UTILIZATION OF IMPROVED RECURSIVE-SHORTEST-SPANNING-TREE METHOD FOR VIDEO OBJECT SEGMENTATION

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Ertem Tuncel

11 August 1997

UTILIZATION OF IMPROVED RECURSIVE-SHORTEST-SPANNING-TREE METHOD FOR VIDEO OBJECT SEGMENTATION

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Ertem Tuncel

11 August 1997

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

Prof. Dr. Levent Onural(Supervisor)

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

Assoc. Prof. Dr. M. İrşadi Aksun

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

Assoc. Prof. Dr. Ömer Morgül

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

UTILIZATION OF IMPROVED
RECURSIVE-SHORTEST-SPANNING-TREE METHOD FOR VIDEO
OBJECT SEGMENTATION

Ertem Tuncel

M.S. in Electrical and Electronics Engineering

Supervisor: Prof. Dr. Levent Onural

11 August 1997

Emerging standards MPEG-4 and MPEG-7 do not standardize the video object segmentation tools, although their performance depends on them. There are a lot of still image segmentation algorithms in the literature, like clustering, split-and-merge, region merging, etc. One of these methods, namely the recursive shortest spanning tree (RSST) method, is improved so that a still image is approximated as a piecewise planar function, and well-approximated areas on the image are extracted as regions. A novel video object segmentation algorithm, which takes the previously estimated 2-D dense motion vector field as input, and uses this improved RSST method to approximate each component of the motion vector field as a piecewise planar function, is proposed. The algorithm is successful in locating 3-D planar objects in the scene correctly, with acceptable accuracy at the boundaries. Unlike the existing algorithms in the literature, the proposed algorithm is fast, parameter-free and requires no initial guess about the segmentation result. Moreover, it is a hierarchical scheme which gives finest to coarsest segmentation results. The proposed algorithm is inserted into the current version of the emerging "Analysis Model (AM)" of the Europan COST211^{ter} project, and it is observed that the current AM is outperformed.

Keywords : Video object segmentation, recursive shortest spanning tree method, 2-D motion estimation, hierarchical segmentation, MPEG-4, MPEG-7.

ÖZET

VİDEO NESNE BÖLÜTLEMESİ İÇİN GELİŞTİRİLMİŞ ÖZYİNELEMELİ-EN-KISA-AĞAÇ YÖNTEMİ KULLANIMI

Ertem Tuncel

Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans

Tez Yöneticisi: Prof. Dr. Levent Onural

11 Ağustos 1997

Geliştirilmekte olan MPEG-4 ve MPEG-7 standartları, video nesne bölütleme metodu olarak neyin kullanılacağını belirlememektedirler. Oysa bu standartların performansları, kullanılacak metodun başarısına doğrudan bağlıdır. Literatürde, gruplama, bölüp-birleştirme, bölge birleştirme gibi bir çok imge bölütleme metodları vardır. Bu metodlardan özyinelemeli-en-kısa-ağaç (RSST) metodu, bir imgeyi parçalı düzlemsel fonksiyon olarak yaklaştıracak ve iyi yaklaştırılmış alanları bölütleme bölgesi olarak verecek biçimde iyileştirilmiştir. Önceden kestirilmiş 2-B sık hareket vektörlerini alıp, iyileştirilen RSST metodunu bu vektörlerin herbir bileşeni üzerinde kullanan yeni bir video nesne bölütleme yöntemi sunulmaktadır. Bu metod, görüş alanındaki 3-B düzlemsel nesnelerin sınırlarını yeterince doğru bulmaktadır. Aynı metod, literatürde bulunan diger benzer metodlarda olmayan bazı iyi özelliklere de sahiptir; örneğin hızlıdır ve parametre veya baslangıç tahmini istememektedir. Ayrıca, hiyerarşik, yani kabadan ayrıntılıya dogru birçok bölütleme sonucunu birden sunmaktadır. Önerilen metod, Avrupa COST211^{ter} projesinde geliştirilen "Analiz Modeli (AM)"nin şu andaki haline sokulmuş, ve AM'nin daha başarılı olmasını sağlamıştır.

Anahtar kelimeler : Video nesne bölütlemesi, özyinelemeli en kısa ağac metodu, 2-B hareket kestirimi, hiyerarşik bölütleme, MPEG-4, MPEG-7.

ACKNOWLEDGMENTS

I would like to express my deep gratitude to my supervisor Prof. Dr. Levent Onural for his guidance, suggestions, and invaluable encouragement throughout the development of this thesis.

And special thanks to Aydın, Tunç, Tolga, for their assistance, Kürşat, Kubilay, for their friendship, and my wife, Sūreyya, for everything.

TABLE OF CONTENTS

LIST OF FIGURES

To My Shelf

Chapter 1

Introduction

1.1 The Segmentation Problem

Throughout the history of *image and video processing*, segmentation has always been a challenging problem [1], [2], [3], [4].

Evaluating the performance of a segmentation tool is difficult since different applications may need different segmentation results for the same image or video. For example, in an object-based compression algorithm, the segmentation step is followed by the coding step where the internal textures of the objects are described. In that case, a given segmentation result is successful if the textures of the objects can *easily* be described, hence can *efficiently* be coded. In other words, for this case the performance of two given segmentation results can be compared either by comparing the reconstructed image qualities for a fixed bit rate, or by comparing the bit rates for a fixed reconstructed image quality. However, for an application where the user is allowed to interact

with the video by choosing and manipulating objects, the previously done segmentation is successful only if the extracted objects have *semantic* meanings, i.e., they must correspond to objects like a car, a woman, a ship, etc.

Note that, the criterion of success may conflict for the two kind of applications mentioned above. For example, the shirt of a woman in the scene may possess some different textured parts. For a coding application, a successful segmentation should extract those parts as distinct objects, whereas for a user-interactive service application, it should extract the shirt as a single compact region.

As a consequence of this fact, there is no *universal* numeric performance evaluator. Hence, the evaluations are usually subjective.

1.2 Increasing Focus on Object Segmentation

Recent trends in the digital video world are led by two emerging standards, namely the MPEG-4 and the MPEG-7 [5], [6].

MPEG-4 is a standard for object-based multimedia services. The user is allowed to interact with the video, by choosing a specific object (object-based interactivity). Then, the decoder is capable of displaying only the chosen object (object scalability), or increasing/decreasing the spatial or temporal resolution of it (spatial or temporal scalability).

In MPEG-4, the video is assumed to be composed of some *video object planes* (VOP) which correspond to distinct objects in the scene. However,

automatic (or at least, semi-automatic) extraction of the VOPs from a given input sequence is not standardized, and is still an open issue.

MPEG-7, or more formally, the "Multimedia Content Description Interface" [6], standardizes the *description* of various types of multimedia information. This description shall be associated with the content itself, to allow fast and efficient searching for material in which the user is interested.

In MPEG-7, a standardized description of different information types can take on many forms, and can exist at a number of semantic levels. Visual material can be described in low abstraction levels in terms of size, shape, texture, color; or in higher (semantic) levels by a sentence like "A man with a green hat standing on his right leg in a yellow room;" or in levels somewhere in between.

Although MPEG-7 does not standardize the *feature extraction* that has to be done before the description step, its success depends heavily on it. A fully or semi-automatic segmentation of video sequences would be a good initial step for extraction of features like color, shape, texture, etc.

There is also another emerging standard, namely the JPEG 2000, for object-based still image compression whose performance obviously depends on successful segmentation in the sense mentioned in the previous section for coding applications.

In short, all these emerging standards somehow involve a still image or video segmentation tools, and hence there is an increasing trend all around the image and video processing world for the segmentation problem.

1.3 Scope and Outline of the Dissertation

The outline of the dissertation is as follows:

In Chapter 2, a survey on various still image segmentation methods in the literature is given. These methods include the early and classical methods as well as the newly emerging morphological methods.

In Chapter 3, a new understanding of one of the methods mentioned in Chapter 2, namely the recursive shortest spanning tree (RSST) method, is given. Then, an improvement to RSST, based on this new understanding, is proposed; RSST is extended to an algorithm which fits *surfaces* to the texture inside the regions in a controlled manner. Experimental justification for this novel algorithm is also given.

Mathematical formulations for geometric image formation, and for projection of motion field of the 3-D objects onto the 2-D image plane are given as introduction at the beginning of Chapter 4. These are well-known formulations, and are given for completeness. Then, video object segmentation methods which are popular in the literature are introduced. These methods are based on 6 or 8-parameter projected motion field models. Equivalently, they can be seen as surface-fitting methods because of the fact that the extracted parameters define a surface for each motion vector component. Utilization of the improved RSST for the segmentation of the estimated motion field is proposed at the end of Chapter 4, and it is observed that the experimental results are promising. Improved RSST is advantageous over the existing algorithms since it is fast and free of *ad hoc* weights or initial values for parameters.

In Chapter 5, the current version of the "Analysis Model" (AM) developed by the Europan Cost211^{ter} project is introduced. The AM contains two segmentation modules using the RSST algorithm; one segments the color information, and the other segments the estimated motion information. The replacement of the so called motion segmentation module by the one proposed in Chapter 4 is experimented, and it is observed that the current version of the AM cannot handle planar objects making 3-D motion, whereas the experimented one does.

Finally, Chapter 6 concludes the dissertation, by summarizing the contributions.

Chapter 2

Image Segmentation

The objective of image segmentation is to partition a still image into connected and disjoint regions, where the resultant regions are homogeneous enough, and adjacent regions have enough contrast, in terms of the features of pixels extracted from the image. Examples of features are; pixel gray level, pixel RGB color, range of the pixel from the camera, position of the pixel, local covariance matrix, etc.

The objective of image segmentation can be described more conveniently by the following conditions: [1], [2]

(a) $\cup_{i=1}^{n} R_i = R$,

(b) R_i is connected for all i,

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

(d) $P(R_i) = TRUE$ for all i,

(e) $P(R_i \cup R_j) = FALSE$ for $i \neq j$ and R_i adjacent to R_j

where R is the entire image, n is the number of regions, $P(R)$ is a boolean operator on regions, and is called the homogeneity predicate.

Usually, the homogeneity predicate $P(R)$ is evaluated by thresholding another function, $h(R)$, which maps regions to real numbers. A simple example of $h(R)$ is the variance of the gray level values of the pixels inside the region, if the only feature used is the gray level. If the conditions (d) or (e) are violated, the image is said to be *undersegmented* or *oversegmented*, respectively.

In the following sections, various image segmentation techniques are described.

2.1 Classical Methods

The methods described in this section approximate the features of interest as piecewise constant functions on the image plane, and try to extract those constant-valued regions. The approximated features are called *synthesized features.*

2.1.1 Clustering

The extracted feature vectors of the pixels inside a homogeneous region are expected to form groups, known as clusters, in the feature space. If the features are scalar, such as pixel intensities, clustering reduces to a simpler method known in the literature as *thresholding*, [3] i.e., the problem is to find $K - 1$

thresholds that define the decision boundaries in the 1-D feature space, where K is the number of clusters.

A standard procedure for clustering is to run the iterative method, known as the K-means algorithm [7]. The objective of the K-means algorithm is to minimize

$$D = \sum_{i=1}^{K} \sum_{(x,y)\in R_i} \|\mathbf{s}(x,y) - \mu_i\|^2 \tag{2.1}$$

where $\mathbf{s}(x,y)$ is the feature image, μ_i is the average (synthesized) feature vector of region R_i, and $\|\|$ is the L_2 norm.

The algorithm is as follows:

1. Choose K initial cluster means $\mu_1, \mu_2, \ldots, \mu_K$

2. Assign each pixel to one of the K clusters according to

$$\|\mathbf{s}(x,y) - \mu_i\| < \|\mathbf{s}(x,y) - \mu_j\| \Rightarrow (x,y) \in R_i$$

   where $j \neq i$

3. Update the cluster means according to

$$\mu_i = \frac{1}{N_i} \sum_{(x,y)\in R_i} \mathbf{s}(x,y)$$

   where N_i is the number of pixels assigned to R_i.

4. If all μ_i's are converged to fixed points, the algorithm is converged, so terminate. Otherwise, go to step 2.

Note that the distortion D is decreased in both steps 2 and 3. So, the algorithm is guaranteed to converge at least to a local minimum.

The biggest problem is the determination of K. One solution is to iterate on K, and evaluate the clustering quality measures, such as *within-* and *between-cluster* scatter measures [7], which can be used as tests for objectives (d) and (e). There is also the problem of determining initial cluster means.

Another problem is that the resultant clusters may not correspond to connected regions. The solution can be the inclusion of pixel coordinates into the feature vectors, and declare every connected component of a cluster as a distinct region. However, Bayesian methods described in the next section solves this problem more conveniently, although they resemble clustering, too much.

2.1.2 Bayesian Methods

The *a priori* probability model for the segmentation label field is assumed to be a Gibbs random field (GRF), [8], [4], which expresses the expectations about the spatial properties of the segmentation, i.e., the GRF assigns higher probabilities to the segmentation fields having connected regions.

The feature image is explicitly assumed as the summation of two parts; one is a piecewise constant function, and the other is a Gaussian white noise with zero mean and variance σ^2. The segmentation is achieved by maximizing the *a posteriori* probability of the segmentation field, given the observed feature image. The mathematical formulation is as follows [8]:

The segmentation label field $\mathbf{Z}(x, y)$ is modeled by

$$P(\mathbf{Z} = z) \propto \exp\{-U(z)\}$$

where $U(z)$ is the Gibbs potential and is defined by

$$U(z) = \sum_{C \in \mathbf{C}} V_C(z) \, .$$

Here $\mathbf{C}$ is the set of all cliques, and V_C is the individual clique potential whose value depends only on $z(x, y)$ where $(x, y) \in C$. Spatial connectivity of regions can be imposed by assigning low values to $V_C(z)$, if $z(x, y)$ is constant for all $(x, y) \in C$, and high values otherwise.

According to the above assumption about the formation of the image, the conditional probability of the observed feature image $\mathbf{S}$, given $\mathbf{Z}$ is modeled by

$$P(\mathbf{S} = s|\mathbf{Z} = z) \propto \exp\{\frac{-1}{2\sigma^2}\left[\sum_{i=1}^{K} \sum_{(x,y)\in R_i} \|s(x, y) - \mu_i\|^2\right]\} \, .$$

The *a posteriori* probability can be manipulated using the Bayes rule:

$$P(\mathbf{Z} = z|\mathbf{S} = s) = \frac{P(\mathbf{S} = s|\mathbf{Z} = z)P(\mathbf{Z} = z)}{P(\mathbf{S} = s)} \, .$$

Then, maximizing $P(\mathbf{Z} = z|\mathbf{S} = s)$ is equivalent to minimizing

$$D' = \sum_{i=1}^{K} \sum_{(x,y)\in R_i} \|\mathbf{s}(x, y) - \mu_i\|^2 + \lambda \sum_{C \in \mathbf{C}} V_C(z) \tag{2.2}$$

with respect to the segmentation mask $z(x, y)$.

Note the similarity with the clustering scheme. The exhaustive searching of the global minimum for D' is prohibited because of the excessive number of possibilities for $z(x, y)$. So, generally a suboptimal method of *iterated conditional modes* (ICM) is used to reduce the complexity. Another problem is the determination of λ [8].

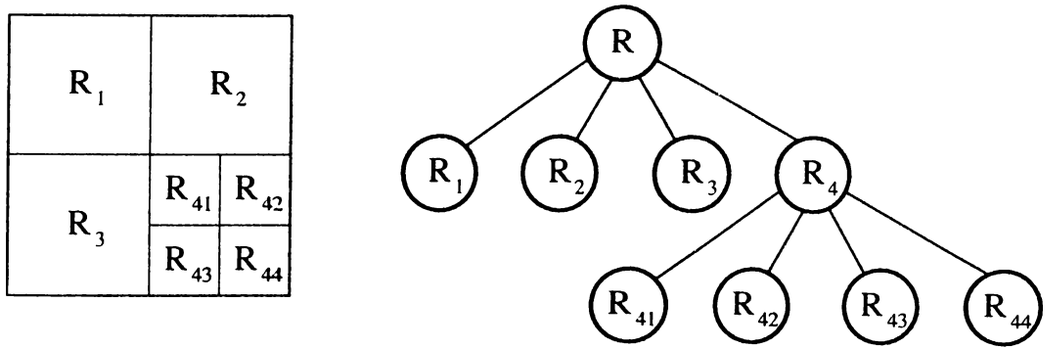

Figure 2.1: Recursively splitted image and the corresponding quadtree.

2.1.3 Split-and-Merge

The image plane is divided successively into quadrants when needed until for any region R_i, $P(R_i) = TRUE$. More clearly, every subquadrant is divided into four subquadrants if $P(R) = FALSE$. A *quadtree* is formed by this successive *splitting*, as shown in Figure 2.1, where every region corresponds to a *leaf* node of the tree.

The final partition at the end of this splitting satisfies all segmentation objectives except for the one stated in (e). To remedy this, *merging* is also allowed at intermediate stages, whenever (e) is violated, i.e., whenever the merging of two adjacent regions yields a homogeneous region.

