Video Object Segmentation

D I S S E R T A T I O N

for the Degree of

Doctor of Philosophy (Electrical Engineering)

Fatih Murat Porikli

April 2001

Video Object Segmentation

D I S S E R T A T I O N

Submitted in Partial Fulfillment
of the Requirement for the
Degree of

Doctor of Philosophy (Electrical Engineering)

at the

POLYTECHNIC UNIVERSITY

by

Fatih Murat Porikli

April 2001

Approved:

____
Department Head

__________________ 20___

Copy No. ____________

Approved by the Guidance Committee:

Major: Electrical Engineering

Yao Wang
Professor of
Electrical Engineering

Date

Ivan Selesnic
Assistant Professor of
Electrical Engineering

Date

Huifang Sun
Deputy Director,
Mitsubishi Electric Research Labs

Date

Minor: Computer Science

Edward K. Wong
Associate Professor of
Computer and Information Science

Date

VITA

Fatih Murat Porikli received the B.S. and M.S. degrees in Electrical Engineering from Bilkent University, Ankara, Turkey, and Polytechnic University, Brooklyn, NY, in 1992 and 1996 respectively. Since then he has been working towards the Ph.D degree in Electrical Engineering under the supervision of Prof Yao Wang, while also working full-time.

In 1992, he started his graduate studies in both Electrical Engineering and Sociology at Middle East Technical University, Ankara, Turkey. In 1993, he was entitled to receive an overseas doctorate education scholarship. Upon arriving the USA, he pursued towards his master degree at University of Southern California, Los Angeles, CA, and then transferred to Polytechnic University, Brooklyn, NY in 1994.

While his doctorate studies, he was employed at AT&T Research Laboratories, Holmdel, NJ where he worked on stereoscopic depth estimation techniques, and Hughes Research Laboratories, Malibu, CA where he developed methods of unsupervised road extraction from aerial imagery.

He joined Mitsubishi Electric Research Laboratories, Murray Hill, NJ in 2000. Upon joining Mitsubishi, he worked on algorithms for down-conversion decoding, network management, and optimal bandwidth allocation. More recently, his work has focused on the segmentation of video sequences with emphasis on automatic detection, semi-automatic tracking, and content analysis.

He holds twelve U.S. patent applications that are pending, and has published a number of papers in these areas. He is a member of IEEE that he served as a reviewer for many journal and conference papers.

To Sevda

ACKNOWLEDGEMENT

I would like to thank God for the many days that I woke up and did not know what to do, where to turn and how I would make it through. God has carried me this far by faith, joy, peace, love and satisfaction, and I wish to continue being His servant. I praise my Lord for blessing us beyond anything I could have ever imagined.

I am deeply indebted to my thesis advisor, Prof. Yao Wang, for her informed guidance, understanding, support and encouragement through the years. Her depth and vision in this area is truly admirable and she has been an inspiration to me in many ways. Her patience and willingness to discuss the details of the different obstacles I encountered while working on this thesis were invaluable. I greatly appreciate the many impromptu lectures she conducted on any available writing surface, whether it be scratch paper or a white-board.

I would like to thank Prof. Edward K. Wong and Prof. Ivan Selesnic for their precious time spent in reviewing this work and for their valuable input during its development.

Upon joining Mitsubishi, I was fortunate to work under the supervision of Dr. Huifang Sun. I am grateful to him for having the confidence to hire me, for creating such a stimulating work environment and providing guidance on a many things. He has had a significantly positive influence on my development through his role as both my manager and my advisor. Dr. Ajaj Divakaran became a mentor of mine while we worked together to develop content analysis projects. His extensive knowledge and enthusiasm as an educator introduced me to the breadth and variety of the field of content analysis. I express my gratitude to our dedicated EVP, Dr. Tommy Poon, for his leadership. I appreciate his vivid expertise and keen attention to detail, and I thank him for the encouragement and support he has given me. I also appreciate the efforts of our current CEO, Dr. Dick Waters, as he continues to redefine and unify the MERL. Working with our patent attorneys, Mr. Dirk Brinkman and

Mr. Andrew Curtin has been an enriching experience. I would especially like to thank the following people for their collaborative efforts: Mr. Tokumichi Murakami, Mr. Kohtaro Asai, Dr. Paul Ratliff, Dr. Miroslaw Bober and Dr. Leszek Cieplinski.

I feel very honored to have worked with so many distinguished colleagues during the course of this thesis. I would especially like to thank Prof. Yucel Altunbasak and Prof. Levent Onural, Prof. Enis Cetin for their valuable comments. I would also like to acknowledge my previous collaborations, which continue to provide a very integral part of my educational experience. I appreciate the various discussions with Dr. David Schwartz, Dr. Roy Matic, and Dr. Cassandra Swain. I enjoy the friendly interactions that I now have with them.

I would like to express my gratitude to Dr. Recep Senturk, for his precious friendship and guidance throughout the years. He has never tired of answering my questions. I feel very honored have good friends, especially Mr. Musab Kocak, Mr. Mustafa Egilmezbilek, Mr. Goksen Elkas, Mr. Erhan Yilmaz, Dr. Sami Ayyorgun, Dr. Abdulkadir Balikci, Mr. Levent Sendur, Dr. Kadir Peker, Dr. Zafer Sahinoglu, and many others who remind me of the value of having good friends. Mr. Musab Kocak has been a phenomenal source of new ideas and direction.

I also wish to thank everybody not mentioned here personally, but who contributed in one way or another to the success of this thesis.

I would like to thank my loving and supportive wife, Sevda, who has always supported me in my endeavors, always given me the strength and encouragement to follow my dreams, and has never left me in doubt of her love for me. Without her support, love and occasional kick in the pants, this thesis would have never been written.

Finally, and above all, I thank my mother Kudret, my father Hanefi, my brothers Ersoy, Ugur, and my sister Arzu, for their love, understanding and never-ending support.

Fatih Murat Porikli

AN ABSTRACT

Video Object Segmentation

by

Fatih Murat Porikli

Advisor: Yao Wang

Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy (Electrical Engineering)

April 2001

In this thesis, the problems associated with the automatic object segmentation of the video sequences are considered. Towards this goal, a unique framework that combines various disciplines of image and video processing techniques ranged from noise filtering to data clustering is developed. The framework also addresses a number of challenging issues associated with computational complexity, accuracy, generality, and robustness.

One of the primary aims of this thesis is to study the fusion of color, texture, motion, shape, frame difference, and other methods of video segmentation for automatic detection considering the real-time processing requirements. In contrast to frame-wise tracking techniques, the employment of a spatiotemporal data that is constructed from multiple video frames introduces new degrees of freedom that can be exploited in terms of object extraction and content analysis. The current notions of region segmentation are extended to the spatiotemporal domain, and new models to estimate the object motion are derived.

Another objective of this work is to explore techniques and algorithms that provide efficient means of preprocessing of input video sequence. The newly designed image simplification and reconstruction filters enable us to develop efficient algorithms for noise removal, and they prevent from over segmentation difficulties. The problem of adaptation of system parameters and thresholds is formulated and solved. Also, the relation between the color space and the similarity functions is investigated.

As a final objective of this thesis, a clustering problem that considers construction of meaningful video objects from color homogeneous regions is examined. Specifically, the fine-to-coarse and coarse-to-fine strategies are discussed. Novel descriptors to evaluate the quantitative and relational attributes of the extracted objects are introduced. Additionally, new sources of motion information is considered, such as the trajectory definition of an object. Also, the area of multi-resolution object representation is explored.

Contents

Chapter 1

Introduction

If you will be observant and vigilant, you will see at every moment the response to your action. Be observant if you would have a pure heart, for something is born to you in consequence of every action.

Rumi

There has been light from the beginning. In all its forms - visible and invisible - it saturates the universe. Usually, though, we don't see light, we merely see *with it.* Ancient Greeks thought that our eyes acted as lanterns, sending out rays that made objects visible when struck. This concept amazingly held for more than 15 centuries until Arab scholar Al-Hazen about A.D. 1000 made convincing arguments otherwise.

Visual information reaching us by light plays an important role in our ability to interact with the world. The eyes are probably the most important and amazing of the body senses. Without the ability to process visual information, we would be severely handicapped and it is therefore not surprising that approximately half of the primate cortex is devoted to visual information processing.

Now, we are able to capture the light not only as invaluable memories but also into various forms of media. By the turn of the century, the advancement in digital video technology is determined to change our everyday lives since it has fueled numerous applications and services which has never been possible with the analog video technology. The most noticeable is videoconferencing, and more importantly, digital television, which has been marked to replace the analog one that we are so accustomed to and has become so much a part of our everyday life.

Video grew into arguably the most popular means of communication and entertainment. With this popularity comes an increase in the volume of video data and the need for the effective techniques for analysis, mining, manipulation, and description of visual information to sift through it and search for relevant material automatically. Thus, advanced image processing, motion estimation, and object segmentation techniques have turned into highly active fields of research.

1.1 Motivation of Video Segmentation

More and more visual information is available in digital form, in various places and on various media. The emergence of digital video and its proliferation in multimedia applications has created a significant demand for content-based representation of visual information. Main purpose of video segmentation is to enable content-based representation by extracting objects of interest from a series of consecutive video frames. Briefly, motivation behind video segmentation can be categorized as the applications in indexing and retrieval, compression and coding, recognition, identification, and understanding of video scenes, editing, manipulation, and animation. Some sample uses of video segmentation are illustrated in Fig.1.1.

Video databases on the market today allow only limited capability of or domain limited searching for video using characteristics like color, texture, and simpler motion statistics. If video can be stored in the form of individual objects, indexing and retrieval of visual information is as simple as that of textual information. An essential tool in the management of visual records is the ability to automatically describe and index the content of video sequences in a meaningful manner. Such a facility would allow recovery of desired video segments or objects from a very large database of video sequences. The efficient use of stock film archives and identification of specific activities in surveillance videos are among the potential applications.

From compression point of view, video segmentation is essential for object-based video coding standards, i.e. MPEG-4. Due to the vast data size of video sequences, communicating digital video over the bandwidth limited network sources demands competent coding techniques. Having an object-based representation scheme that identifies the important parts of image frames, video sequences can also be encoded efficiently to satisfy transmission requirements. Videoconferencing is one of the applications benefit from object-based coding.

Video segmentation is key to many robotic vision applications. Most vision based autonomous vehicles acquire information on their surroundings by analyzing video. Particularly, it is required for high-level image understanding and scene interpretation such as spotting and tracking of special events in surveillance video. For instance, pedestrian and highway traffic can be regularized using density evaluations obtained by segmenting people and vehicles. By object segmentation, speeding and suspicious moving cars, road obstacles, strange activities can be detected. Forbidden zones, parking lots, elevators can be monitored automatically. Gesture recognition as well as visual biometric extraction can be done for user interfaces.

With a good segmentation, it is possible to access and manipulate objects in video. To illustrate, traffic enforcement currently employs supervised video segmentation tools to acquire identity of speeding or trespassing cars. Infotainment industry utilize video segmentation for editing, manipulating, and animation.

Although the human being can quickly interpret the embedded semantic content from the information carried by different modalities, computer understanding of visual information is still in its primitive stage. Good segmentation tools are crucial to the success of the future standards. But tasks of automatically segmenting image sequences into semantic meaningful objects prove to be very challenging. We have currently a reasonably good understanding of the basic mechanisms underlying visual information processing, still, many questions are still open to investigation, some desperately waiting for an answer.

1.2 Object: A Bridge from Pixels to Semantic

We define an object as a collection of image regions which have been grouped together under some criteria across several frames. A region is defined to be a contiguous set of pixels that is homogeneous in the features i.e., texture, color, motion, and shape. It can also be specified by a class membership function defined in color space; as a plane, a cell, or fuzzy rules. Namely, a video object is a collection of regions exhibiting consistency across several frames in at least one feature. For example, a shot of a walking person "object" would be segmented into a collection of adjoining regions differing in criteria such as shape, color, and texture, but all the regions may exhibit consistency in their motion attribute. An object may be represented by the union of several regions that may be connected or not.

Motion is a primary discriminating criterion of an object in video sequences. In this thesis, an object implies a collection of consistently moving regions that have distinguishable motion from the rest of the regions. For head and shoulders type of sequences, skin color features are integrated into definition of object to determine human body.

1.3 Elementary Categorization

There are common measures for all segmentation applications regardless of their environments: generality, quality, adaptability, complexity, and promptness. A segmentation framework is expected to meet these intervening requirements.

Ideally, an object extraction system should be generic, without any specific prior knowledge of the type of color, shape, and motion to deal with. Whereas, constraining video segmentation to specific applications enables development of robust

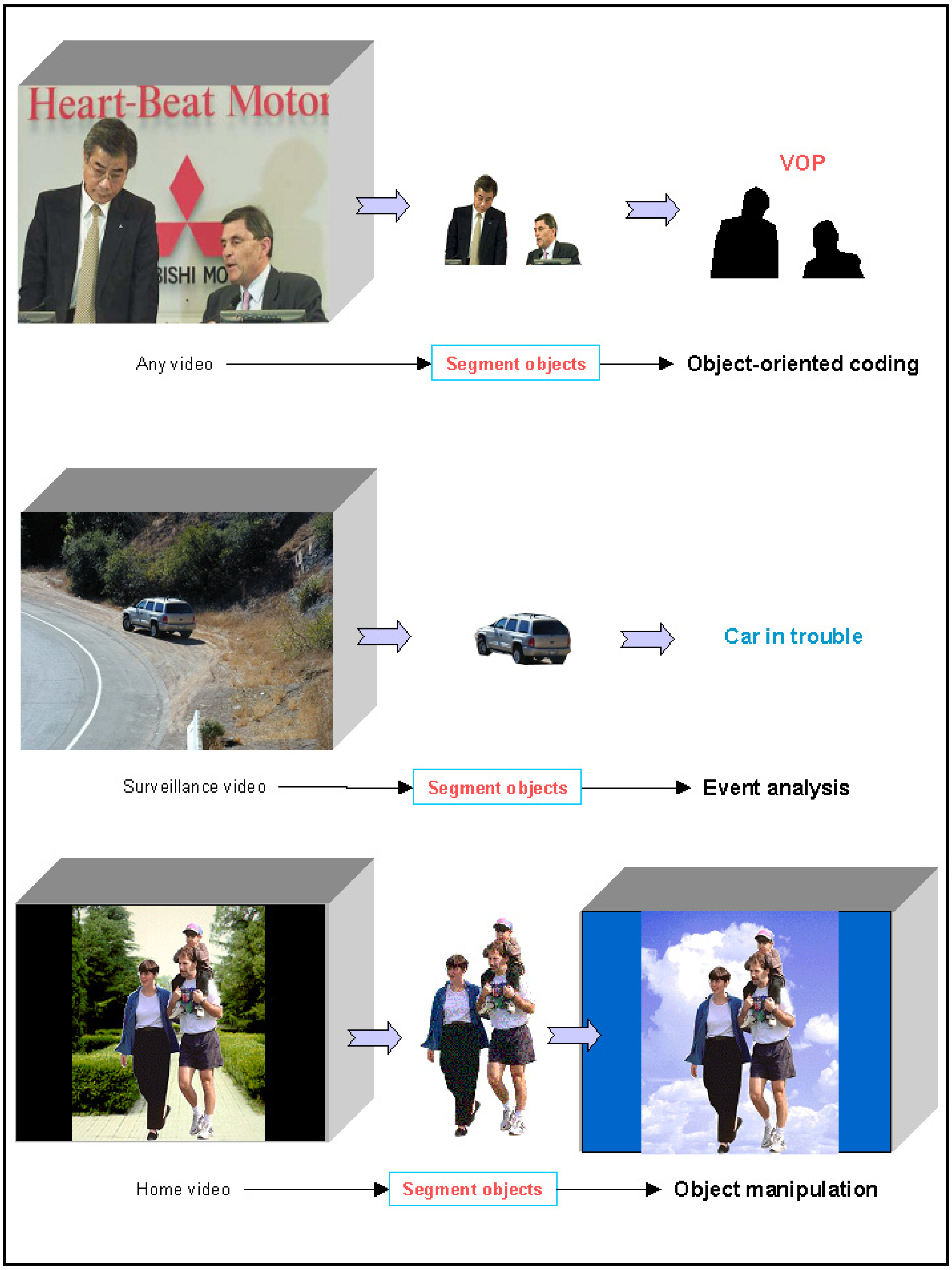


Figure 1.1: By object segmentation, it is possible to (a) extract object shape for coding, (b) analyze a video scene for event detection, (b) edit and manipulate a video.

schemes. This is particularly true for image sequences having well defined figure-background separation.

It should be able to provide accurate boundaries corresponding to a semantic entity. Applications such as in entertainment where individual objects are manipulated, require a high level of accuracy. However, in other applications, precision of segmentation depends on the context.

It is desirable to achieve automatic segmentation with no human assistance while allowing minimal user interaction to correct the possible errors produced by the system. For instance, a surveillance system requires automatic tracking and event analysis due to the cumbersome and costly monitoring of images from multiple cameras by human operators. Still, when an event is detected, the system should allow user to select object of interest, manipulate, track, etc. on it. On the other hand, some applications, i.e. coding of videoconferencing sequences, demand constant automatic segmentation.

Segmentation process should not consume an overwhelming computation power to do the job either. The efficiency and complexity of the required segmentation algorithm depend considerably on the field of application.

Real-time implementation of video segmentation algorithms is not trivial because of the intensive computations and memory requirements involved. Not all applications demand real-time segmentation. Server-side segmentation, i.e. object extraction for video libraries, MPEG-4 video object plane extraction can be carried out without pushing for promptness constraint. On the other side, applications such as videoconferencing, surveillance, gesture recognition, auto-cruising vehicle systems highly depend on prompt segmentation.

One way to speed up video segmentation is to use compressed domain clues, or achieving a pre-segmentation in the compressed domain. Since the decoding is a relatively expensive process, segmenting the object and extracting its features directly from the compressed domain would be an effective way to achieve fast algorithms for searching a large database and indexing the object. The computation in uncompressed domain is often formidable for large video databases.

1.4 Video Coding Standards

In order to compress video data for both efficient transmission and storage, various techniques and international standards have been developed. To understand why video compression is so important, one has to consider the vast bandwidth required to transmit uncompressed pictures.

MPEG is an encoding and compression system for digital multimedia content defined by the Motion Pictures Expert Group. MPEG-1 is a standard for storage and retrieval of moving pictures and audio on storage media. MPEG-2 is a backward compatible version of the basic MPEG-1 system to provide compression support for television quality transmission of digital video. The current MPEG-1 and MPEG-2 standards divide the images in small square blocks for processing to be able to adapt their performance to the non-stationary image nature. Parameters of the algorithm can be changed on block basis but real objects in a scene are never a collection of square regions. When the compression is increased this annoying block structure becomes visible in the decoded image which is also called as blocking effect.

MPEG-4 is an international standard for multimedia applications, and provides high interactive object-based functionalities, universal accessibility and robustness in error-prone environments, and compression efficiency. Unlike MPEG-1 and MPEG-2 where the coding is performed on a frame by frame basis, MPEG-4 provides an “object-layered” coding where each object is coded separately into a bitstream layer. This feature allows the access and manipulation of individual audio-visual objects in the scene. MPEG-4 standard introduces the concept of video object planes (VOP’s). Each frame of the sequence may be segmented into a number of arbitrary shaped VOPs where each one of these covers a particular video content of interest.

Thus, in object-based coding, the input to be coded is no longer a rectangular region as in block-based MC-DCT coding. Successive VOP belonging to the same object in a scene are referred to as video objects. The shape, motion, spatial coordinates and coding information of each video object are encoded into separate video object layer in order to support separate decoding of objects. The user can either reconstruct the entire sequence by decoding all video object layers, or reconstruct only some of the scene objects. The manipulation of objects, such as translation, rotation, scaling and zoom, is possible using information coded in bitstream layers. In addition, new objects that did not belong to the original scene can be included, or objects may be neglected.

MPEG-7 is a content representation standard for information search, and it is also facing the same kind of challenges. Among the large set of functionalities involved in a retrieval application, let us consider browsing. One would like to have access to a table of contents of the video and to be able to jump from one item to another. This kind of functionality implies at least a structuring of the video in terms of individual shots and scenes. Of course, indexing and retrieval involve also a structuring of the data in terms of objects, regions, semantic notions, etc.

1.5 Scope of Thesis

In this thesis, a segmentation framework is developed to find moving objects of a color video sequence in uncompressed domain. The targets of this study can be itemized as following:

- One primary objective is unifying advantages of color based region growing, frame difference based change detection masks, shape descriptors, and motion based segmentation methods into a computationally simple, adaptive, modular, and effective algorithm.
- We present an algorithm that is automatic and unsupervised. Such supervision

covers specification of number of visible moving objects as well. The algorithm determines an optimal number of objects by measuring similarity functions in a clustering stage.

- Improving computational complexity is succeeded by preventing from dense motion vector estimation, instead engaging in a simple but effective motion extraction method. We accomplished approximation of translational motion without using optical flow calculations.

- To attain generality, we target utilizing essential attributes of objects, and supplied common video processing functions. To achieve adaptability for specific applications, we devise object descriptors and adaptable system parameters.

- Modularity is another principal consideration. The segmentation framework is set into two main stages that the first stage copes with low-level video processing functions to provide homogeneous video components, and the second stage constructs video objects. As output, an object-wise multi-resolution segmentation tree that represents segmentation results for different object numbers is generated. Segmentation of the input video all over again, e.g. when the object number is changed, is avoided by employing self and mutual descriptors of homogeneous components to determine the video objects in the construction stage.

- We evaluate various video attributes such as color, motion, shape, texture to characterize video objects in terms of assigned descriptors. Descriptors are structured to incorporate priori information about the nature of input video such as MPEG-4 motion vectors as well as MPEG-7 descriptors.

- Color spaces are investigated to understand the effects of a color space on the performance of volume growing based object segmentation. Suitable color space is determined for specific applications.

1.6 Outline of Thesis

In the next chapter, the existing methodologies related to the image and video segmentation such as region growing, color histograms, motion estimation, spatiotemporal segmentation, object tracking, etc. are discussed. Their merits and disadvantages are investigated from an object segmentation point of view.

Among the segmentation methods, region segmentation has often been regarded as a first step in image analysis with applications in scene interpretation, object recognition and compression. Out of several 2D region segmentation technologies, histogram thresholding and clustering in color space, region growing, split-and-merge, texture segmentation are summarized in detail. Estimating the motion of articulated objects in image sequences is an important problem in computer vision with many potential applications including image segmentation and interpretation. Motion estimation is also utilized to eliminate the large amount of temporal and spatial redundancy that exists in video sequences. Main issues related to block-matching, feature-matching, optical flow, and motion models are highlighted. To overcome the drawbacks of motion based approaches, it is important to incorporate spatial information into motion segmentation. Under the spatiotemporal segmentation title, change detection masks, stochastic approaches, morphological techniques, and other hybrid methods that utilize both motion and spatial features are analyzed. In the remainder of this chapter, we addressed the object tracking, compressed domain segmentation, scene-cut detection, and data clustering, which are essential tools in video processing and information-theoretic.

The following preprocessing chapter evaluates statistical and structural attributes of a video sequence. A 3D spatiotemporal data structure that will serve as a basis in the segmentation framework is defined and constructed from the video sequence using its attributes. To determine a suitable color space, the effects of color spaces on the performance of region growing based segmentation are evaluated.

