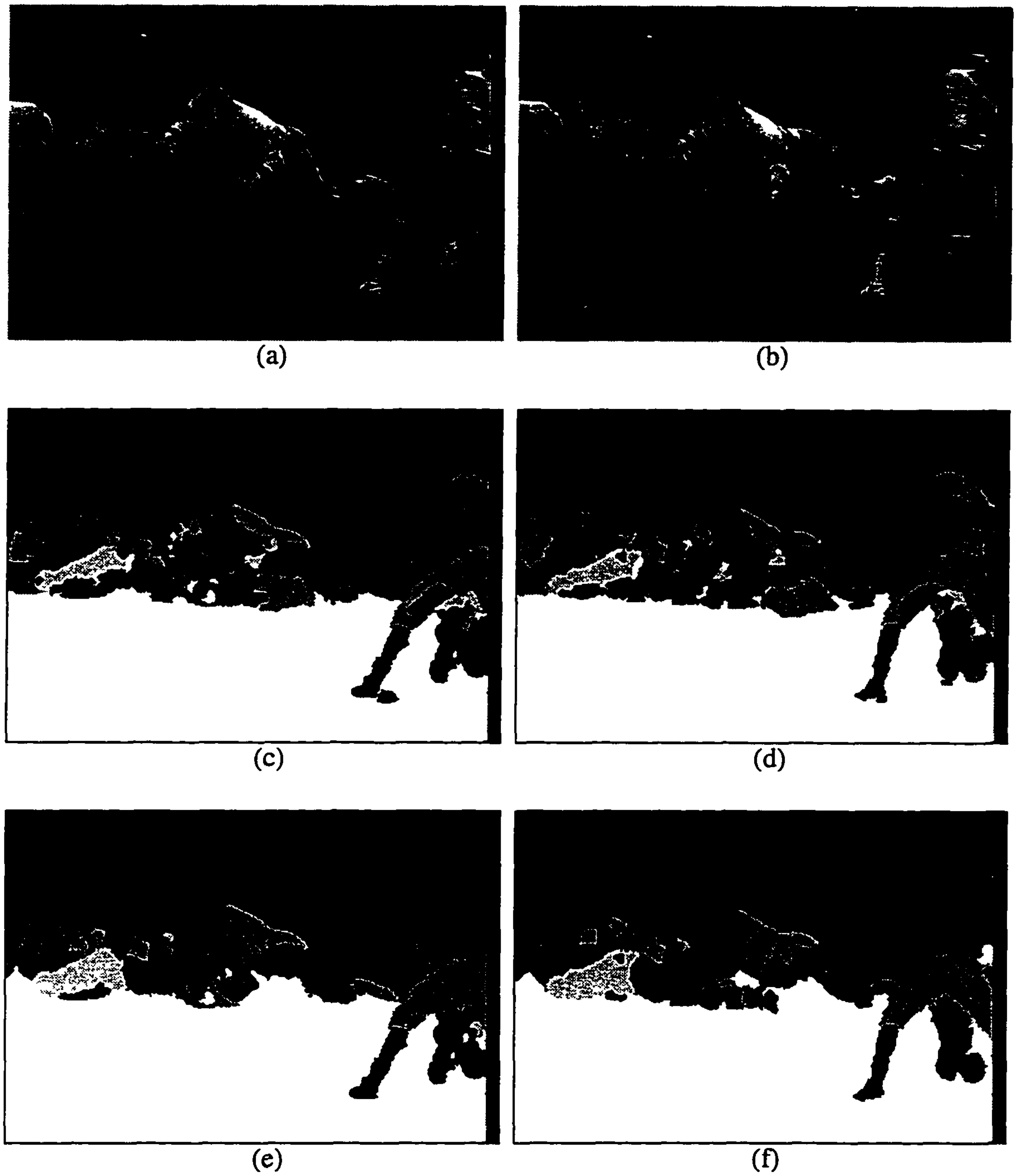


Figure 5.7. (a)-(b) The second and third frames of the football sequence. (c)-(f) Same as Fig. 5.6(c)-(f).

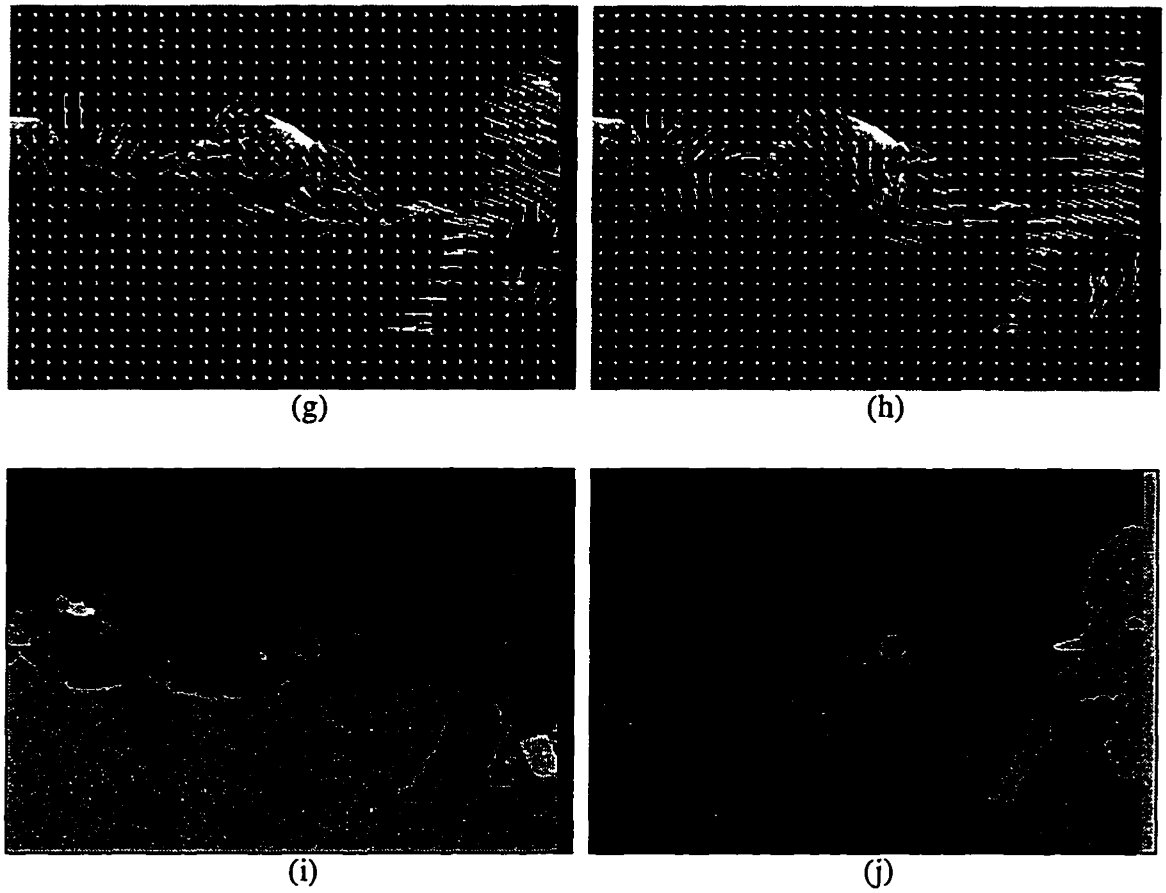
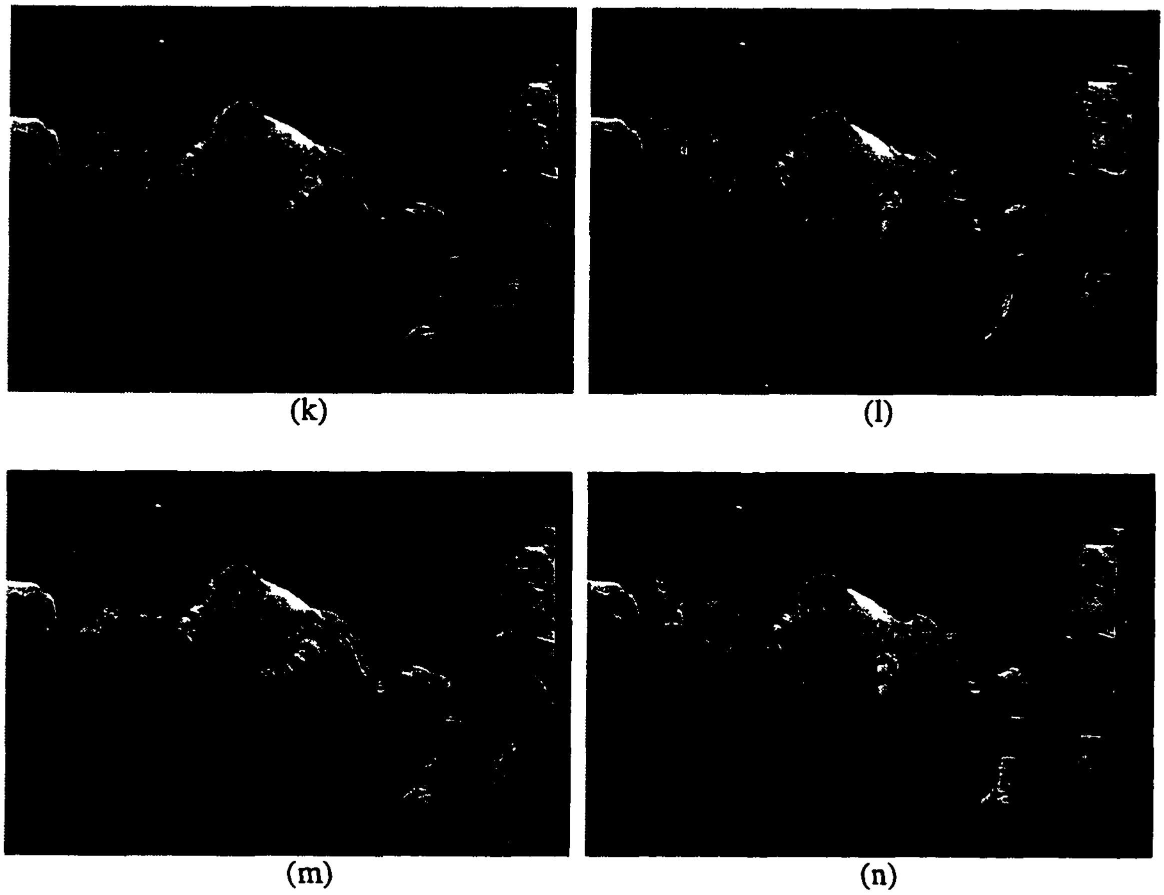