The procedure can be summarized as follows[1]:

1. If for any region R_i, $P(R_i) = FALSE$, then **split** R_i into four subquadrants.

2. If for any adjacent regions R_i and R_j, $P(R_i \cup R_j) = TRUE$, **merge** them.

3. If no further splitting or merging is possible, stop. Else go to step 1.

Note that, each region corresponds to a *collection* of some leaves, and when two adjacent regions are merged, the corresponding collections are concatenated. After the merging, there is no need for the resultant region to be split any more, since it satisfy the objective (d). So, at any time, split regions are necessarily the ones corresponding to a leaf node, i.e., a collection with a single member.

Split-and-merge method does not suffer from predetermination of number of regions, or any other constants. However, the main drawback is the artificial blocking effects on the resultant region boundaries.

2.1.4 Seeded Region Growing

The algorithm [9] has two modes; supervised and unsupervised. In the supervised mode, the user declares some *seeds* as an incomplete segmentation. Then the algorithm is to grow those seeds until the segmentation objective (a), i.e., the condition for a complete labeling, is satisfied.

In the unsupervised mode, another algorithm may give the seeds as an input to the supervised mode algorithm. Examples for extracting seeds from a feature image are histogram mode (or cluster mean) extraction [1], and, flat zones extraction using morphological filtering [10], [11], [12], [13].

Once the seeds are ready, priorities are assigned to all pixels in the image that are not yet assigned to any region, but adjacent to at least a growing one. The lower the *distance* between the pixel and its adjacent region, the higher

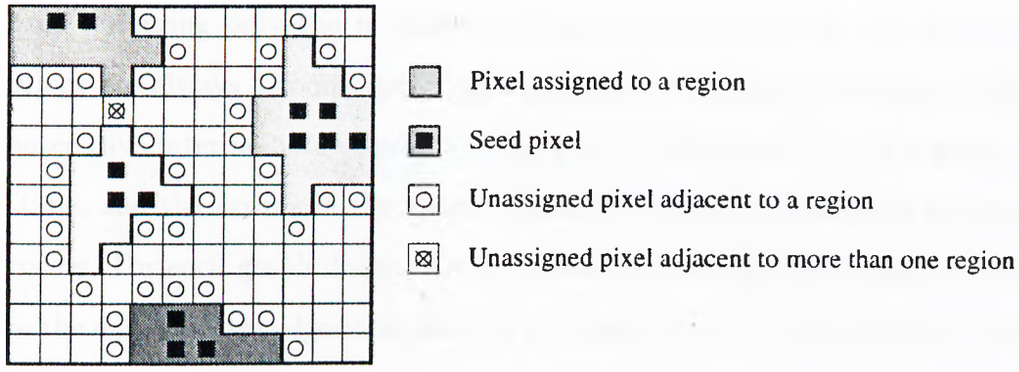

Figure 2.2: Region Growing at an intermediate stage.

the priority assigned to it. At each step, the pixel with the highest priority is merged to the *closest* region (if there is more than one adjacent region). The distance of the pixel to an adjacent region is evaluated by calculating the L_2 distance between the feature vector of that pixel and the average feature vector of the region. Figure 2.2 shows an intermediate stage of growing.

Another remedy for the unsupervised mode is to grow one region at a time. In that procedure, a pixel is merged to the only growing region, if it is not yet assigned to any region, and merging of that pixel will not cause the region to violate the segmentation objective (d). If it violates, it is declared as a new seed and is grown after the growing of the current region is finished.

A final merging of some adjacent regions may be necessary in order to obey the objective (e).

Obviously, there is the disadvantage of the determination of seeds.

them. Merging only the neighboring regions guarantees that the resultant regions are always 4-connected. Figure 2.3 shows examples of merging at two consecutive intermediate stages. The merging is performed for two regions at a time, and the arrows in the figure indicate the flow of the ongoing merging process. For each graph drawn, the two nodes covered by the rectangular box are the ones that the algorithm decided to merge, i.e., that are tied by the link with minimum weight in the whole graph. After the merging, the link between these nodes should be deleted, because it becomes useless. That link is drawn thicker than the other links in the figure. The joint node is to be assigned a weight, which is the average of the feature vectors of the pixels inside the corresponding merged region. The weights of the links between the joint node and other nodes are to be updated, because as seen from Equation 2.3, the link weights depend on the weights of the corresponding nodes one of which is the joint node. These links are indicated by three parallel lines in the figure. Finally, some redundant links may occur after the merging, i.e., two distinct links may come out to tie the same pair of nodes after the chosen pair of nodes are merged. An example of this phenomenon occurs on the second graph, where the link which will be redundant after the merging is indicated by a scissors. That redundant link should be deleted before the next merging.

Repeating this procedure, the number of regions can be reduced down to one. The removed links construct a spanning tree of the original graph [14]. By noting the order in which the links are eliminated, the image can be segmented into an arbitrary number of regions, say K, by using the last removed $K-1$ links.

Opposed to the other algorithms mentioned in previous subsections, RSST

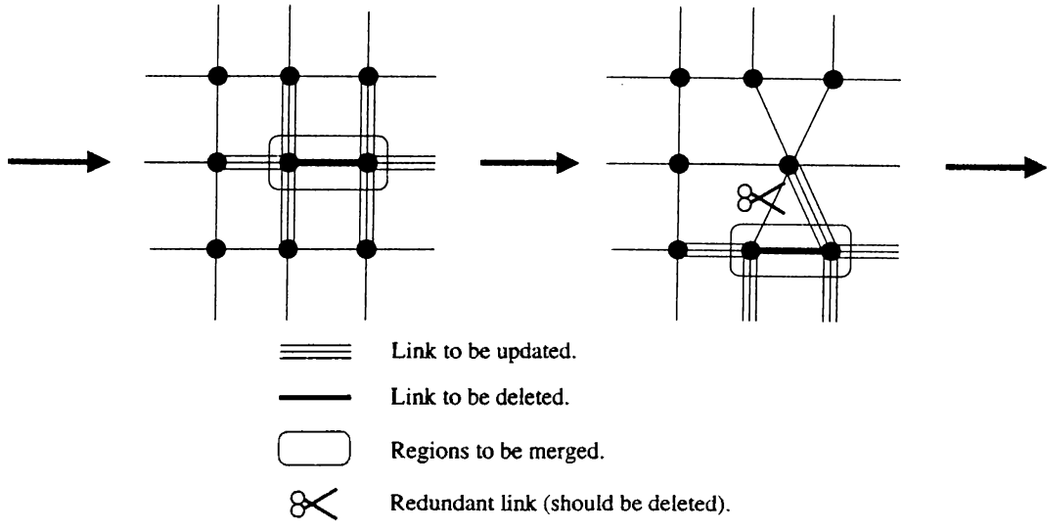

Figure 2.3: RSST at an intermediate merging stage.

has the advantage of not imposing any external constraints on the image. Furthermore, it has a hierarchical structure which permits simple control over the number of regions.

2.2 Morphological Methods

Mathematical morphology [15] is an efficient tool for image analysis, and especially for image segmentation. The reason is that its highly nonlinear operations and/or filters directly operate on size, shape, contrast, connectivity, etc. Moreover, morphological transformations can be efficiently implemented in both software and hardware.

In this section, first the basic morphological operators are described, and then the *filtering by reconstruction* and the *watershed* algorithm, which are together the heart of the morphological segmentation, are defined. Finally,

some variations of the watershed is discussed.

2.2.1 Basic Morphological Operators

A large number of morphological tools relies on two basic operators known as *erosion* and *dilation*. If $I(x,y)$ denotes an input signal and $B(x,y)$ is the so-called *structuring element* of the operator, the erosion $\epsilon_B(I)$ and the dilation $\delta_B(I)$ are defined as follows [15]:

$$\epsilon_B(I) = \min_{(x',y')\in D_B} I(x+x', x+y') - B(x',y')$$

$$\delta_B(I) = \max_{(x',y')\in D_B} I(x-x', y-y') + B(x',y')$$

where D_B is the domain of the structuring element B, and $I(x,y)$ is assumed to be $-\infty$ wherever it is undefined. When the structuring element is zero throughout its domain, then it is called to be *flat* and the erosion and the dilation reduce to

$$\epsilon_B(I) = \min_{(x',y')\in D_B} I(x+x', x+y')$$

$$\delta_B(I) = \max_{(x',y')\in D_B} I(x-x', y-y')$$

Now, the morphological filters such as the *morphological opening* (γ_B) and *closing* (ϕ_B) can be defined in terms of erosion and dilation:

$$\gamma_B(I) = \delta_B(\epsilon_B(I))$$

$$\phi_B(I) = \epsilon_B(\delta_B(I)).$$

2.2.2 Filtering by Reconstruction

Reconstruction filters are a subfamily of a wider class of morphological filters, called the *connected operators*, [12]. Connected operators fundamentally have the property of interacting with the signal by producing *flat zones*. A flat zone is a connected component of the image where the gray-level value is constant (Note that a flat zone may be a single point). When the image is passed from a filtering by reconstruction process, its flat zones are either *preserved* or *merged*, but never *split*. In other words, the contours of the flat regions are either preserved or removed by the filter; introducing new contours is not allowed.

In filtering by reconstruction terminology, there is always a *marker* image $M(x, y)$, extracted from the original image $I(x, y)$ which is taken as a reference during the reconstruction process. The reconstruction process uses the operators called the *geodesic dilation* and *geodesic erosion* of size one, which are defined by

$$\delta_B^1(M, I) = \min\{\delta_B(M), I\}$$

and

$$\epsilon_B^1(M, I) = \max\{\epsilon_B(M), I\}$$

respectively [12]. Usually B is equal to zero for a 3x3 box whose center is origin, and undefined elsewhere. From this point on, if the structuring element is omitted, it will be taken as this B, unless otherwise stated.

Geodesic dilations and erosions of arbitrary size are defined by recursion, i.e., $\delta^n(M, I)$ means $\delta^1(\delta^{n-1}(M, I), I)$. Based on this definition, *reconstruction*

by dilation or *erosion* can be defined by

$$\gamma^{rec}(M, I) = \delta^{\infty}(M, I) \tag{2.4}$$

and

$$\phi^{rec}(M, I) = \epsilon^{\infty}(M, I) \tag{2.5}$$

respectively [12].

The most popular filter by reconstruction is the *opening by reconstruction* filter $\gamma^{rec}(\epsilon_A(I), I)$, i.e., the marker image is extracted from the original by eroding it with an arbitrary structuring element $A(x, y)$. Of course, by duality, a closing by reconstruction can be defined: $\phi^{rec}(\delta_A(I), I)$. These filters have a shape/size-oriented simplification effect on the image but preserve the contour information.

Other examples of filters by reconstruction are the *h-maxima*, and its dual *h-minima* operators, which are used for contrast-oriented simplification. They can be defined in terms of reconstruction by dilation or erosion. If h is a constant,

$$h_{max}(I) = \gamma^{rec}(I - h, I)$$

$$h_{min}(I) = \phi^{rec}(I + h, I).$$

An efficient implementation method for the reconstruction process can be found in [16]. In the following sections, the use of filtering by reconstruction for segmentation purposes is described.

2.2.3 The Watershed Algorithm

The classical morphological approach to segmentation relies on the *watershed* algorithm [17] applied on the *morphological gradient* image $G(x, y)$:

$$G = \delta(I) - \epsilon(I).$$

Note that the gradient image G is non-negative everywhere. Its amplitude is high around the edges, and low around smooth regions in the original image. This means that, by thresholding the gradient image, a good edge detection is achieved.

However, edge detection does not complete the segmentation process since the edges do not necessarily form closed regions. The remedy is the watershed algorithm [17] which can be seen as a post-processing tool for the completion of detected edges to closed curves.

The watershed algorithm partitions the morphological gradient image G into *catchment basins* whose dividing lines are called the *watershed lines*, by flooding the surface of the image from its regional minima[1]. Starting from the global minimum, the water progressively fills up the catchment basins. When the water level reaches the altitude of other minima, these minima start to be active, and the flooding process also originates from them. When the water coming from two different minima would merge, an imaginary dam is built to prevent any mixing of water. The procedure is ended when the water level is higher than the global maximum. In this case, each minimum is surrounded by water, that is its catchment basin, and a dam delimiting its border, that is its

[1] A regional minima is a flat zone whose value is lower than its surrounding flat zones.

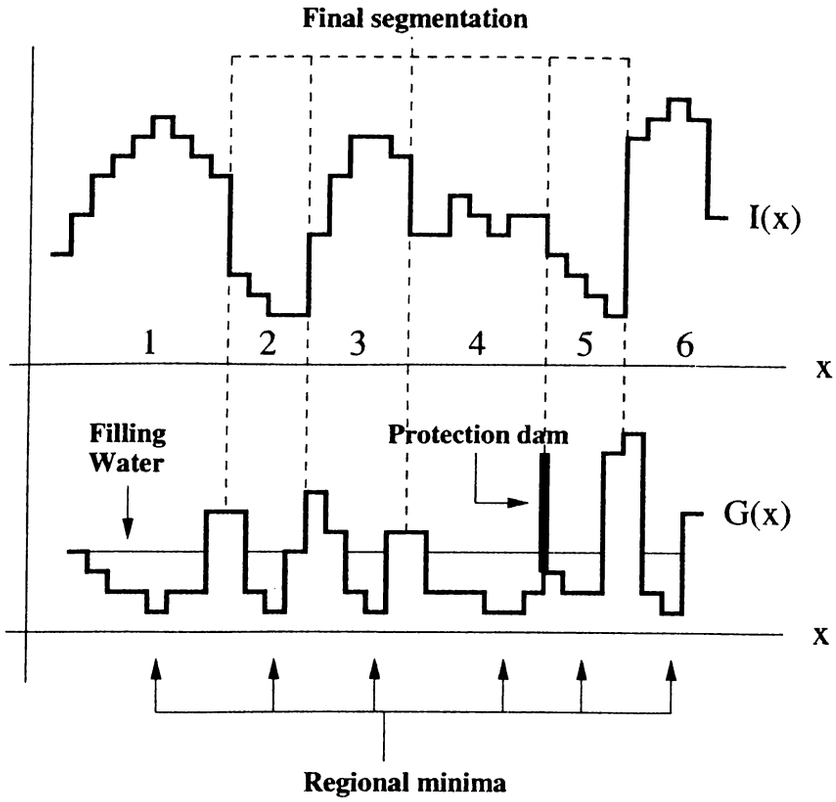

Figure 2.4: A 1-D discrete signal $I(x)$, and its gradient $G(x)$ applied to the watershed algorithm. The water level is at an intermediate stage. Final segmentation result is also shown.

watershed line. See Figure 2.4, where an intermediate stage of the algorithm for a 1-D signal is shown.

The catchment basins at the end of the algorithm constitute a segmentation for the original image $I(x, y)$. Here, the homogeneity predicate P, defining the segmentation objectives (d) and (e) is given by

$P(R) = TRUE \Leftrightarrow R$ covers exactly **one** regional minimum of the gradient image G.

The flooding process shows many similarities with the seeded region growing algorithm. The regional minima can be seen as the seeds of the growing process, priorities should be assigned to any pixels that are not yet assigned to any region, but adjacent to at least one (i.e., the lower the amplitude of the pixel at the gradient image, the higher its priority), etc.

Watershed on Size/Contrast-Filtered Gradient Image

The main problem with the watershed algorithm is its very sensitive nature to observation noise, because of being an edge-based paradigm. For example, if the image is corrupted by the so called *salt and pepper* noise, every salt or pepper grain will be a separate region at the end of the growing process.

One remedy to this weakness is to apply a filtering by reconstruction process to the gradient image G, before the application of watershed.

Application of a *closing by reconstruction* filter $\phi^{rec}(\delta_A(G), G)$ eliminates, or fills in the regional minima whose shape does not cover the structuring element A. $A(x, y)$ is usually chosen as zero in an $N \times N$ block, and undefined elsewhere.

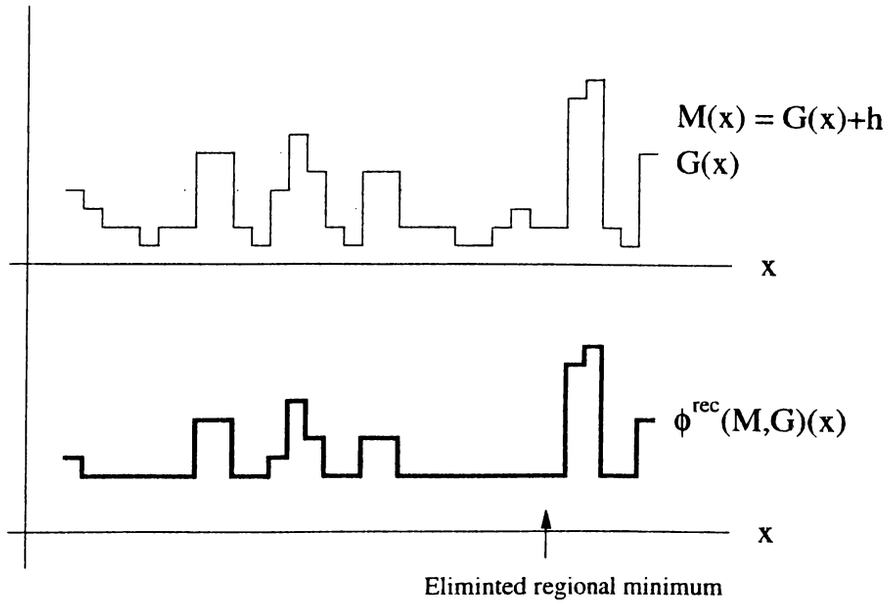

Figure 2.5: The 1-D gradient signal $G(x)$, the extracted marker signal $M(x)$ and the output of the *h-minima* filter.