Several noise removal and simplification filters, and change detection masks are also presented in this chapter.

For an automatic segmentation framework, filtering has great importance on the quality of the final segmentation results. Strong noise and certain spatial texture induce over-partitioning. Over segmentation has several disadvantages; it slows down the algorithm, increases memory load by increasing the number of regions, more importantly it causes an additional problem of clustering small regions of slightly textured image parts. Although image noise can be removed by using low-pass filtering, median filtering, and morphological operators, such filters often disturb the object boundaries by smearing or completely changing the edge structure.

In the last chapter, an automatic segmentation framework is presented. This framework takes a color video sequence between two scene-cuts as an input, and generates a multi-resolution object tree as an output. By using a three dimensional volume growing technique it finds the smallest homogeneous parts of the video. These volumes are expanded from the marker points using dual or centroid linkage methods. Quantitative descriptors that represents each volume, and relational descriptors that capture the mutual properties of a pair of volumes are determined by evaluating the shape, trajectory and parameterized motion. Then, these descriptors are utilized in clustering methods to construct objects. Adaptive thresholds, distance metrics, extraction of homogeneous video components, refinement processes, estimation of trajectorial motion, quantitative and relational descriptors, construction of object tree are explained in the rest of this chapter.

Chapter 2

Background on Video Segmentation

There are two kinds of intelligence: One acquired, as a child in school memorizes facts and concepts from books and from what the teacher says, collecting information from the traditional sciences as well as from the new sciences. With such intelligence you rise in the world. You get ranked ahead or behind others in regard to your competence in retaining information. You stroll with this intelligence in and out of fields of knowledge, getting always more marks on your preserving tablets.

There is another kind of intelligence, one already completed and preserved inside you. A spring overflowing its stem. A freshness in the center of the chest. This other intelligence does not turn yellow or stagnate. It's fluid, and it doesn't move from outside to inside through the conduits of plumbing-learning.

This second knowing is a fountainhead from within you, moving out.

Rumi

Segmentation, especially video segmentation, has been a very active research topic recently. Since several disciplines are included within, even a brief of the previous work needs sizeable explanation.

Basically, video object segmentation techniques can be grouped into three classes: region-based methods using a homogeneous color criterion, motion-based approaches utilizing a homogeneous motion criterion, and object tracking. Some typical works in the color oriented domain can be classified as single-level methods or multi-level approaches. In single-level methods, people traditionally use edge-based methods, the nearest neighbor algorithm [35], or treat it as an estimation problem [28]. Although these techniques work well in some situations where the input data set is relatively simple, clean, and fits the model well, they lack generality and robustness. Recently, multi-level methods have received significant attention in the research community, e.g., split and merge [71], pyramid linking [21], and morphological methods [105]. Despite these new technologies provide better performance than single-level methods, the results are still far from perfect. The main problem arises from the fact that a video object can contain totally different colors. On the other hand, works in the motion oriented segmentation domain start with an assumption that a semantic video object has homogeneous motion. These motion segmentation works can be simply separated into two broad classes: boundary placement schemes [117] and region extraction schemes [128], [2], [90], [18], [42]. Most of them are based on rough optical flow estimation or unreliable spatiotemporal segmentation. As a result, they may suffer from the inaccuracy of motion boundaries. These two classes of methods will also fail when a semantic video object have different motions in different parts of the object. The last class of methods that is related to semantic video object extraction is "tracking" [3]. Tracking is the process to estimate the current dynamic state based on previous ones. It is the trajectories of the dynamic states that are linked in a temporal form. Many types of features, e.g., points [107], intensity edges [40], textures [15], and regions [87] can be utilized for tracking.

In summary, a single homogeneous color or motion criterion does not lead to satisfactory extraction of complete semantic visual information because each homogeneous criterion can only deal with a limited set of scenarios. A semantic video object may contain multiple colors and multiple motions. Therefore, any single criterion could only lead to a partial solution for semantic visual information extraction. To solve this problem, detecting shapes via user-selected points using an energy formulation has been suggested [52]. This however requires human assistance to obtain the final results.

In the remaining of this chapter, several segmentation approaches are discussed in detail. The next section is dedicated to image segmentation, which is one of the most fundamental concepts of video processing and encompasses a wide spectrum of techniques; vector-based methods, region growing, split-and-merge, morphological operators, edge and texture-based segmentation. These intra-frame techniques often utilize local color statistics, spatial aspects of color distribution, neighborhood constraints as well as color histograms. Vector-based methods can be broadly categorized as color histogram thresholding and clustering of color vectors. Inter-frame motion estimation is a well known problem in computer vision with many potential applications. Out of several motion estimation algorithms, block-matching, feature point matching, optical flow, parametric and non-parametric motion models are presented next. On the other hand, spatiotemporal segmentation methods attempt to utilize both motion and spatial information to partition image sequence. These methods are assorted as change detection mask approaches, stochastic approaches, morphological approaches, and hybrid approaches. Another type of spatiotemporal segmentation tracks video objects between the consecutive frames depending whether objects are rigid, nonrigid, or have no regular shape. Unlike the above methods, compressed domain segmentation methods use only compressed domain video features such as DCT coefficients, motion vectors. The following section introduces an essential factor of video segmentation; detection of scene-cuts. In the final section, data clustering methodologies that are extensively utilized in segmentation are summarized.

2.1 Region Segmentation

Image segmentation is a partitioning of an image into related regions, and has often been regarded as a first step in image analysis with applications in scene interpretation, object recognition and compression. Color similarity, texture coherence, and edge properties are some of the metrics used to determine locally homogeneous image regions. In general, most of the region segmentation algorithms fall under color-based and texture-based methods.

Approaches to color-based segmentation range from empirical evaluation of various color spaces [95], to clustering in feature space [14], to physics-based modeling [79]. An appropriate color representation provides a more efficient way of dealing with color information. Thus, the first step of color processing is the selection of a suitable color space. There are a wide variety of segmentation techniques that have been used for color image segmentation, their algorithms are generally edge oriented, region oriented and clustering oriented, which are either based on the concepts of similarity or discontinuity of feature values at the pixel level.

Edge-based segmentation takes into account that edges are boundaries between segments, edge detection is used to detect edges based on searching local discontinuities, tracing algorithms are applied to link edges into continuous and connected segment boundaries. Edge detection techniques are suitable for detecting linear features in the image, they have the disadvantage of producing low level segments even after considerable processing, detected edges sometimes may not form a set of closed curves which surround connected regions.

Region-based segmentation separates color image into homogeneous regions by its color feature similarities. Thresholding, region growing, region split-and-merge, dilation/erosion are standard region-based segmentation techniques. Region-based segmentation has the advantage of producing high level segments, however, extracted region may not correspond to actual physical objects unless the intensity of each pixel in regions differs from background.

Vector-based methods are divided into two groups; histogram thresholding and vector clustering. Histogram thresholding identifies one or more peaks within the color histograms. Surrounding intervals in these histograms are then utilized in a pixel classification process. Clustering-based segmentation is in fact a procedure of unsupervised pattern classification. It refers to grouping a given set of color vectors into subsets. A subset is required to contain vectors that are similar to one another in color properties than to vectors in other subsets. Similarity is measured using color distance from a vector to a subset center. Still, the result of clustering is seriously affected by the initial values at the beginning of clustering. In some techniques, fuzzy membership functions is evaluated for all pixels and for all fuzzy subsets defined. Then, hard subsets of pixels are obtained by defuzzification process and subdivided into connected regions.

2.1.1 Histogram Thresholding in Color Space

Histogram based segmentation techniques attempt to remedy a number of deficiencies on simple thresholding by automating threshold selection and coping with multi-modal distributions as illustrated in Fig. 2.1. Additionally, histogram segmentation compensates for shifts in mean intensity level since the image intensity histogram distribution is considered, rather than examining the intensity values directly, as in thresholding. Hence, histogram segmentation is invariant to additive intensity variation. Histogram segmentation has no explicit notion of connectivity, here is an implicit assumption that pixels with similar intensities belong to the same regions, while this may not true in general.

Chu [33] models the image intensity histogram distribution of the object and background as the combination of two normal distributions, the background being the most significant. After determining the background mean and standard deviation, a threshold for a given pixel belonging to the background is chosen. By thresholding, pixels are segmented as object or background.

Bonsiepen [16] proposed single threshold computation in bimodal histograms. A scalar feature is extracted from the rgb parameters to unify three color bands. A threshold was found in the minimum point of the very well separated bimodal histogram built for the feature. Obviously, due to the confidence limit, some pixels

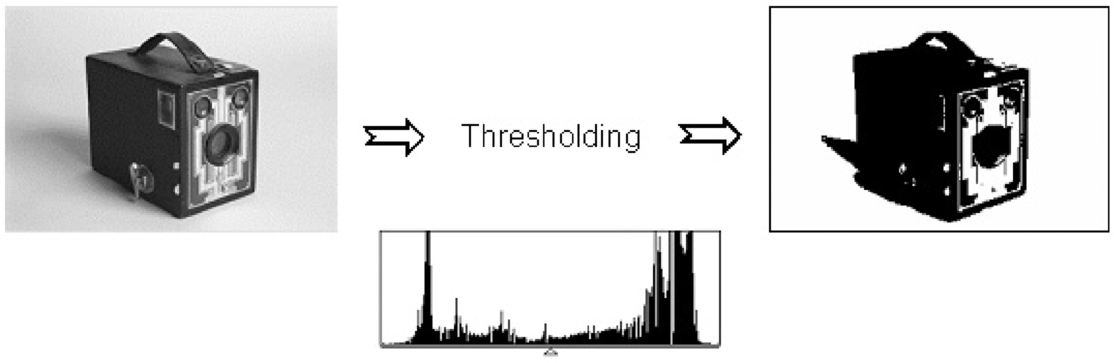


Figure 2.1: By thresholding in color space, an image is divided into two parts with respect to its color histogram.

will always be labeled as object, irrespective of whether an object is actually imaged.

Ohlander [94] used nine features collected from RGB, HSI, and YIQ color systems to select the best peak from color histograms. First, all peaks in the set of histograms are located. The list of peaks of the lowest priority is built. The best peak on this list is determined and threshold values are chosen on each side of this peak. Connected regions are selected by using the thresholds, and this process repeated until no image point remains. Basic scheme is improved by removing of small regions, working on reduced image version, adding density of edges as another feature.

Ohta's [96] algorithm is essentially very similar to Ohlander's, but he suggested new data structures for more efficient implementation in both time and space. He applied the Karhunen-Loeve transformation to color images to derive features with large discriminatory power. Ohta determined from experiments on eight color images that an effective set of color features is given by

$$f_1 = \frac{R+G+B}{3}, \quad f_2 = R - B, \quad f_3 = \frac{2G - R - B}{2} \tag{2.1}$$

where f_1 is the most effective for segmentation and f_3 is the least effective. If any of the histograms show conspicuous peaks, a pair of cut-off values which separate the peak in the histogram are determined at the positions of valleys, and the image of the color feature corresponding to that histogram is thresholded using the cut-off values. Ohta experienced with Ohlander's algorithm to determine empirically whether a small number of color features might be adequate for segmentation. Though Ohta's analysis improves the Ohlander algorithm, it is insufficient to claim that eight images will be enough to lead to the determination of color features which are universally effective for segmentation. Another misleading argument on Ohta's color features is that the most significant color feature f_1 depends on the object geometry, i.e., intensity changes due to the shape of the objects.

Finding peaks and basis in the 2D histogram of the opponent color pairs is proposed by [57]. RGB values are transformed to the opponent color pairs red-green RG, yellow-blue YB, and the intensity function I. The three channels are smoothed by applying band-pass filters. Then peaks and basis in the 2D RG-YB histogram are searched for. Peaks and basis points determine areas in the RG-YB plane. Pixels falling into one of these areas create one region. Pixels falling into another area create another region. Due to this definition there remain some non-attached parts in the image. Holla suggests to include additional features as luminance or the local connection of pixels into the segmentation process to enhance the result. A similar approach is reported by Stein [111] that an additional refinement process is employed. If one or more pixels in the $3x3$ neighborhood of a non-assigned pixel are assigned to the same region, the pixel is marked for assignment to this region. No decision is made if none of the pixels in the $3x3$ neighborhood is assigned or if several pixels in the $3x3$ neighborhood belong to different regions.

Lin and Chen [74] selected the HSI space for road following and compared the results with those computed in the RGB space. Assuming that roads appear bright and with low saturation and non-road areas correspond with low intensity and high saturation, they reduced the segmentation process to a 1D search problem.

Tominaga [118] presented a method that finds peaks and basis in the 1D histograms of the three components HSV of the Munsell space. Since no analytical formula exists for the conversion between the CIE standard system and the Munsell system, conversion is based on a table. First, the entire image is regarded as one region, and histograms are computed for each attribute of H, S, and V. The histograms are smoothed by an average operator. Then, the most significant peak is found in a set of three histograms. The peak selection is based on the shape analysis on the histogram. After some clear peaks are selected, a criterion function is calculated for each candidate peak

$$f = \frac{S_p T}{R F_p} \tag{2.2}$$

where S_p denotes a peak area between two valleys v_1 and v_2, F_p is the full-width at half-maximum of the peak, and T denotes the overall area of the histogram, that is the total number of pixels in the specified image region. R is the full range of the histogram. The image is thresholded using two thresholds derived from the lower bound v_1 and the upper one v_2 for the most significant peak in the set of three histograms. And then image region is partitioned into two sets of subregions; one consists of subregions corresponding to the color attributes within the threshold limits, and the other is a set of subregions with the remaining attribute values. The thresholding process is repeated for the extracted subregions. If all the histograms become monomodal, the cluster detection is to be finished and a suitable label is assigned to the latest extracted subregions. The segmentation process is terminated when an area of the regions is sufficiently small in comparison to the original image size or no histogram has significant peaks.

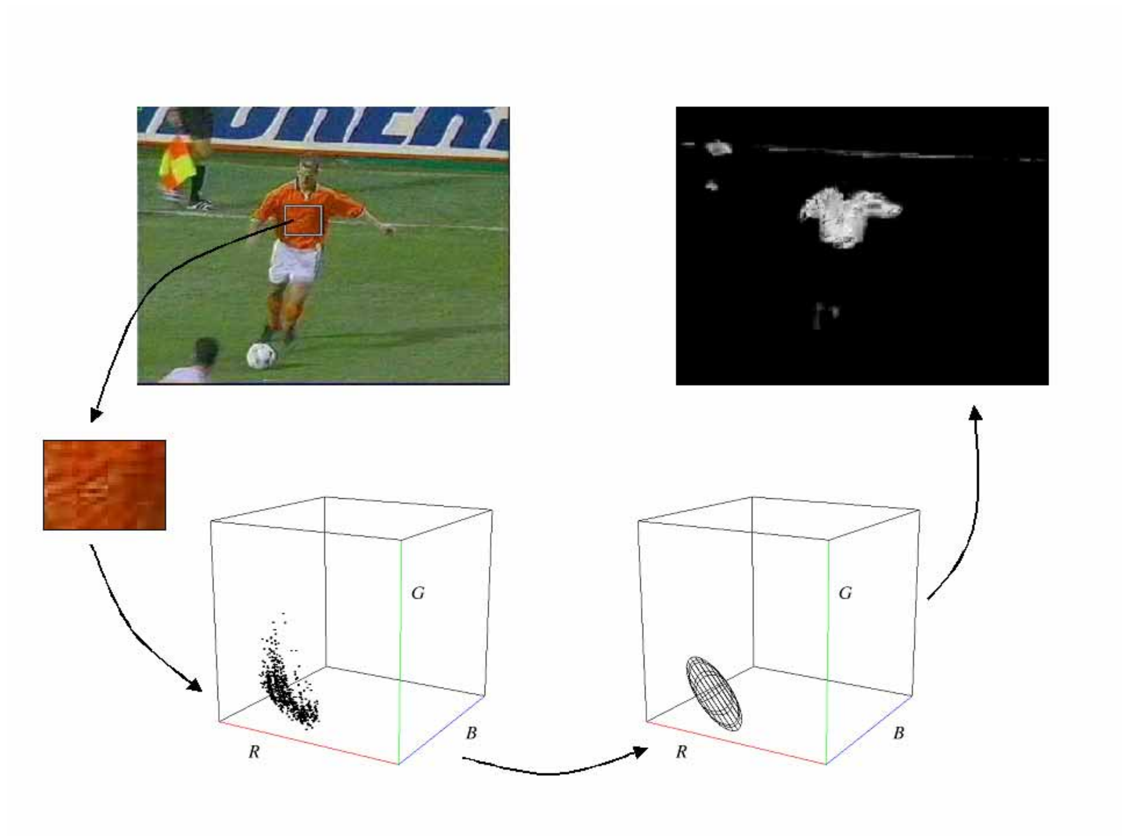


Figure 2.2: The color cluster of the target region is obtained and modeled for segmentation.

2.1.2 Clustering in Color Space

Clustering-based segmentation refers to grouping of a given set of color vectors into subsets, and is a procedure of pattern classification as illustrated in Fig. 2.2. A color clustering-based method using the nearest neighbor algorithm is proposed by Ferri [45]. Supervised clustering is performed in a 10 dimensional color feature space consisting of feature vectors based on the chromatic components of YUV color space. He extracted prototypes of objects in terms of color feature vectors using manually segmented training images. The number of the prototypes is decreased by applying a condensation technique. This algorithm is specified for certain images, e.g., a scene consists of leaves, fruits, and sky. It requires information of the object shape to decide prototypes, therefore not suitable for general purpose, automatic segmentation.

Another specific purpose color segmentation algorithm is developed by Umbaugh [121] for skin tumor identification. Representatives are obtained by median splitting process in Karhunen-Loeve transform in color space. Namely, at each subdivision step the most occupied box is chosen. The axis with the maximum range is taken and splitted in median point on that axis. The subdivision is continued until the specified number of boxes is obtained. Then representatives are taken as gravity centers. Pixels are classified by minimum distance to gravity centers. The authors report the best classification results for chromatic coordinates, but from the text we can only guess that a sort of rgb was used. The knowledge base was defined by dermatologists.

Skarbek [109] used principal component analysis. The rgb space is preferred and color statistics, e.g., directional variance, are computed by using all the image points. Pixels in a region are assigned to a leaf in the median tree. This tree is created by splitting of the color space by a plane perpendicular to the direction of biggest variance going through the median point. Splitting is terminated if a uniformity condition becomes true.

In [24], segmentation is considered as a recursive cluster detection problem.

The method operates in the *Lab* color space and detects image clusters by fitting some cylindrical decision volumes. The peaks and valleys in the 1D histograms of L, a, and b are found, and boundaries of the clusters are determined. Boundaries of the decision elements consist of two planes of constant lightness, two cylinders of constant chrominance, and two planes of constant hue. The detected clusters are then isolated from their neighbors by projecting their estimated color distributions onto the Fisher linear function for 1D thresholding.

Lucchese and Mitra [76] used 2D k-means clustering using color information, and then associating these clusters with appropriate luminance values, using 1D k-means algorithm.

In [56], Hild defines knowledge indexing as finding relevant pieces of knowledge in a knowledge base with the help of a set of descriptive properties, called index. There is an affinity between indexing and matching. While matching refers to establishing correspondences between a set of properties of a stored model, the role of indexing is to establish connections to models or parts of models without verifying the appropriateness of these connections. Thus, an index can be used as a hypothesis of the object depicted in the image. The segmentation process is used to find relevant features that can be used as indices into a knowledge base. The author uses the HSI color space, and propose to divide the hue into 19 different partitions. Then, the image is projected into these partitions and the pixel density is calculated in each of the partitions. Hue clusters are grouped since they define chromatic features that can be used as indices into a knowledge base.

2.1.3 Region Growing

Taylor [116] proposed a region growing technique starting from seeds and extending regions by one pixel in each step as simulated in Fig.2.3. Then, a boundary relaxation method is used as a correction technique for already coarsely segmented image. He utilized an HSI alike color space. Region growing starts from a seed in

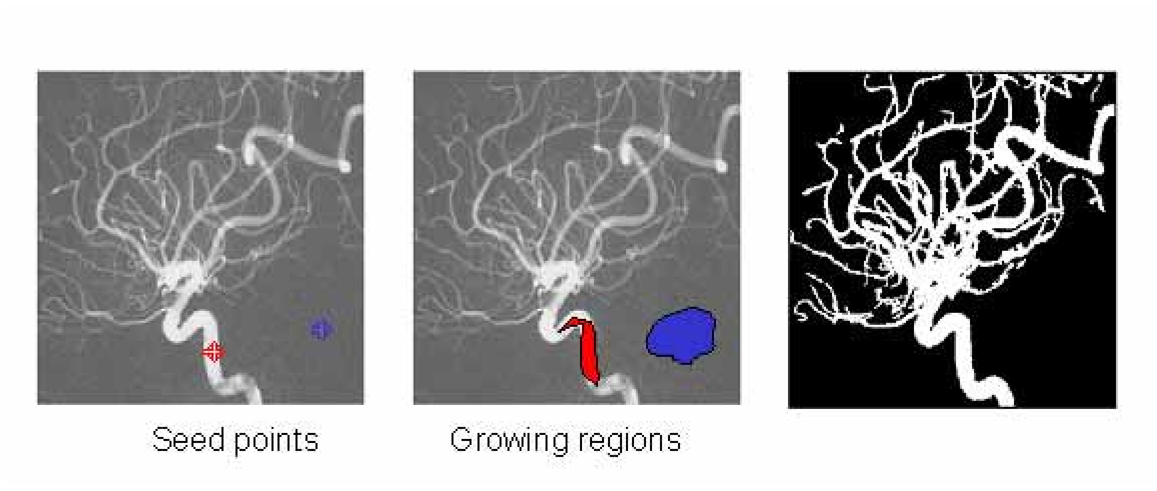


Figure 2.3: In region growing, a region is grown by similarity criteria around a seed point.

the form of 2×2 image blocks. An adjacent pixel p is added to region R if its color distance $\Psi(R, p)$ from the centroid is less than a threshold ϵ that is experimentally set to 10% of the distance between black and white points in the color space. Boundary relaxation is done by each pixel as follows: If $p \in R_a$ and p is adjacent to R_b and moving p from R_a to R_b reduces $\Psi(R_a, p) + \Psi(R_b, p)$ and maintains $\Psi(R_b, p) < \epsilon$ then move p from R_a to R_b. If there is more than one possible move then the one giving the lowest Ψ across all regions involved, is performed.

In Meyer's [85] algorithm, region growing is done using the topographic watershed algorithm in the RGB space. The basic idea of the watershed algorithm may be understood as a flooding process of this topographic surface. At each minimum of the map a water source is defined. Then water is pumped from each water source. During this procedure, the level of each lake increases with uniform speed. Water sources or seed points for region growing, respectively, might be found by calculating the minima of the gradient image. Starting from these seed points a hierarchy for region growing is established based on the differences between the intensity value of the seed point and the intensity values of the neighboring pixels. First, all neighboring pixels of a seed point get the same label as the seed point itself if their intensity difference is zero. Afterwards, all neighboring pixels of a seed point get the same label as the seed point if their intensity difference is 1. Then the difference level is further increased and the procedure is continued until all pixels in the image are labeled. When this scheme is applied to color images, the seed points are found by calculating the minima of the gradients in all three color channels. Due to computational efficiency the author employed the maximum norm to measure the color difference between two pixels. Some additional information is needed to avoid over-segmentation. It has to be known in advance which minima are relevant and which are not.