Figure 5.7 (cont.). (g)-(h) The predicted and calculated motion fields, respectively, shown subsampled. (i) Motion field segmentation. (j) Corresponding layer image.



(k)

(l)

(m)

(n)

Figure 5.7 (cont.). (k)-(l) Predicted occlusion and disocclusion areas, respectively. (m)-(n) Computed occlusion and disocclusion areas, respectively.

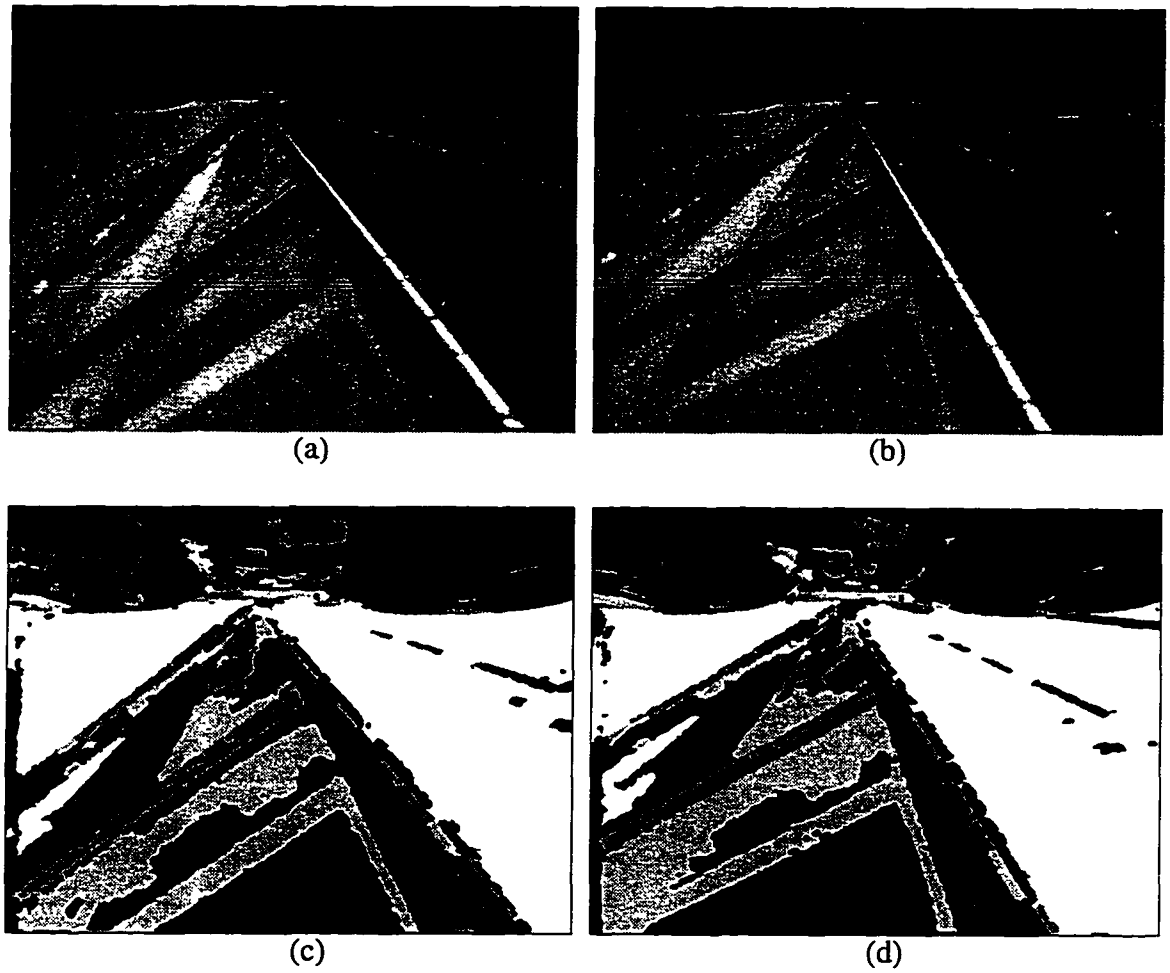
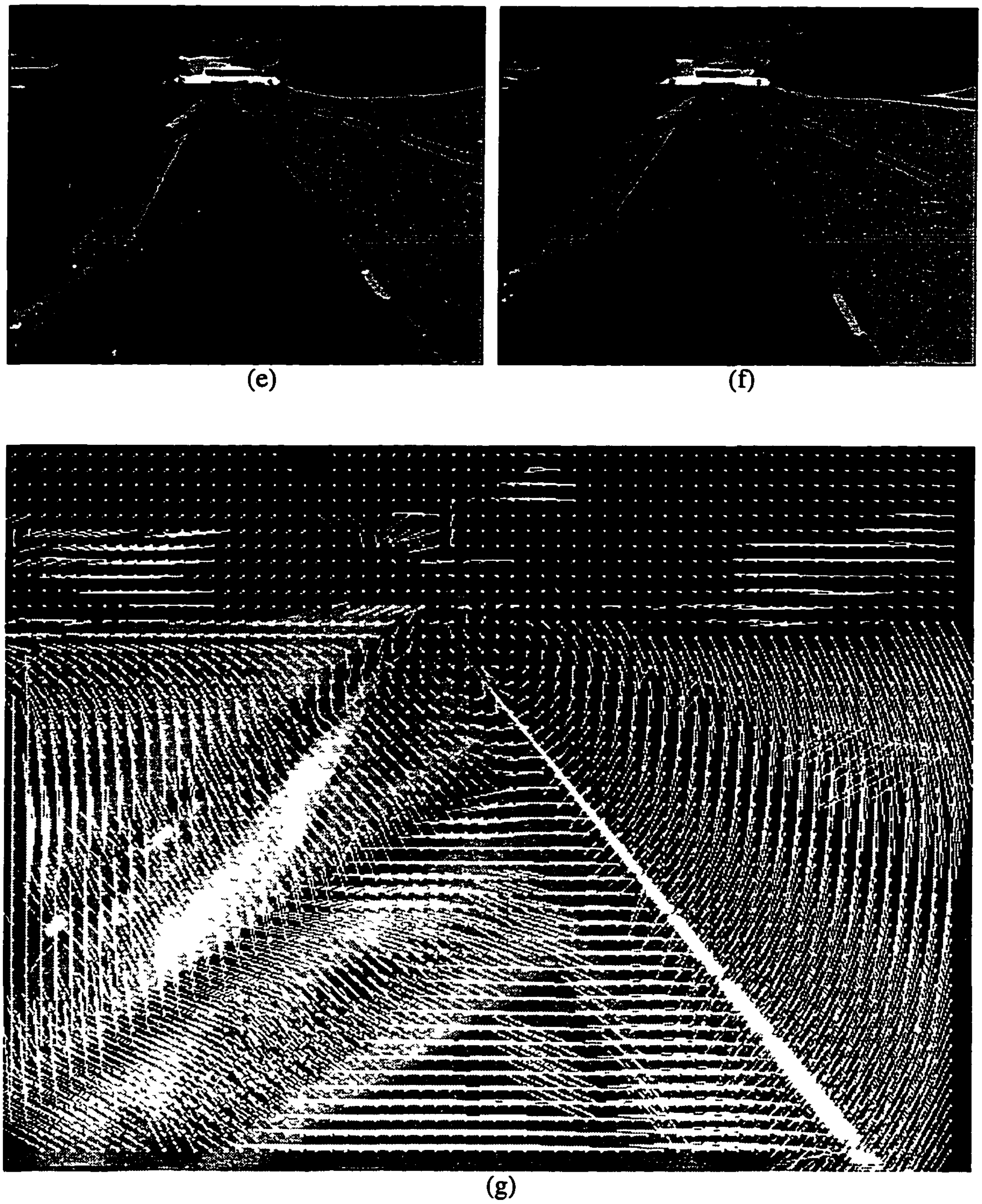