On the other hand, application of a *h-minima* filter $\phi^{rec}(G+h, G)$ fills in the regional minima whose depth (or contrast) is lower than h. Figures 2.5 and 2.6 illustrate the effects of these filters to the regional minima of the gradient image G.

A new connected filter $\beta(\cdot, A, h)$ is defined by

$$\beta(G, A, h) = \min[\phi^{rec}\{\delta_A(G), G\}, \phi^{rec}\{G+h, G\}]$$

which eliminates the small **and** low-contrast regional minima. In other words only large enough **or** deep enough regional minima survive and are used as seeds of the watershed process, with the hope that the eliminated regional minima had owed their existence only to noise .

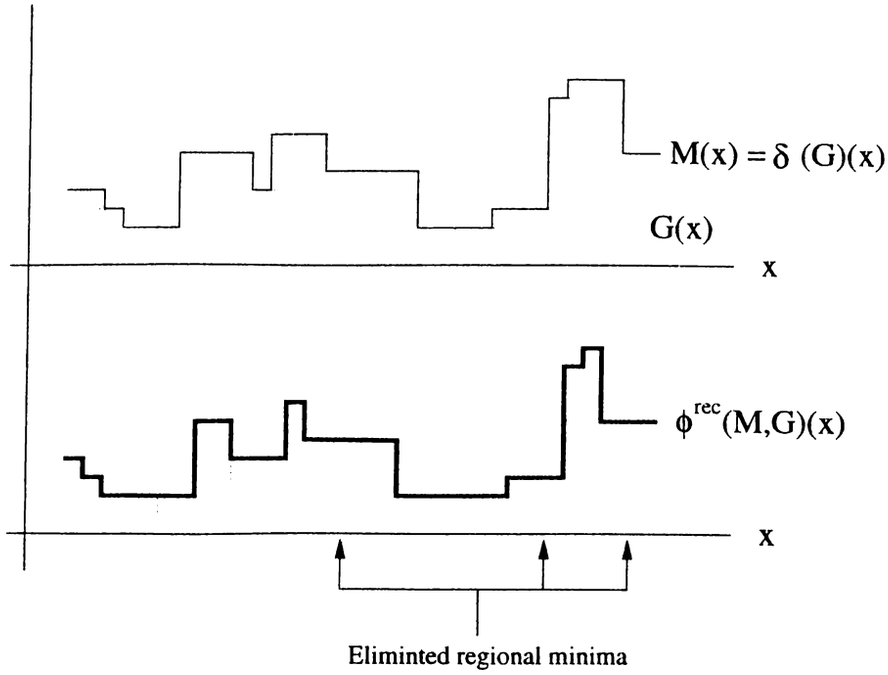

Figure 2.6: The 1-D gradient signal $G(x)$, the extracted marker signal $M(x)$ for a structuring element of size 3, and the output of the *closing by reconstruction* filter.

Watershed on Size/Contrast-Filtered Original Image

Another problem with the watershed algorithm is that by taking the morphological gradient of the original image, some of the information is lost. For example, the high-amplitude curves in the gradient image, corresponding to the edges in the original image, are two-pixel wide, which brings some randomness as to where the protection dams will be located during the flooding process.

The gradient image was used in the original procedure with the hope that each regional minimum in the gradient image would correspond to a local extremum in the original image. Note that the local extrema mentioned include regional minima and maxima, and wide enough flat zones.

As an alternative of flooding the gradient image, a modified watershed algorithm applied to the simplified original image is defined in [10], which takes the local extrema of the simplified image as seeds of the growing process. The simplification is achieved by the application of the size/contrast filter $\beta(\cdot, A, h)$ defined above, followed by its dual $\alpha(\cdot, A, h)$:

$$\alpha(I, A, h) = \max[\gamma^{rec}\{\epsilon_A(I), I\}, \gamma^{rec}\{I - h, I\}].$$

As for the watershed process, the seeds are defined by the regional extrema plus the flat regions wider than a predetermined size, in the simplified image. The growing process is identical to the one described in seeded region growing.

Chapter 3

Simultaneous Segmentation and Reconstruction

In object-based image coding algorithms [18], the segmentation step is followed by a lossy compression step. The coding scheme usually approximates the texture inside of the regions in terms of some predefined 2-D basis functions. The efficiency of the coding algorithm heavily depends on the performance of the segmentation step, i.e., if the image is undersegmented, the reconstructed image quality deteriorates, or if it is oversegmented, the bit rate of the coder is increased, compared to the case of successful segmentation.

So, a good homogeneity predicate candidate P is the so-called *goodness-of-fit* criterion, that is, the measure of how well the approximation in terms of the 2-D basis functions **fit** the original image. The usage of goodness-of-fit criterion leads to the concept of *simultaneous segmentation and reconstruction (SSR)* of the image [19], [20], [21], [22], that is, controlling the segmentation

scheme by the quality of the reconstructed image. Of course, the bit rate is an implicit control thanks to the objective (e), for if it were not, the trivial segmentation in which every individual pixel is a distinct region would be the output.

Even if the segmentation scheme is not followed by coding, that procedure is still useful, since the statement "easily codable" is equivalent to "easily describable", which is what is sought by segmentation algorithms. (In this case, the reconstructed image becomes a by-product of the scheme.) This chapter aims to justify this by comparing the performance of the RSST algorithm, and some proposed algorithms based on RSST and the concept of goodness-of-fit, for some test images.

3.1 Segmentation through Surface Fitting

A gray-level image can be viewed as a 3-D surface $z = I(x, y)$ and is assumed to be piecewise smooth. The aim of the segmentation algorithm is to extract those pieces. For this purpose, the 2-D basis functions mentioned above are chosen as low-ordered bivariate polynomials $x^m y^n$ in [19], [20], and [21], because they are smooth, easy to handle, and defined everywhere.

Once the basis functions, and a distortion measure between the original and the approximated images, are determined, it is straightforward to find the approximated texture inside the regions if the regions are known. However, the very aim is the determination of the regions, and this leads to a so-called "chicken & egg" problem. Different solutions are proposed previously, for example, in [19], first some seeds are extracted by searching surface curvature

signs, and then they are refined and grown. In [21], a multiresolution approach, where at each resolution inherits its initial segmentation from the upper level and refines it, is presented.

In the following section, RSST is treated as a surface fitting method, and based on this, the algorithm is improved by setting proper distance measures between the nodes. Only 1, x, y, and at most xy is used as basis functions, that is, the image surface is tried to be approximated by piecewise planar, or at most bilinear surfaces for the sake of computational simplicity.

3.2 RSST as a Surface Fitting Method

In the RSST algorithm, every node i holds some parameters in its memory. These parameters are namely the coordinates of the pixels belonging to the represented region and the average, μ_i, of the feature vectors of that pixels.

If an approximated feature image is to be constructed by assigning a constant vector ν_i for every pixel inside region R_i, and if the approximation error is measured by the squared error as in (2.1), namely

$$D = \sum_{i=1}^{K} \sum_{(x,y)\in R_i} \|\mathbf{s}(x,y) - \nu_i\|^2,$$

then it is a well known fact that $\nu_i = \mu_i$ minimizes D. So, RSST does its best in terms of approximation quality by holding the average, if squared error is considered, and if the regions are known.

At every intermediate stage, RSST is to merge two regions and to assign a new average to the merged region. A good strategy for choosing the regions to

be merged among all adjacent pairs is the following:

- For all adjacent pairs R_m and R_n, evaluate the increase ΔD that would come out in the total squared error D, if they were merged.
- Choose the pair achieving the minimum increase.

Of course, ΔD depends only on the pixels inside R_m and R_n. So, it can be written as

$$\Delta D = \sum_{(x,y)\in R_{mn}} \|\mathbf{s}(x,y)-\mu_{mn}\|^2 - \sum_{(x,y)\in R_m} \|\mathbf{s}(x,y)-\mu_m\|^2 - \sum_{(x,y)\in R_n} \|\mathbf{s}(x,y)-\mu_n\|^2 \tag{3.1}$$

where R_{mn} is the merged region and μ_{mn} is its feature average vector. Noting that N_m, N_n, and N_{mn} are the number of pixels assigned to regions R_m, R_n, and R_{mn} respectively, ΔD can be simplified if $\sum_{(x,y)\in R_i} \|\mathbf{s}(x,y) - \mu_i\|^2$ is replaced by $(\sum_{(x,y)\in R_i} \|\mathbf{s}(x,y)\|^2) - N_i\|\mu_i\|^2$:

$$\Delta D = -N_{mn}\|\mu_{mn}\|^2 + N_m\|\mu_m\|^2 + N_n\|\mu_n\|^2.$$

Further simplification follows if it is noted that $N_{mn} = N_m + N_n$ and

$$\mu_{mn} = \frac{N_m\mu_m + N_n\mu_n}{N_m + N_n},$$

as

$$\Delta D = \frac{N_m N_n}{N_m + N_n}\|\mu_m - \mu_n\|^2,$$

which is nothing but the link weight (in other words, distance between nodes) mentioned in [14], that is given by Equation 2.3.

As a consequence, each 2-D function $s_i(x,y)$, formed by the ith component of $\mathbf{s}(x,y)$, is piecewise approximated in the least squares sense in terms of the

only basis function available: $f_1(x, y) = 1$; therefore, each region has a constant value. The approximation is performed by a suboptimal iterative minimization method, that is "merge the two regions merging of which increases the distortion (2.1) the least". It starts from a *zero* distortion case where every pixel is a distinct region, and ends with a single region where distortion is maximized. At every intermediate stage, it outputs a complete segmentation mask which means that it is a hierarchical algorithm.

3.3 Improvements to RSST

As a trade-off to its simplicity, RSST, as described in Section 3.2, suffers from the problem of unnecessary contours. If $I(x, y)$ is smoothly varying over a large surface on the scene, the false contours become inevitable, since RSST tries to reconstruct this surface as a piecewise constant function. One may expect to eliminate the false contours by decreasing the number of regions. However, the increase in the total distortion, ΔD, corresponding to the merging of the regions constituting the large surface is much greater than that corresponding to the merging of some other *small* regions. Moreover, some necessary contour information can be lost by doing this.

To overcome this problem, two types of variations from the conventional RSST are possible; changing the modeling strategy, that means the collection of 2-D basis functions involved, and changing the distortion measure. These are briefly discussed and some experimental results are given in the next section.

Modeling Strategies: The simplest variation is the inclusion of $f_2(x, y) = x$ and $f_3(x, y) = y$, in addition to $f_1(x, y) = 1$, into the collection of 2-D basis

functions. This means that, $s_i(x,y)$ is approximated as *piecewise planar*. One level higher surface fitting strategy is the inclusion of $f_4(x,y) = xy$, which means that the approximation is *piecewise bilinear*. Since each of its components $s_i(x,y)$ is approximated by some basis functions, the feature vector field $\mathbf{s}(x,y)$ is said to be approximated by some *vector-surfaces*, e.g., vector-planes, vector-bilinear surfaces, etc. Surely, these extensions will result in better *approximated* textures inside the regions compared to the conventional RSST. So, it is expected that large smooth surfaces can be approximated in a single region, because this means the elimination of at least some false contours.

Distortion Measures: The squared distortion measure is a standard measure because it gives the *energy* of the representation error. However, it results in a ΔD expression as above, i.e., which involves sizes of the candidate regions to be merged. More specifically, it prevents large regions from merging until smaller regions merge, which may be the cause of false contours. So, the distortion measure below, independent of region sizes is experimented:

$$D_{max}\{\mathbf{s}(x,y), \mathbf{r}(x,y)\} = \sum_i \max_{x,y} |s_i(x,y) - r_i(x,y)| \tag{3.2}$$

where $s_i(x,y)$ and $r_i(x,y)$ are the ith components of $\mathbf{s}(x,y)$ and $\mathbf{r}(x,y)$, respectively.

3.4 Experimental Results

There are four cases which are different combinations of the modeling strategies and distortion measures mentioned above:

	Basis Functions	Distortion Measure
Case 1	1	squared error
Case 2	1	maximum error
Case 3	1, x, y	squared error
Case 4	1, x, y, xy	squared error

There are four grayscale images over which the experiments are performed; the famous *Lena* image, and arbitrary frames from three MPEG-4 test sequences *Akiyo*, *Hall Monitor*, and *Mother & Daughter*. The images are segmented to 256 regions first, and then by continuing the merging, to 50 or 64 regions. The only feature used in all experiments is the gray-level values of the pixels. Figures 3.1 through 3.4 show the segmentation results.

For the *Lena* image, it is observed that the conventional RSST joins some part of the hat into the background. In case 2, the situation is much worse, even for the 256-region result. However, in cases 3 and 4, the contour of the hat is preserved even for the 50-region result.

For the *Mother & Daughter* image, the homogeneous background is split into many false regions in case 1 with 256 regions. Decreasing the number of regions down to 64 yields less number of false regions, but does not solve the problem completely. Case 2 is handling the problem in an uncontrolled manner, as observed from the 64-region result. For example, it removes some parts of the borders of the face of the mother and the hair of the daughter. Only the top-left part of the background remains as a false region in cases 3 and 4 with 64 regions. However, some part of the face of the daughter is joined to the chair behind her in case 3.

The false contour problem is most clearly observed on the walls and the floor inside the *Hall Monitor* image. Again, case 2 removes false contours in an uncontrolled manner, and moreover, it is unsuccessful in extracting the wall on the right truely, as seen from the 64-region result. Case 3 and 4 with 64 regions are successful in eliminating the false contours such as the ones on the walls, without sacrificing some necessary contours like the borders of the walls.

In the *Akiyo* image, the same homogeneous background phenomenon is observed. Again cases 3 and 4 with 64 regions are the most successful ones in eliminating the false contours without deleting the true ones.

Some other cases could also be experimented; for example, maximum deviation error could be a good choice when the basis functions are 1, x, y. However, the maximum deviation error does not offer analytical solutions to the problem of finding the "best approximated surface" in terms of these basis functions.

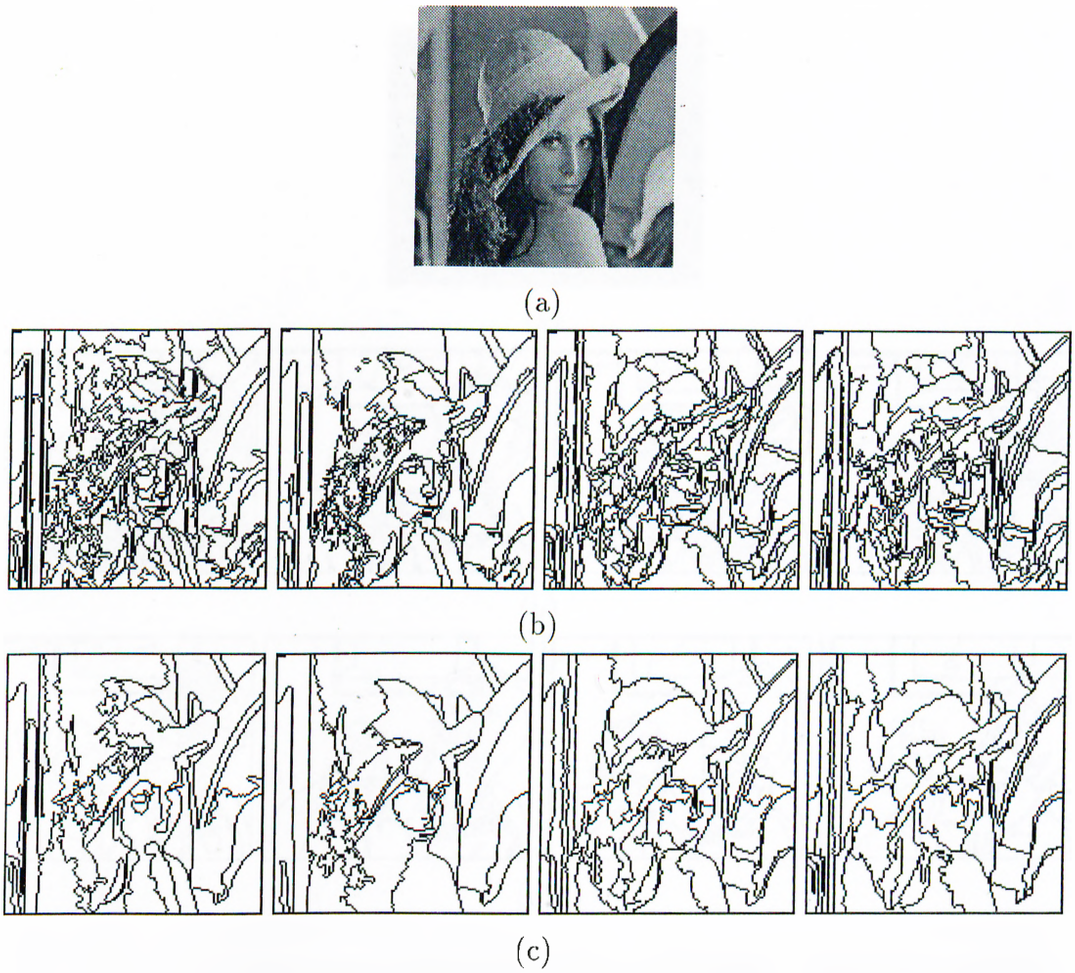

Figure 3.1: (a) The original *Lena* image. (b) and (c) RSST results with 256 and 50 regions, respectively. Results are given in the order of their **case** numbers.