Priese [99] designed a bottom-up region growing and top-down separation technique. In a preprocessing step, color values are smoothed by replacing them by the mean value of the adjacent pixels in a hexagonal structure. The segmentation is based on a hierarchical organization of hexagonal structures. At the lowest level, local region growing is applied and connected regions of small size are obtained. These connected regions are treated as pixels at higher levels. The distance between the color contents of R_1 and R_2 is measured to decide whether two neighbored regions R_1 and R_2 become connected. The decision whether to combine or not is formulated as a predicate $\Psi(R_1, R_2)$. As the color models are 3D, distance metric Ψ is a 6D predicate. Instead of using metrics, Priese suggests to use a navigation in these 6D spaces to compute whether Ψ holds. As the level n in the hierarchy may influence Ψ; Ψ is treated as a 7D predicate $\Psi(R_1, R_2, n)$. This predicate is only computed for those regions that are candidates for merging to avoid mismatches. One simple criterion for being candidates is that both regions have a common boundary. To create a second criterion, both regions R_1 and R_2 are linked by a chain $p_1 ... p_k$ of pixels that are pairwise connected due to $\Psi(p_i, p_{i+1}, 0)$ for $1 \leq i < k$, $p_1 \in R_1, p_2 \in R_2$. If two candidates R_1 and R_2 are not similar according to $\Psi(R_1, R_2, n)$, a mismatch of their aggregated color values is detected despite a linking chain between R_1 and R_2.

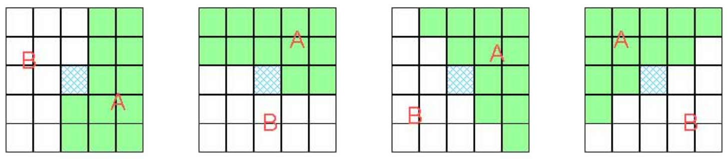


Figure 2.4: The partitioning of the 5×5 blocks in Fesharaki's method.

Image pixels are merged based on the minimum Euclidean color distance in a four-connected neighbourhood in Vlachos's work [124]. The process of image segmentation is considered in a graph-theoretic context. The suppression of artificial contour is formulated as a dual graph-theoretic problem. A hierarchical classification of contours is obtained which facilitates the elimination of the undesirable contours. Regions are represented by vertices in the graph and links between geometrically adjacent regions have weights that are proportional to the color distance between the regions they connect. The link with the smallest weight determines the regions to be merged. At the next iteration of the algorithm the weights of all the links that are connected to a new region are recomputed before the minimum-weight link is selected. The links chosen in this way define a spanning tree on the original graph and the order in which links are chosen defines a hierarchy of image representations.

The algorithm presented by Fesharaki [46] is based on testing the homogeneity of pixels around a centre pixel by using statistical inference techniques. A 5×5 window around each pixel is partitioned into two subregions in four different orientations as shown in the Fig. 2.4. Two hypotheses are formulated about pixels in the neighborhood. H_0: all pixels in the given neighborhood belong to one homogeneous region, or H_1: they belong to at least two different homogeneous regions. If the null hypothesis is accepted, the center pixel in the window is considered to be part of a homogeneous region. Assuming that A and B are mutually independent and come from two populations, the null hypothesis may be stated as $H_0 : P(A \leq \alpha) = P(B \leq \alpha)$ for all confidence levels α. This means if the difference between the two cumulative distributions is less than some threshold that is correlated to α, the adjacent subregions are similar. To verify this hypothesis, the cumulative distribution functions of two subsamples are compared with each other. The Kolmogorov-Smirnov statistics considers the base for the verification of the homogeneity of the two subsamples since it is more suited for comparing the cumulative distributions of small samples of populations. The results obtained in the three color channels are blended by Boolean "and" operator. In region growing, each pixel is considered as a region. A 2×2 window containing the left and above pixel is moved across the image. Neighboring pixels are candidates for merging if the related signals are set. For example, each pixel is allowed to merge with its adjacent left pixel, if either its left merge signal or the homogeneous signal computed from its left pixel is set. The same procedure is carried out for merging the current pixel with the pixel above. If it is not allowed to merge with its adjacent pixels, a new label is assigned. If it is allowed to merge with both its left and above pixels, the minimum label of those two pixels is assigned to that pixel.

Ji and Park [66] used the artificial neural networks to merge homogeneous regions based on the information of the luminance, the chrominance difference, and the region proximity. Wu and Reed [131] used a region-growing method, a Gibbs-Markov random field model and contour relaxation.

Westman's approach [130] consists of two processing steps. First, pixel values are substituted by averaging the values of four nearest pixels. A nearest neighbor of a pair is the one closest in color value to the centre value. In the modified version, a tolerance for local directional contrast is used to detect when a center pixel is part of a thin structure that outlies the symmetric pair lying in a background. If the center is an outlier (ridge, valley) with respect to a pair, then the center value is left

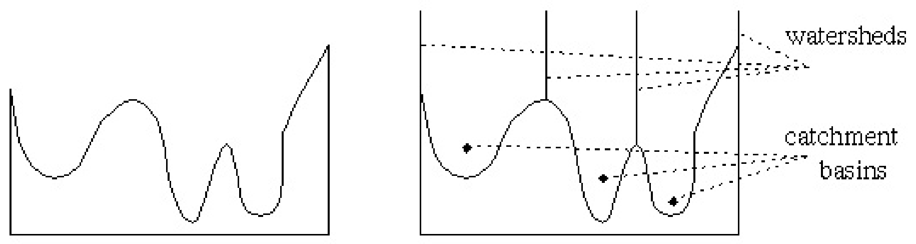


Figure 2.5: Watersheds correspond to boundaries between regions.

fixed. After, a region growing technique is applied based on hierarchical connected components analysis. In the first pass, all pixels are assigned labels by comparing them with the four adjacent labeled pixels above and to the left (8-connectivity). Adjacent pixels are merged if their color difference is lower than a predetermined threshold value. For each region the labels of adjacent regions, the average edge contrast between adjacent regions, and the lengths of the borders between adjacent regions are stored in a region adjacency graph. In the second stage, for each region starting from the one with the smallest label, the neighboring regions with an average edge contrast smaller than a threshold epsilon are merged and the adjacency graph is updated. The region-merging step is iterated using gradually increased epsilon values if the multi-stage version of the algorithm is employed. The authors obtained about equally good results employing the maximum metric as employing the Euclidean color metric.

2.1.4 Morphological and Edge Based Techniques

Watershed is a morphological technique derived from elevation maps, the physical analogy is to regard the image intensity as a height, a “rain drop” falling on a given pixel will collect at a given point, pixels sharing the same collection point will be grouped together. The boundaries between collection regions are termed watersheds, image segments correspond to the catchment basins delineated by the watersheds as in Fig.2.5. Watersheds are formed from the influence of two or more "catchment basins", hence the location of a particular watershed cannot be determined from consideration of a single catchment basin, i.e. watershed location is entirely dependent on the interactions between image structures. If an image contains only a single catchment basin, then no watersheds will be formed. Watersheds of image intensity appear to be independent of gradient magnitude, but dependent on gradient direction and curvature. Due to the independence of watersheds to gradient magnitude, they may fail to provide a physically desirable segmentation in the presence of a flat plateau. The watershed technique is invariant to additive and multiplicative intensity variation and does not require an explicit threshold. From the examples shown in [123], it appears that watersheds may tend to "overthreshold" intensity images, but may be more applicable to gradient images.

In edge based segmentation, edge pixels are joined to delineate image regions, e.g. [44] which uses a technique known as sequential edge linking. Edge linking techniques suffer from a number of problems, since not only must edge pixels be linked into edge lists, these lists must also be linked so as to extract closed regions. Indeed, the closure problem is not straightforward to solve, since many edges may terminate in the same location or conversely large gaps may need to be bridged between edges.

2.1.5 Split and Merge Techniques

Split-and-merge algorithms start from nonuniform regions, and subdivide them until uniform ones are obtained, then apply some merging heuristics to fit them into a maximal possible uniform area.

One class of algorithm attempts to merge "atomic" image elements so as to form image segments, avoiding the necessity for an explicit split function, e.g. by using uniform intensity image regions as "atomic" elements. Generally some form of mergestrength function is defined for adjacent regions, based on e.g. similarity in intensity distribution, gradient/edge strength between segments, etc. Lim and Park [73] merge adjacent regions with the highest mergestrength first, followed by a recalculation of the new inter region mergestrengths. These steps are iterated until some appropriate threshold is obtained. Quadtree techniques (Fig.2.6) are a subclass of split and merge segmentation schemes, they perform probability tests on image regions as

1. Split into four disjoint quadrants any current image region R_i if $P(R_i) = 0$,
2. Merge any adjacent regions R_j and R_k for which $P(R_j \cup R_k) = 1$,
3. Stop when no further merging or splitting is possible.

where $P(R)$ is the probability that the image region R is a single image segment. The probability test P is generally based on statistical properties of the region intensities under consideration, e.g. Choo *et al.* [32] use mean and standard deviation of image intensities, however, any region test should be invariant to block size.

In Schettini's work [106], splitting is made by sequential histogramming for five color features from the *Luv* space as in [94], but with another peak detection technique. At the fixed scale peak of histogram is chosen using Tominaga's [118] criterion. The best peak is chosen for all five histograms. All pixels with color between arguments giving half peak are taken to one region from which connected components are extracted. The procedure is repeated recursively for all remaining pixels. At the end of splitting phase, the very small regions are merged to adjacent one with the closest color. Global dissimilarity measure $\psi(ij)$ of two adjacent regions is defined using distance Ψ_{ij} and region adjacency measure C_{ij}

$$\psi(ij) = \frac{\Psi_{ij} C_{ij}}{4} \tag{2.3}$$

$$C_{ij} = \frac{\min(P_i, P_j)}{4P_{ij}} \tag{2.4}$$

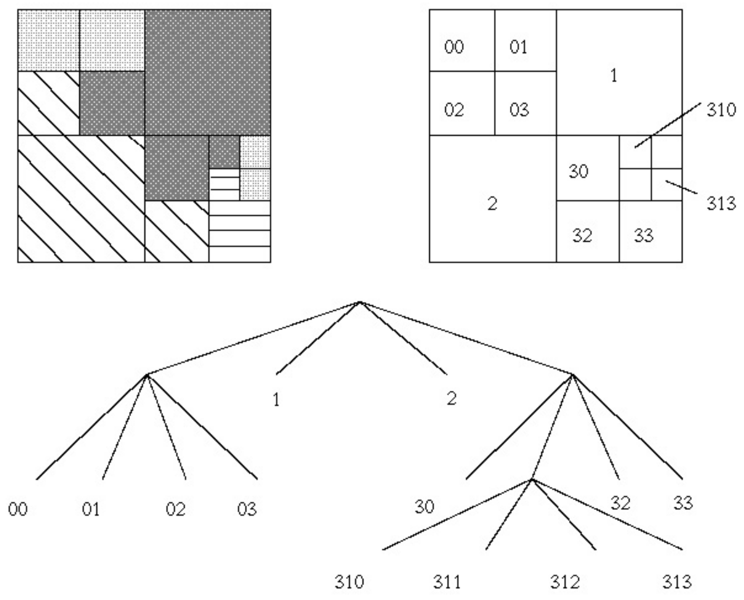


Figure 2.6: Quadtree representation of block-wise region splitting.

$$\Psi_{ij} = \max_{k=L,u,v} \Psi_{ij}(k) \tag{2.5}$$

$$\Psi_{ij}(k) = \frac{\sqrt{N_i + N_j}\bar{x}_{ik} - \bar{x}_{jk}}{\sqrt{N_i s_{ik}^2 + N_j s_{jk}^2}} \tag{2.6}$$

where P_i is the perimeter of i^{th} region and P_{ij} is the perimeter of common boundary i^{th} and j^{th} regions. In one step of the merging phase two adjacent regions are joined if their dissimilarity measure $\psi(ij)$ is minimum. The merging phase is terminated if there is no candidates for merging. Two adjacent regions are candidates for merging if the following inequalities are true

$$\Psi_{ij} < 4, \quad 0.5 \leq C_{ij} \leq 2, \quad \psi(ij) < 1. \tag{2.7}$$

The perceptual HSI space is employed in [120]. He suggests to split up the color image into chromatic and achromatic areas to determine effective ranges of hue and saturation. The criteria for achromatic areas were measured by experimental observation of human eyes and are defined as follows

$$\begin{aligned}
case\ 1: &\quad I > 95 \quad \vee I \leq 25\\
case\ 2: &\quad 81 < I \leq 95 \quad \wedge S < 18\\
case\ 3: &\quad 61 < I \leq 81 \quad \wedge S < 20\\
case\ 4: &\quad 51 < I \leq 61 \quad \wedge S < 30\\
case\ 5: &\quad 41 < I \leq 51 \quad \wedge S < 40\\
case\ 6: &\quad 25 < I \leq 41 \quad \wedge S < 60
\end{aligned}$$

After an image has been segmented into chromatic and achromatic regions, chromatic regions are further segmented using hue histogram thresholding. Achromatic regions are segmented using intensity histogram thresholding. If the saturation histogram of a chromatic region has obvious variation distribution, the region will be split based on the saturation histogram. The significance of a peak is measured according to a sharpness function

$$S = \frac{T_p}{W_p} \tag{2.8}$$

where T_p denotes the total number of pixels between two valleys and W_p is the distance between those valleys. A peak is chosen if its S value is greater than the predefined thresholds (in the experiments $S > 256$ for hue histograms and $S > 1536$ for intensity histograms). A further process is applied to avoid over segmentation. An 8×8 mask is evenly divided into 16 2×2 submasks. If there is at least a chromatic and an achromatic pixel in the 2×2 submask, then the submask has a vote to the dispersion of the mask. A special label is assigned to the 8×8 region if the mask possesses more than seven votes. After convoluting the mask throughout the image, region growing is used to merge the labeled regions with the segmented regions or to form some new regions based on the idea of clustering in the HSI space.

2.1.6 Texture Segmentation

Among other region properties such as color, shape, or motion, texture is one of the most prominent image attributes in both human and automatic image analysis. Fig. 2.7 shows a sample texture segmentation result. There are a number of different definitions for texture. What all definitions have in common is the fact that they describe texture as an attribute of an image window. This attribute represents spatial arrangement of the gray levels of the pixels in a region, and provides a measure of properties such as smoothness, coarseness, and regularity. Texture is also described as extended patterns based on the more or less accurate repetition of some unit cell. As such, textures have statistical properties, structural properties, or both. They may consist of the structured or random placement of elements, but also may be without fundamental unit cells. With respect to relevance with biological vision, automated texture segregation efforts have resulted in two broad categories: human-vision-related and pure machine vision texture segregation models. In the first belong models that are designed to produce results that correlate well with human performance, when tested with the same classes of stimuli. The algorithms and models in the second category perform texture segregation on input images without necessarily employing neurophysiological principles. There are numerous broader theoretical reviews on texture analysis algorithms [54], [101]. The feature set of Haralick et al. [53] is probably one of the most famous methods of texture analysis. It is based on the calculation of the co-occurrence matrix, a second-order statistics of the gray levels in the image window. In order to reduce the computation time necessary to calculate the co-occurrence matrix, Unser has suggested a method to estimate its coefficients using a first-order statistic on the image window [122]. He suggests sum and difference histograms on the gray levels. Galloway has proposed a run-length based technique, which calculates characteristic textural features from gray-level run lengths in different image directions [50]. The basis of the calculation of the features is a run-length matrix. Sun and Wee [115] define five features from a modified Haralick approach.

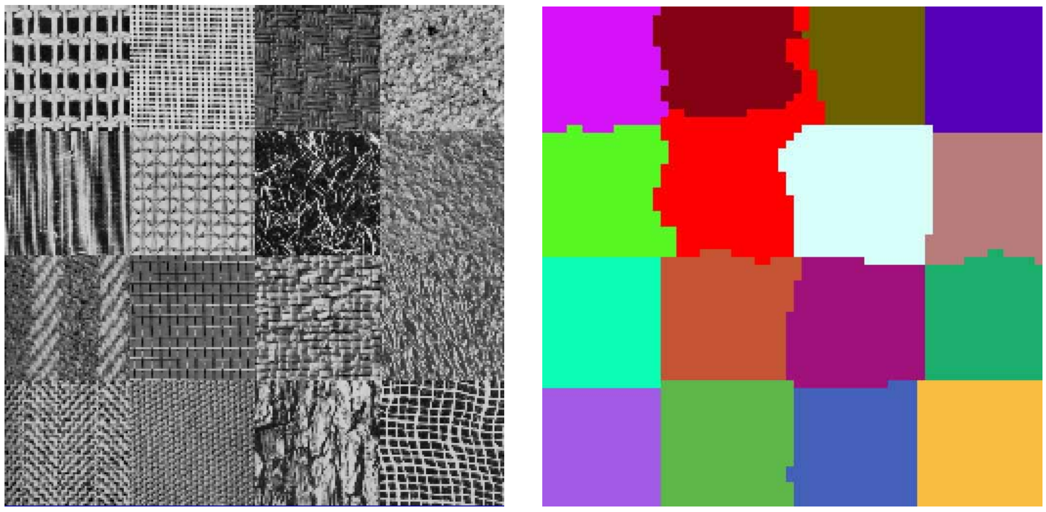


Figure 2.7: An image consisting 16 textures, and its texture segmentation.

They calculated statistics in an 8-neighborhood and are therefore less dependent on image rotations. Chen [29] decomposes a gray-level image into a sequence of binary images and calculates features from geometric properties of the resulting blob regions. Laws has suggested a set convolution masks for feature extraction [72]. There are five 1D filter masks with the labels “level,” “edge,” “spot,” “wave,” and “ripple”. From them, various 2D filter masks can be constructed. Pikaz and Averbuch [98] calculate textural features from a sequence of graphs of the number of 4-connected structures in the binarized image. In addition to the above methods, local texture features that contain statistical features and gradient features are also calculated directly from the original image window.

Bergen and Adelson [12] use the local energy of linear oriented filters at various scales to account for scale invariance in texture segmentation. Good fits are also achieved by Sagi and his colleagues, who use a similar energy approach [103]. Sperling [110] differentiates between first-order linear and second-order nonlinear rectifying regimes. The same analogy is made by Clark, Bovik, and Geisler [19], who use the spatially pooled amplitude and phase responses of Gabor filters; they recognize that their implementation offers primarily a machine vision method, as does that of Voorhees and Poggio [125]. Fogel and Sagi [48] and many others suggest the use of Gabor wavelets for feature extraction. Gabor wavelets can be used to extract a certain wavelength and orientation from an image with a specified bandwidth. Because the Gabor filter is a quadrature filter, the "energy" of the signal in the filter band can be determined by computing the square magnitude of the complex filter response. Laine uses a wavelet decomposition with Daubechies wavelets and takes the energies in the different filter bands as features [49]. While many wavelet transforms do a multi-resolution analysis by successively transforming only the lowest-resolution subimage of each decomposition step, Laine has decided to do the decomposition for all subimages successively. For extracting textural features, an energy value is calculated for each subimage in a decomposition sequence by summing over the squares of all pixel values in the subimage.

In Kashyap et al. [68], a Markov random field approach is chosen to extract seven different textural features. Markov random fields model a texture by expressing all gray values of an image as a function of the gray values in a neighborhood of each pixel. Amadasun and King [5] use an approach similar to that of co-occurrence matrix, but do not construct a 2D co-occurrence matrix. Instead, they average the differences for each gray-level value of the central pixel and construct features from this histogram distribution. Malik and Perona [77] model human pre-attentive texture perception in three stages which have close parallels with Caellis [22] model: i) Convolution of the image with a bank of orientation- and frequency-tuned filters, followed by half-wave rectification, to "model outputs of V1 simple cells." ii) Inhibition, localized in space within and among the responses, to suppress spurious responses in the presence of stronger ones. iii) Spatial pooling and texture edge extraction by odd-symmetric mechanisms.

2.2 Motion Estimation

Estimating the motion of articulated objects in image sequences is an important problem in computer vision with many potential applications including image segmentation and interpretation. Motion estimation is also utilized to eliminate the large amount of temporal and spatial redundancy that exists in video sequences. In conventional predictive coding the difference between the current frame and the predicted frame based on the previous frame is coded and transmitted. There are a large number of motion estimation algorithms for predictive coding and video segmentation.

2.2.1 Block-Matching

The most frequently used motion estimation algorithms are block-matching algorithms. The main idea of block-matching is to divide the current frame into blocks and then find the best match block in the previous or next frame for a given matching criterion. All pixels of a block are assumed to undergo the same translation, and are assigned the same correspondence vector. The selection of the block size is crucial. Large windows might contain more than one motion and cannot accurately locate motion boundaries, whereas small windows often results in wrong matches within uniform regions in the presence of noise. Because real objects in real scenes do not coincide with the block boundaries, block-matching algorithms suffer from certain drawbacks. This is particularly obvious for the blocks that include multiple moving objects within. A weakness of block-matching algorithms is their inability to cope with rotations and deformations. Block-matching results in block patterns in the motion-vector field. The human visual system is very sensitive to such artifacts, especially abrupt changes which are located in the high frequencies. Nevertheless, their simplicity, suitability for online applications and relative robustness make it very popular technique. To speed block-matching, sub-optimal solutions are also

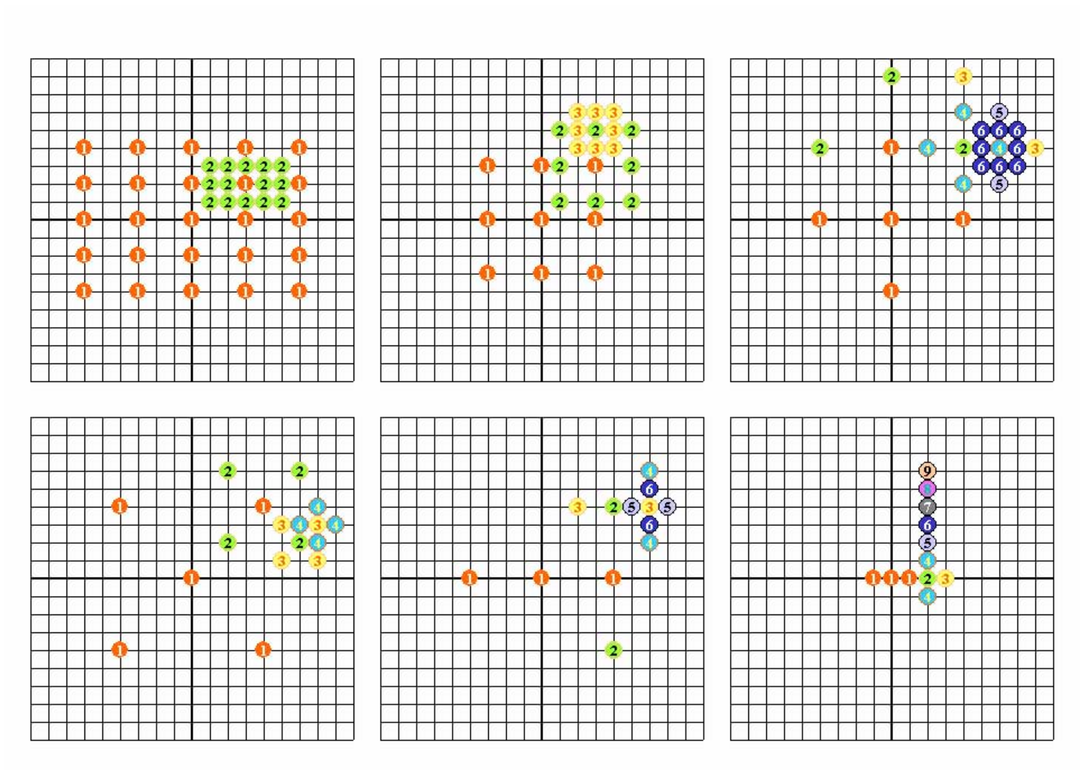


Figure 2.8: Sub-optimal block matching methods. Colored numbers represent consecutive stages of search algorithm.

developed as demonstrated in Fig.2.8.