Figure 5.8. (a)-(b) The first two frames of an aerial sequence. The camera is mounted beneath an airplane. From the camera's perspective, the plane is stationary and the ground is rotating, however, the ground is reflected onto some areas of the surface of the plane, thereby inducing motion in those areas. (c)-(g) Same as Fig. 5.6(c)-(g).

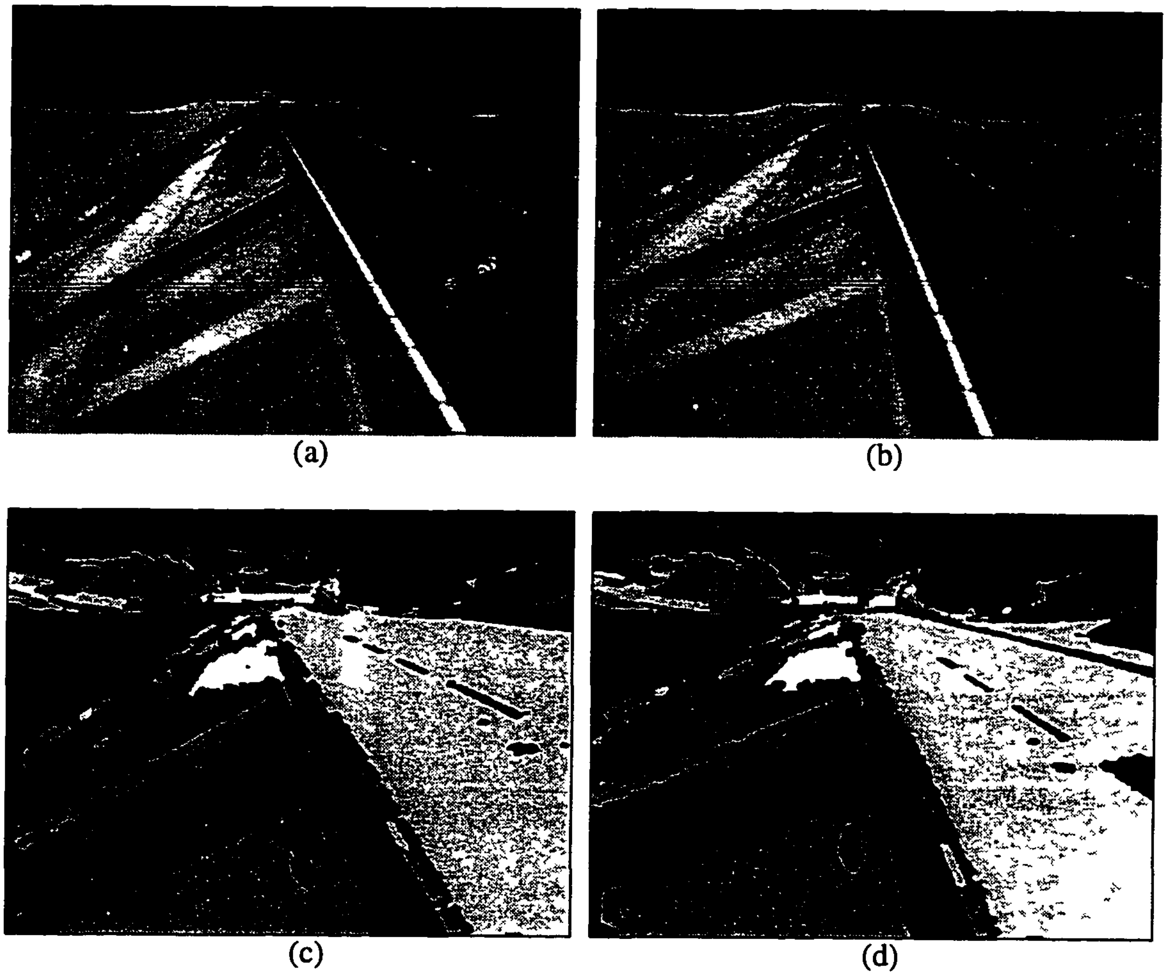


(e)

(f)

(g)

Figure 5.8 (cont.).