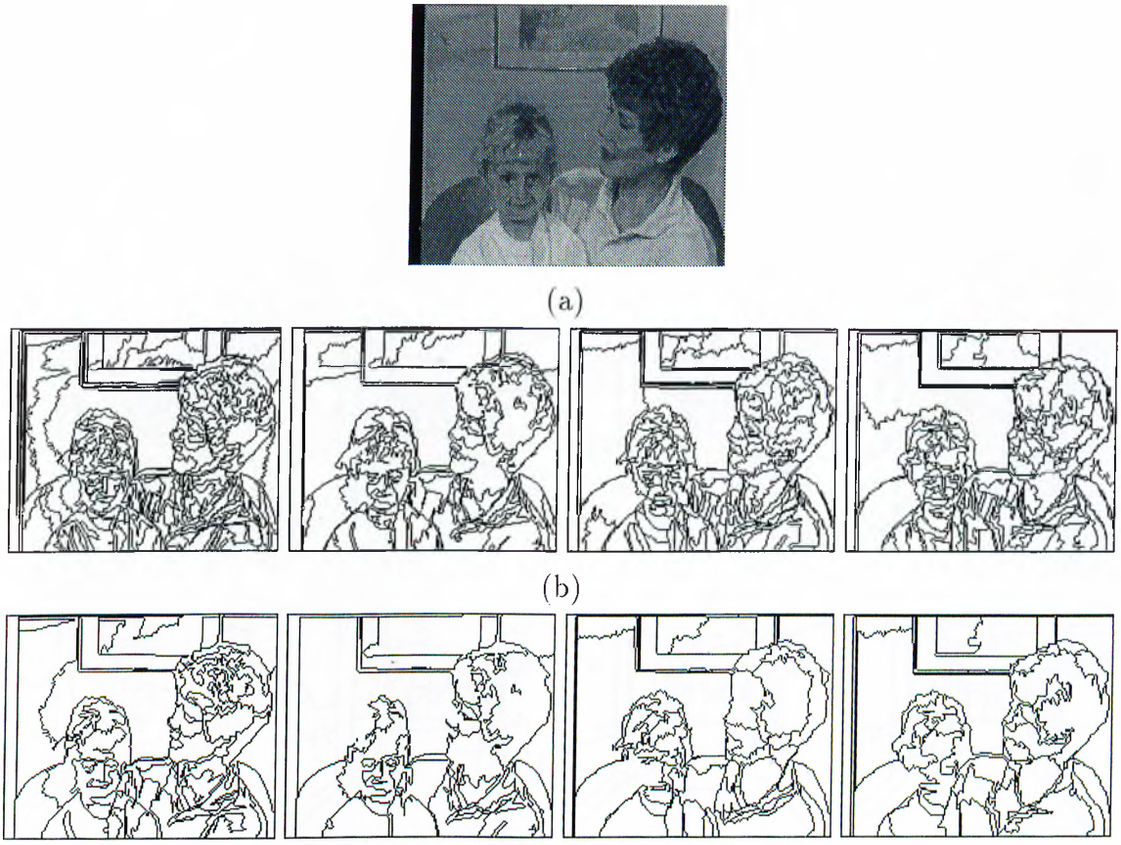

Figure 3.2: (a) The original *Mother & Daughter* image. (b) and (c) RSST results with 256 and 64 regions, respectively. Results are given in the order of their **case** numbers.

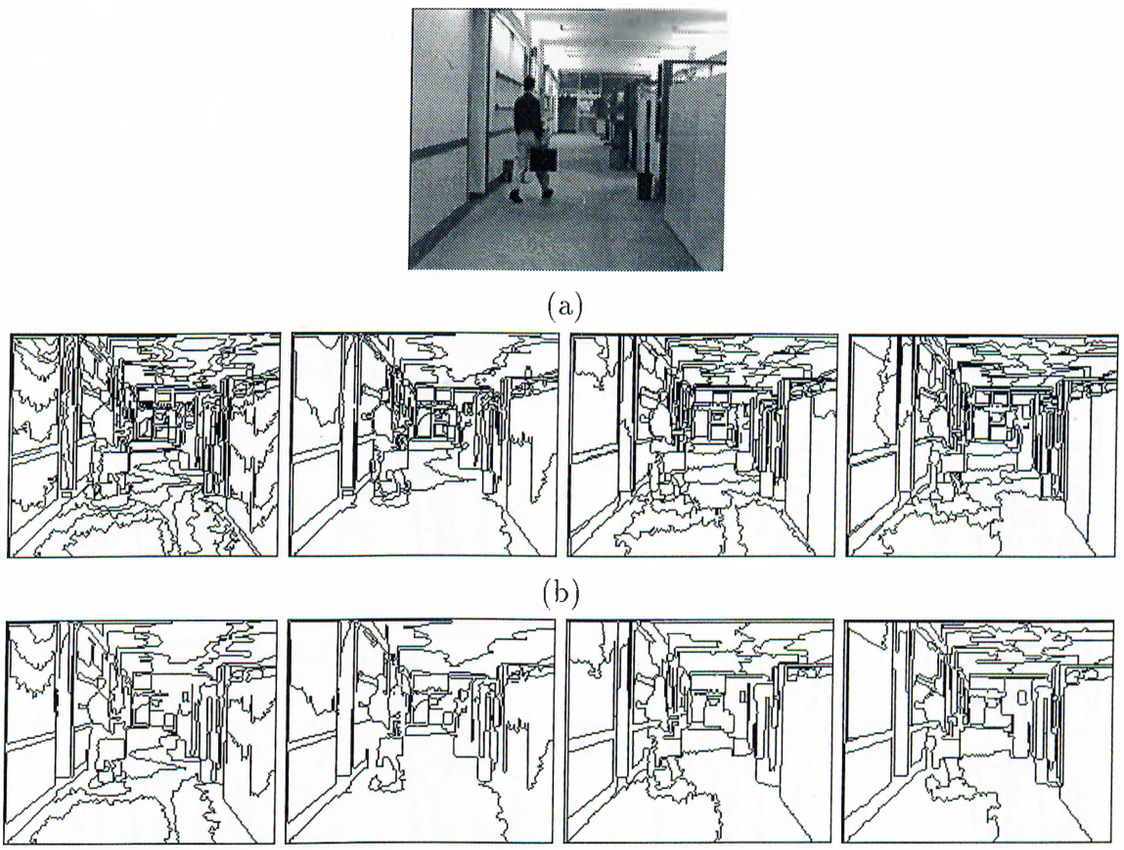

Figure 3.3: (a) The original *Hall Monitor* image. (b) and (c) RSST results with 256 and 64 regions, respectively. Results are given in the order of their **case** numbers.

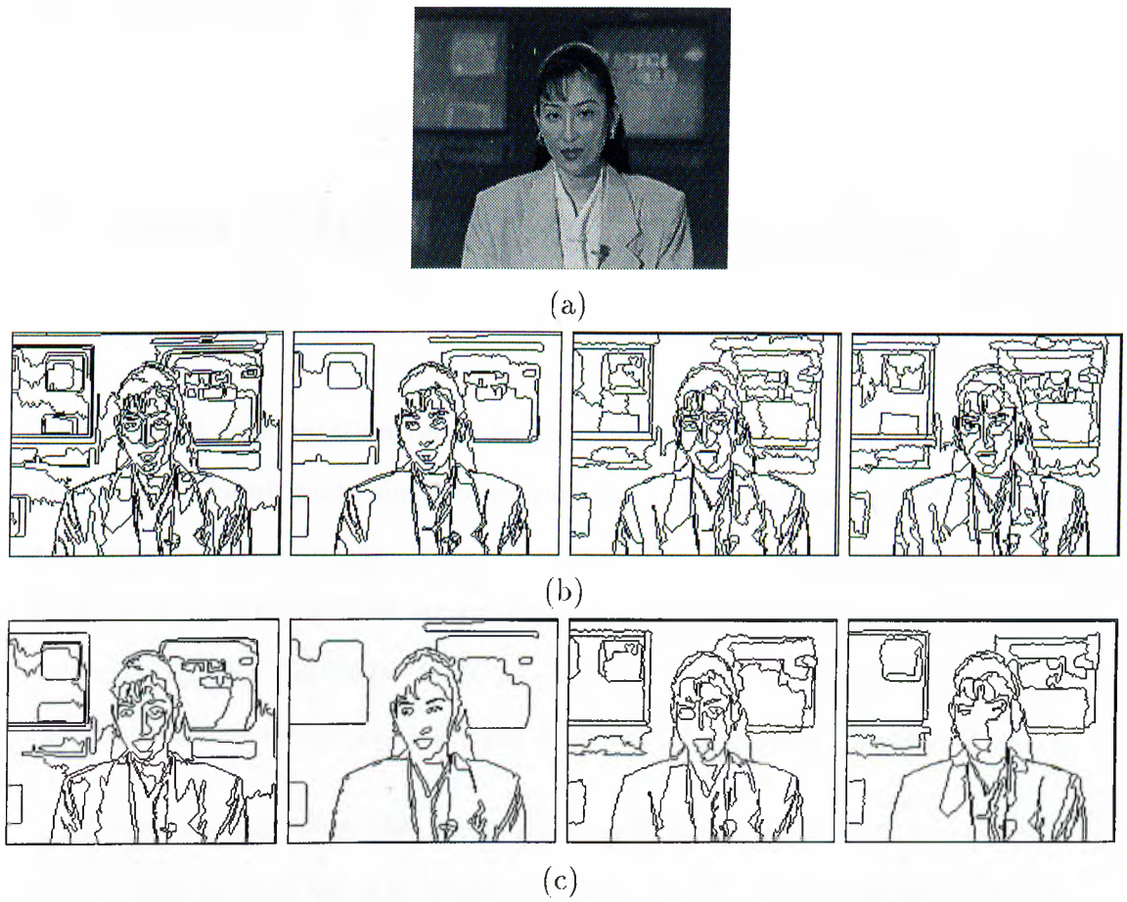

Figure 3.4: (a) The original *Akiyo* image. (b) and (c) RSST results with 256 and 64 regions, respectively. Results are given in the order of their **case** numbers.

Chapter 4

Video Object Segmentation

Video object segmentation refers to partitioning of the frames in a video sequence [4]. The ultimate aim is to extract the *semantically* meaningful objects, e.g., woman, car, ship, etc. from the scene. The *temporal* information is also used, as well as the spatial information, in the segmentation process (see for example [4], pp. 198–199). This intuitively promises a better segmentation compared to still image segmentation schemes mentioned in Chapters 2 and 3.

In this chapter, first the geometric image formation and the projection of 3-D motion onto the 2-D image plane are given as mathematical formulations, then the methods in the literature about video object segmentation are discussed. After this discussion, an alternative method based on the facts mentioned in Chapter 3, is proposed with some experimental justifications on both artificially generated and natural image sequences.

4.1 Geometric Image Formation

Imaging systems capture 2-D projections of time varying 3-D real world. This projection can be modeled by

$$P : (X, Y, Z, t) \rightarrow (x, y, t)$$

where X, Y, Z are the 3-D world coordinates, x, y are the 2-D image plane coordinates, and t is the time.

The most popular types of projection models are the *perspective projection* and the *orthographic projection* (see for example [4], pp. 28–31.) They are good approximations for some real cases, and they are mathematically simple.

Perspective Projection

Perspective projection (see for example [4], pp. 28–30) reflects the image formation process using the ideal pinhole camera model. All the rays from the object pass through the lens center as shown in Figure 4.1. The corresponding algebraic relations follow from *similar triangles* as

$$\begin{aligned} x_0 &= \frac{fX_0}{f - Z_0} \\ y_0 &= \frac{fY_0}{f - Z_0}, \end{aligned} \tag{4.1}$$

where f is the distance from the lens center to the image plane.

A simplified but equivalent model which comes out by introducing a change of variable $f - Z_0 \rightarrow Z_0$, is drawn in Figure 4.2. The corresponding projection formulas are

$$x_0 = \frac{fX_0}{Z_0}$$

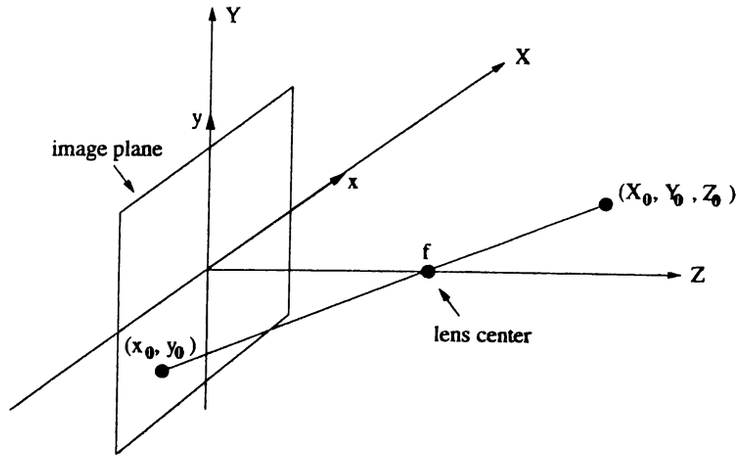

Figure 4.1: Perspective Projection Model

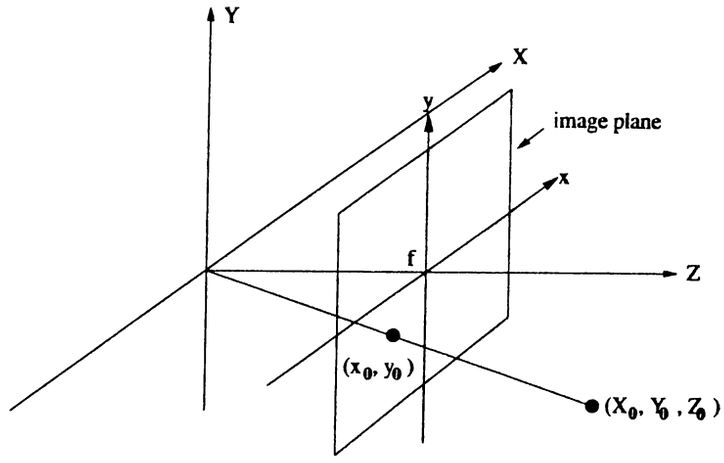

Figure 4.2: Alternative Perspective Projection Model

$$y_0 = \frac{fY_0}{Z_0}. \tag{4.2}$$

Orthographic Projection

Orthographic projection (see for example [4], pp. 30–31) is a special case of the perspective model represented by equation (4.1) , where $f \to \infty$ and $Z_0 \to \infty$. In this case, all the rays from the object to the image plane travel parallel

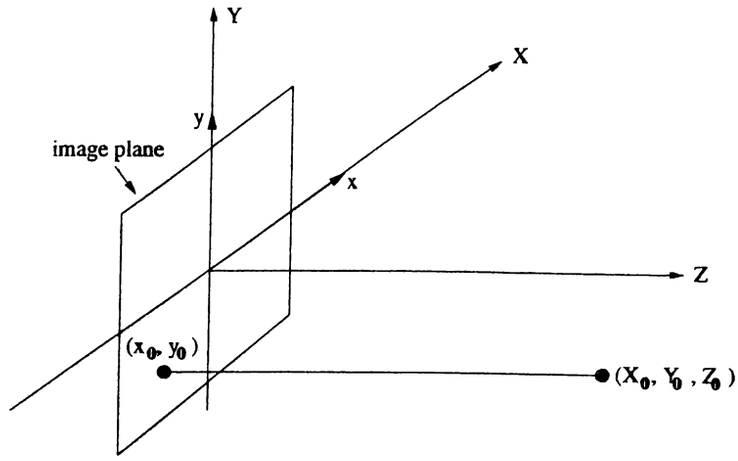

Figure 4.3: Orthographic Projection Model

to each other. This phenomenon is shown in Figure 4.3. The orthographic projection can be described by

$$\begin{aligned} x_0 &= X_0 \\ y_0 &= Y_0 \,. \end{aligned} \tag{4.3}$$

The distance of the object to the camera does not affect the image plane intensity distribution, that is, the object always yields the same image no matter how far it is from the camera.