Block-matching is mainly developed for motion-compensated prediction and block-based video coding. Since hardware implementation of block-matching is quite simple, block-based motion representation and compensation have favorably been adopted in video compression standards such as H.261, MPEG-1, and MPEG-2. However, in order to limit the number of motion vectors that need to be transmitted, a single motion vector is estimated for each 16×16 square block.

2.2.2 Feature Point Matching

Feature-matching algorithms are based on extracting a set of relatively sparse, but highly discriminatory features such as corners, occlusion boundaries of surfaces, and boundaries delimiting changes in surface reflectivity. Such points, lines or curves are extracted from each image. Inter-frame correspondence is then established between these features. Constraints are formulated based on assumptions such as rigid body motion, i.e., the 3D distance between two features on the rigid body remains same after object-camera motion. Such constraints usually results in a system of nonlinear equations. The observed displacements of the 2D image features are used to solve these equations leading ultimately to the computation of motion parameters of objects in the scene.

Feature-based approaches require that correspondence be established between a sparse set of features extracted from one image and those extracted from the next image in the sequence. Although several methods have been discussed for extracting and establishing feature correspondence, the task is difficult and only partial solutions that are suitable for simplistic situations have been developed. In general, the process is complicated by occlusion which may cause features to be hidden, false features to be generated and hidden features to reappear. Much work needs to be done in this area before the advent of one or more general techniques that can be reliably applied to real imagery.

Sensitivity to noise is also a problem with the feature based techniques. The techniques reported in the literature have all been only marginally tolerant to the noise. One method of decreasing the sensitivity to noise has been to use more than the required minimum number of features in an iterative least-squares techniques. Although this usually has a smoothing effect, it can cause additional complications. For example, if all the additional points chosen are coplanar, then all that has been achieved is a significant increase in the computation time and probable instability of the solution. The establishment of correspondence also becomes computationally expensive.

2.2.3 Optical Flow

In comparison, optic flow based approaches do not require any feature correspondence to be established. It also assumes pixels belong to the same objects in successive frames have the same brightness as the conservation of mass in fluid dynamics. Optical flow is a differential method based on the idea that the brightness is continuous for most points in the image, neighboring points have approximately the same brightness. In other words, the world is made up of continuous objects over which brightness varies smoothly. It also assumes pixels belong to the same objects in successive frames have the same brightness as the conservation of mass in fluid dynamics. To understand this property, let's write the continuity equation for the optical term by omitting the second order terms

$$\frac{\partial g}{\partial t} + \mathbf{u}\nabla\mathbf{g} = \mathbf{0} \tag{2.9}$$

where g is the brightness function, and $\mathbf{u}$ is the velocity vector. In one-dimensional case, the above equation takes the simple form

$$\frac{\partial g}{\partial t} + u_x\frac{\partial g}{\partial x} = 0 \tag{2.10}$$

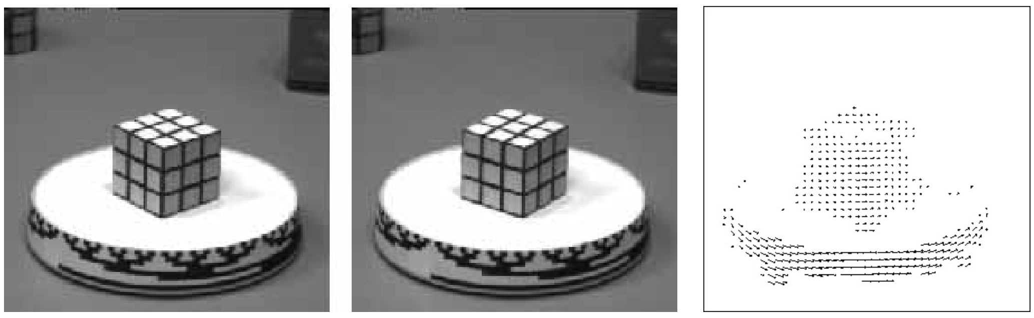


Figure 2.9: Simulation of optical flow estimation for rubics cube sequence.

from which we can directly determine one-dimensional velocity

$$u_x = -\frac{\partial g}{\partial t} / \frac{\partial g}{\partial x} \tag{2.11}$$

provided that the spatial derivative does not vanish, i.e. brightness is continuous. Note that apparent motion and 2D motion are not equivalent. Consider a static scene with varying illumination. The 2D motion is obviously zero because no 3D motion is present; however, the change in illumination induces optical flow, and therefore apparent motion. Further, moving objects or regions must contain sufficient texture to generate optical flow, because the luminance in the interior of moving regions with uniform intensity remains constant. Besides these difficulties, motion estimation algorithms have to solve so-called aperture problem (Fig.2.10). Thus, additional assumptions are necessary to obtain a unique solution. Usually some smoothness constraints on the optical flow field are imposed to achieve continuity. Sample motion vectors obtained by optical flow is presented in Fig.2.9.

The computation of the optical flow as well as the interpretation of the motion and structure from optic flow requires the evaluation of first and second partial derivatives of image brightness values and also of the optic flow. Real images are, in general, noisy. The evaluation of the derivatives is a noise enhancing operation. The higher the order, the more sensitive to noise is the derivative. Hence, even in cases where closed-form solutions for the 3D structure and motion exist, the opti-

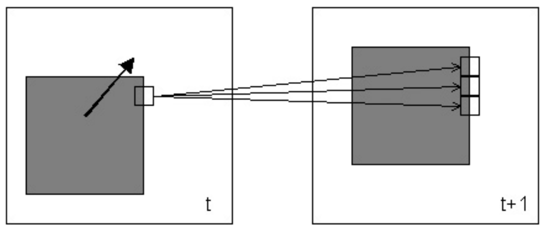


Figure 2.10: Aperture problem: which is the corresponding block?

cal flow techniques do not produce usable results because of the sensitivity to noise. Also, there are discontinuities in the optical flow due to occlusion, and these regions must be detected reliably otherwise violations of the continuity assumptions will have adverse and global effects on the estimate of the optical flow.

In contrast to the method of global minimization, another approach depends upon solving a set of constraints in a small neighborhood. However, the local and global methods rely on similar assumptions of smoothness of optical flow field. The common weakness of both method is the inaccurate estimates at the points where the flow changes sharply or discontinuous. The global method propagates the errors across the entire image, while the neighborhood size limits the propagation in local methods. Kearney *et. al* [62] identify three main sources of error: i) Poor estimation of brightness gradients in highly textured image regions. The problem is especially severe for temporal gradients in moving regions. ii) Variations in optical flow across the image violates assumptions of locally constant flow. Significant error arises at discontinuities in the flow field. iii) Insufficient local variation in the orientation of the brightness gradient which causes error propagation in the ill conditioned system.

2.2.4 Nonparametric and Parametric Motion Models

Two ways of describing motion fields are possible. In the nonparametric representation, a dense field is estimated where each pixel is assigned a correspondence or flow vector. Nonparametric dense field representation is generally not suitable for segmentation because an object moving in the 3D space generates spatially varying 2D motion field even within the same regions, except for the simple case of pure translation. That is the reason why parametric models are commonly used in segmentation algorithms. However, dense field estimation is often the first step in calculating the model parameters. Parametric models require a segmentation of the scene, which is our ultimate goal, and describe the motion of each region by a set of a few parameters. The motion vectors can then be synthesized from these model parameters. A parametric representation is more compact than a dense field description and less sensitive to noise, because many pixels are treated jointly to estimate a few parameters. In order to derive a model or transformation that describes the motion of pixels between successive frames, assumptions on the scene and objects have to be made. Parametric models describe each regions by one set of parameter that is either estimated by fitting a model in the least squares sense to a dense motion field obtained by a nonparametric method or directly from the image. Although parametric representations are less noise sensitive, they still suffer from the intrinsic problem of motion estimation. It should be noticed that one has to be careful when interpreting an estimated flow field. Most likely, it is necessary to include additional information such as color to accurately and reliably detect boundaries of moving objects.

2.3 Spatio-Temporal Segmentation

Using motion attributes of video sequence for segmentation has been a very challenging research problem, and many algorithms are proposed in the literature [42], [43], [67], [129]. A classical approach to motion segmentation is to estimate a dense field followed by a clustering of pixels into regions of coherent motion [2]-[27]. Many approaches use optical flow methods [58] to estimate motion vectors. Using motion information for segmentation is a good idea that exploits the underlying nature of the video data, but there are two major drawbacks to this approach. The optical flow method does not cope well with large motion. Besides, regions of coherent motion may contain multiple objects and need further segmentation for object extraction. To overcome these drawbacks, it is important to incorporate spatial information into motion segmentation. One feasible approach is to spatially segment the first frame to obtain initial segmentation results, and then motion segment subsequent frames using affine region matching. However, the new objects entering the scene and the propagation error due to affine region matching must be handled. Affine region matching employs a coordinate transformation such that it corresponds to the orthographic projection of a 3D rigid motion of a planar surface. By using 6 transformation parameters $a_1, .., a_6$

$$x' = a_1 x + a_2 y + a_3 \tag{2.12}$$

$$y' = a_4 x + a_5 y + a_6 \tag{2.13}$$

$$\tag{2.14}$$

a triangular region is transformed to another triangular region, and a square is mapped to a parallelogram. Similar to the block-matching, the difference score is computed between the original region and its transformed correspondence.

2.3.1 Change Detection Mask

Several methods employ a change detection mask (CDM) instead of a motion field. Although easy to compute, this approach has two drawbacks. First, unless moving objects contain sufficient texture, only occlusion areas will be marked as changed, while the interior of objects will be unchanged. Second, objects or parts of objects that stop moving for a certain period of time will be lost, which is not acceptable in content-based applications. To prevent this, a memory would have to be incorporated. Unfortunately, this would cause background that is becoming uncovered to remain classified as an object for the length of the memory, and the resulting video object planes would be larger than the actual objects, depending on the speed of movement and length of memory.

Automatic segmentation is formulated by Neri *et al.* [93] as the problem of separating moving objects from a static background. In a preliminary stage, potential foreground regions are detected by applying a higher order statistics test to a group of inter-frame differences. The nonzero values in the difference frames are either due to noise or moving objects, with the noise being assumed to be a Gaussian in contrast to the moving objects, which are highly structured. In the case of moving background, the frames must first be aligned by motion compensation. For all difference frames, the zero-lag fourth-order moments are calculated because of their capability to suppress Gaussian noise. These moments are then thresholded, resulting in a preliminary segmentation map containing moving objects and uncovered background. To identify uncovered background, the motion analysis stage calculates the displacement of pixels that are marked as changed. If the displacement of a pixel is zero for all lags, it is classified as background and as foreground otherwise. Finally, morphological opening and closing operators are applied to achieve spatial continuity and to remove small holes inside moving objects of the segmentation map. The resulting segmented foreground objects are slightly too large, because the boundary location is not directly determined from the gray level or edge image.

Mech and Wollborn [80] generate a video object plane from an estimated change detection mask. Initially, a change detection mask is generated by taking the difference between two successive frames using a global threshold. It is then refined in an iterative relaxation that uses locally adaptive threshold to enforce spatial continuity. Temporal stability is increased by incorporating a memory such that each pixel is labeled as changed if it belonged to an object at lest once in the last change detection masks. The simplification step includes a morphological close and removes small regions to obtain the final CDM. The object mask is calculated from the CDM by eliminating uncovered background and adapting to gray-level edges to improve the location of boundaries.

Nevertheless, change detection masks are sometimes more useful than motion fields. For example, in head-and-shoulder sequences, there is only little movement of the person, and the occlusion regions are very small. Thus, a relatively long memory can be attached without getting video object planes that are significantly larger than the object. Estimating a motion field would be more difficult because the motion is simply too small.

2.3.2 Stochastic Approaches

The Bayesian framework provides an elegant formalism and is among the popular approaches to motion segmentation [90, 113]. The key idea is to find the maximum *a posteriori* estimate of the segmentation for some given observation which is the video sequence.

Murray and Buxton [90] used an estimated flow field as observation. As it is common, the segmentation or priori model is assumed to be a sample of Markov random field (MRF) to enforce continuity of the segmentation labels, and thus, probability density function of segmentation is a Gibbs distribution [13]. The energy function of the MRF consists of a spatial smoothness term, a temporal continuity term, and a line field as in [51] to allow for motion discontinuities. To define the observation model, the parameters of a quadratic flow model [2] are calculated for each region by linear regression. The mismatch between this synthesized flow and the flow field given as observation is assumed to be zero-mean white Gaussian noise. The resulting probability function is maximized by simulated annealing [51]. Major drawbacks of this proposal are the computational complexity and the number of objects likely to be found has to be specified. A similar approach was taken by Bouthemy and

Francois [18]. Their observation function contains the temporal and spatial gradients of the intensity function, which is essentially the same information as the optical flow. For each region, the affine motion parameters are computed in the least-squares sense. The optimization is performed by iterated conditional modes [13], which is faster than simulated annealing, but likely to get trapped into a local minimum. The techniques [2]-[18] include only optical flow data into the segmentation decision, and hence, their performance is limited by the accuracy of the estimated flow field. This means that they inevitably suffer from the problems such as noise sensitivity and inaccuracy at motion and therefore object boundaries.

It is possible to treat motion estimation and segmentation jointly in the Bayesian framework [26, 113]. In this case, the observation function consists only of the gray-level intensity, and both the segmentation and the motion field have to be estimated. Chang *et al.* [26] used both a parametric and a dense correspondence field representation of the motion, with the parameters of the eight-parameter model being obtained in the least squares sense from the dense field. The objective function resulting from the MAP criterion consists of three terms, each derived from an MRF. Since the number of unknowns is three times higher when the motion field has to be estimated as well, the computational complexity is significantly larger. The technique proposed by Stiller in [112] and extended in [113] is similar, but no parametric motion field representation is necessary. In [112], the objective function consists of two terms. The displaced frame difference generated by the dense motion field is modeled by a zero-mean generalized Gaussian distribution, and an MRF ensures segment-wise smoothness of the motion field and spatial continuity of the segmentation. Although ICM is used to obtain the MAP estimate, the computational burden of this algorithm is enormous. A classification based approach was proposed for jointly segmenting moving objects and their corresponding optical flow by Bors [17]. The classification of regions is done accordingly to the Bayesian theory, which is used to express the probabilities from one frame with respect to the previous frames such that the conditional probability of the estimated vectors in all of the frames of the sequence is blended into three conditional probability term. The first probability is associated with the reconstruction of a frame based on the previous frames and their correspondent feature vectors. The components of the second term represents the dependency of a feature vector on the values of the same future vector in the previous frames. The third probability models the moving object characteristics evaluated in the first frames. The median radial basis function network is then employed for modeling the moving object characteristics. The initial segmentation and assignment of regions into objects are main drawbacks of this approach.

Techniques that make use of Bayesian inference and model images by Markov random fields can easily incorporate mechanisms to achieve spatial and temporal continuity. On the other hand, these approaches suffer from high computational complexity, and many algorithms need the number of objects or regions in the scene as an input parameter.

2.3.3 Morphological Approaches

Morphological tools such as watershed algorithm and simplification filters are becoming increasingly popular for segmentation [86, 78]. An introduction, discussion of potential problems, and several applications to segmentation are presented by Meyer and Beucher [86]. Salembier and Pardas [105] described a segmentation algorithm that has a typical structure for morphological approaches. In a first step, the image is simplified by the morphological filter "open-close by reconstruction" to remove small dark and bright patches. An attractive property of these filters is that they do not blur or change contours like low-pass or median filters. The following marker extraction step detects the presence of homogeneous areas, for example by identifying large regions of constant color or luminance. This step often contains most of the knowhow of the algorithm. Each extracted marker is then the seed for a region in the decision step, the so-called watershed algorithm, which is a technique similar to region growing. A quality estimation is performed in [105] as a last step to determine which regions require resegmentation. The proposed segmentation by Salembier *et al.* in [104] is very similar, but an additional projection step is incorporated that warps the previous partition onto the current frame. This projection, which is also computed by the watershed algorithm, ensures temporal continuity and linking of the segmentation.

Another morphological video segmentation algorithm was proposed by Choi *et al.* [30]. Their marker extraction step detects areas that are not only homogeneous in luminance, but also in motion, so-called joint markers. For that, intensity markers are extracted as in [105], and affine motion parameters are calculated for each marker by linear regression from a dense flow field. It starts with a global motion estimation and compensation step. The thresholded morphological gradient image, which is also called as morphological edge, serves as input for the watershed algorithm that detects the location of the object boundaries. Every region for which more than half of its pixels are marked as changed in a change detection mask is assigned to the foreground. After that, the segmentation is simplified by merging regions with similar affine motions. In a last stage, the frame is simplified with a morphological open-close by reconstruction filter. A drawback of this technique is the lack of temporal correspondence to enforce continuity in time.

A double-partition approach based on morphology was suggested by Marques and Molina [78]. Initially, objects of interest have to be selected interactively, leading to partition at object level that corresponds to a decomposition into video object planes. These objects are normally not homogeneous in color or motion and are resegmented to obtain a fine partition that is spatially homogeneous. After estimating a dense motion field by block matching, the fine partition is projected onto the next frame using motion compensation. These projected regions are used to extract the markers for the next frame, which is then segmented by the watershed algorithm based on luminance. To improve the temporal stability, the segmentation process is guided by a change detection mask that prevents markers of static areas to overgrow moving areas and vice versa. Finally, the new object level partition is computed from the projected and segmented fine partition.

Morphological segmentation techniques are computationally efficient, and there is no need to specify the number of objects as with some Bayesian approaches, because this is determined automatically by the marker of feature extraction step. However, due to its nature, the watershed algorithm suffers from the problems associated with region-growing techniques.

2.3.4 Hybrid Approaches

In his early work, Adiv [2] proposed a hierarchically structured two-stage algorithm. The flow field is first segmented into connected components using the Hough transform such that the motion of each component can be modeled by an affine transformation. Adjacent components are then merged into segments if they obey the same eight-parameter quadratic motion model. In the second stage, neighboring segments are consistent with the same 3D motion are combined, resulting in the final segmentation.

Hierarchically structured segmentation algorithms were proposed by Hotter and Thoma [59], Musmann *at al.* [91], and Diehl [41]. A change detector divides the current frame into changed and unchanged regions, and each connected changed region is assumed to correspond one object. Starting from the largest changed region the motion parameters for this object are estimated directly from the image intensity and gradient. If the prediction error after motion compensation is too large, this object is further subdivided and analyzed in subsequent levels of hierarchy. The algorithm sequentially refines the segmentation and motion estimation until all changed regions are accurately compensated. Because these techniques alternate between analyzing the image and synthesizing, they have been described as object-oriented analysis-synthesis algorithms. In [59] and [91], the eight parameter motion model is used, and the parameters are obtained by a direct method. A 12-parameter quadratic motion model that describes a parabolic surface undergoing the 3D motion under parallel projection is proposed in [41]. An iterative technique that is similar to the Newton-Raphson algorithm estimates the parameters by minimizing the MSE between the motion-compensated and the current frame. Edge information is incorporated into the segmentation algorithm to improve the accuracy of boundaries.

Wang and Adelson [128] proposed a layered representation of image sequences. The current frame is segmented based on motion with each layer being modeled by an affine transformation. The algorithm starts by estimating the optical flow field, and then subdivides the frame into square blocks. The affine motion parameters are computed for each block. The pixels are then regrouped by adaptive k-means clustering algorithm. A pixel is assigned to a layer if the difference between the optical flow at that pixel and the flow vector synthesized from the affine parameters of that layer is smaller than for any other hypothesis. The frames are warped according to the affine motion of the layers such that coherently moving objects are aligned. A temporal median filter is then applied to obtain a single representative image for each object. This proposal has several disadvantages. To construct the layers, the information of a longer sequence is necessary. If in a sequence different views of the same object are shown, it is not possible to represent that object by a single image that is warped from frame to frame. Further, the affine transformation might not be able to describe the motion of a complete layer in the presence of strongly nonrigid motion. The algorithm also depends completely on the accuracy of the optical flow estimates since no color or intensity information is used.

Meier and Ngan [83] extend the techniques proposed in [82] to scenes with a moving camera or background. After a binary model for the object of interest has been derived from the edge image, an object tracker matches the model against subsequent frames in the sequence using the Hausdorff distance and updates the model every frame to accommodate for rotation and changes in shape of the object. The proposed algorithm is improved by a filtering technique that removes the stationary background. The output of the tracker is a sequence of binary models that will guide the extraction of VOP's. The location of the object boundaries is determined based on the binary model, which in turn is derived from the edge image. Although matching the binary model using Hausdorff distance is remarkably robust, the most difficult task is to distinguish between background and objects in the initialization and update stage.

Qui [100] fuses information from color image segmentation, motion segmentation and active contour to achieve accurate extraction of moving objects. They employ the sequential labeling algorithm based on using reflectance ratio as a measure of similarity of two neighboring pixels. A dense motion field by using non-parametric approach that uses local ordering of intensities which is called as census transform. Its main idea is to first apply an ordering-based local transform to the image, and then to use correlation. The disparity values are assigned as the most popular disparity of boundary pixels of the portions found by color segmentation. Finally a mask is derived for each object.

Layered representation of regions to construct motion consistent objects is addressed by Siggelkow *et al.* [108]. In order to achieve a segmentation conformable to subjective image partition, a hierarchical merging of regions is done. The first layer corresponds the regions detected by inter-frame segmentation based color clustering and translational region matching. In the second layer, regions of the first layer are merged with respect to similar chrominance and motion. The highest layer represents a semantic segmentation, where regions of the second layer are merged in case of similar motion. However, using only translational motion model degrades the performance of tracking and significantly lowers the success of correct merges in the layered representation stage.

Torres et al [119] used a motion segmentation algorithm. They partitioned the image into rectangular regions and computed affine motion parameters for each region. These motion parameters are clustered using k-means algorithm to form homogeneous regions with similar motion parameters. Dufaux et al [43] used spatiotemporal segmentation algorithm based on luminance information and motion parameters. The luminance is filtered by morphological operator, and then clustered using k-means algorithm. At the end, regions with similar motions are merged using k-means clustering. Moscheni et al [89] used the spatiotemporal similarity as their merging criterion. Their spatial similarity is obtained from the test statistic of the gradient value along the boundary of the regions. Their temporal similarity is derived from test statistic of the residual distribution and motion parameters.

The algorithms described so far mainly focused on segmenting video sequences into regions that are homogeneous with respect to motion and possibly color or luminance. For content-based functionalities, it is desirable to partition the frames into objects that are semantically meaningful to the human observer. Thus, the above techniques will fail in many practical situations where objects do not correspond to partitions based on features like motion or color.