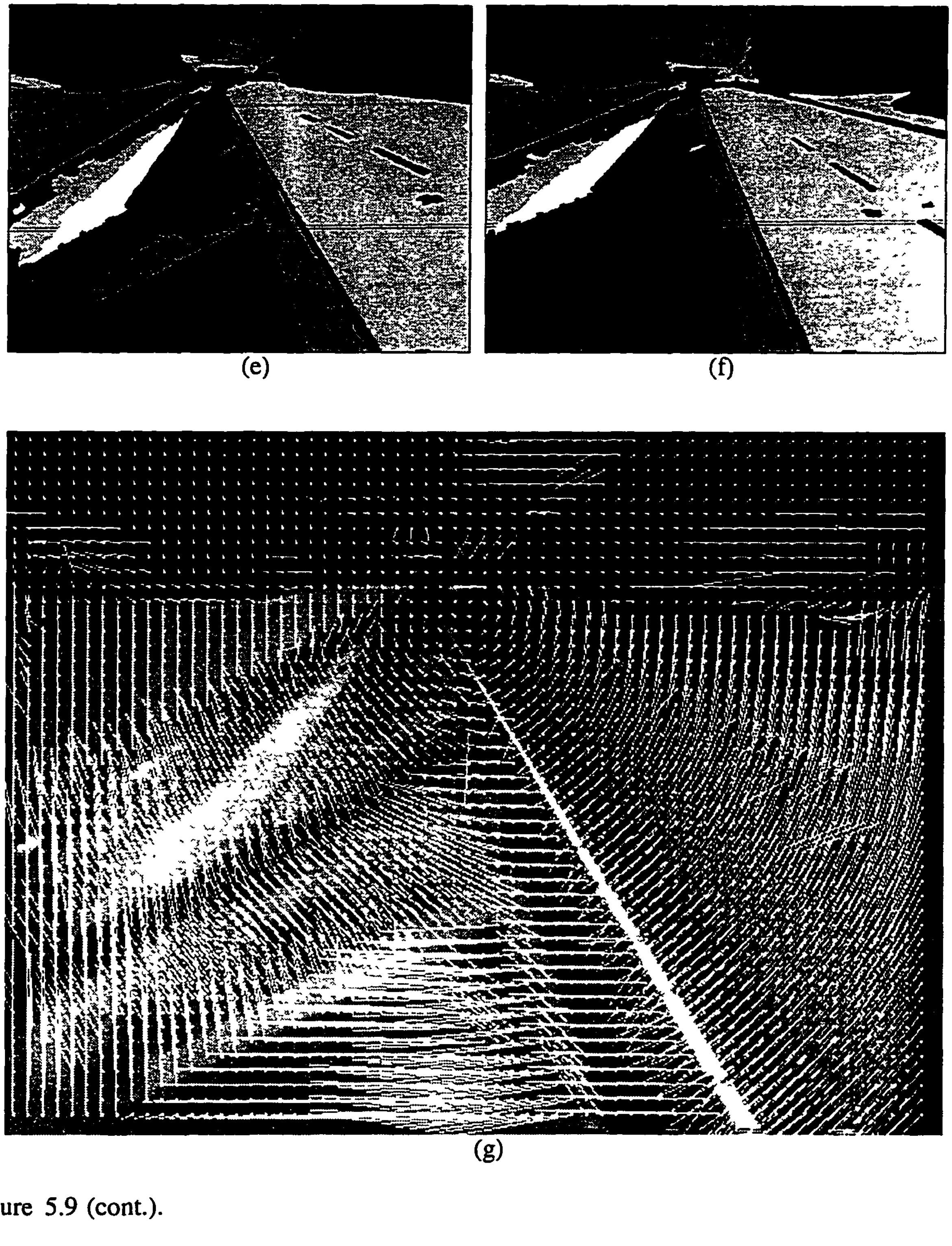
(a)

(b)

(c)

(d)

Figure 5.9. (a)-(b) The second and third frames of the aerial sequence. (c)-(g) Same as Fig. 5.6(c)-(g).

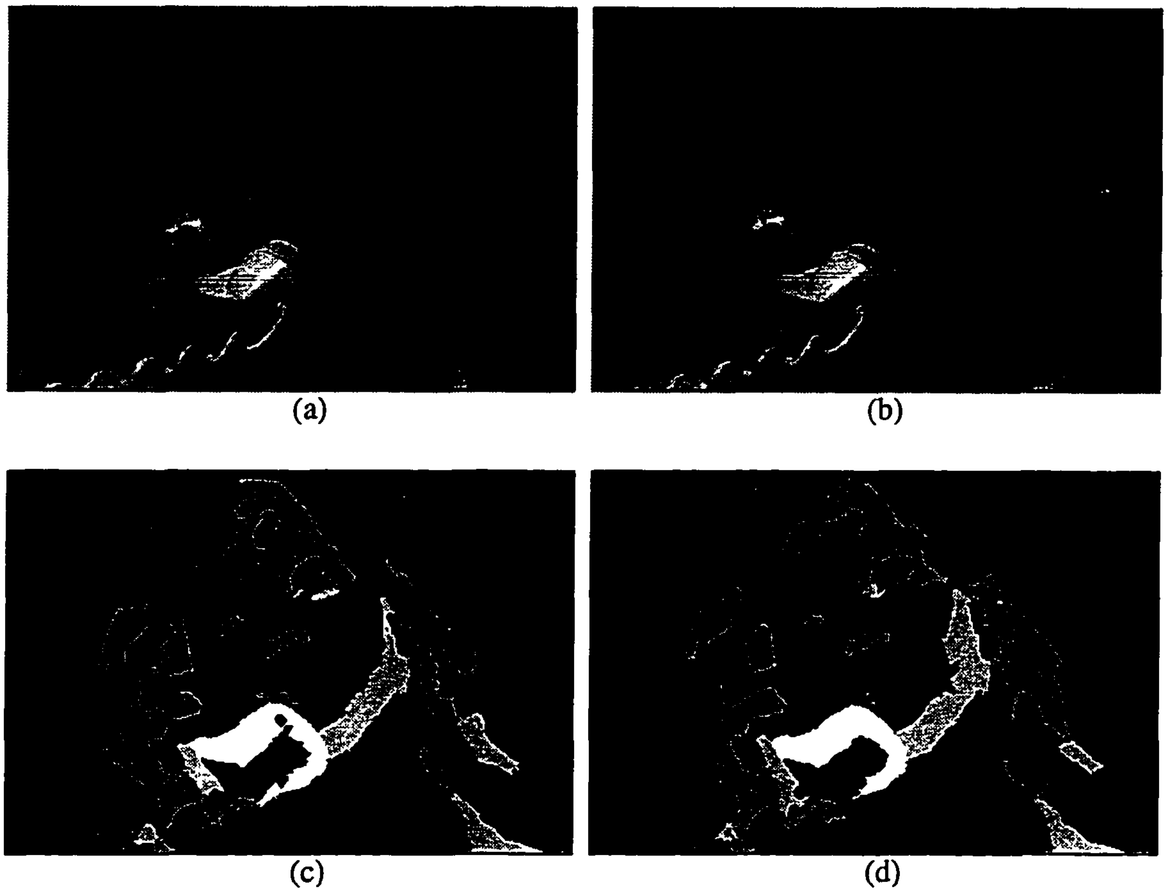


(e)

(f)

(g)

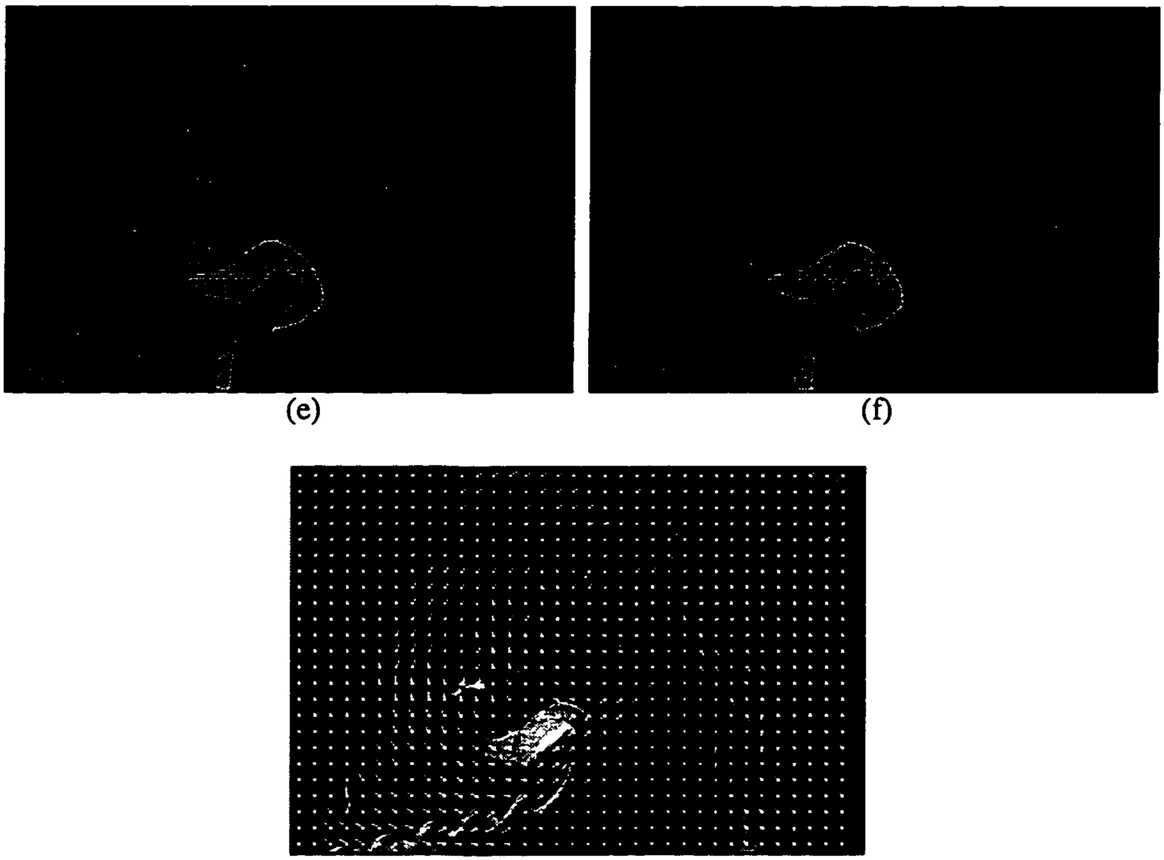
Figure 5.9 (cont.).