4.2 Modeling the Projected Motion Field

The 3-D motion of an object is assumed to be rigid, that is, purely rotational and translational, and hence, can be represented by an affine transformation which has 6 degrees of freedom (see for example [4], pp. 153.)

Suppose a point $\vec{\mathbf{X}} = [X \; Y \; Z]^T$ on a rigid object at time t moves to

$\vec{\mathbf{X}}' = [X'\ Y'\ Z']^T$ at time t' subject to a rotation matrix $\mathbf{R}$ and a translation vector $\vec{\mathbf{T}}$, that is,

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \vec{\mathbf{X}}' = \mathbf{R}\vec{\mathbf{X}} + \vec{\mathbf{T}} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix} . \tag{4.4}$$

Note that $\mathbf{R}$ should be a *unitary* matrix for the 3-D motion to be rigid.

The corresponding motion field $\vec{\mathbf{V}} = \vec{\mathbf{X}}' - \vec{\mathbf{X}}$ can also be written as an affine transformation

$$\vec{\mathbf{V}} = \mathbf{A}\vec{\mathbf{X}} + \vec{\mathbf{T}} , \tag{4.5}$$

where

$$\mathbf{A} = \mathbf{R} - \mathbf{I} .$$

Let $\vec{\mathbf{x}} = [x\ y]^T$ and $\vec{\mathbf{x}}' = [x'\ y']^T$ be the projections of $\vec{\mathbf{X}}$ and $\vec{\mathbf{X}}'$ onto the 2-D image plane, respectively. Then $\vec{\mathbf{v}} = \vec{\mathbf{x}}' - \vec{\mathbf{x}}$ is called the *projected motion field.* In following subsections, the behavior of projected motion field of an object under two types of projections is discussed.

4.2.1 Perspective Motion Field Model

The perspective motion field (see for example [4], pp. 154) can be derived by substituting X', Y', and Z' from (4.4) into the perspective projection model given by (4.2):

$$
\begin{aligned}
x' &= f\frac{r_{11}X + r_{12}Y + r_{13}Z + T_1}{r_{31}X + r_{32}Y + r_{33}Z + T_3} \\
y' &= f\frac{r_{21}X + r_{22}Y + r_{23}Z + T_2}{r_{31}X + r_{32}Y + r_{33}Z + T_3} .
\end{aligned} \tag{4.6}
$$

Further simplification follows by dividing the numerator and the denominator by Z (see for example [4], pp. 154):

$$
\begin{aligned}
x' &= f\frac{r_{11}x + r_{12}y + r_{13}f + \frac{T_1}{Z}f}{r_{31}x + r_{32}y + r_{33}f + \frac{T_3}{Z}f} \\
y' &= f\frac{r_{21}x + r_{22}y + r_{23}f + \frac{T_2}{Z}f}{r_{31}x + r_{32}y + r_{33}f + \frac{T_3}{Z}f} .
\end{aligned} \tag{4.7}
$$

Notice that this model is valid for moving surfaces with arbitrary shape in 3-D.

4.2.2 Orthographic Motion Field Model

The orthographic motion field (see for example [4], pp. 153) can be derived by substituting X' and Y' form (4.4) to $x' = X'$ and $y' = Y'$, that define the orthographic projection. The resultant formulation is:

$$
\begin{aligned}
x' &= r_{11}x + r_{12}y + (r_{13}Z + T_1) \\
y' &= r_{21}x + r_{22}y + (r_{23}Z + T_2)
\end{aligned} \tag{4.8}
$$

or equivalently

$$
\begin{aligned}
v_x &= (r_{11} - 1)x + r_{12}y + (r_{13}Z + T_1) \\
v_y &= r_{21}x + (r_{22} - 1)y + (r_{23}Z + T_2) ,
\end{aligned} \tag{4.9}
$$

where $\vec{\mathbf{v}} = [v_x\ v_y]^T$. As with the perspective motion field model, this model is also valid for moving surfaces with arbitrary shape in 3-D.

4.2.3 Special Case of 3-D Planar Surfaces

Planar surfaces play an important role, because most real-word surfaces can be approximated as planar at least on a piecewise basis. The main purpose for treating planar surfaces as a special case is that, it is possible to simplify the projection formulas in that case.

Let the 3-D points that we observe all lie on a plane described by

$$aX + bY + cZ = 1 \ ,$$

where $[a\ b\ c]^T$ denotes the normal vector to the plane.

Then, in the perspective motion field model (4.6), T_i can be replaced by $T_i(aX + bY + cZ)$ and by dividing again both the numerator and the denominator by Z, one gets (see for example [4], pp. 165-166):

$$\begin{aligned} x' &= \frac{a_1 x + a_2 y + a_3}{a_7 x + a_8 y + 1} \\ y' &= \frac{a_4 x + a_5 y + a_6}{a_7 x + a_8 y + 1} \ , \end{aligned} \tag{4.10}$$

which is known as the 8-parameter or *pure* motion model in 2-D.

The same substitution can be done in the orthographic motion field model (4.8) which results in

$$\begin{aligned} x' &= a_1 x + a_2 y + a_3 \\ y' &= a_4 x + a_5 y + a_6 \ , \end{aligned} \tag{4.11}$$

which is known as the 6-parameter or *affine* motion model in 2-D. The affine motion model plays an important role in this dissertation, so the following subsection is devoted to other special cases which yield the affine model.

4.2.4 Other Cases Yielding Affine Motion in 2-D

From (4.8), it can be seen that there is another way to achieve 2-D affine motion model under orthographic projection, this time for arbitrary 3-D surfaces : just turn off r_{13} and r_{23}; that is, let $r_{13} = 0$ and $r_{23} = 0$. Note that, this means that $r_{33} = 1$, $r_{31} = 0$, and $r_{32} = 0$, since the rotation matrix $\mathbf{R}$ in (4.4) is unitary. But, this implies the rotation $\mathbf{R}$ is *purely* 2-D.

There is also a way to achieve an affine motion model in 2-D under perspective projection, but for 3-D planar surfaces only. If the rotation and the translation are purely 2-D, that is, $r_{31} = 0$, $r_{32} = 0$, $r_{33} = 1$, $r_{23} = 0$, $r_{13} = 0$, and $T_3 = 0$, then in (4.10), a_7 and a_8 are turned off, which obviously means that the 8-parameter model reduces to an affine model.

As a summary, the following cases result in 2-D affine motion model:

- Under orthographic projection, 3-D planar surfaces, arbitrary rotation and translation,
- Under orthographic projection, arbitrary surfaces, 2-D rotation, arbitrary translation,
- Under perspective projection, 3-D planar surfaces, 2-D rotation and translation.

4.3 Use of Motion as a Feature

If the segmentation algorithms mentioned in Chapters 2 and 3 are applied to the frames of the video, because they exploit only spatial information, they are bound to yield semantically meaningless objects. This phenomenon can be observed from Figures 3.1 through 3.4.

However, semantically meaningful objects usually make rigid motion in the 3-D world. The projection of this kind of motion onto the 2-D image plane constitutes a so-called *parametric* model throughout the 2-D projection range of the object. These models are already studied in the previous section.

So, if the estimated 2-D motion vectors at each pixel are used as the features of interest, *segmentation through surface fitting* is anticipated to extract the regions for which a good parameter set (explaining the observed motion well) exists. There are justifications of the use of surface fitting in the literature, and the next section is devoted to some methods using 6 or 8-parameter models. In Section 4.5, a novel method, using the improved RSST proposed in Chapter 3 applied on estimated motion vector field, is introduced.

4.4 Methods in the Literature

4.4.1 Modified K-Means Algorithm

In [23], a modified K-means algorithm is proposed. Suppose we have K regions/clusters that are known a priori. The region R_i will have its motion parameters $a_1^i, a_2^i, \ldots, a_n^i$, which define a so-called *motion vector surface* $\vec{\mathbf{w}}^i(x,y)$ throughout the image plane, and are optimum in the sense that the squared sum error

$$\sum_{(x,y)\in R_i} \|\vec{\mathbf{v}}(x,y) - \vec{\mathbf{w}}^i(x,y)\|^2 , \tag{4.12}$$

is minimized [23]. The samples of the approximated motion vector surface $\vec{\mathbf{w}}^i(x_0,y_0)$ are referred as the *synthesized motion vectors* of cluster i at site (x_0,y_0).

Once again suppose that we have K different parameters at hand, but this time the regions are unknown. A pixel (x_0,y_0) can be assigned to region R_i if

$$\|\vec{\mathbf{v}}(x_0,y_0) - \vec{\mathbf{w}}^i(x_0,y_0)\|^2 \tag{4.13}$$

is minimized over i.

In both of the situations, the total squared error

$$D = \sum_{i=1}^{K} \sum_{(x,y)\in R_i} \|\vec{\mathbf{v}}(x,y) - \vec{\mathbf{w}}^i(x,y)\|^2 \tag{4.14}$$

is minimized over the freedom (either unknown regions, or unknown parameters) at hand.

Based on these facts, a generalization of the K-means algorithm such that, instead of the cluster means, the cluster parameters are iterated, was proposed

in [23]. In fact, if the motion model is such that $\vec{\mathbf{w}}^i(x,y) = c^i$, where c^i is the only parameter of region i, this method reduces to the classical K-means algorithm, applied to the estimated motion vectors.

The classical problem of the clustering algorithm is not eliminated: the resultant regions may not be connected, since spatial connectivity is not involved.

The algorithm proposed in [23] to determine the initial cluster parameters is the following: first, the image is divided into square blocks, and the parameters and the corresponding synthesized motion vectors of each block are computed. Then the reliability of that synthesis is estimated by calculating the sum of squared error between the actual and the synthesized motion vectors over the block and a decision is made in a boolean manner to take the parameters into the sample set, or not. Finally, the accepted (reliable) parameters are clustered into K groups, using the conventional K-means algorithm.

4.4.2 Bayesian Segmentation

A Bayesian framework [24], [25], [26], [27], [28] can be a remedy for the problems of clustering scheme; i.e., spatial connectivity is supported by an appropriate Gibbs random field model, and minimization of the distortion function via simulated annealing or similar approaches guarantees avoidance from being trapped into a *local* minimum.

The MAP-based segmentation method proposed in [24] searches for the maximum of the *a posteriori* probability of the segmentation labels, given the

motion vector data. The a posteriori probability measures how well the segmentation explain the observed motion vectors.

If $\mathbf{Z}$ denotes the scalar segmentation label field, and $\mathbf{V}$ denotes the observed motion vector filed, the probability $P(\mathbf{Z} = z|\mathbf{V} = \mathbf{v})$ to be maximized is easily shown to be proportional to $P(\mathbf{V} = \mathbf{v}|\mathbf{Z} = z)P(\mathbf{Z} = z)$, from Bayes rule, and from the fact that $P(\mathbf{V} = \mathbf{v})$ is a constant.

$\mathbf{Z}$ is modeled by a Gibbsian distribution in [24], that is

$$P(\mathbf{Z} = z) = k \exp\{-U(z)\}, \tag{4.15}$$

where

$$U(z) = \sum_{C \in \mathbf{C}} V_C(z). \tag{4.16}$$

The conditional pdf $P(\mathbf{V}|\mathbf{Z})$ is a measure of how well the piecewise model depending on $\mathbf{Z}$, fits the estimated motion field $\mathbf{V}$. Assuming that the mismatch between the estimated motion v and the synthesized motion w (which is a function of z) is a Gaussian white noise with zero mean and variance σ^2, that conditional probability can be expressed as

$$P(\mathbf{V} = \mathbf{v}|\mathbf{Z} = z) = k \exp\left[-\frac{1}{2\sigma^2} \sum_{x,y} \|\vec{\mathbf{v}}(x,y) - \vec{\mathbf{w}}(x,y)\|^2\right]. \tag{4.17}$$

Then, maximization of the *a posteriori* probability is equivalent to minimization of

$$E(z) = \lambda \sum_{C \in \mathbf{C}} V_C(z) + \sum_{x,y} \|\vec{\mathbf{v}}(x,y) - \vec{\mathbf{w}}(x,y)\|^2 \tag{4.18}$$

The MAP-based segmentation alternates between estimation of the model parameters and assignment of the segmentation labels based on a simulated annealing procedure, i.e., by perturbing the segmentation field and accepting

or rejecting it depending on the change in the cost function (4.18) and on the temperature T.

4.4.3 Simultaneous Segmentation and Motion Estimation

The success of performing the segmentation based on the *estimated* motion information is closely related to the accuracy of the estimation, and at the same time, motion estimation is bound to give inaccurate results in the proximity of object boundaries if these boundaries are not known. So, the quality of motion estimation depends on the quality of the motion segmentation and vice versa (see for example [4], pp. 210.) This phenomenon is known as the *chicken and egg* problem in the literature.

What follows is that the motion estimation and motion segmentation must be addressed simultaneously as in [27], [28], [29], [30], for best results.

The simultaneous MAP framework proposed in [29] tries to maximize the *a posteriori* probability given by

$$P(\mathbf{V},\mathbf{Z}|\mathbf{I}_{k-1},\mathbf{I}_k) = \frac{P(\mathbf{I}_k|\mathbf{I}_{k-1},\mathbf{V},\mathbf{Z})P(\mathbf{V}|\mathbf{Z},\mathbf{I}_{k-1})P(\mathbf{Z}|\mathbf{I}_{k-1})}{P(\mathbf{I}_k|\mathbf{I}_{k-1})}\,, \qquad (4.19)$$

where $\mathbf{I}_k$ and $\mathbf{I}_{k-1}$ are the current and reference intensity frames respectively.

The first conditional pdf $P(\mathbf{I}_k|\mathbf{I}_{k-1},\mathbf{V},\mathbf{Z})$ measures how well the present motion and segmentation fields conform with the observed **frame** $\mathbf{I}_k$ given frame $\mathbf{I}_{k-1}$. It is given by a Gibbsian distribution in [29] as

$$P(\mathbf{I}_k = I_k|\mathbf{I}_{k-1} = I_{k-1},\mathbf{V}=\mathbf{v},\mathbf{Z}=z) = c_1 \exp\left\{-U_1(I_{k+1}|I_k,\mathbf{v})\right\}\,, \qquad (4.20)$$

where

$$U_1(I_{k+1}|I_k, \mathbf{v}) = \sum_{x,y} [I_k(x,y) - I_{k-1}(x + v_x(x,y), y + v_y(x,y))]^2 \,. \qquad (4.21)$$

The second conditional pdf $P(\mathbf{V}|\mathbf{Z}, \mathbf{I}_{k-1})$ is assumed to be equivalent to $P(\mathbf{V}|\mathbf{Z})$ which measures how well the estimation of motion conforms with the present segmentation field, and is given in [29] by

$$P(\mathbf{V} = \mathbf{v}|\mathbf{Z} = z) = c_2 \exp\{-U_2(\mathbf{v}|z) - U_3(\mathbf{v}|z)\} \,, \qquad (4.22)$$

where

$$U_2(\mathbf{v}|z) = \alpha \sum_{x,y} \|\vec{\mathbf{v}}(x,y) - \vec{\mathbf{w}}(x,y)\|^2 \qquad (4.23)$$

and

$$U_3(\mathbf{v}|z) = \beta \sum_{x,y} \sum_{(s,t) \in N^z_{(x,y)}} \|\vec{\mathbf{v}}(x,y) - \vec{\mathbf{v}}(s,t)\|^2 \qquad (4.24)$$

for which $\vec{\mathbf{w}}(x,y)$ is the synthesized motion vector field implied by the segmentation and motion fields, $N^z_{(x,y)}$ is the set of neighboring pixels to (x,y) which are in the same segment with it, and α and β are the weights of the two terms.

Finally, the conditional pdf $P(\mathbf{Z}|\mathbf{I}_{k-1})$ is assumed to be equal to $P(\mathbf{Z})$, which models the *a priori* probability of the segmentation field, as usual given by

$$P(\mathbf{Z} = z) = c_3 \exp\{-U_4(z)\} \,, \qquad (4.25)$$

where

$$U_4(z) = \gamma \sum_{C \in \mathbf{C}} V_C(z) \,, \qquad (4.26)$$

and $V_C(z)$ is the clique potential which is only a function of $z(x,y)$ for which $(x,y) \in C$.

The Algorithm

Maximizing (4.19) is equivalent to minimizing

$$E(z, \mathbf{v}) = U_1(I_{k+1}|I_k, \mathbf{v}) + U_2(\mathbf{v}|z) + U_3(\mathbf{v}|z) + U_4(z) \; . \qquad (4.27)$$

Direct minimization with respect to all unknowns is a difficult problem. So, in [29], an iterative algorithm is proposed, so that for the given motion vector field, segmentation field is iterated, and vice versa. The algorithm is as follows:

1. Given the best available estimates of a_1^i, a_2^i, ..., a_n^i for $i = 1,\ldots,K$, and $z(x,y)$, update the motion vector field $\vec{\mathbf{v}}(x,y)$. This step involves the minimization of a sub-cost function $U_1(I_{k+1}|I_k, \mathbf{v}) + U_2(\mathbf{v}|z) + U_3(\mathbf{v}|z)$.