2.4 Object Tracking

A tracking process can be interpreted as the process of search for a target within the reduced scope. This process is usually embodied through model matching. Three main approaches have been developed to track objects depending on their type; whether they are rigid, nonrigid, or have no regular shape. For the two first approaches the goal of the tracking process is to compute carefully the correspondences between objects already tracked and the newly detected moving regions, whereas the goal of the last approach is handling the situations where correspondences are ambiguous.

The major difficulty in a tracking problem is to deal with the inter-frame changes of moving objects. It is clear that the image shape of a moving object may

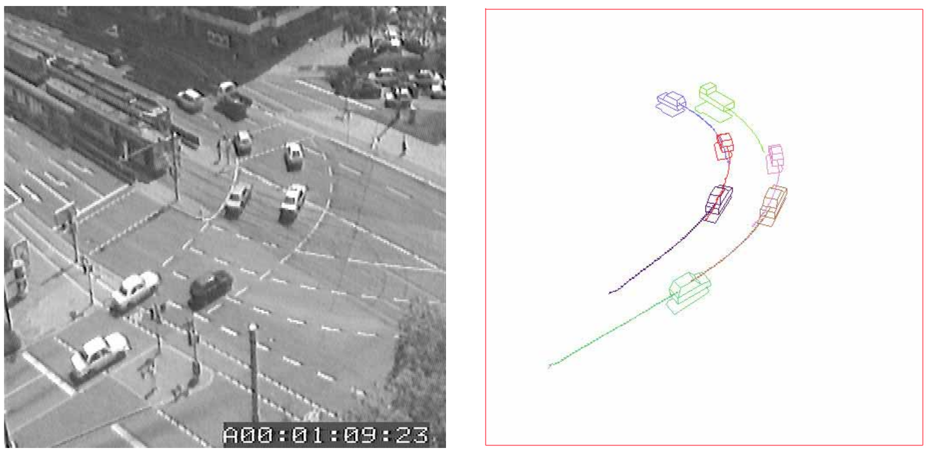


Figure 2.11: Cars are found, modeled and tracked using models .

undergo deformation, since a new aspect of the object may become visible or an actual shape of an object may change. Thus a model needs to evolve from one time frame to the next, capturing the changes in the image shape of an object as it moves.

2.4.1 Object Models for Tracking

When objects are rigid, like manufactured objects, the tracking process can take advantage of accurate knowledge on their shape as illustrated in Fig.2.11. A first method consists of detecting particular primitives, such as corners and edges, and in tracking these primitives from one image to another [126]. This method can be used only with objects owning numerous primitives easy to detect, like vehicles. Another method computes the correspondences between a 3D model of mobile objects and the 2D moving regions corresponding to their perception [69]. For example, the center of the moving region and the direction of its motion are computed, then the correspondences between the line segments of the 3D model and the edges detected inside the region are established. This method is more reliable when the 3D model is accurate.

When objects are nonrigid, like humans, no accurate model of their shape is available. Instead, dynamic templates of the perception of object motion are used. These templates that we call models are regularly updated during the tracking process to compensate the evolution of the object perception. Three types of dynamic model can be used.

The first type of model corresponds to the parameterized shape of the mobile objects. These models can be applied to rigid or nonrigid objects. For example in [87], polygons are used to represent the outline of vehicles. For similar applications in [10] the authors use cubic B-splines instead of polygons. In both cases these models have been successfully applied to track rigid objects. In [11], the authors extend the method to nonrigid objects. Their model is made of cubic B-splines but it also contains the authorized deformations that correspond to the outline of a walking pedestrian. By this way the outline of tracked objects can be distorted only in certain directions making the tracking process more reliable. The second type of model contains a template of the moving region corresponding to the detection of the object motion. This template is defined by the color distribution of the pixels belonging to the moving region. For example in [31], the authors use a color histogram and thanks to it they are able to track in the same time several football players and to cope with dynamic occlusions in certain situations. In [9], each pixel of the template is associated to the temporal color distribution of the intensity function resulting in a more robust tracking. The third type of model is also made of a template of the moving region. However this template is defined by the set of edges detected in the moving region. In [60], the authors use this template and define a distance between two sets of edges to allow them to compare parts of templates.

When no a priori model of objects is available, the tracking process can only use the coordinates of object locations to compute the correspondences between already tracked objects and newly detected ones. As ambiguous correspondences may arise, several methods have been proposed to solve these ambiguities. For example the method of multiple hypothesis tracking (MHT) generates hypotheses to perform all possible combinations of correspondences. These hypotheses define different worlds where the correspondences are coherent. When a hypothesis becomes incoherent, the associate correspondences are discarded. To avoid a combinatorial explosion, only a few levels of hypotheses are computed in real-world applications. In [36], the authors propose an efficient implementation of MHT. The beam search method also proposes a mechanism to handle ambiguities [134]. It duplicates all the ambiguous tracked objects and makes the correspondences with the newly detected objects. Then this method consists of tracking all these objects and verifies whether or not they are coherent at every new frame arrival. If their tracking is coherent, the correspondences relative to these objects are considered as the true ones. These methods are two examples of tracking methods that can be applied to any kind of objects. Since they do not use a priori knowledge on objects, they cannot compute accurate correspondences and may lead to tracking errors.

2.4.2 Tracking Techniques

In [55], Heisele proposes a cluster based tracking method that for each image clusters of the previous image are adapted iteratively by a parallel k-means clustering algorithm. Instead of tracking single points, edges, or areas over a sequence of images, only the centroids of the clusters are tracked. For the very first image a set of prototypes is determined by the divisive clustering. For each following image the prototypes of the preceding image serve as seeds for the parallel k-means clustering. This clustering provides a new set of prototypes for the next image.

A model based tracking system is developed by Jang *et al.* [65]. After the initialization is done manually, the least enclosing rectangle of the target is found, and partitioned into cells that are classified later as boundary and internal cells. The selected cells become nodes of a model graph, and they are linked to the neighboring nodes. A feature vector composed of color, Gabor based wavelet coefficients, geometric disposition of the node, and edges is assigned to each node. An energy function similar to Snake model interprets matching process as an energy minimization process. Tracking is decomposed in two steps: 1) estimating a possible area of a target at the current time frame by a Kalman filter that predicts motion parameters, and 2) updating a model of target through energy minimization. The new model is constructed by associating feature values of the found cells with nodes of a graph. A drawback of this approach is that new model is isomorphic to old one in terms of the number of nodes and the linkage among nodes.

Another region tracking algorithm heavily depends on color correspondence was proposed by Deng [39]. In his approach, color is assumed to be the most dominant visual feature, and it is used to rank a distance measure. Other features are only used as constraints to eliminate false matches, and this is achieved by the use of thresholds. In other words, the goal is to find the most similar region in color that is also close in texture, size, and location.

For every region in the next frame, the size and texture differences of current region are compared with the preset thresholds for differences in size and texture. If both values are less than the thresholds, the region is considered as a candidate for color rank ordering. Affine motion compensation is estimated to predict the approximate location of the current region in the next frame.

The distance between the location of a candidate region and the predicted location of the current region is calculated. The distance has to be less than an image size-dependent threshold, for the region to remain in candidacy. The color feature distance between a candidate region and the current region is weighted such that, the further away a candidate region, the less likely that it will be the true match. The particular weighting by location difference is needed because there might be some objects nearby with similar color, texture, and size. The weighted color distance is checked against a threshold to validate candidacy. The candidate regions are ranked in terms of their color distances. The region having the minimum value is determined to be the match for the current region. If a match satisfying all of the above conditions cannot be found, the current region is determined to have no match in the next frame. Some obvious drawbacks of this method are its dependency to initial segmentation as well as intra-frame spatial segmentation, and determination of optimum threshold values.

A semiautomatic object tracking algorithm that depends user assistance to obtain the initial object-segmented frame of the sequence was presented by Gu [52]. His method starts by asking user to mark up the boundary of the object to be tracked. Then the pixels along a certain neighborhood of this initial boundary are classified as object or background with respect to their distance to cluster centers by using either distance or morphological watershed algorithm. Unfortunately, it is not explained how to derive these cluster centers. After the initial user depended segmentation, the algorithm calculates global perspective motion parameters between the current and next frame, updates the boundary of the detected object by the same operators used for the initial boundary adjustment. The need for assistance, the inability to handle multiple objects, and the motion dependent nature of region estimation for the consecutive frames make this scheme unsuitable for ordinary video sequences.

Bremond [20] presents a method to track multiple non-rigid objects in video sequences. To handle the particularities of nonrigid objects, he proposes an appearance-based approach for the tracking method. He defines the target model thanks to the height and width of the bounding box surrounding the moving regions associated to the target. The height and width are average values regularly updated during the process. Five generic points are defined as the middle of the sides of the bounding box and as its center. They are tracked separately and the point that best matches the newly detected moving region defines the track of the global object. The tracking of moving regions is performed in a prediction-matching-update loop. To allow changes he supposes that the motion of a target is piece-wise linear and he represents its trajectory by a polygonal approximation. Short segments of line for the trajectory are computed by considering that the speed of the target is constant on segments. So the trajectory gathers an approximation of all past locations of the target. The new location of a target is computed during the matching process between the target and a moving region detected in the current frame. Then, every estimated location is matched with the location of the corresponding generic point of the moving region. The target generic point with the best match is chosen to compute the location of the target center. The matching process compares the predicted location of targets with the location of newly detected moving regions through the use of an ambiguity matrix. The ambiguity matrix gives the number of moving regions that are close to the predicted location of each target. The matrix elements measure the distance between targets and moving regions. The task of the matching process is to solve the correspondences of the ambiguous targets. This paper presents a tracking method that gathers several characteristics. The method is based on the tracking of the appearance of scene objects instead of their real structure. This approach allows us to handle several cases of partial static occlusion. In particular, it helps in tracking scene objects even if they are partially detected. It uses elementary dynamic models of targets that need no a priori knowledge on scene objects, and allows user to solve several cases of ambiguous correspondences. The method systematically uses two other types of information: contextual information and information computed by the scenario recognition module.

2.5 Segmentation in Compressed Domain

Performing analysis in the compressed domain reduces the amount of effort involved in decompression. MPEG coding standards convert the bitstream in terms of I, B, and P-frame. The B and P frames store the motion information and residues after motion compensation. The I-frame stores DCT information of the original frame. Though I-frame provides no motion information, still color and texture information can be grasped and propagated to the B, P frames by inverse motion compensation. Compressed-domain video possesses several important characteristics attractive for object analysis. First, motion information is readily available without incurring cost of estimation of motion field. Second, DCT forms relay information on image characteristics. On the other hand, the motion vectors are often contaminated with mismatching and quantization errors. Comparably, motion processing in uncompressed image-sequence domain is better suited for accuracy and precision. On top of that, the motion fields in MPEG streams are quite prone to quantization errors.

Only few researchers have proposed the segmentation algorithms in compressed domain. De Queiroz [38] segmented JPEG documents into specific regions such as those containing halftones, text, and continuous-tone pictures using the encoding cost map based segmentation. Wang [127] proposed a fast algorithm to automatically detect faces in MPEG compressed video. He used skin-tone statistics, shape constraints, and energy distribution of the luminance DCT coefficients to detect and locate the face position.

In a related work in compressed domain, Meng and Chang [84] employ a block count method to estimate parameters in a three-parameter affine global motion model. Then they perform global motion compensation to get object mask and perform histogram clustering to deal with multiple objects.

Sukmarg [114] propose a fast algorithm to detect and segment objects in MPEG compressed video. His segmentation algorithm consists of four main stages, initial segmentation using sequential leader and adaptive k-means clustering, region merging based on spatiotemporal similarities, foreground-background classification, and object detail extraction. The initial segmented regions are generated from 3D spatial information based on DC image and AC energy information that is used to cluster the image using sequential leader clustering. After clusters are obtained, adaptive k-means is applied until no more changes occur in each cluster. The resulting sequences contain gradient estimates along their associated dimensions. The temporal similarity is derived based on the hypothesis test of the distribution of the temporal gradient. This hypothesis test is the Kolmogorov-Smirnov test, which measures the overall difference between two cumulative distribution functions [6]. The spatiotemporal similarities are calculated and used to create a similarity graph between regions. This graph is thresholded and clustered. First clustering stage is used to merge regions, which form cycles in graph. The second clustering stage is used to merge regions based on the number of graph edges connecting between an interested cluster and its neighbor cluster, and those connecting within the interested cluster itself.

2.6 Scene-Cut Detection

An essential step involved in video segmentation is partitioning videos into short sequences called shots. A shot is a sequence of images that is consistent in terms of object content. Shots have been identified as the fundamental unit of video and their detection is the foremost task of scene segmentation. Once shots are extracted it is possible to analyze their content based on motion, color, texture and others features.

The shot boundary detection techniques may be data driven or model driven. The model driven approach is essentially based on mathematical models. The data driven methods for detecting shot boundary essentially fall into two classes: those based on global features without any spatial information, i.e. color histograms, and those based on spatially registered features of the images. The former are insensitive to motion but they can fail to detect scene cuts when the images before and after the scene cut have similar global features. Spatially registered feature based approaches are often too sensitive to moving objects and some false cut may be detected when the image motion is very fast.

In [47], scene-cut detection is addressed using global representation like color histogram and spatially related features. Corridoni and Del Bimbo [34] proposed a technique based on a relative difference between frames. They expect a scene cut when the difference between two frames is much larger than the standard difference between frames belonging to the same shot. The threshold value has to be set experimentally. Nagasaka and Tanaka [92] applied the template matching technique and the X^2 test to the color histograms of two subsequent frames. Arman et al. [8] proposed techniques which operate directly on compressed video to detect scene cuts by using known properties of the coefficients of the DCT. The previous mentioned approaches are essentially data driven. Recent schemes based on video content encoded in DCT coefficients and motion vector information [133, 61], neural architecture [7], and reduced image sequences [132] have been reported in the literature. These schemes are sufficiently accurate in segmenting the video into shots.

2.7 Data Clustering

Clustering is unsupervised classification of patterns that are referenced as observations, data items, or feature vectors into groups called as clusters. Typical pattern clustering activity involves the following steps [64]: i) pattern representation optionally including feature extraction and selection, ii) definition of a pattern proximity measure appropriate to the data do-main, iii) clustering or grouping, iv) data abstraction if needed, and v) assessment of output if needed.

The most challenging step in clustering is feature extraction or pattern representation. Pattern representation refers to the number of classes, the number of available patterns, and the number, type, and scale of the features available to the clustering algorithm. Some of this information may not be controllable by the practitioner. Feature selection is the process of identifying the most effective subset of the original features to use in clustering. Feature extraction is the use of one or more transformations of the input features to produce new salient features. Either or both of these techniques can be used to obtain an appropriate set of features to use in clustering. In small size data sets, pattern representations can be obtained based on previous experience of the user with the problem. However, in the case of large data sets, it is difficult for the user to keep track of the importance of each feature in clustering. A solution is to make as many measurements on the patterns as possible and use them in pattern representation. But it is not possible to use a large collection of measurements directly in clustering because of computational costs. So several feature extraction and selection approaches have been designed to obtain linear or nonlinear combinations of these measurements which can be used to represent patterns. Most of these approaches are typically iterative and cannot be used on large data sets due to prohibitive computational costs.

The second step in clustering is similarity computation. Pattern proximity is usually measured by a distance function defined on pairs of patterns. A variety of distance measures are in use in the various communities. A simple distance measure like Euclidean distance can often be used to reflect dissimilarity between two patterns, whereas other similarity measures can be used to characterize the conceptual similarity between patterns. A variety of schemes have been used to compute similarity between two patterns. They use knowledge either implicitly or explicitly. Most of the knowledge-based clustering algorithms use explicit knowledge in similarity computation. However, if patterns are not represented using proper features, then it is not possible to get a meaningful partition irrespective of the quality and quantity of knowledge used in similarity computation. There is no universally acceptable scheme for computing similarity between patterns represented using a mixture of both qualitative and quantitative features. Dissimilarity between a pair of patterns is represented using a distance measure that may or may not be a metric.

The next step in clustering is the grouping step. There are broadly two grouping schemes: hierarchical and partitional schemes. The hierarchical schemes are more versatile, and the partitional schemes are less expensive. The partitional algorithms aim at maximizing the squared error criterion function. Motivated by the failure of the squared error partitional clustering algorithms in finding the optimal solution to this problem, a large collection of approaches have been proposed and used to obtain the global optimal solution to this problem. However, these schemes are computationally prohibitive on large data sets. The grouping step can be performed in a number of ways. The output clustering can be hard as a partition of the data into groups, or fuzzy where each pattern has a variable degree of membership in each of the output clusters. Hierarchical clustering algorithms produce a nested series of partitions based on a criterion for merging or splitting clusters based on similarity. Partitional clustering algorithms identify the partition that optimizes a clustering criterion. Additional techniques for the grouping operation include probabilistic and graph-theoretic clustering methods.

In some applications, it may be useful to have a clustering that is not a partition. This means clusters are overlapping. Fuzzy clustering and functional clustering are ideally suited for this purpose. Also, fuzzy clustering algorithms can handle mixed data types. However, a major problem with fuzzy clustering is that it is difficult to obtain the membership values. A general approach may not work because of the subjective nature of clustering. It is required to represent clusters obtained in a suitable form to help the decision maker.

Knowledge-based clustering schemes generate intuitively appealing descriptions of clusters. They can be used even when the patterns are represented using a combination of qualitative and quantitative features, provided that knowledge linking a concept and the mixed features are available. However, implementations of the conceptual clustering schemes are computationally expensive and are not suitable for grouping large data sets. The k-means algorithm and its neural implementation, the Kohonen net, are most successfully used on large data sets. This is because k-means algorithm is simple to implement and computationally attractive because of its linear time complexity. However, it is not feasible to use even this linear time algorithm on large data sets. Incremental algorithms like leader and its neural implementation can be used to cluster large data sets. But they tend to be order-dependent. Divide and conquer is a heuristic that has been rightly exploited by computer algorithm designers to reduce computational costs. However, it should be judiciously used in clustering to achieve meaningful results.

2.8 Summary

This chapter discussed several aspects of segmentation from image segmentation to data clustering. Specifically, we focused on region-based and motion estimation methods, and also explored the problem of object tracking.

With regard to region segmentation, we presented common histogram thresholding and clustering approaches. Although histogram based segmentation is invariant to additive intensity variation, it has no explicit notion of connectivity. It depends an implicit assumption that pixels with similar intensities belong to the same regions, while this may not true in general. Besides, the result of clustering is seriously affected by the initial values at the beginning of clustering.

Edge-based approaches utilize the nearest neighbor algorithm or treat segmentation as an estimation problem. Although these techniques work well in some situations where the input data set is relatively simple, clean, and fits the model well, they lack generality and robustness. Color-based split-and-merge, pyramid linking, and morphological methods provide better performance than edge methods, however the main problem arises from the fact that a video object can contain totally different colors.

On the other hand, works in the motion oriented segmentation domain start with an assumption that a semantic video object has homogeneous motion. Most motion based methods are based on optical flow estimation or unreliable initial segmentation. As a result, they may suffer from the inaccuracy of motion boundaries.

This class of methods will also fail when a semantic video object have different motions in different parts of the object. Computational complexity is another main concern.

The major difficulty in object tracking is to deal with the inter-frame changes of moving objects. It is clear that the image shape of a moving object may undergo deformation, since a new aspect of the object may become visible or an actual shape of an object may change. Thus a model needs to evolve from one time frame to the next, capturing the changes in the image shape of an object as it moves.

In summary, a single homogeneous color or motion criterion does not lead to satisfactory extraction because each homogeneous criterion can only deal with a limited set of scenarios. A semantic video object may contain multiple colors and multiple motions. Therefore, any single criterion could only lead to a partial solution for semantic visual information extraction.

Chapter 3

Preprocessing of Color Digital Video for Segmentation

Knowledge should mean a full grasp of knowledge
Knowledge means to know your heart and soul
If you have failed to understand yourself
Then all of your reading has missed its call.

What is the purpose of reading those books?
So that man can know the All-Powerful
If you have read, but failed to understand
Then your efforts are just a barren toil..

Yunus Emre

Preprocessing prepares an input video sequence for the following object segmentation stage by filtering the video frames. In this chapter, statistical and structural attributes of a video sequence are evaluated, and several color spaces, noise removal filters, simplification filters, change detection masks, etc., are discussed.

The most common video attributes can be counted as color values, texture scores, edge properties, and frame differences. A 3D spatiotemporal data structure that will serve as a basis in the segmentation framework is constructed from the video sequence using its attributes. To determine a suitable color space, the effects of several color spaces on the performance of region growing based segmentation are evaluated. Out of several differences, sensitivity to illumination changes and having computationally simple distance metric are two main considerations.

For an automatic segmentation framework, filtering has great importance on the quality of the final segmentation results. Video sequences acquired through sensors may be contaminated by a variety of noise sources associated with electro-optical devices and detection processes. By noise it is often referred to stochastic variations as opposed to deterministic distortions such as shading or lack of focus. Some distortions are inherited to camera lens system as in the wide-angle field lenses, others are related to shooting conditions, i.e. camera shake, jitter, flickers, sparks. In that sense, noise removal is a prerequisite for most segmentation algorithms.

Similar to the noise, certain spatial texture is undesirable in segmentation since it induces over-partitioning and increases computational load. For instance, an object having chess-like pattern may be segmented into multiple smaller partitions if a color homogeneity criterion is applied. An image simplification or reconstruction method that preserves the object boundaries would solve excessive segmentation problem. In this chapter, various noise removal filters are presented and compared. Two novel methods that employ robust estimators and regressive band-suppression filters are introduced. The filtering performances of these methods are analyzed.

Frame difference is implemented as a change detection mask that detects moving pixels between two consecutive frames. A dual window based change detection mask extraction is explained.

3.1 Analysis of Suitable Attributes

Color is important to living creatures, not only because it beautifies our world but also gives more information about our surroundings as well. To each distinct color, there is a corresponding wavelength in the electro-magnetic spectrum of visible light. A range of colors, which is called as color gamut, is obtained by taking three

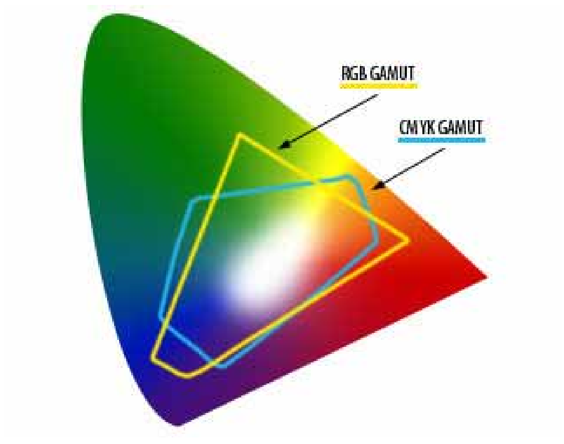
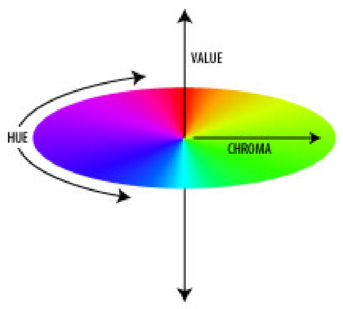


Figure 3.1: Gamuts of the RGB and CMY color spaces within the visible gamut.

arbitrary colors from the spectrum as illustrated in Fig.3.1. The colors themselves are called as primary colors.