(a) (b)

(c) (d)

Figure 5.10. (a)-(b) The first two frames of a sequence of a woman talking on the telephone. The telephone and the woman's head are rotating, while the background is stationary. (c)-(g) Same as Fig. 5.6(c)-(g). Note that the lack of subtexture within the background region prevents its motion from being estimated correctly.

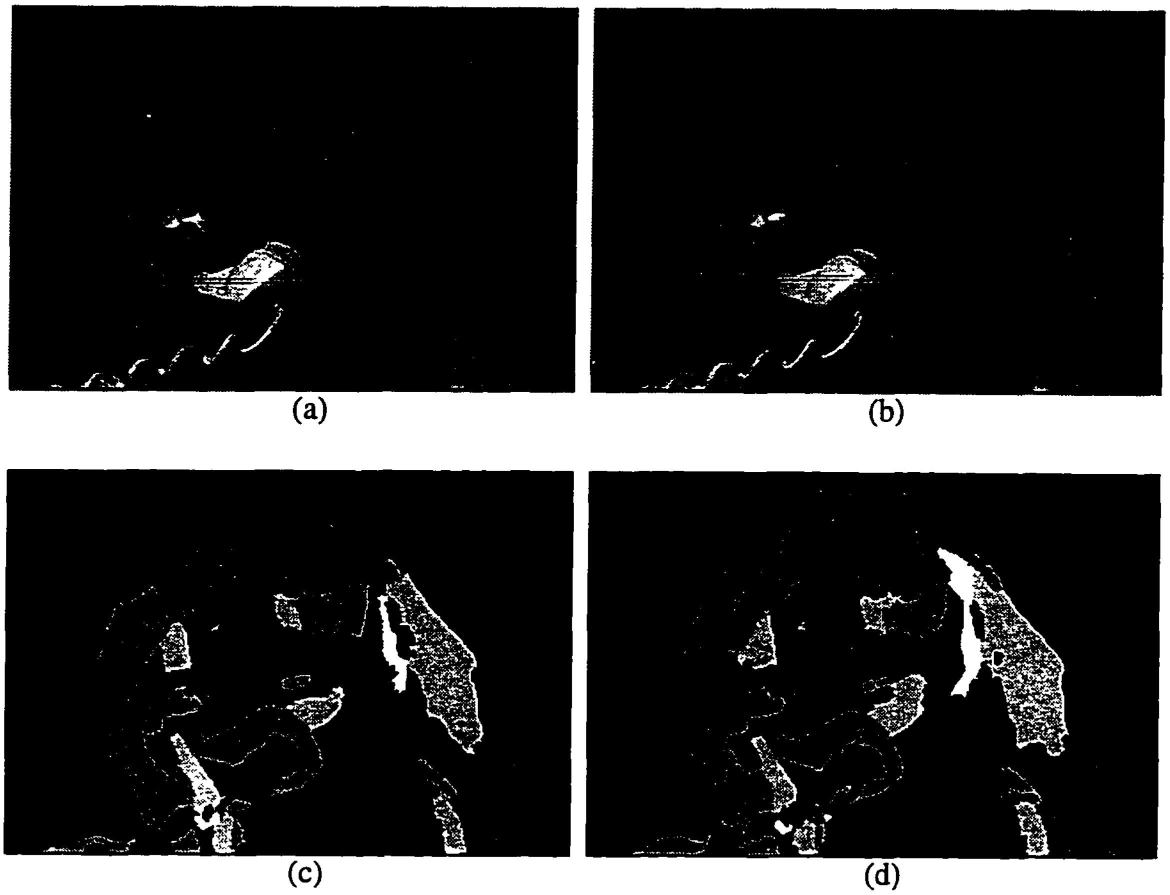


(e)

(f)

(g)

Figure 5.10 (cont.).



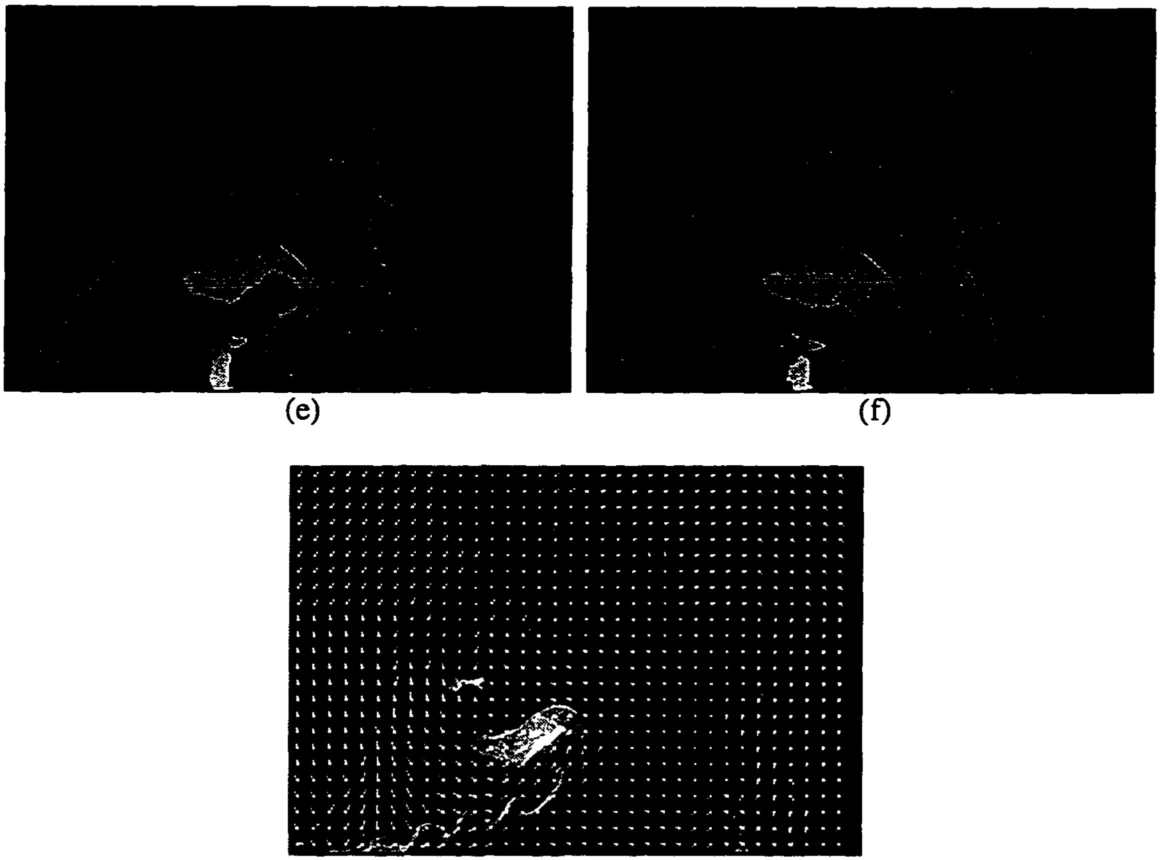
(a)

(b)

(c)

(d)

Figure 5.11. (a)-(b) The second and third frames of the telephone sequence. (c)-(g) Same as Fig. 5.6(c)-(g).



(e)

(f)

(g)

Figure 5.11 (cont.).