2. Update the segmentation field $z(x,y)$, assuming that the motion vector field $\vec{\mathbf{v}}(x,y)$ is known. This step involves the minimization of another sub-cost function $U_2(\mathbf{v}|z) + U_4(z)$.

The algorithm needs an initial motion vector field and initial segmentation labels for the sake of fast convergence. The initial motion vector field can be found by a fast block-based motion estimation scheme. Then given this estimate, the segmentation labels can be initialized by using the technique in [23].

4.5 Proposed RSST-based Method

The methods mentioned in the last section are designed elaborately and their mathematical background is rather sophisticated. However, they either suffer

from determination of "initial parameters", or presetting of "weights" assigned to different penalizing parts of the involved cost function. Furthermore, being iterative algorithms, it takes a long time for them before convergence, even for dedicated machines.

So, what is proposed in this dissertation is a more practical scheme, in the sense that it involves

- *no* ad hoc weights balancing the significance of the various parts of the cost function,
- *no* initial parameters affecting the performance of the result,
- *no* iterative procedures slowing down the convergence,
- a hierarchical scheme which promises a multiscale segmentation, that is, from *finest* to *coarsest* segmentation results. This means that at a single run, one obtains K-region results for all $K = 1, 2, \ldots, N$.

The algorithm, as depicted in Figure 4.4, is simply to estimate the dense motion vector field between two consecutive frames, and to run the *improved* RSST algorithm which tries to control the segmentation scheme by the quality of the vector-surfaces (namely vector-planes) fitted in the least squares sense to the estimated motion field. In short, the improved RSST algorithm merges at each recursion step the two regions merging of which increase the distortion given by

$$D = \sum_{i=1}^{K} \sum_{(x,y)\in R_i} \|\vec{\mathbf{v}}(x,y) - \vec{\mathbf{w}}^i(x,y)\|^2 \tag{4.28}$$

the least. The details of the improved RSST algorithm is given in Chapter 3.

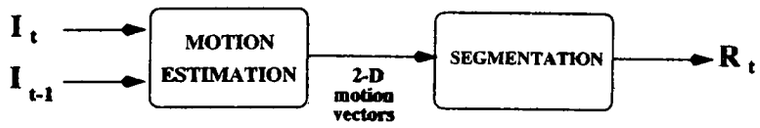

Figure 4.4: Block diagram of the proposed scheme.

Fitting vector-planes to the motion fields inside regions is equivalent to modeling the motion field as an *affine* model, which is a realistic model in certain cases, as explained in Section 4.2.4.

4.6 Experimental Work

Two kinds of experiments are performed:

1. An artificial sequence, consisting of pure 3-D planar objects which are orthographically projected onto the 2-D image plane, is created. This sequence serves the testing of the algorithm under the condition where the motion vector field is known *a priori*. This guarantees that the performance of the algorithm is not affected by the motion estimation step. Furthermore, the motion field already consists of piecewise vector-planes, and as a primary test, the algorithm should extract those pieces successfully.

2. A natural sequence where the surfaces of objects of primary importance are planar in 3-D, is produced. This is to test the algorithm in more realistic cases. This includes the estimation of the motion that involves some errors, especially near the object boundaries, and untextured areas.

Some frames from these sequences are given in Figures 4.5 and 4.6.

In both experiments, results are compared to the application of the conventional RSST, as described in Chapter 2, on the motion vector field. This application implicitly assumes 2-D pure translational motion model, which is a very special case. It is to show that, in object segmentation routines, conventional image segmentation routines fail, because they try to attach a 2-parameter model to objects, whereas improved routines that promotes the model to at least 6-parameters, give promising results.

Figures 4.7 and 4.8 show respectively the results with conventional RSST and plane-fitting RSST for the artificial sequence. It is easily observed from these figures that the modified (plane fitting) scheme is successful when the motion estimation part is bypassed. However, the result of the conventional (homogeneity seeking) scheme is disastrous, as expected, since the movie consists of rotating objects.

In Figures 4.9 and 4.10, the results for the natural QCIF sequence are displayed when the conventional and the plane-fitting RSST are applied, respectively. Although there are some inaccuracies in the motion estimation scheme because of either the occlusion problems or the areas lacking enough texture, the resultant segmentation masks for the plane-fitting RSST experiment are successful in the sense that they more or less locate the objects precisely. However, they are inaccurate in terms of object boundary precision.

A post-processing algorithm, which is later described in Chapter 5, can be applied on these segmentation results to improve the precision of the object boundaries. This algorithm uses an oversegmented color segmentation result to correct the boundaries given by the motion vector field.

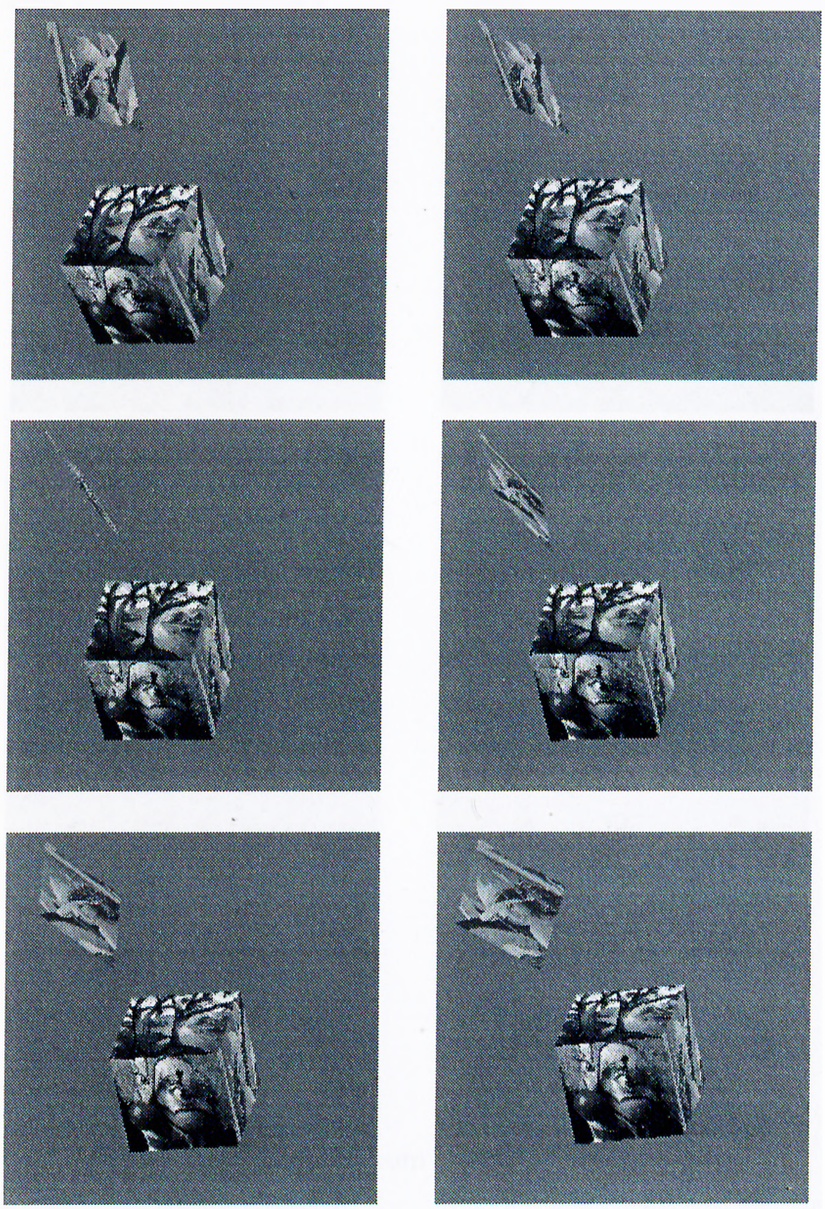

Figure 4.5: Samples from the artificially generated sequence.

Figure 4.6: Samples from the natural sequence.

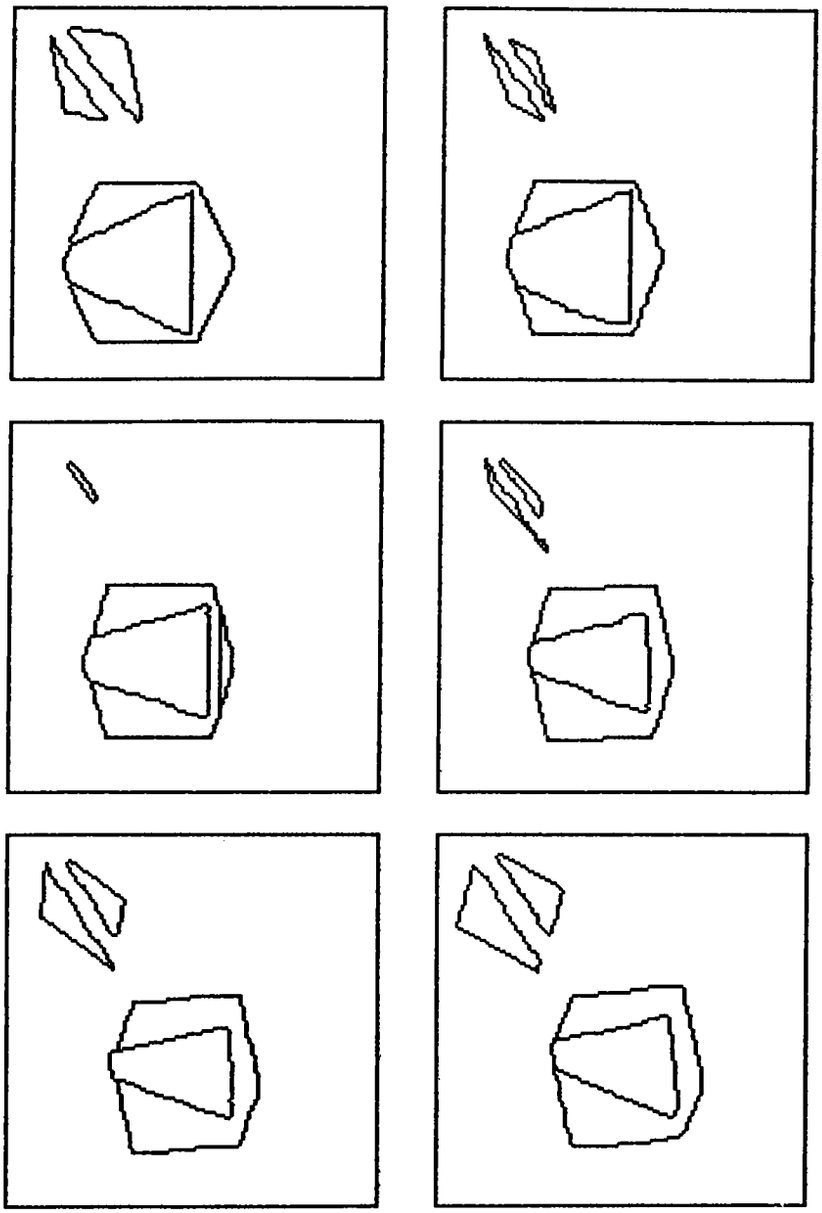

Figure 4.7: Segmentation of the artificially generated sequence with the conventional RSST algorithm.

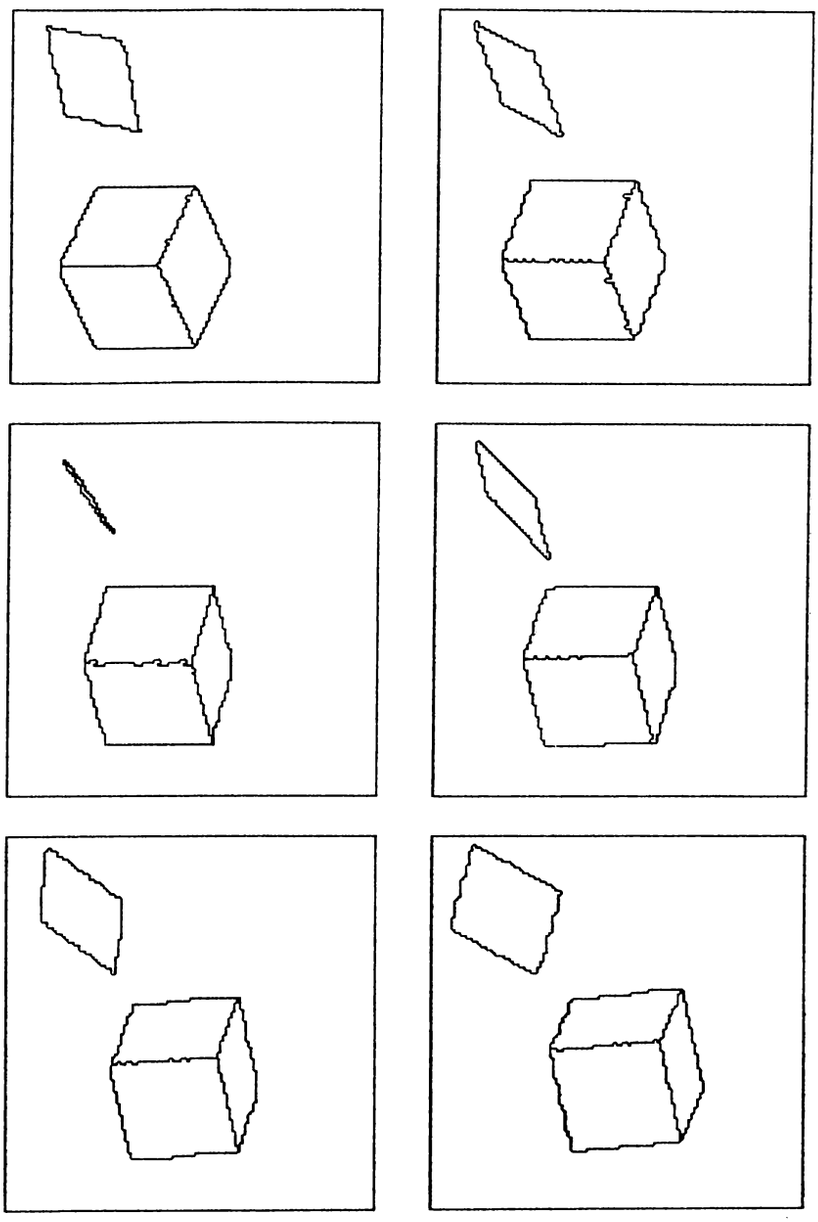

Figure 4.8: Segmentation of the artificially generated sequence with the proposed RSST algorithm.

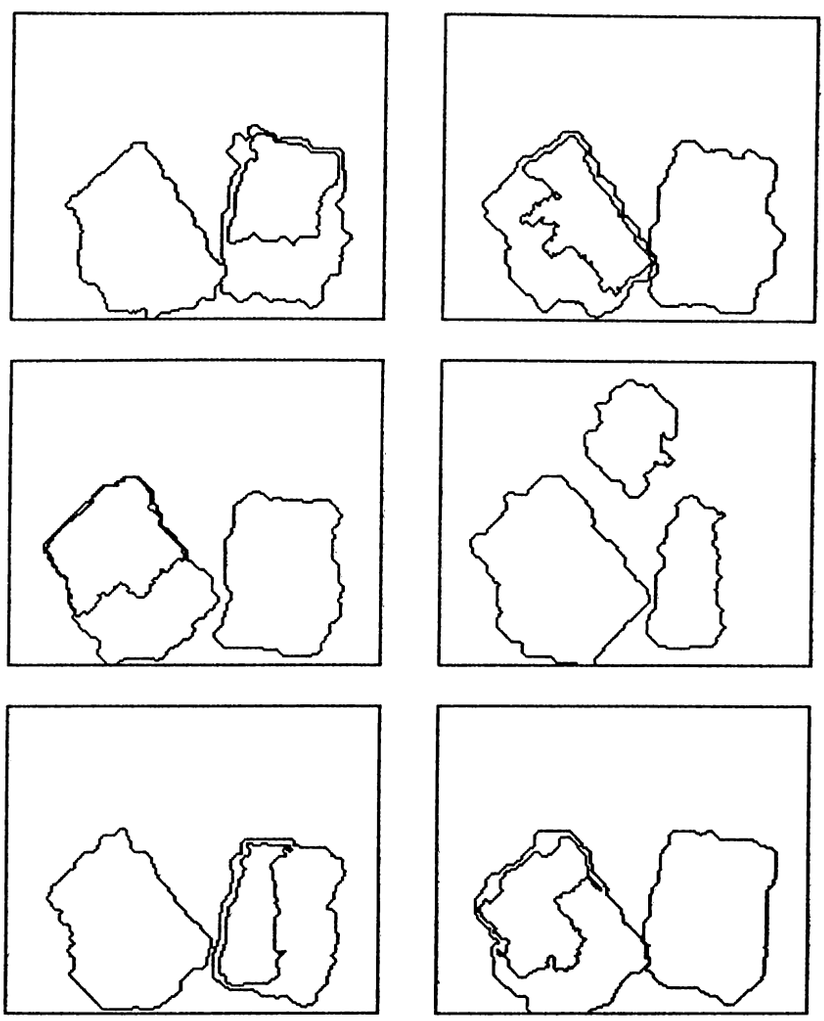

Figure 4.9: Segmentation of the natural sequence with the conventional RSST algorithm.