One of the most influential color modeling systems was devised by A. Munsell. He identified colors in terms of three attributes: hue, value and chrominance as shown in Fig.3.2. Hue is defined as the dominant wavelength of the color. Hue is the attribute of a color by which we distinguish red from green, blue from yellow, etc. There is a natural order of hue: red, yellow, green, blue, purple. One can mix paints of adjacent colors in this series and obtain a continuous variation from one color to the other. For example, red and yellow may be mixed in any proportion to obtain all the hue range from red through orange to yellow. Chrominance is the degree of departure of a color from the neutral color of the same value. Colors of low chrominance are sometimes called weak, while those of high chrominance are said to be highly saturated, strong or vivid. Chrominance can be viewed as the distance from the white; it is a measure of how pure the color is. In other words, the amount of white a hue is mixed with that represents the vividness of the color. Value corresponds to the shading of the color. In other words, it is related to the emitted energy of a color. The higher the emitted energy, the brighter the color appears. Value was defined by Munsell as "the



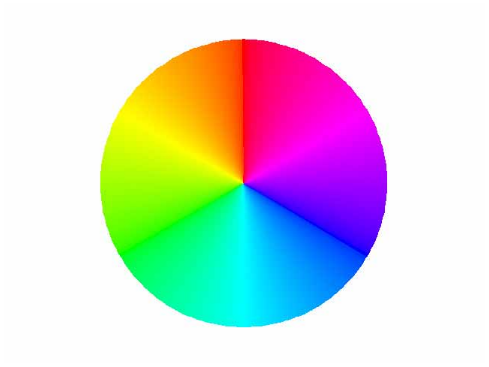


Figure 3.2: Munsell color space and its hue diagram.

quality by which we distinguish a light color from a dark one." Value is a neutral axis that refers to the grey level of the color.

3.1.1 Color Spaces

Color space is a model for representing color attributes of image pixels numerically in terms of primary colors. No color system can be considered as universal, because perception can be interpreted and modeled in different ways. Each color system has its own color features, which are the parameters of a color system. In any problem involving color quantification, the first step toward the solution is to define the color space. There exist several color spaces used for the description of color attributes. An international standard of primary colors was established in 1931 by the Commision Internationale de l'Eclairage (CIE). The chosen standard primaries colors are not real but imaginary colors, because they are too saturated to be seen by a human eye or produced. The advantage, however, is that those primary colors can describe each perceived color mathematically.

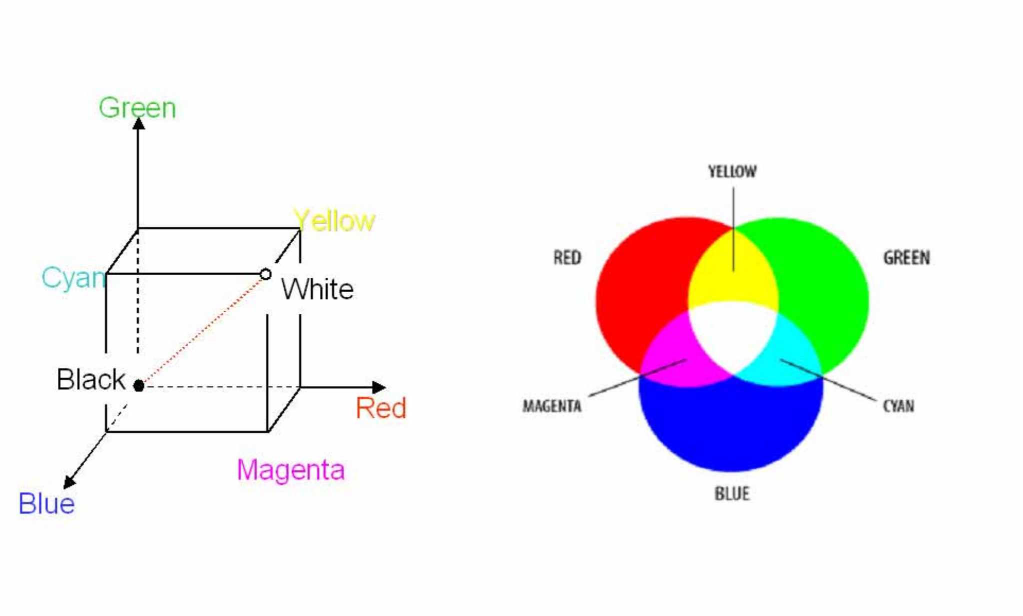


Figure 3.3: The RGB color space and additive property.

RGB Space

By defining three primary colors as 700nm for red, 546.1nm for blue, and 435.8nm for green, RGB color space is established. The RGB is an additive color model that is used for video screens, it has machine oriented chromatics rather than human oriented chromatics. One other major disadvantage of the RGB space is the dependency of all three parameters from the light intensity. Thus, it is sensitive to intensity changes, shadows and irregular illumination. The RGB gamut is smaller, hence certain visible colors (e.g. pure yellow, pure cyan) cannot be seen on monitors.

CMY Space

CMY (cyan, magenta, yellow) is a subtractive model that is used often for printers. Just as the primary colors of CMY are the secondary colors of RGB, the primary colors of RGB are the secondary colors of CMY. But the colors created

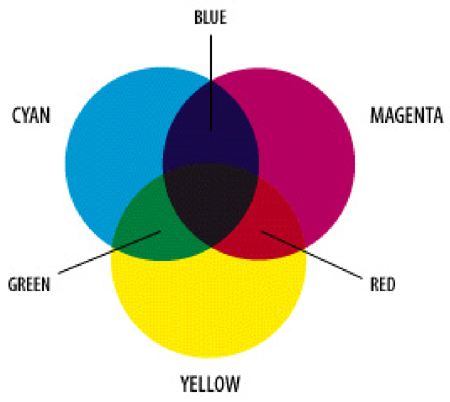


Figure 3.4: The CMY color space and subtractive model.

by the subtractive model of CMY don't look exactly like the colors created in the additive model of RGB. Particularly, CMY cannot reproduce the brightness of RGB colors.

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = 1 - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.1}$$

XYZ Space

The XYZ color space defines virtual primary colors X, Y, and Z, which are the wavelengths that the rods and cones in the human eye are most sensitive. The Y primary was specifically designed to follow the luminous efficiency function, illumination, of human eyes. The most important characteristic of this system is that all colors can be defined by a suitable arrangement of the three primary colors. The transformation from the RGB to XYZ is defined as

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.618 & 0.177 & 0.205 \\ 0.299 & 0.587 & 0.114 \\ 0.000 & 0.056 & 0.994 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.2}$$

xyz and rgb Spaces

If we are only interested in the ratio of the standard primary colors the xyz color space is obtained. Here, x, y, and z are called the chromaticity coordinates and are calculated by dividing respectively the X, Y, and Z coordinates by their total sum. It is obvious that the intensity information is factored out of the system, because the chromaticity coordinates reject only the ratio of the three standard primary colors. Since the sum of the chromaticity coordinates equals unity, two of these three quantities are sufficient to describe a color. However, the xyz representation becomes unstable when intensity is small. It is defined as

$$x = \frac{X}{X+Y+Z}, \quad y = \frac{Y}{X+Y+Z}, \quad z = \frac{Z}{X+Y+Z} \tag{3.3}$$

Similarly, the rgb color space transforms are specified as

$$r = \frac{R}{R+G+B}, \quad g = \frac{G}{R+G+B}, \quad b = \frac{B}{R+G+B} \tag{3.4}$$

Lab Space

In 1976, the CIE created a refined model of the XYZ called the Lab (Fig.3.5). In the Lab, L is the luminance value, a is such that $-a$ is green and $+a$ is red, b is a value for which $-b$ is blue and $+b$ is yellow. The Lab gamut covers all colors in visible spectrum. The Lab color space is also called as YIQ in some context. If we define X_0, Y_0, Z_0 as the values of a nominally white object-color stimulus, i.e.

$$X_0 = \frac{X}{\max(X)}, \quad Y_0 = \frac{Y}{\max(Y)}, \quad Z_0 = \frac{Z}{\max(Z)} \tag{3.5}$$

then transformation from XYZ to Lab can be formulated as

$$L = \begin{cases} 116(Y_0)^{\frac{1}{3}} - 16 & Y_0 > 0.008856 \\ 903.3(Y_0) & Y_0 \leq 0.008856 \end{cases}, \tag{3.6}$$

$$a = 500[(X_0)^{\frac{1}{3}} - (Y_0)^{\frac{1}{3}}], \tag{3.7}$$

$$b = 200[(Y_0)^{\frac{1}{3}} - (z_0)^{\frac{1}{3}}]. \tag{3.8}$$

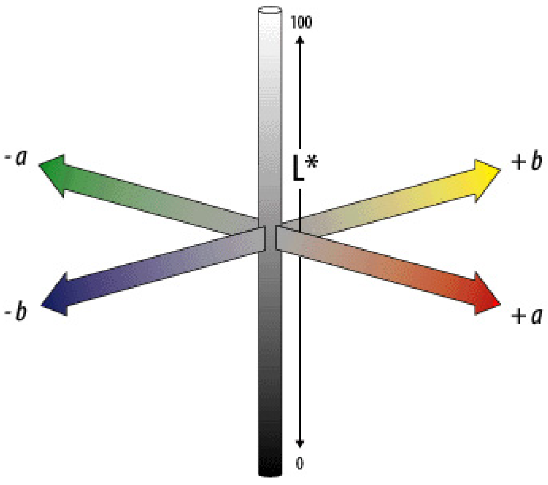


Figure 3.5: The CIE-*Lab* color space.

YUV Space

The YUV color space is the basic color space used by the composite color video standards. Instead of separating colors, the YUV signal separates the intensity Y from the color components U, V that correspond to the hue and saturation aspects. The human eye is less sensitive to the hue and saturation. Therefore the U, V components, which are also called as chrominance, are usually subsampled in image applications. The YUV parameters in terms of RGB is

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.9}$$

The CIE-YC_rC_b is a scaled and offset version of the YUV color space. A similar color space is defined by simply calculating color ratios $Q_{rg} = R/(R+G)$ and $Q_{rb} = R/(R+B)$. This color representation is referred to as YQQ space.

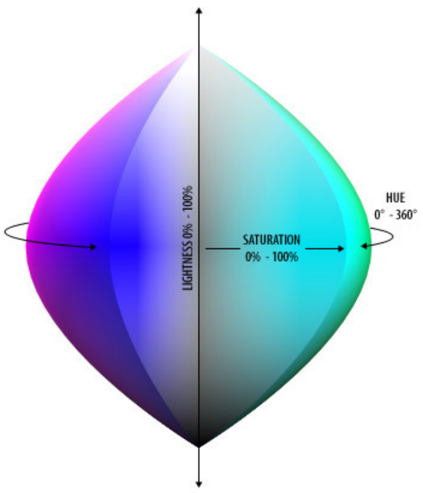


Figure 3.6: The HSI color space.

HSI Space

The formulation of YUV space in polar coordinates, the HSI space, distinguishes hue, specified by the angle, saturation as radial component, and intensity again vertical to the UV plane. Thus, the HSI space is basically an offshoot of the Munsell system. Hue is described by the angle about the vertical axis, beginning with red 0° through 360°. At every 60° another hue is defined; respectively yellow, green, cyan, blue, and magenta. Saturation is a value between 0 and 1, and describes the ratio of the saturation of the selected hue to its maximum. Intensity is also a value between 0 and 1. 0 is the value representing the bottom of the hexcone and 1 represents the top of the hexcone. Selecting a hue and setting $V = 1$ and $S = 1$, shades are obtained by decreasing V and tints are obtained by decreasing S. The HSI is formulated as

$$H = \arctan\left(\frac{\sqrt{3}(G-B)}{2R-G-B}\right) \tag{3.10}$$

$$S = 1 - 3\min(r,g,b), \tag{3.11}$$

$$I = \frac{R+G+B}{3}. \tag{3.12}$$

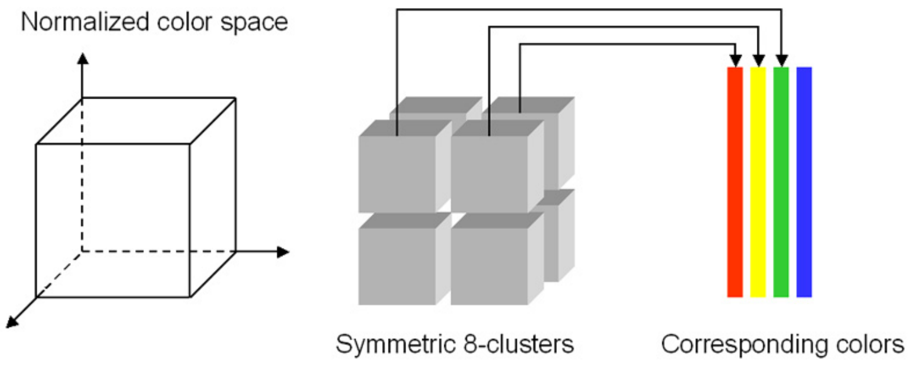


Figure 3.7: Clustering a cubic color space into 8 identical bins.

The problem of HSI is that H becomes unstable when S is near zero due to the non-removable singularities in the nonlinear transformation, which a small perturbation of the input can cause a large jump in the transformed values.

3.1.2 Comparison of Color Spaces

The selection of color space is a first task for a video processing framework, particularly for video segmentation. We investigated extensively the properties of color spaces to determine suitable color space for video segmentation. Each segmentation technique has its own unique characteristics. The parameters and relations such as color similarity and distance functions, linkage thresholds, gradients, etc., are all depend the choice of the color space and segmentation technique.

We designed three tests to examine segmentation performances in each color spaces. The first test inspects vector clustering of color spaces that have orthogonal domains; namely RGB, XYZ, YUV, and YC_bC_r. The target color space is divided into identical bins, and in each bin the included colors are inspected. Using 8-identical cubes is the best way to divide an orthogonal space as in Fig. 3.7. The results are shown in Fig. 3.8. In the figure, the first set of 8-columns corresponds the colors of 8-bins in the RGB space. The other sets are for the YUV, YC_bC_r, XYZ respectively.

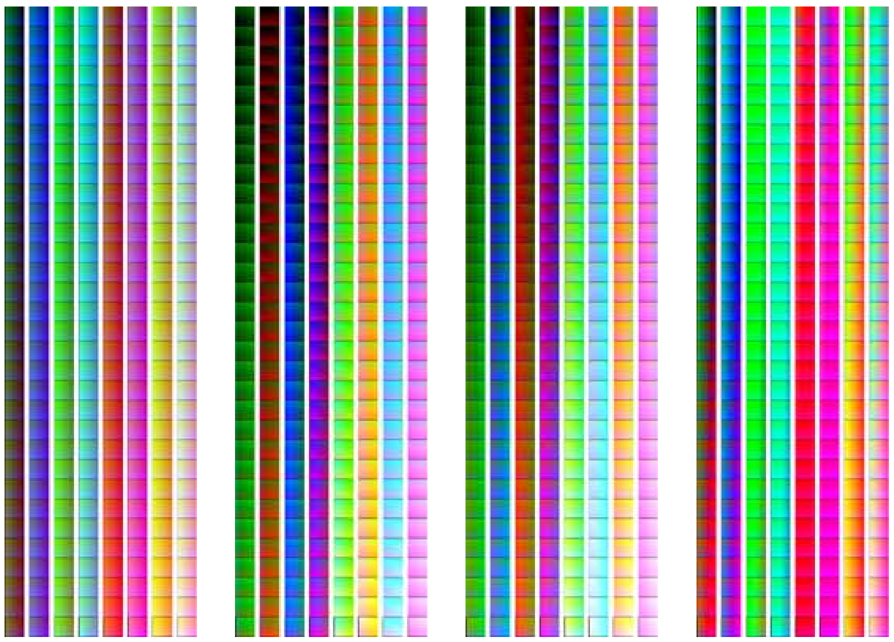


Figure 3.8: The colors in each one of the 8-clusters in: (a) RGB, (b) YUV, (c) YC_bC_r, and (d) XYZ color spaces. As visible, RGB has the most illumination depended clustering.

Clustering of some other color spaces, e.g. HSI, is not orthogonal, therefore dividing into identical bins is not straightforward. We concluded that XYZ and RGB clusters differ in luminance. XYZ has the worst hue homogeneity. Hue homogeneity is better in YC_bC_r. YUV is not luminance sensitive as RGB, however, RGB has better hue separation.

In a second test, we used adaptive-k means clustering of color vectors from an test image. Each input image is clustered into 2, 4, and 8 segments. Again, the magnitude and Euclidian distance norms are used in clustering. We presented magnitude norm based results in Figures 3.9 - 3.10 for standard MPEG test sequences, yet the test was conducted with dozens of image. We examined the RGB, rgb, YUV, and HSI color spaces that represent a fair spectrum of all color spaces. Each color channel is normalized before clustering. The segmented images are color coded for illustration purposes. Inspection of the results established that there is a marginal difference between the YUV and HSI spaces in terms of segmentation performance. The RGB and rgb are inferior expect some specific cases.

In our segmentation framework, the smallest constituents of video objects are obtained by point-wise color statistics. To manage spatial connectivity, a growing based approach is utilized. Therefore, a color space that endows better segmentation for region growing would give superior performance in our method. Our third test examines the region growing performances for three different cases:

1. hue and saturation are changing, but illumination is constant (Fig.3.11-a),
2. saturation is changing but hue and illumination are constant (Fig.3.11-b),
3. illumination is changing but hue and saturation are constant (Fig.3.11-c).

For each case, a test image is generated. By starting from the center point, regions are grown. A centroid-linkage method which is explained in the next chapter, is used to grown regions. When a region does not grow any further, another region is initiated. The seed of the new region is searched in a circular region that extends from center to outer boundary around the center of the image. Each region is color coded in the figure. The color distance threshold is kept same for all color spaces. In this way, we were able to evaluate performances of color spaces for segmentation. We tested the RGB, YUV, HSI, and rgb color spaces. The segmented regions as given in Fig.3.11. The second row is the region growing results for RGB color space. Results shows that this color space is highly sensitive to the illumination changes as expected. However, it is also sensitive to the hue differences more than any other space. In RGB space, saturation difference of same color may ignite another region as well.

The rgb color space uses illumination scaled color values. It was visible on the test results too; when the illumination was changing as in the Fig.3.11-c, the

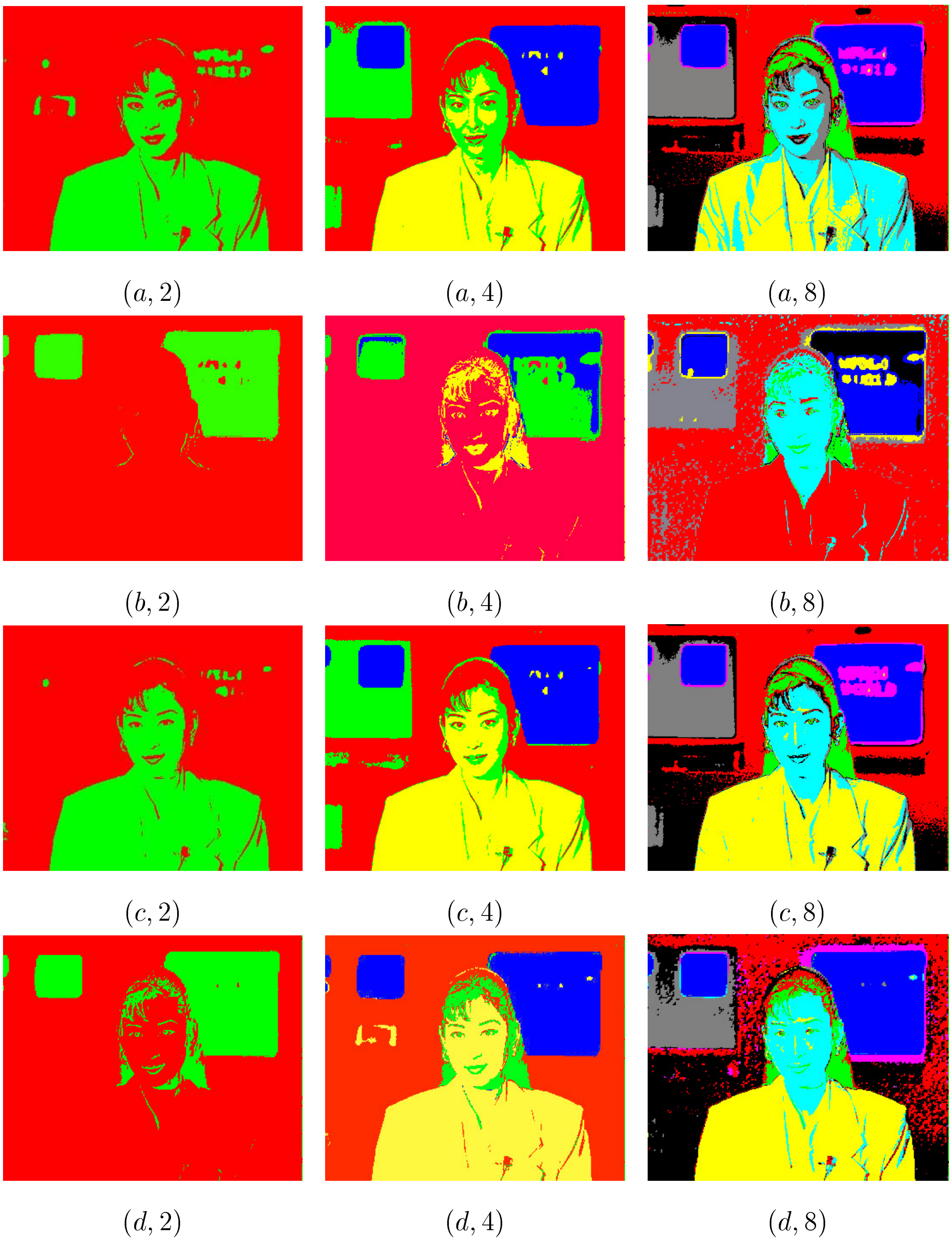


Figure 3.9: *Akiyo* is segmented into 2 (1^{st} column), 4 (2^{nd} column), and 8 clusters (3^{rd} column). Row (a) corresponds to the RGB, (b) rgb, (c) YUV, and (d) HSI color spaces.