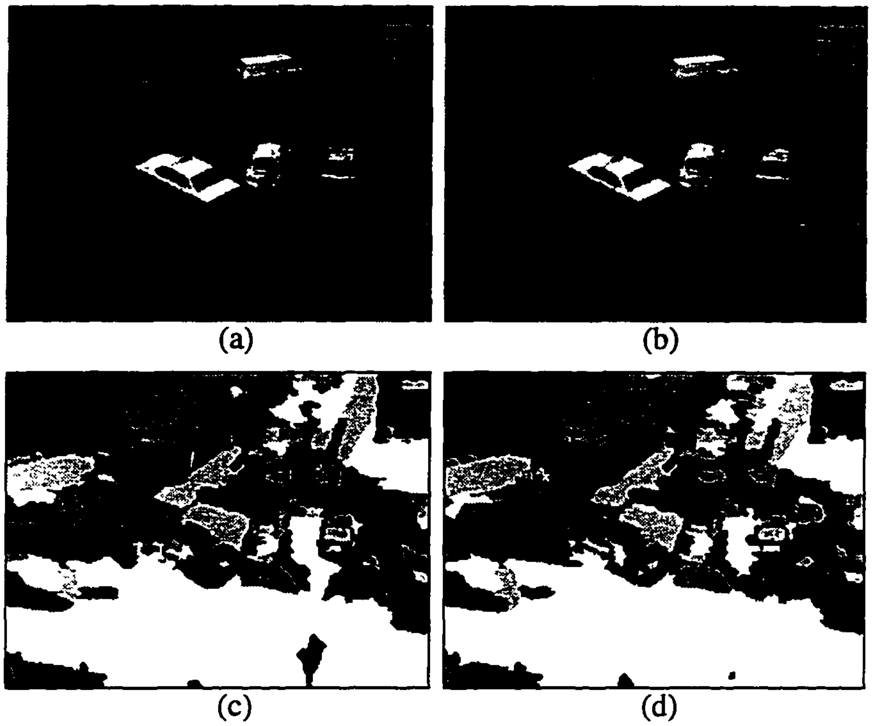
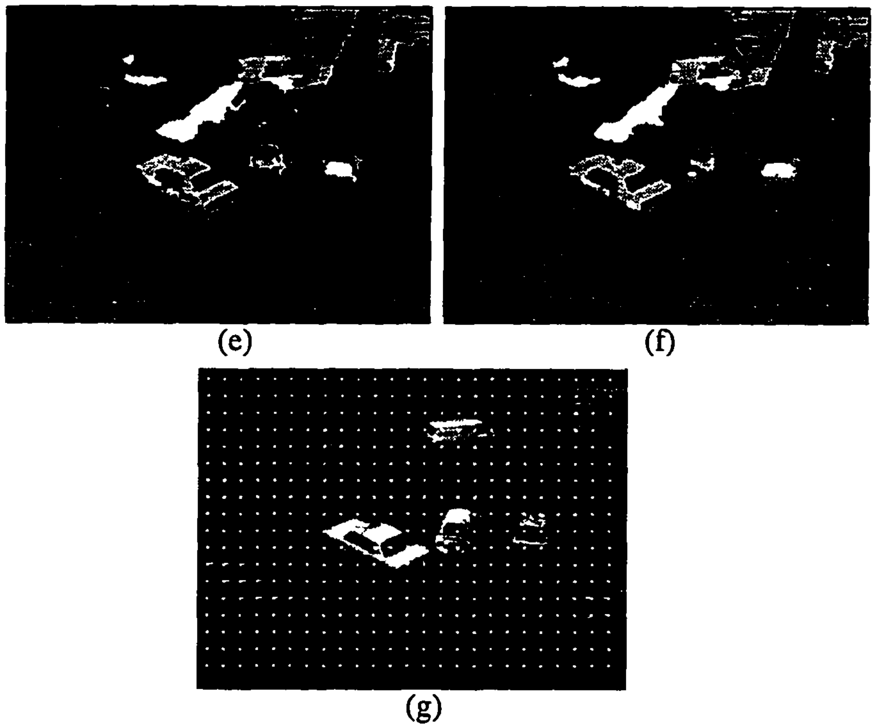


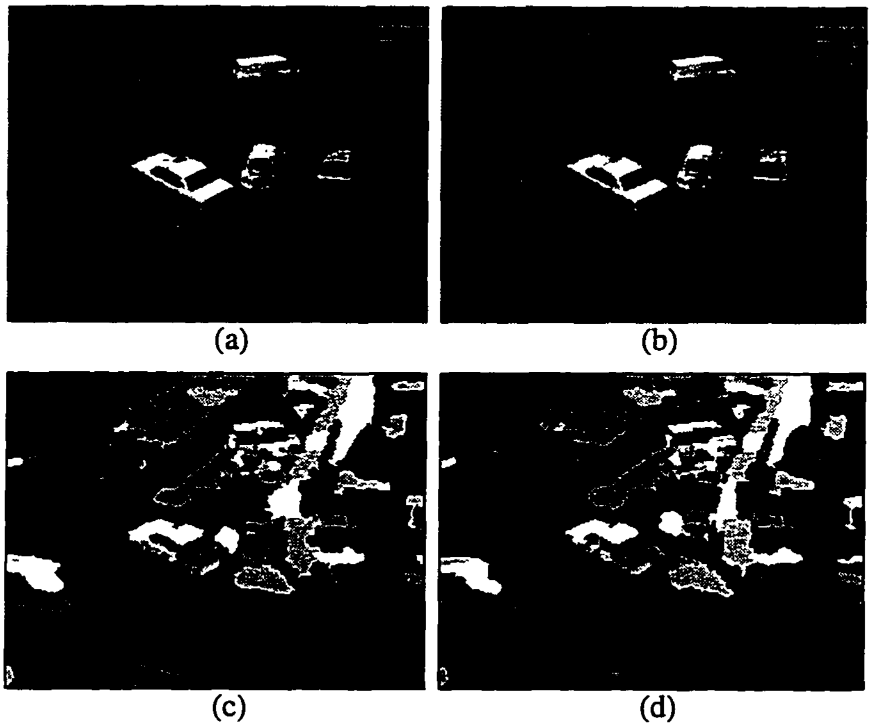
Figure 5.12. (a)-(b) The first two frames of a sequence of a traffic scene. Three cars are moving fairly slowly (0.5 – 2 pixels/frame). (c)-(g) Same as Fig. 5.6(c)-(g).



(e) (f)

(g)

Figure 5.12 (cont.).



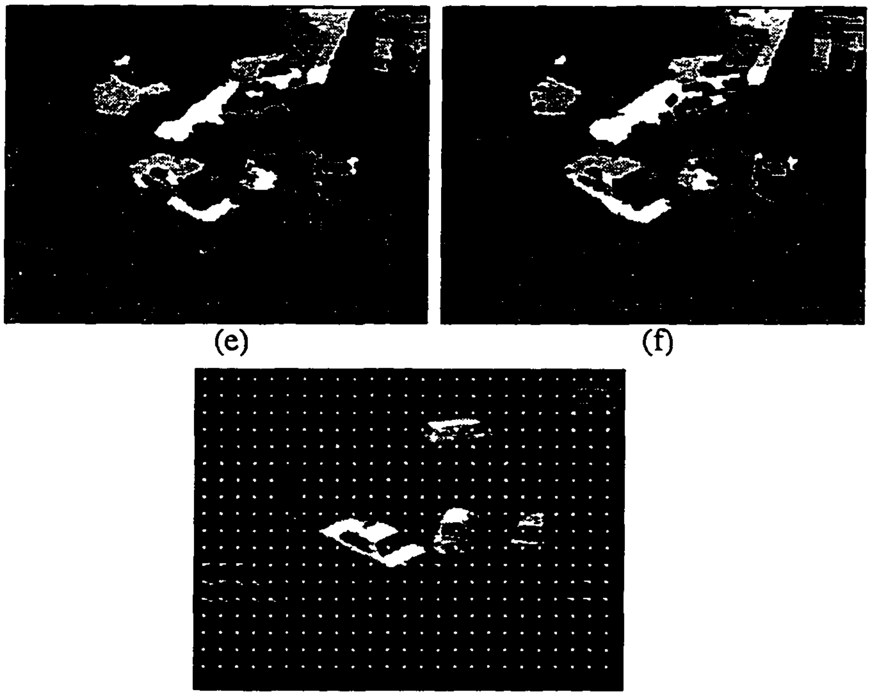
(a)

(b)

(c)

(d)

Figure 5.13. (a)-(b) The second and third frames of the traffic sequence. (c)-(g) Same as Fig. 5.6(c)-(g).