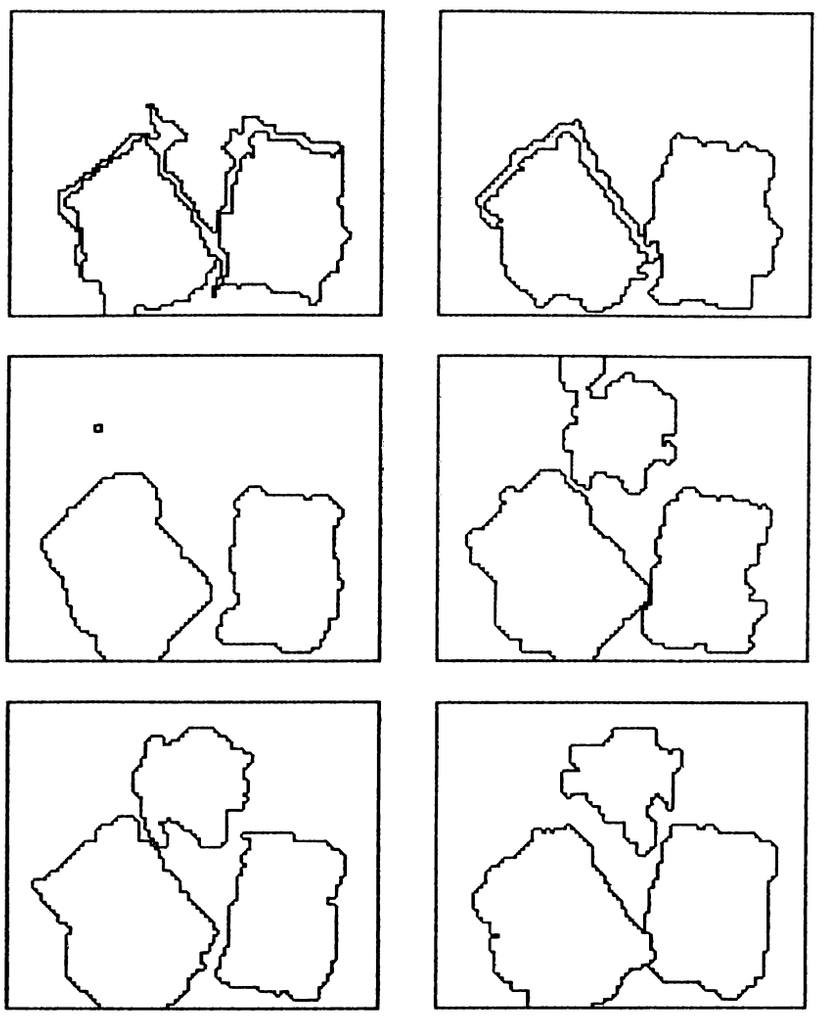

Figure 4.10: Segmentation of the natural sequence with the proposed RSST algorithm.

Chapter 5

Rule-Based Video Object Segmentation and Tracking

"Object tracking" refers to the determination of "which object in the current frame corresponds to which one in the previous frame." It is an essential tool especially for object-based coding purposes, or for other functionalities addressed by MPEG-4, for instance, *object scalability.*

If object tracking is not provided, an object-based coding algorithm loses the *temporal* information, and hence loses its chance to eliminate the redundancies in the temporal domain. And as for the object scalability, fulfillment of manipulation of the objects separately is impossible without achievement of object tracking.

Object tracking includes the handling of newly occurring objects, or of some objects temporarily seem to join the background because that they do not move any more.

In the Europan COST211^{ter} project, the first version of an "Analysis Model", which tries to fulfill the functionality of *Object Definition and Tracking* is already agreed on. This first version was proposed in [31], and then described in more detail in [32].

In this chapter, first this Analysis Model is described and then some improvements based on the object segmentation algorithm proposed in Chapter 4 are sought.

5.1 The Analysis Model

The first version of the Analysis Model (AM) offers a novel approach for object segmentation and tracking, where motion, color, and accumulated segmentation information can be fused at the "region level" by the help of some predefined rules. The color-based and motion-based segmentation results, and the segmentation result of the previous frame are given as inputs to the rule-based decision box which yields the segmentation result for the current frame.

Fusion of segmentation results via a rule-based decision process leads to good segmentation results by utilizing the motion-based regions to *locate* the objects in the scene, and the color-based regions to extract the *true boundaries* of these objects. The segmentation result of the previous frame serves as a temporally accumulated segmentation information, which is essential for *tracking.*

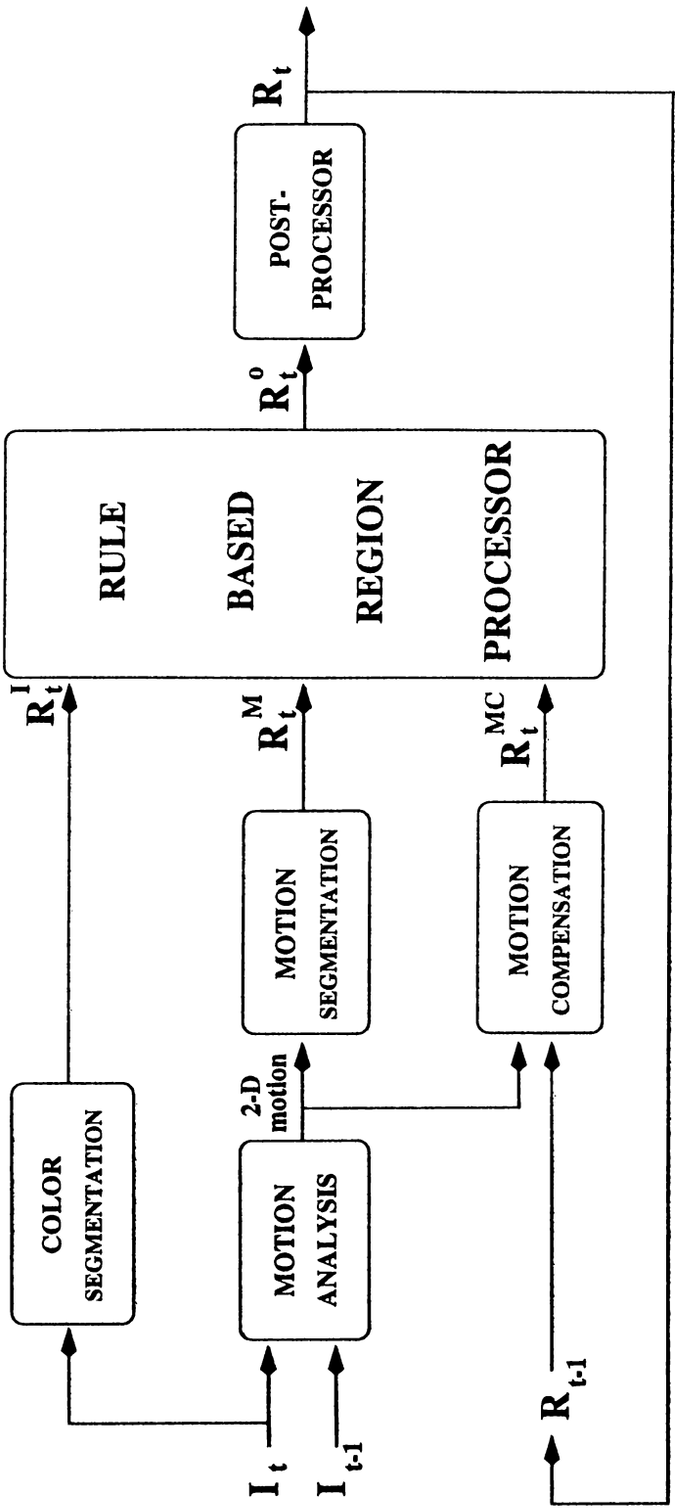

Figure 5.1: The Block Diagram of the Analysis Model

The Algorithm

The block diagram of the algorithm is displayed in Figure 5.1. The six blocks drawn are generic in the sense that the inner work performed inside a box can be changed without disturbing the overall strategy. For example, various motion estimation algorithms existing in the literature can be put inside the 'Motion Analysis' block. The performance of the overall algorithm depends on the power of the individual blocks. Nevertheless, the functions of the blocks and current algorithms inside are described below:

Color Segmentation: Current frame is divided into regions using only the color information. The recursive shortest spanning tree (RSST) [14] based method, described in Chapter 2 is used.

Motion Analysis: Backward motion between two consecutive frames is estimated. A 3-level hierarchical block matching (HBM) [33] algorithm is used due to its acceptable results with low computation. To reduce the computational effort, filtering part in HBM is omitted.

Motion Segmentation: Current frame is divided into regions using only the motion values calculated in the Motion Analysis block. This segmentation is performed by the same RSST algorithm described above. This time, motion vectors are used instead of color values of the pixels.

Motion Compensation: Using the estimated motion vectors, the previous segmentation result is motion compensated.

Rule-Based Region Processor: The segmentation information supplied by Color Segmentation, Motion Segmentation and Motion Compensation blocks

are fused according to some pre-defined rules, which are described in the next section. The output still has to be post-processed.

Post-Processor: Regions are merged if they are too small or have similar motion characteristics. The edges of the regions are modified using a morphological open-close filter. The output is the final segmentation result of the current frame.

5.2 Data Fusion via Rule-Based Region Processing

The three different segmentation results which are input to the region processor have different properties. Although usually results in oversegmentation, Color Segmentation gives the most reliable boundaries. Motion Segmentation locates distinct objects with some semantic meanings, although the extracted object boundaries are usually incorrect. Motion Compensation block outputs a rough prediction of what the current segmentation result will look like. The Rule-Based Region Processor tries to exploit all these facts.

The region processing routine can be split into three phases: Projection, Labeling, and Decision.

From now on, regions in the segmentation results supplied by Color Segmentation, Motion Segmentation and Motion Compensation will be referred as **I**, **M**, and **MC** regions, respectively.

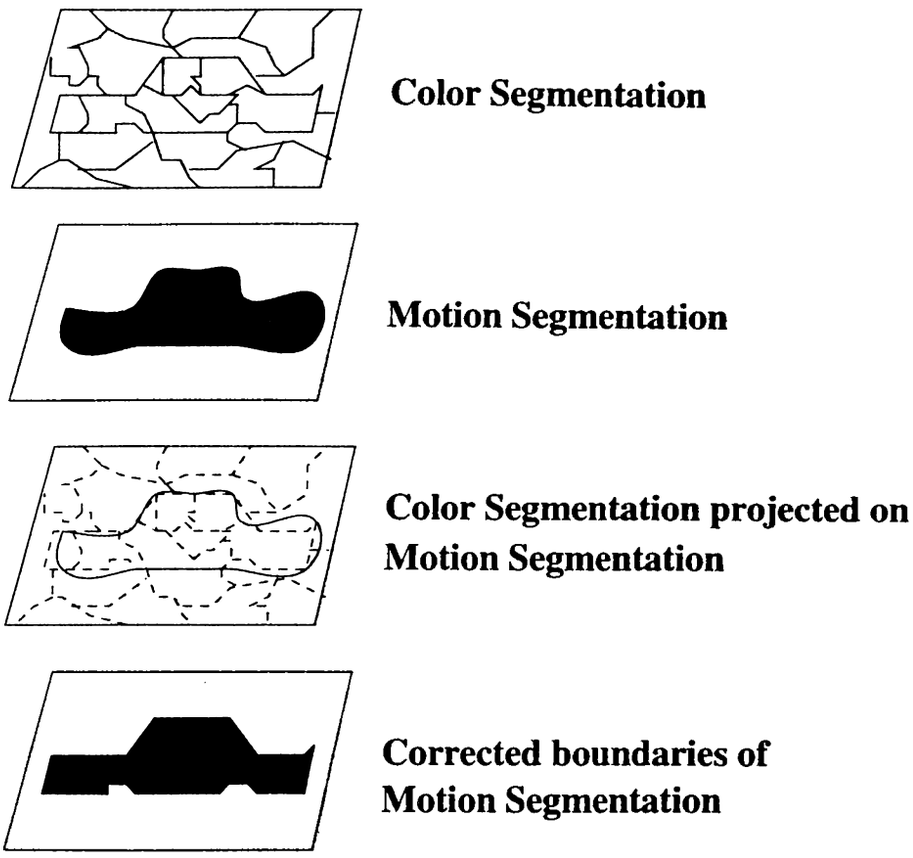

Figure 5.2: The mapping of **I** regions onto the Motion Segmentation result and correction of boundaries

Projection

Every I region is projected onto one M region and one MC region. This mapping is done as follows:

- For an I region, calculate and find the maximum of the areas of the intersections with all M(MC) regions.
- Map the I region onto the M(MC) region with the maximum intersection area.

Figure 5.2 describes the projection step and its possible consequences. In that figure, each I region is painted with the same color with its corresponding M region (either black or white), in order to visualize the "boundary correction" phenomenon. The same projection step is applied for the natural sequence in Chapter 4, and the results are shown in Figure 5.3,

If the projection phase were the mere one, the algorithm would yield a good object segmentation result, but would not fulfill the object tracking functionality.

Labeling

Each M region is labeled as 'moving' or 'stationary' by comparing its average motion with a threshold. Each I region takes the same label as its corresponding M region. Every MC region also has the same type of label borrowed from its counterpart in the previous segmentation result.

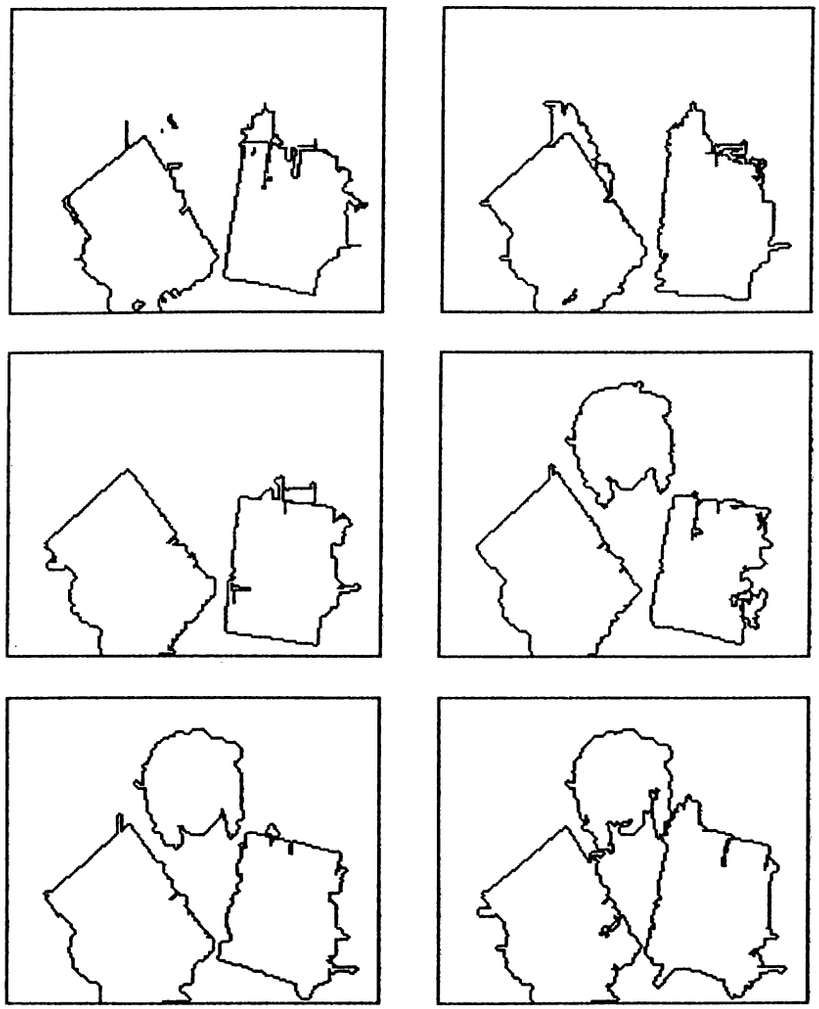

Figure 5.3: The projection phase applied on the natural sequence.

Decision

For an MC region;

- If all I regions mapped onto the MC region have the same label
  - merge those I regions.
- Else
  - if the MC region is moving,
    - * merge the stationary I regions mapped onto the MC region,
    - * merge the moving I regions mapped onto the MC region.
  - else
    - * merge the stationary I regions mapped onto the MC region,
    - * merge moving I regions mapped onto the same M region.

The overall effect of the region processor at first sight is to split the MC regions into several parts, while correcting their boundaries. So traditionally, there must be a merging phase, too. The first task of the post-processor is to merge some regions: some small regions are joined to their larger neighbors, and, some regions which are close in motion are also joined together.

The second post-processing need may come from boundary coding issues, i.e. if the aim is to implement an object-based coder, the coding efficiency will increase with a negligible amount of loss if the boundaries are smoothed with a small morphological open-close filter.