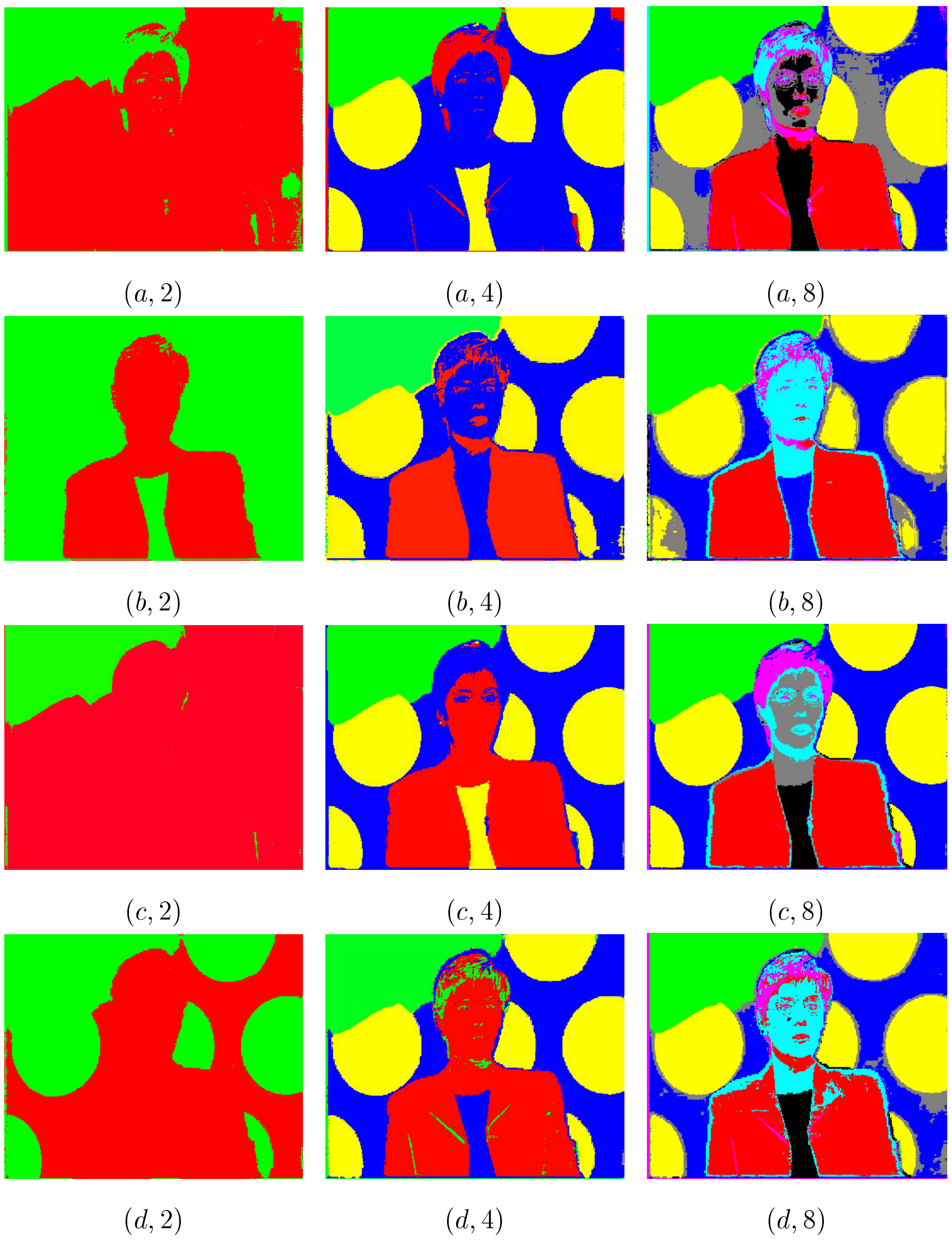


Figure 3.10: *Hanna* is segmented into 2 (1st column), 4 (2nd column), and 8 clusters (3rd column). Row (a) corresponds to the RGB, (b) rgb, (c) YUV, and (d) HSI color spaces.

segmentation still grown a large region. The very small regions in the center are a result of scaling with very small RGB values. The rgb space was not sensitive enough to the hue differences either. However, it performed well in case of the saturation changes.

The forth row shows the results of the YUV color space. The sensitivity to the hue was satisfactory, it was between the above color spaces. In addition, it was able to catch saturation and illumination changes without over segmentation. The last row presents the HSI space results. In this color space, the difference computation is more complicated due to the polar coordinates. HSI fails in case of dim or very highly saturated colors. Also, for whitish colors it diverges since the hue component has different values, which is a disadvantage. The HSI performed less than acceptable, not perceptively as good as the YUV space.

Note that specifying a color space is a big challenge, yet finding the best feature for segmentation is even a bigger challenge. Since the complexity of the analysis demands cumbersome chores, we concentrated on using identical color channels blended in a magnitude or L^2 distance norm.

In addition to the above test, we had the following observations on color spaces:

1. The metric that is used for computing the inter-color distances is the major factor of selecting a color space. Most of the processing time of a region growing algorithm is spent by computing the color distances between image pixels. A simple metric can accelerate segmentation considerably. Color distance computation in the HSI color space requires use of trigonometric functions or conditionals due to the coordinates. On the other hand, color distance is computed using the magnitude or L^2-norms for the RGB, YUV, XYZ, and rgb.

2. The RGB suffers from an important drawback for many vision applications where features of the environment are marked with identifying colors, such as "orange", "magenta", etc. We would like our segmentation to be robust in the

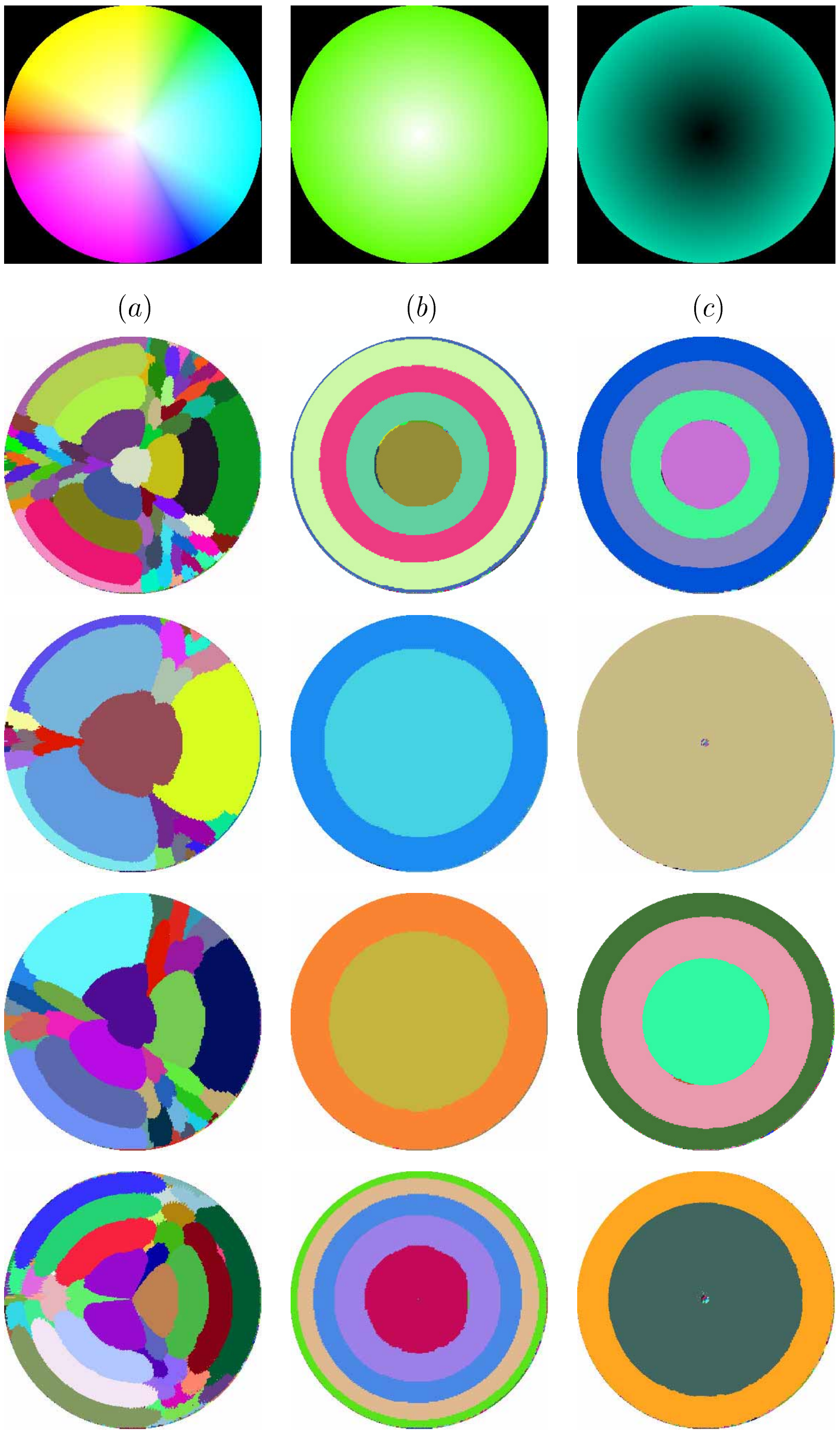


Figure 3.11: The test images (1st row), the grown regions in the RGB space (2nd row), the grown regions in the rgb space (3rd row), the grown regions in the YUV space (2nd row), the grown regions in the HSI space (2nd row).

face of variations in the brightness of illumination, so it would be useful to define "orange" in terms of a ratio of the intensities of R, G and B in the pixel. This can be done in the RGB color space, but the volume implied by such a relation is conical and cannot be represented with simple thresholds.

3. In contrast, the HSI and YUV have the advantage that chrominance is coded in two of the dimensions (H and S for HSI or U and V for YUV) while intensity is coded in the third. Thus a particular color can be described as a "column" spanning all intensities.

4. The rgb and xyz are also robust to illumination changes but more expensive transformations where each of the component colors is specified as a fraction of the intensity, and the intensity should be added as another dimension. This projection into a 4D space improves accuracy, but with the cost of the extra dimension to process and three divisions per pixel to calculate the fractions.

5. The YUV, Lab, and HSI enable easier clustering of the human skin color tones, and perform in accordance with human reception with reference to the opponent-colors theory.

6. MPEG standards utilize the YUV color space. In case of processing an MPEG encoded video, using a different color space rather than the YUV requires extra computation for color space conversions.

As a result, we preferred to use the YUV color space in our segmentation framework. By considering the hue discrimination capability, a second tier would be the HSI color space.

3.1.3 Texture Elements

We preferred to compute the texture components by the Gabor transform [63]. The Gabor filters are quadrature filters and can be used to extract a certain

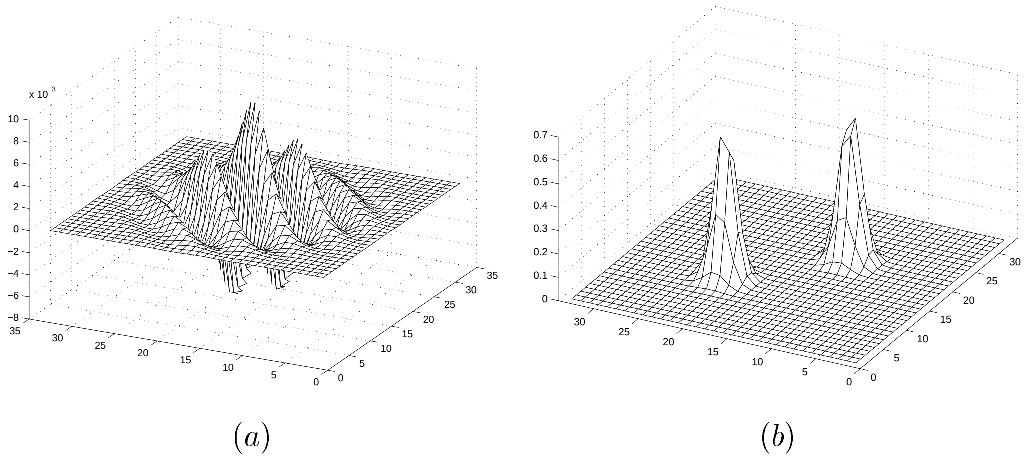


Figure 3.12: A sample Gabor function that detects diagonal lines: (a) in spatial domain, (b) corresponding response in frequency domain.

wavelength and orientation from an image with a specified bandwidth. A sample Gabor function in both spatial and frequency spaces are shown in Fig. 3.12. Because the Gabor filter is a quadrature filter, the "energy" of the signal in the filter band can be determined by computing the square magnitude of the complex filter response. Two-dimensional Gabor filters $h(x, y)$ have the functional form

$$h(x,y) = g(x,y)e^{-2\pi(px+qy)}, \quad g(x,y) = \frac{1}{2\pi\sigma_g^2}e^{-\frac{x^2+y^2}{2\pi\sigma_g^2}} \tag{3.13}$$

where σ_g^2 specifies effective width, p, q specify modulation that has spatial frequency $f = \sqrt{p^2 + q^2}$ and direction $\theta = tan^{-1}(q/p)$, and $g(x, y)$ is the Gaussian kernel. The texture scores are computed convolving image $I(x, y)$ by the Gabor filters frame-wise

$$h(x,y) \otimes I(x,y). \tag{3.14}$$

It is acceptable to chose the values for the spatial frequency $f = 2, 4, 8$ and the direction $\theta = 0, \pi/4, \pi/2, 3\pi/4$, which leads to a total of 12 texture features. The computed scores are then normalized as described in [97]. We give 12-texture scores for a frame of *Akiyo* in Fig. 3.13. The computed scores are then normalized as described in [97].

The computation of the texture elements is computationally very expensive to be real-time for the existing state-of-art technology. Moreover, blending texture and color components into a similarity measure demands elaborately extracted weighting parameters for each component. Selection of proper weights is intrinsically a troublesome task, yet the normalization of each different type of texture score is not straightforward either. Despite using texture may improve the segmentation results for specific cases, it is omitted in practice.

3.1.4 Edge Elements

Edge magnitude and orientation are among the other elements of spatiotemporal data. To compute the edge magnitude and orientation, the Canny operator [23] is used.

The Canny operator takes as input a gray scale image, and works in a multistage process. First of all the input image is smoothed by Gaussian convolution. Then a simple 2D first derivative operator, somewhat like the Roberts Cross, is applied to the smoothed image to highlight regions of the image with high first spatial derivatives. Edges give rise to ridges in the gradient magnitude image. The algorithm then tracks along the top of these ridges and sets to zero all pixels that are not actually on the ridge top so as to give a thin line in the output, a process known as non-maximal suppression. The tracking controlled by two thresholds: α_1 and α_2, with $\alpha_1 < \alpha_2$. Tracking can only begin at a point on a ridge higher than α_1. Tracking then continues in both directions out from that point until the height of the ridge falls below α_2. Dual thresholds helps to ensure that noisy edges are not broken up into multiple edge fragments. Increasing the width of the Gaussian kernel reduces the detector's sensitivity to noise, at the expense of losing some of the finer detail in the image. The localization error in the detected edges also increases slightly as the Gaussian width is increased. The results of edge detection is demonstrated in Fig. 3.14. The Canny operator is only applied to the luminance component. Usually, a 7×7 window with

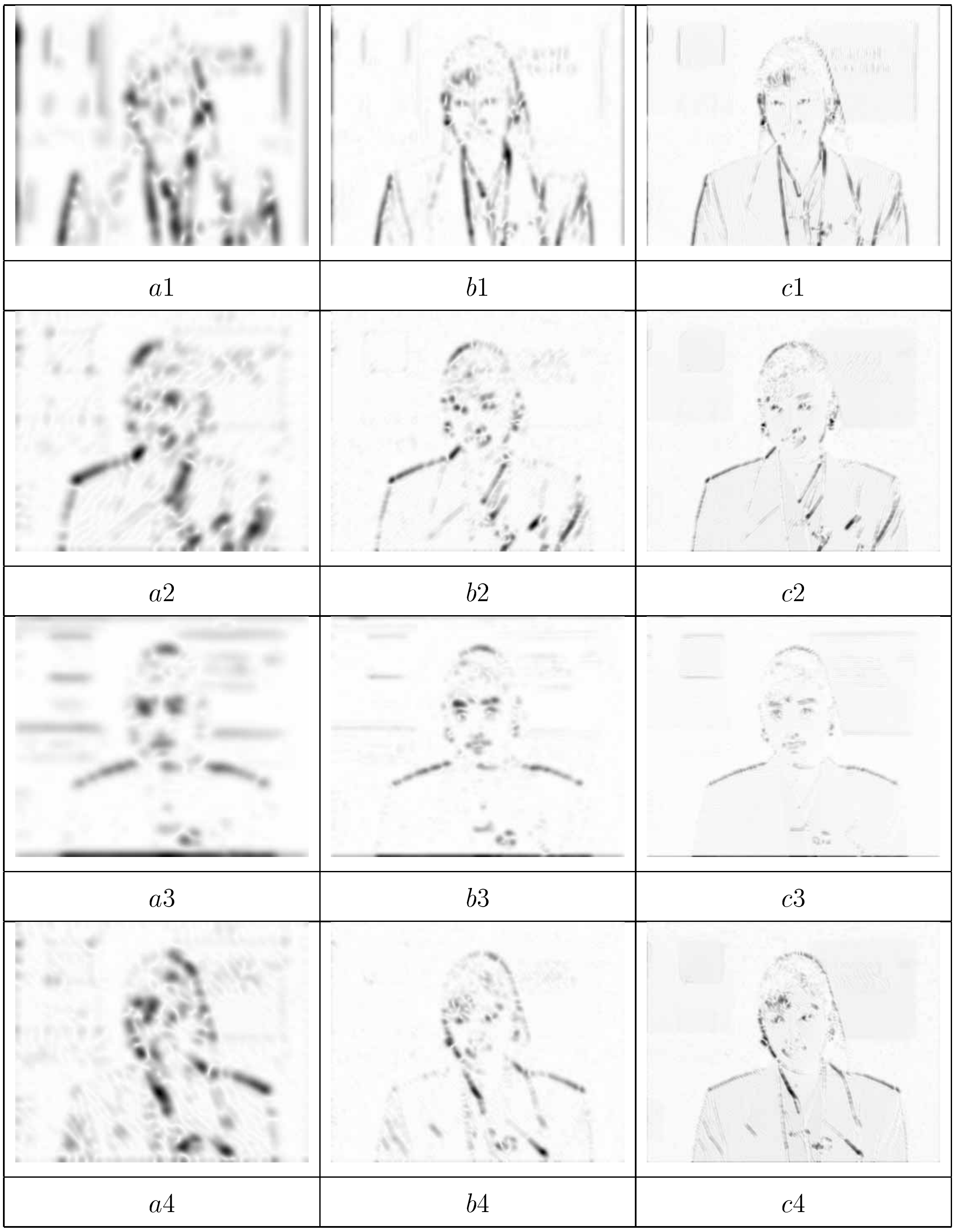


Figure 3.13: Texture features obtained by the Gabor filters for an image from *Akiyo*.

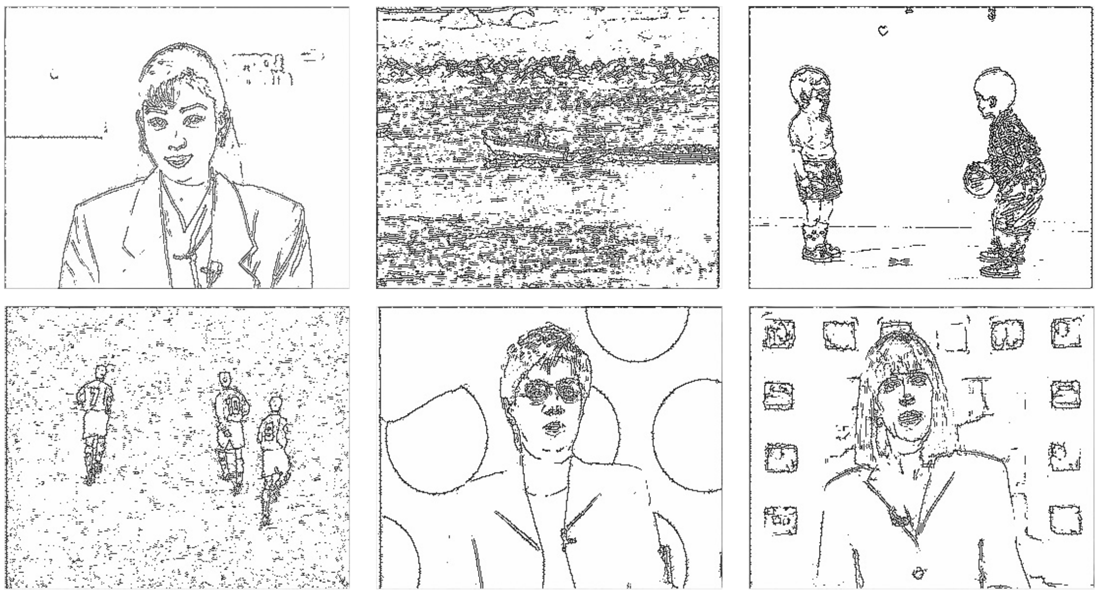


Figure 3.14: Edge magnitudes of sample frames.

a smoothing factor $\sigma = 0.5$ would fulfill most edge estimation tasks.

3.2 Filtering and Simplification

Over segmentation has several disadvantages; it slows down the algorithm, increases memory load by increasing the number of regions, more importantly it causes an additional problem of clustering small regions of slightly textured image parts. Although image noise can be removed by using low-pass filtering, median filtering, and morphological operators, such filters often disturb the object boundaries by smearing or completely changing the edge structure.

3.2.1 Implementation of Fast Median Filter

Median filter sorts a list with respect to the magnitudes of the list elements, then finds the magnitude of the element at the middle of the list, e.g., magnitude

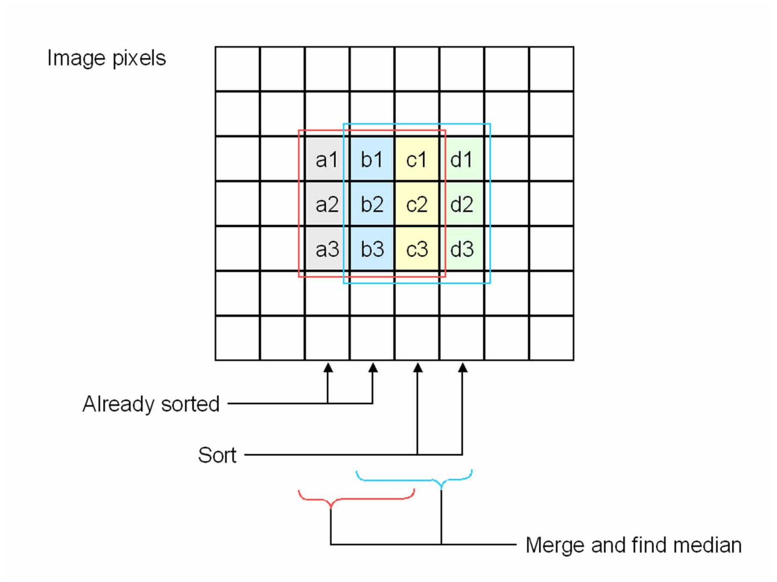


Figure 3.15: Fast 3×3 median filtering exploits 2D coherence.

of the third element in a five elements list. We utilized a fast 3×3 median filter implementation [70] that exploits 2D coherence by dividing the selection process into several 1D sorting problem to take the advantage of working in smaller lists. Due to the fact that sorting algorithms are logarithmically complex, using such filters are computationally very expensive. In the implementation, two horizontal adjacent medians are computed in one step as shown in Fig.3.15. Let a, b, c, d be 3×1 columns of a 3×4 image window. Let b_{3x3}, c_{3x3} represent the 3×3 windows around the center points of b and c.