(e)

(f)

(g)

Figure 5.13 (cont.).

is given in (g), and the actual computed motion field is shown in (h). The corresponding motion segmentation and layer map are given in (i) and (j), respectively. Finally, (k)-(l) represents the predicted occlusion and disocclusion areas, and (m)-(n) represents the actual identified areas. Note that even though the motion is quite fast and nonrigid, both the predicted motion field and (dis)occlusion areas are fairly close to the actual estimated results. The computed motion fields qualitatively resemble the actual motion quite well, especially near the occluding contours. Also, from the motion segmentation results, one can see that the identified regions should not be regarded as constituting moving objects. In image sequences where the motion is more rigid, this is a better assumption, but in situations where objects move very slowly or nonrigidly, it is necessary to examine the motion fields over multiple frames in order to identify the moving objects. However, the motion regions identified by segmenting individual motion fields are useful as a coarsest scale image partition.

Figures 5.8 and 5.9 display results from an aerial sequence taken from a camera mounted beneath an airplane. For both figures, (a)-(g) contain the same information as is present in Fig. 5.6(a)-(g). Over the three frames shown, the ground is rotating and the plane is stationary as viewed from the camera's perspective. The motion estimated for the plane is not zero because in some areas on the plane the ground is reflected onto the plane's surface. With regard to the ground motion, one can see that the estimated motion is not entirely smooth. This is caused, not by region mismatches, but by the fact that many of the regions on the ground have little subtexture and large fractions of the regions' boundaries are occluding contours. This causes a certain amount of noise in the estimated affine transformation parameters. This same problem also can be seen in the sequence displayed in Figs. 5.10 and 5.11. This sequence consists of a woman talking on a telephone. The woman's head is rotating, while the background is stationary.

The motion of the woman's head is estimated correctly, but the background motion is not. The background is segmented into two regions, both of whose entire boundaries consist of occluding contours. Thus, the motion information is contained solely in the background region subtexture. The leftmost background region has no subtexture, and, hence, its motion is undefined, while the rightmost region has a small amount of subtexture, but not enough for the algorithm to determine the correct motion. This situation can be contrasted with that of the football sequence, where the background regions also consist entirely of occluding contours, but which have significant subtexture, thereby allowing the correct motion to be estimated.

Results from the last sequence are displayed in Figs. 5.12 and 5.13. This sequence consists of a traffic scene within which three cars are moving quite slowly (0.5 – 2 pixels/frame) on a stationary background. The motion is estimated accurately except for a small patch on the center car where a shadow causes the estimated motion to deviate from the true motion by a couple of pixels.

5.12. Discussion

A method for region-based 2–D motion estimation has been given in this chapter. A video sequence is processed two frames at a time, although information from the previous frame pair is used in guiding the region matching and affine estimation processes at the current frame. A multiscale segmentation algorithm is applied separately to each frame. This provides a rich set of regions available for matching. The process of region matching is formulated as an inexact region adjacency graph matching problem. Pixel correspondences are obtained for each pair of matched regions by estimating the parameters of the best affine transformation relating the regions. Before computing the affine transformation parameters, the distortion in region shapes caused by occlusion is compensated. A motion field is then estimated from the affine parameters calculated for the set of regions identified as being the best matched.

6. CONCLUSION

This thesis has described the problem of structure detection and its relationship to scale. Three distinct kinds of scale have been identified that are necessary to characterize structure. These scales are utilized by a transform that represents structure within a force field as areas of contracting flow. In the case when the structures are connected, the boundaries of the structures are indicated by pairs of diverging field vectors. Boundaries that form closed contours at some scale, and which are stable to changes in scale, are identified as perceptually important. A pyramidal representation is constructed that contains each salient structure along with the range of homogeneity scale for which it exists. The identified pyramids agree fairly closely with human perception over a wide variety of different images. In addition, the application of this pyramidal representation to the problem of estimating 2–D motion fields has been discussed. The pyramids corresponding to a pair of temporally adjacent image frames are matched by formulating the matching process as an inexact matching of attributed region adjacency graphs. An affine motion model is then applied to the pixels within each pair of matched structures to give motion information. Although the motion algorithm operates two frames at a time, it feeds back information obtained from the previous matched frame pair to guide the region matching and motion estimation modules.

6.1. Primary Contributions of this Work

The relevant results of the presented research include:

1. A framework within which the general problem of structure detection can be solved.
2. A formulation of scale that is directly related to image structure.
3. A transform that is able to make the structure at a given scale explicit.

4. A method for integrated automatic scale selection and region structure detection.
5. Identification of region structures with unsmoothed boundaries, regardless of the scale of the structure.
6. An integration of the processes of region and edge detection.
7. A 2–D motion estimation algorithm capable of identifying accurate motion at occlusion boundaries and in areas possessing little texture.

6.2. Future Extensions of this Work

The most obvious extension of this work is to apply the given framework to the general structure detection problem. This requires developing a region-based method for extracting both connected and disconnected zones of attraction. One possible approach is to treat the field domain as a true flow, and allow pixels to move according to the force they experience. Structures then naturally coalesce. With regard to texture, more work has to be done on using integration scales to capture local texture statistics. In addition, although the algorithm as applied to image segmentation does seem to indicate that the problem is indeed well-posed, the model used is implicit. Although using a single homogeneity scale for each structure seems reasonable, one would like to know what this truly implies, and whether, for example, there might be any advantages in allowing σ_g to be a slowly varying function within each region. It does not appear that the structure model can be explicated for any but the most trivial (and, hence, uninteresting) cases, but further attempts have to be made. Also, modifications to the transform that result in the vector magnitudes as well as directions agreeing with that of the gradient at a particular scale should be explored. A reasonable definition of a multiscale gradient may have interesting mathematical ramifications. With regard to motion analysis, the motion segmentation process should be extended to consider similarity of motion over multiple frames. This is necessary to allow both nonrigid and slowly moving objects to be segmented. Finally, an analysis has to be done of both the performance advantage and computational penalty involved in extending the algorithm to perform region matching and motion estimation over a batch of frames simultaneously, instead of in pairs.

REFERENCES

[1] R. Ohlander, K. Price, and D. Reddy, "Picture segmentation using a recursive region splitting method," *Comput. Graph. Image Process.*, vol. 8, no. 3, pp. 313–333, 1978.

[2] R. Haralick and L. Shapiro, "Survey: Image segmentation techniques," *Comput. Vis. Graph. Image Process.*, vol. 29, no. 1, pp. 100–132, 1985.

[3] S. Zucker, "Survey region growing: Childhood and adolescence," *Comput. Graph. Image Process.*, vol. 5, no. 4, pp. 382–399, 1976.

[4] O. Monga, "An optimal region growing algorithm for image segmentation," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 1, no. 4, pp. 351–375, 1987.

[5] S. Horowitz and T. Pavlidis, "Picture segmentation by a directed split-and-merge procedure," *Proc. 2nd Int. Joint Conf. Pattern Recognit.*, pp. 424–433, 1974.

[6] F. Meyer and S. Beucher, "Morphological segmentation," *J. Vis. Commun. Image Represent.*, vol. 1, no. 1, pp. 21–46, 1990.

[7] A. Nazif and M. Levine, "Low level image segmentation: An expert system," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 5, pp. 555–577, 1984.

[8] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 11, pp. 721–741, 1984.

[9] T. Lindeberg, *Scale-Space Theory in Computer Vision.* Boston: Kluwer Academic, 1994.

[10] R. Whitaker and S. Pizer, "Geometry-based image segmentation using anisotropic diffusion," in *Shape in Picture: Mathematical Descriptions of Shape in Grey-Level Images*, pp. 641–650, New York: Springer-Verlag, 1992.

[11] P. Salembier, "Morphological multiscale segmentation of images," *Proc. SPIE Visual Commun. Image Process. '92*, vol. 1818, no. 3, pp. 620–631, 1992.

[12] B. Horn and B. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, no. 1, pp. 185–203, 1981.