5.3 An Improvement to AM

As seen from the description of the first version of the AM, the Motion Segmentation part uses the conventional RSST applied on the estimated motion field. But, this yields bad motion segmentation results for certain cases (e.g., for 3-D rigid motion) as shown at the end of the previous chapter. This is because, a motion segmentation algorithm assuming such a motion model, i.e., a constant motion vector for each region, can handle only the special case of "2-D and purely translational" motion. So, here, replacing the Motion Segmentation tool by the method based on the improved RSST, explained in Section 4.5, is proposed.

However, if the motion vectors at hand are not reliable enough, as is the case for the HBM used by the Motion Estimation block, then it is meaningless to compare the performance of this or that motion segmentation tool. Therefore, during the experiments, the Motion Estimation block should also be replaced by a more reliable tool (which preferably estimates the dense motion field,) such as the regularized Gibbs formulated motion estimator proposed in [27]. As a matter of fact, that Gibbs formulated tool was used in the original proposal for AM [31], but it was discarded for computational purposes.

In the next section, for testing purposes, the HBM is replaced by the Gibbs formulated motion estimator [27]. Then, the results of the Analysis Model with the conventional RSST and the improved RSST inside the Motion Segmentation block, are compared.

5.4 Experimental Work & Results

Figures 5.4 and 5.5 show the results of the Analysis Model using the conventional and the improved RSST for the Motion Segmentation block, respectively. As mentioned in the previous section, for the Motion Estimation block, the GRF-based algorithm in [27] is used instead of the HBM.

This time, the regions are painted with distinct gray levels in order to visualize the fulfillment of the object tracking functionality. An object is successfully tracked if and only if it is painted with the same gray level value throughout the sequence. Although it is not what is tested in this experiment, the success of the AM to track objects can be seen from the figures. In this experiment, for the two cases mentioned, the capability of extracting objects making 3-D rigid motion, without splitting them into several parts, is tested.

Because of the drawbacks of the conventional RSST applied on motion field mentioned before, the Analysis Model tends to split 3-D objects into several parts whose motion can be approximated more or less as purely 2-D translational. This can be observed for the rotating books on both sides of the man. However, if the improved RSST based on plane fitting is used by the Motion Segmentation block, these objects are captured as a whole.

For the other objects whose motion can be approximated as 2-D and translational, the performance of the compared tools are more or less the same. For example, the shirt of the man is extracted as a single object most of the time (if it constitutes a connected region in 2-D image plane.)

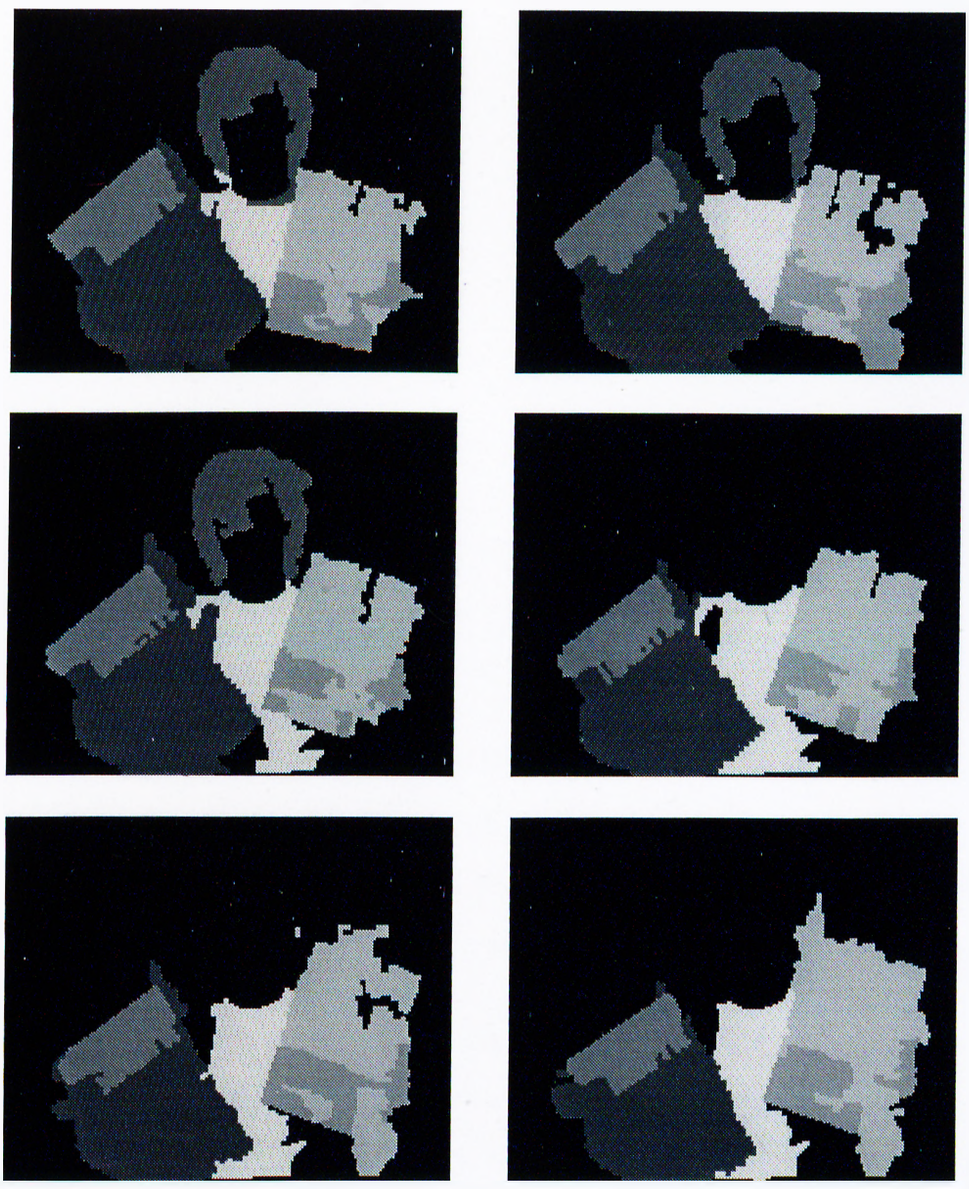

Figure 5.4: The segmentation result of the Analysis Model using the conventional RSST for the Motion Segmentation Block.

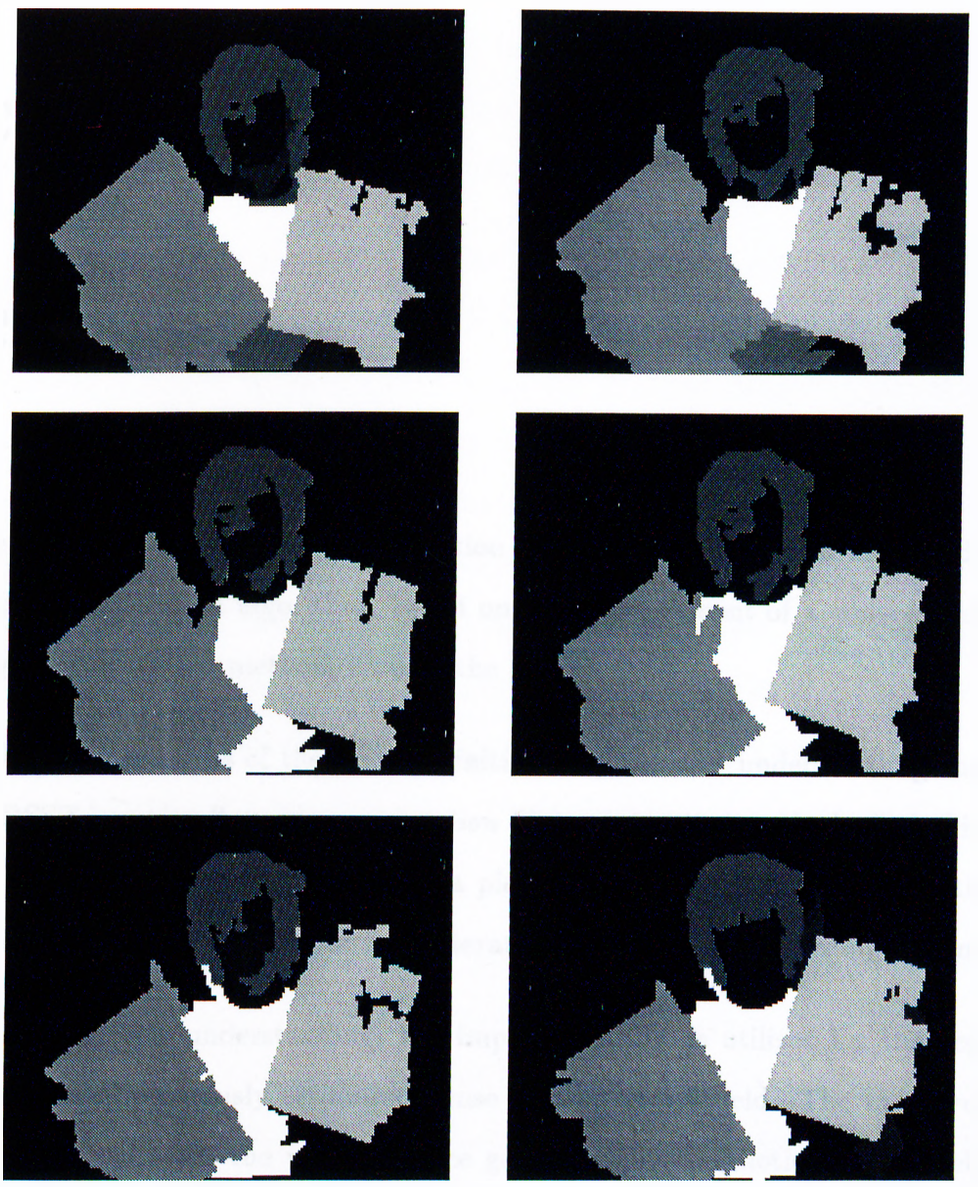

Figure 5.5: The segmentation result of the Analysis Model using the improved RSST for the Motion Segmentation Block.

Chapter 6

Conclusions

The main work done in this dissertation is the development of a novel video object segmentation algorithm, based on the improvement of a conventional image segmentation method, namely the RSST.

The improvement of the RSST is initiated by the new understanding that the RSST is a *simultaneous segmentation & reconstruction* method which tries to approximate the image signal as a piecewise smooth surface through the minimization of a cost function in an iterative manner, i.e., by merging regions.

Through this understanding, the improved RSST is utilized for the segmentation of previously estimated dense motion vector field. The improved RSST approximates the vector-surface generated by the motion vector field by a piecewise vector-planar function, i.e., a vector-valued function which is piecewise planar in each component. This plane-fitting strategy is known in the literature as the extraction of the 6 parameters of the assumed affine motion model.

If the motion vectors at hand are reliable enough, then the resultant segmentation is successful in locating the 3-D planar objects in the scene correctly, with acceptable accuracy at the boundaries for real-life video. Moreover, thanks to the RSST, the algorithm is free from the determination of "initial" parameters, and from presetting "weights" of different parts of the involved cost function, whereas similar algorithms in the literature suffer from those.

Although neither the MPEG-4, nor the MPEG-7 standardizes the video object segmentation, their performance obviously depends on whether the segmentation is successful or not. The emerging "Analysis Model" (AM) of the Europan COST211^{ter} project aims to achieve the functionality of the unsupervised segmentation and *tracking* of the objects, for this purpose. The proposed video object segmentation tool can readily be inserted into the AM which has a modular structure and whose object segmentation module uses the conventional RSST. The replacement of the conventional RSST in the AM by the improved RSST results in a better segmentation, as expected. However, a motion estimation tool which is more reliable than that of the current AM, is needed in order to justify this.

As a future work, the order of the approximation can be increased in order to handle the case of more complex 3-D objects making arbitrary rigid motion. For example, a basis including all $x^m y^n$ with $m, n \leq 2$ can be utilized. Since RSST is much faster than any of the iterative algorithms in the literature, the slowing down caused by increasing the order of the surfaces can be ignored.

Another possible future work is the utilization of the improved RSST for an object-based still image compression algorithm which can be a proposal to the emerging JPEG 2000 standard.

REFERENCES

[1] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Addison-Wesley Pub. Co., 1992.

[2] D. H. Ballard, C. M. Brown. *Computer Vision.* Prentice Hall, 1982.

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

[4] A. M. Tekalp. *Digital Video Processing.* Prentice Hall, 1995.

[5] "MPEG-4 proposal package description (PPD) revision 2,". ISO/IEC JTC1/SC29/WG11 N0937, March 1995.

[6] "MPEG-7: Context and objectives,". ISO/IEC JTC1/SC29/WG11 N1425, November 1996.

[7] G. B. Coleman, H. C. Andrews "Image segmentation by clustering," *IEEE Proceedings*, vol. 67, pp. 773–785, May 1979.

[8] H. Derin, H. Elliott "Modeling and segmentation of noisy and textured images using gibbs random fields," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 39–55, January 1987.

[9] R. Adams, L. Bischof "Seeded region growing," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 641–647, June 1994.

[10] P. Salembier, M. Pardas "Hierarchical morphological segmentation for image sequence coding," *IEEE Trans. on Image Processing*, vol. 3, pp. 639–651, September 1994.

[11] P. Salembier, L. Torres, F. Meyer, C. Gu "Region-based video coding using mathematical morphology," *IEEE Proceedings*, vol. 83, pp. 843–857, June 1995.

[12] P. Salembier, J. Serra "Flat zones filtering, connected operators, and filters by reconstruction," *IEEE Trans. on Image Processing*, vol. 4, pp. 1153–1160, August 1995.

[13] P. Salembier, P. Brigger, J. R. Casas, M. Pardas "Morphological operators for image and video compression," *IEEE Trans. on Image Processing*, vol. 5, pp. 881–897, June 1996.

[14] O. J. Morris, M. J. Lee, A. G. Constantinides "Graph theory for image analysis : an approach based on the shortest spanning tree," *IEE Proceedings*, vol. 133, pp. 146–152, April 1986.

[15] J. Serra. *Image Analysis and Mathematical Morphology.* New York: Academic, 1982.

[16] L. Vincent "Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms," *IEEE Trans. on Image Processing*, vol. 2, pp. 176–201, April 1993.

[17] L. Vincent, P. Soille "Watersheds and digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 583–598, June 1991.

[18] M. Kunt, A. Ikonomopoulos, M. Kocher "Second-generation image coding techniques," *IEEE Proceedings*, vol. 73, pp. 549–574, April 1985.

[19] P. J. Besl, R. C. Jain "Segmentation through variable-order surface fitting," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 167–192, March 1988.

[20] A. Leonardis, A. Gupta, R. Bajcsy "Segmentation as the search for the best description of the image in terms of primitives," in *Third International Conference on Computer Vision '90*, pp. 121–125, 1990.

[21] A. Ackah-Miezan, A. Gagalowicz "Discrete models for energy-minimizing segmentation," in *Fourth International Conference on Computer Vision '93*, pp. 200–207, 1993.

[22] G. Hewer, C. Kenney, B. S. Manjunath "Image segmentation via functionals based on boundary functions," in *Proceedings of ICIP '96*, pp. I 813–816, 1996.

[23] J. Y. A. Wang, E. Adelson "Representing moving images with layers," *IEEE Trans. on Image Processing*, vol. 3, pp. 625–638, September 1994.

[24] D. W. Murray, B. F. Buxton "Scene segmentation from visual motion using global optimization," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 220–228, March 1987.

[25] M. Chang, A.M. Tekalp and M.I. Sezan "Motion field segmentation using adaptive map criterion," in *Proceedings of IEEE ICASSP 93*, pp. 33–36, April 1993.

[26] P. B. Chou, C. M. Brown "The theory and practice of bayesian image labeling," in *International Joint Computer Vision*, volume 4, pp. 185–210, 1990.

[27] A. A. Alatan, L. Onural "Object-based 3-D motion and structure estimation," in *Proceedings of ICIP '95*, pp. I 390–393, October 1995.

[28] A. Alatan, L. Onural "Joint estimation and optimum encoding of depth field for 3-D object-based video coding," in *Proceedings of ICIP '96*, October 1996.

[29] M. Chang, M. I. Sezan, A.M. Tekalp "An algorithm for simultaneous motion estimation and scene segmentation," in *Proceedings of IEEE ICASSP 94*, April 1994.

[30] S. Hsu, P. Anandan, S. Peleg "Accurate computation of optical flow by using layered motion representation," in *Proceedings of International Conference on Pattern Recognition, Jerusalem, Israel*, pp. 743–746, October 1994.

[31] A. Alatan, E. Tuncel and L. Onural "A hybrid method toward automated vop generation and motion estimation," in *COST211ter, Simulation Subgroup Meeting in Ankara*, October 1996.

[32] A. Alatan, E. Tuncel and L. Onural "Object segmentation via rule-based data fusion," in *WIAMIS '97*, pp. 51–55, June 1997.

[33] M. Bierling "Displacement estimation by hierarchical blockmatching," in *Proceedings of SPIE Visual Communications and Image Processing 88*, pp. 942–951, 1988.