[13] A. Verri, F. Girosi, and V. Torre, "Differential techniques for optical flow," *J. Opt. Soc. Am.*, vol. 7, no. 5, pp. 912–922, 1990.

[14] S. Barnard and W. Thompson, "Disparity analysis of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-12, no. 4, pp. 333–340, 1980.

[15] S. Haynes and R. Jain, "Detection of moving edges," *Comput. Vis. Graph. Image Process.*, vol. 21, no. 3, pp. 345–367, 1982.

[16] E. Hildreth, "Computations underlying the measurement of visual motion," *Artif. Intell.*, vol. 23, no. 3, pp. 309–354, 1984.

[17] J. Weng, N. Ahuja, and T. Huang, "Matching two perspective views," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 8, pp. 806–825, 1992.

[18] H. Nagel, "Displacement vectors derived from second-order intensity variations in image sequences," *Comput. Vis. Graph. Image Process.*, vol. 21, no. 1, pp. 85–117, 1983.

[19] C. Fuh and P. Maragos, "Region-based optical flow estimation," in *Comp. Vision Patt. Recog. '89*, (San Diego), pp. 130–135, 1989.

[20] G. Healey, "Hierarchical segmentation-based approach to motion analysis," *Image Vis. Comput.*, vol. 11, no. 9, pp. 570–576, 1993.

[21] D. Kalivas and A. Sawchuk, "A region matching motion estimation algorithm," *CVGIP: Image Underst.*, vol. 54, no. 2, pp. 275–288, 1991.

[22] K. Price and R. Reddy, "Matching segments of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 1, pp. 110–116, 1979.

[23] S. Sull and N. Ahuja, "Integrated 3–d analysis and analysis-guided synthesis of flight image sequences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 4, pp. 357–372, 1994.

[24] M. Pardas and P. Salembier, "3d morphological segmentation and motion estimation for image sequences," *Signal Process.*, vol. 38, pp. 31–43, 1994.

[25] C. Wang and K. Abe, "Region correspondence by inexact attributed planar graph matching," *Fifth Int. Conf. Comp. Vision*, pp. 440–447, 1995.

[26] C.I.E., "Colorimetry proposal for study of color spaces," *J. Opt. Soc. Am.*, vol. 64, no. 6, pp. 896–897, 1974.

[27] B. Julesz, "Experiments in the visual perception of texture," *Sci. Am.*, vol. 232, no. 1, pp. 34–43, 1975.

[28] B. Mandelbrot, *The Fractal Geometry of Nature*. New York: W.H. Freeman, 1977.

[29] G. Cantor, "Uber die Ausdehnung eines Satzes aus der Theorie der Trigonometrischen Reichen," *Mathematische Annalen*, vol. 5, no. 1, pp. 123–132, 1872.

[30] P. Fatou, "Sur les solutions uniformes de certaines equations fonctionelles," in *Comptes Rendus*, vol. 143, (Paris), pp. 546–548, 1906.

[31] A. Witkin, "Scale space filtering," in *Eighth Int. Joint Conf. on Art. Intell.*, (Karlsruhe, West Germany), pp. 1019–1022, Aug. 1983.

[32] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 7, pp. 629–639, 1990.

[33] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 674–693, 1989.

[34] R. Boomgaard and A. Smeulders, "Towards a morphological scale-space theory," in *Shape in Picture: Mathematical Descriptions of Shape in Grey-Level Images*, pp. 631–640, New York: Springer-Verlag, 1992.

[35] N. Ahuja, "A transform for detection of multiscale image structure," in *IEEE Conf. Comp. Vis. Pattern Recognit. '93*, (New York), pp. 780–781, June 1993.

[36] M. Tuceryan and A. Jain, "Texture segmentation using Voronoi polygons," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 2, pp. 211–216, 1990.

[37] B. Julesz, "Texton gradients: the texton theory revisited," *Biol. Cybern.*, vol. 54, no. 2, pp. 245–251, 1986.

[38] T. Rearick, "A texture analysis algorithm inspired by a theory of preattentive vision," *IEEE Conf. Comp. Vision Patt. Recog.*, pp. 312–317, 1985.

[39] T. Reed and H. Wechsler, "Segmentation of textured images and Gestalt organization using spatial/spatial-frequency representations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 1, pp. 1–12, 1990.

[40] D. Dunn, W. Higgins, and J. Wakeley, "Texture segmentation using 2–d Gabor elementary functions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 2, pp. 130–149, 1994.

[41] A. Bovik, M. Clark, and W. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 1, pp. 55–73, 1990.

[42] F. Farrokhnia and A. Jain, "Unsupervised texture segmentation using Gabor filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 12, pp. 1167–1186, 1989.

[43] M. Unser and M. Eden, "Multiresolution feature extraction and selection for texture segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 717–728, 1991.

[44] T. Lindeberg, "Shape from texture from a multiscale perspective," *Fourth Int. Conf. Comp. Vision*, pp. 683–691, 1993.

[45] W. Pratt, *Digital Image Processing*. New York: Wiley-Interscience, 1991.

[46] B. Krose, "Local structure analyzers as determinants of preattentive pattern discrimination," *Biol. Cybern.*, vol. 55, no. 3, pp. 289–298, 1987.

[47] R. Hu and M. Fahmy, "Texture segmentation based on a hierarchical Markov random field model," *Signal Process.*, vol. 26, no. 3, pp. 285–305, 1992.

[48] J. Francos and A. Meiri, "A compound Poisson-cliques random field model for texture singularities," *Proc. Int. Conf. Acoustics, Speech, Sig. Proc.*, pp. 2669–2672, 1991.

[49] H. Derin and H. Elliott, "Modeling and segmentation of noisy and textured images using Gibbs random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 1, pp. 39–55, 1987.

[50] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[51] A. Rosenfeld and A. Kak, *Digital Picture Processing*. New York: Academic Press, 1981.

[52] P. Perona and J. Malik, "Detecting and localizing edges composed of steps, peaks, and roofs," *Third Int. Conf. on Computer Vision*, 1990.

[53] V. Nalwa and T. Binford, "On detecting edges," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, no. 7, pp. 699–714, 1986.

[54] D. Marr and E. Hildreth, "A theory of edge detection," in *Proc. Royal Society London*, (London), pp. 187–217, 1980.

[55] L. Davis, "A survey of edge detection techniques," *Comput. Graph. Image Process.*, vol. 1, no. 2, pp. 248–270, 1975.

[56] V. Berzins, "Accuracy of Laplacian edge detectors," *Comput. Vis. Graph. Image Process.*, vol. 26, no. 2, pp. 195–210, 1984.

[57] R. Haralick, "Digital step edges from zero-crossings of second directional derivative," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 6, no. 1, pp. 58–68, 1984.

[58] J. Aggarwal and N. Nandhakumar, "On the computation of motion from sequences of images- a review," *Proc. IEEE*, vol. 76, no. 8, pp. 917–935, 1988.

[59] J. Barron, D. Fleet, S. Beauchemin, and T. Burkitt, "Performance of optical flow techniques," in *Comp. Vision Patt. Recog. '92*, (Champaign), pp. 236–242, 1992.

[60] T. Huang and R. Tsai, "Image sequence analysis: Motion estimation," in *Image Sequence Analysis* (T. Huang, ed.), New York: Springer-Verlag, 1981.

[61] F. Heitz and P. Bouthemy, "Multimodal estimation of discontinuous optical flow using Markov random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 12, pp. 1217–1232, 1993.

[62] J. Konrad and E. Dubois, "Bayesian estimation of motion vector fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 9, pp. 910–927, 1992.

[63] A. Verri and T. Poggio, "Motion field and optical flow: qualitative properties," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 5, pp. 490–498, 1989.

[64] C. Fuh, P. Maragos, and L. Vincent, "Visual motion correspondence by region-based approaches," in *Proc. ACCV '93*, (Osaka, Japan), pp. 784–789, 1993.

VITA

Mark D. Tabb received the B.S. degree in Electrical Engineering from Cornell University in 1991, the M.S. degree in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign in 1993, and will receive the Ph.D. degree in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign in the spring of 1996. He was a research assistant at the Beckman Institute for Advanced Science and Technology between 1991 and 1996.