1. First, slices c and d are sorted, slices a and b were already sorted in previous step (6 comparisons).

2. Slice b and c are merged to thick slice bc (5 comparisons).

3. Slice a and bc are merged to compute median for b_{3x3} (4 comparisons).

4. Slice d and bc are merged to compute median for c_{3x3} (4 comparisons).

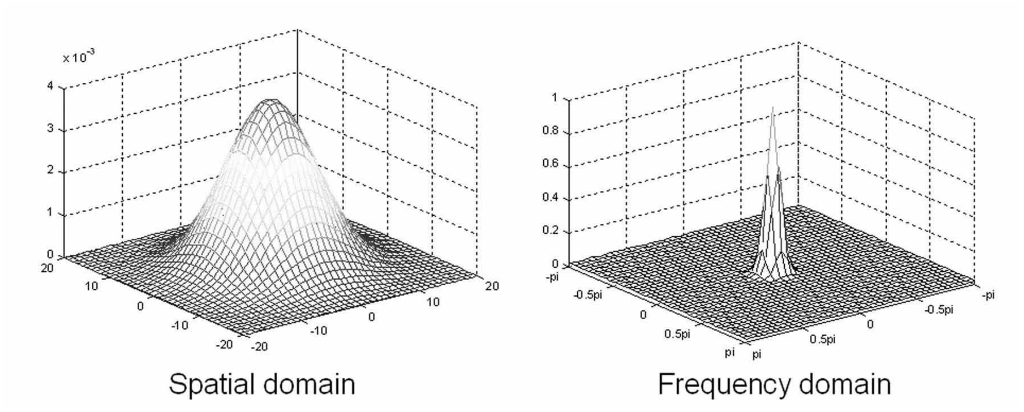


Figure 3.16: The kernel of a Gaussian filter, and its frequency domain correspondence.

Therefore, the number of necessary comparisons is reduced to from 30 to 9.5. The comparisons are carried out using nested if-loops.

3.2.2 Low-Pass Filtering by Gaussian Kernels

Gaussian filter is basically a low-pass filter that is used to blur image and remove detail and noise. Its kernels in both frequency and spatial domains are Gaussian "bell-shaped" kernel,

$$g(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{3.15}$$

where x, y are coordinates, $sigma^2$ is the variance that acts as a smoothing parameter. Gaussian filter outputs a weighted average of each pixel's neighborhood, with the average weighted more towards the value of the central pixels. Thus, it provides gentler smoothing and preserves edges better than a similarly sized mean filter. However, Gaussian filter assumes that noise is normal distributed. For salt-and-pepper type of noise, this filter reduces the intensity of the noise, but also attenuates high frequency detail and smears noise out over a larger spatial region. Another drawback of the low-pass filtering is also obvious at Gaussian filtering; the region boundaries are smeared and disturbed.

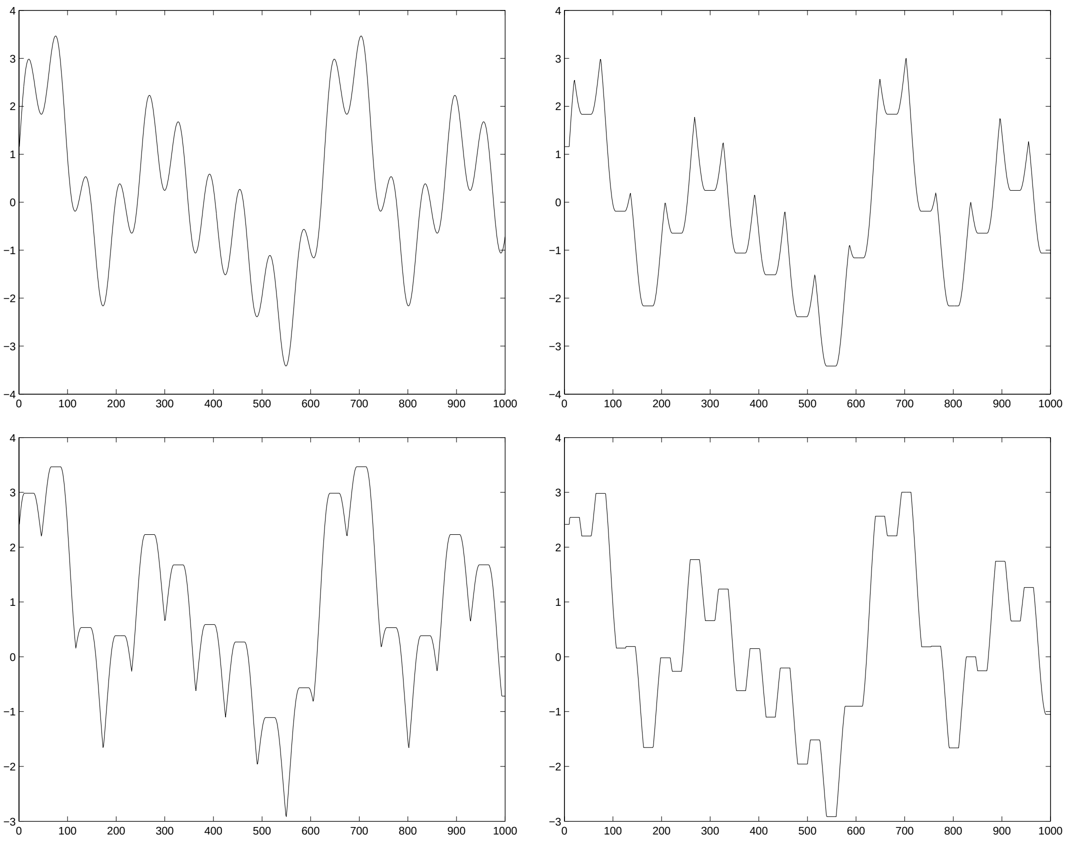


Figure 3.17: Basic binary morphological operators; original, opened, closed, and smoothed.

3.2.3 Smoothing by Morphological Operators

Morphological filters are nonlinear filters suited to the selective removal of image structures. This can be achieved by probing image with another set of given shape called as the structuring element. Erosions and dilations are the two fundamental morphological operators because all other operators are based on their combinations. Erosion answers the question "Does the structuring element fit the set?" The eroded set is the locus of points where the answer to this question is affirmative. We will consider structuring elements, A, that comprise a finite number of pixels and are convex and bounded to simplify matters. Gray-level dilation is given by

$$D^g(I,A) = I \oplus A = \max_{i,j \in A}\{I(x-i, y-j) + A(i,j)\}. \tag{3.16}$$

For a given output coordinate $[x, y]$, the structuring element is summed with a shifted version of the image I and the maximum encountered over all shifts within the domain of A is used as the result. Similarly, gray-level erosion is defined as

$$E^g(I,A) = I \ominus A = \min_{i,j \in A}\{I(x-i, y-j) + A(i,j)\} \tag{3.17}$$

The dilation is the dual operator of the erosion and is based on the following question: "Does the structuring element hit the set?" The dilated set is the locus of points where the answer to this question is affirmative. In many situations the seeming complexity of gray level morphological processing is significantly reduced through the use of symmetric structuring elements where $A(i,j) = A(-i,-j)$. The most common of these is based on the use of $A = \mathit{constant} = 0$. For this important case and using again the domain of A, the definitions above reduce to:

$$\begin{aligned} D^g(I,A) &= I \oplus A = \max_A(I,A) && (3.18)\\ E^g(I,A) &= I \ominus A = \min_A(I,A) && (3.19) \end{aligned}$$

Once an image has been eroded, there exists in general no inverse transformation to get the original image back. The idea behind the morphological opening is to dilate the eroded image to recover as much as possible the original image. The opening by a structuring element A is denoted by O and is defined as the erosion by A followed by the dilation with the transposed A. The idea behind the morphological closing is to build an operator tending to recover the initial shape of the image structures that have been dilated. This is achieved by eroding the dilated image. To summarize

$$\begin{aligned} \mathit{Open}(I,A) &= \{\{I \ominus A\} \oplus A\} && (3.20)\\ \mathit{Close}(I,A) &= \{\{I \oplus A\} \ominus A\}. && (3.21) \end{aligned}$$

Morphological smoothing is based on the observation that a gray-level opening smoothes image from above the brightness surface, and the gray-level closing smoothes from below. Thus, smoothing is achieved by finding maxima and minima consecutively in a window shaped as the structure element

$$\begin{aligned} Smooth\{I, A\} &= Close(Open(I, A), A) \\ &= \{\{\{\{I \ominus A\} \oplus A\} \oplus A\} \ominus A\} \qquad (3.22) \\ &= \min_{a \in A}\{\max_{a \in A}\{\max_{a \in A}\{\min_{a \in A}\{I, A\}\}\}\} \qquad (3.23) \end{aligned}$$

Contrary to linear filters, morphological filters preserve sharp edges. The basic idea behind a morphological filter is to suppress image structures selectively. These structures are either noise or irrelevant image objects. We used a 3×3 block as the structuring element A in the given results.

3.2.4 Image Simplification by Robust Estimators

Simplification can be viewed as a smoothing operator such that object boundaries and object colors are preserved as much as possible. When region growing is the target application, image simplification becomes a useful tool since it can effectively remove unnecessary texture that causes over segmentation.

The term robust is, in general, referring to a statistical estimator, it means "insensitive to small departures from the idealized assumptions for which the estimator is optimized." The word small can have two different interpretations, both important: either fractionally small departures for all data points, or else fractionally large departures for a small number of data points. Out of various sorts of robust statistical estimators, we prefer to employ M-estimates that follow from maximum-likelihood arguments. M-estimates are usually the most relevant class for model-fitting, that is, estimation of parameters.

Given a set of observations, one often wants to condense and summarize the data by fitting it to a model that depends on adjustable parameters. Suppose that

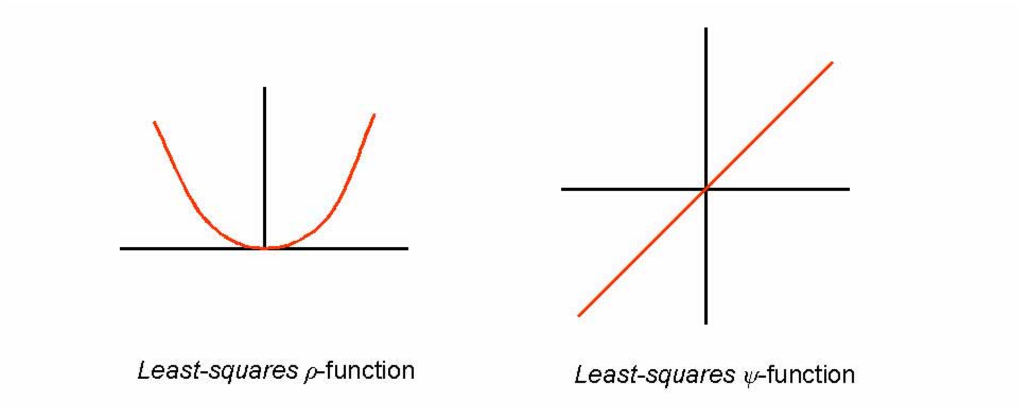


Figure 3.18: The quadratic error norm function, and its derivative. The derivative determines how much the difference will be weighted.

we are fitting N data points $(x_i; y_i)$ $i = 1...N$, to a model that has M adjustable parameters a_j $j = 1...M$. The model predicts a functional relationship between the measured independent and dependent variables,

$$y(x) = y(x; a_1...a_M) \tag{3.24}$$

where the dependence on the parameters is indicated explicitly on the right-hand side.

Least-Squares Estimator

A number of techniques are known to fit data to a model. The most common prior art techniques use the familiar least-squares fit,

$$\min_{a_1 \,...\, a_M} \quad \sum_{i=1}^{N} [y_i - y(x_i; a_1...a_M)]^2 \tag{3.25}$$

If each data point y_i has a measurement error that is independently random and distributed as a normal distribution around the a model $y(x)$, and if the standard deviations of the normal distributions are the same for all data points, then the probability P of the data set is the product of the probabilities

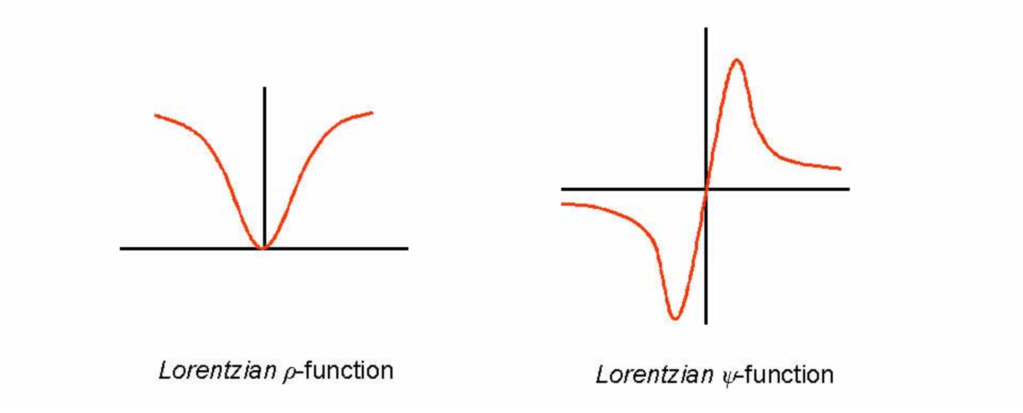


Figure 3.19: The Lorentzian error norm, and its derivative.

$$P \propto \prod_{i=1}^{N} \exp\left[-\frac{1}{2}\left(\frac{y_i - y(x_i)}{\sigma}\right)^2\right] \tag{3.26}$$

Maximizing above equation is equivalent to maximizing its logarithm, or minimizing the negative of its logarithm,

$$\sum_{i=1}^{N}\left(\frac{y_i - y(x_i)}{\sigma}\right)^2 \tag{3.27}$$

If the derivative of the above equation is taken with respect to the parameters a_k, then the resulting equations must hold at the minimum,

$$\sum_{i=1}^{N}\left(\frac{y_i - y(x_i)}{\sigma^2}\right)\left(\frac{\partial y(x_i; a_k)}{\partial a_k}\right) = 0, \qquad k = 1...M. \tag{3.28}$$

This demonstrates that least-squares fitting is a Maximum-likelihood estimation of the fitted parameters when the measurement errors are independent and normally distributed with constant standard deviation.

Robust Estimator

The data fitting analogy can be generalized using an error norm function ρ instead of the least-squares. The fitting formula for the estimated parameters a in a model $y(x;a)$, that is, equation 3.25 becomes,

$$\min_{a_1 \ldots a_M} \quad \sum_{i=1}^{N} \rho(y_i - y(x_i; a_1 ... a_M), \sigma). \tag{3.29}$$

The Maximum-Likelihood formula for the probability P of the data set is the product of the probabilities of each data point can be rewritten as,

$$P \propto \prod_{i=1}^{N} \exp\left[-\rho\left((y_i - y(x_i), \sigma\right)\right] \tag{3.30}$$

To maximize the above equation, it is necessary to minimize the expression,

$$\prod_{i=1}^{N} \rho\left((y_i - y(x_i), \sigma\right). \tag{3.31}$$

The derivative of $\rho(z)$ is defined as a function $\psi(z)$;

$$\psi(z) \equiv \frac{d\rho(z)}{dz}. \tag{3.32}$$

Then the generalization of the case of a general M-estimate is,

$$\sum_{i=1}^{N} \frac{1}{\sigma} \psi\left(\frac{y_i - y(x_i)}{\sigma^2}\right)\left(\frac{\partial y(x_i; a_k)}{\partial a_k}\right) = 0, \qquad k = 1...M. \tag{3.33}$$

Note that for the least-squares the error norm is

$$\rho\left((y_i - y(x_i), \sigma\right) = \left(\frac{y_i - y(x_i; a_k)}{\sqrt{2}\sigma}\right)^2. \tag{3.34}$$

However, the least-squares approach is notoriously sensitive to outliers; the problem being that outliers contribute "too much" to the overall solution. Outlying points are assigned a high weight by the quadratic function ρ. If the measurement errors are not normally distributed, or data may be corrupted by gross error then the least-square fit is substantially less than optimal. Therefore, what is needed is a robust estimator that can be applied to data whose measurement errors are not normally distributed. To describe the structure best fitting the bulk of the data, and to identify deviating data points are the main goals of robust estimators.

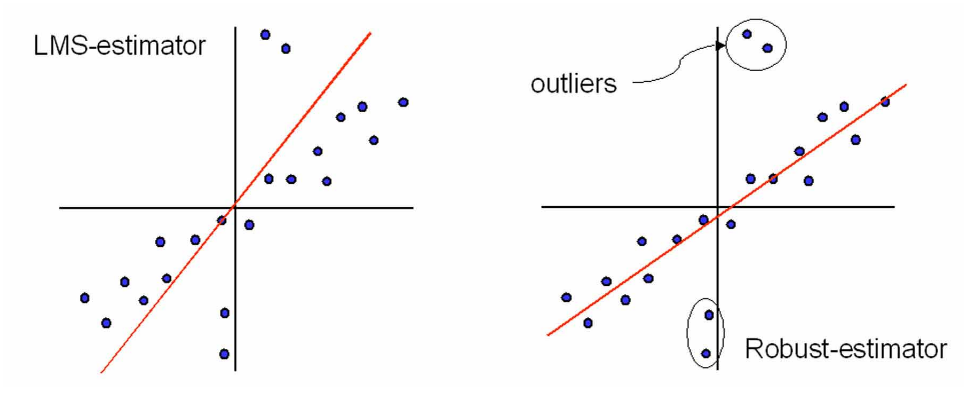


Figure 3.20: Fitting a line to the data points by using LMS and robust estimators.

Consider, for example, the least-squares quadratic ρ-function that has derivative

$$\rho(z) = \frac{1}{2}z^2, \qquad \psi(z) = 2z \tag{3.35}$$

For least-squares estimation, the influence of outliers increases linearly and without bound as shown in Fig.3.18. To increase robustness, an estimator must be more forgiving about outlying measurements. Robustness can be increased by using a ρ-function that falls away from the quadratic more quickly. Consider the following Lorentzian estimator:

$$\rho(z,\sigma) = \log\left(1 + \frac{z^2}{2\sigma^2}\right), \qquad \psi(z,\sigma) = \frac{2z}{2\sigma^2 + z^2}. \tag{3.36}$$

As shown in Fig.3.20-a for normally distributed errors, the Gaussian distribution gives more weight to the deviant data points, when fitting the line to the data. In contrast, when the tails of the distribution are even larger, as in the Lorentzian function, the function $\psi(z)$ increases the weights of moderately deviant data, but then the weights decrease, so that extremely deviant data, the "true" outliers are not counted at all in the estimation of the parameters, as shown by the line in Fig.3.20-b.

We build optimization-based filter by using downhill simplex minimization.

The downhill simplex method requires only function evaluations, not derivatives. It has a geometrical naturalness. A simplex is the geometrical figure consisting, in N dimensions, of $N + 1$ points (or vertices) and all their interconnecting line segments, polygonal faces, etc. In two dimensions, a simplex is a triangle. In three dimensions it is a tetrahedron, not necessarily regular tetrahedron. In general we are only interested in simplexes that are non-degenerate, i.e. which enclose a finite N-dimensional volume. If any point of a non-degenerate simplex is taken as the origin, then the N other points define vector directions that span the N-dimensional vector space. To start the method we need to choose the first point to start. The algorithm is then supposed to make its own way downhill through the unimaginable complexity of an N-dimensional topography, until it encounters (at least local) minimum. The downhill simplex method must be started not just with a single point, but with $N+1$ points, defining an initial simplex. The method now takes a series of steps, most steps just moving the point of the simplex where the function is largest ("highest point") through the opposite face of the simplex to a lower point. These steps are called reflections, and they are constructed to conserve the volume of the simplex (hence maintain its nondegeneracy). When it can do so, the method expands the simplex in one or another direction to take larger steps. When it reaches a "valley floor," the method contracts itself in the transverse direction and tries to ooze down the valley. If there is a situation where the simplex is trying to "pass through the eye of a needle," it contracts itself in all directions, pulling itself in around its lowest (best) point.

For each pixel of the input image, a local region centered on the pixel is assigned. The size of a local region is determined from the simplification parameter. A model function $u(x, y; a)$ where $a = [a_1, , a_M]$, is fitted to the image intensity values $I(x, y)$ within each region. In other words, the image simplification method fits a brightness model u to image data $I(x, y)$. The model function can be a polynomial, a piece-wise continuous function, as well as other surface functions that can be pa-

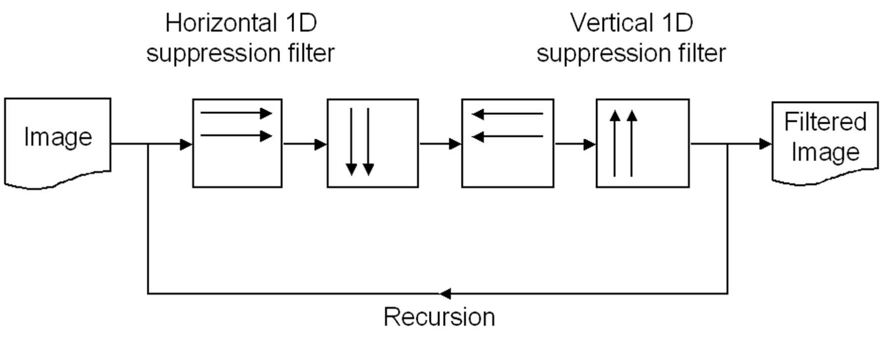

Horizontal 1D suppression filter
Vertical 1D suppression filter
Image
Filtered Image
Recursion

Figure 3.21: Image points are compared with their vertical and horizontal neighbors.

rameterized by a finite number of parameters. Within each region, an error term is computed for each pixel. The error term includes the accumulated differences between the model function's value at a pixel and the pixel's original intensity value. The error term norm is assigned as a Lorentzian function, which was given above.

3.2.5 Recursive Band-Suppression Filters

By recursive band-suppression filters, we reduce the spatial variance without any quantization nor disturbing the edge structure. Its implementation involves very simple operators. Specific but not limited to video processing, filtering tools should be computationally as simple as possible. We developed an iterative suppression technique that can be implemented by parallel programming to decrease the processing time. Let I denotes the input image. We compare the color value of a pixel $I(x, y)$ with its neighbors. If the color distance is less than a threshold δ, the pixel's color value is updated by the average of its neighbors within a local window. For parallel implementation considerations, the pixels are compared to their vertical neighbors at the first pass, and then to their horizontal neighbors as illustrated in Fig.3.21. The horizontal pass along the x-axis that gives the simplified image at the iteration level

$k+1$ is defined as

$$I(x,y)^{k+1} = \begin{cases} \frac{1}{2}[I(x,y)^k + I(x+1,y)^k] & d \leq \delta \\ I(x,y)^k & d > \delta \end{cases} \tag{3.37}$$

$$d = |I(x,y)^k - I(x+1,y)^k| \tag{3.38}$$

Similarly, the vertical pass is done by using the vertical neighbor $I(x, y+1)$. Since filtering operation intends to simplify image texture, smooth spatial variance is assumed to contribute to the noise statistics. The threshold δ is assigned to $\mu + 2\sigma$ where μ, σ are the mean and variance of the noise, respectively. For a zero-mean Gaussian noise model $N(0, \sigma^2)$, the threshold reduces to 2σ. At this level, 95.45% of the noise has smaller magnitude than the threshold. If a noise model is unavailable, the threshold δ is assigned to the maximum color distance that does not correspond to an edge point. It is observed that $\delta \simeq 8$ is a good compromise for 2^8 bit coded channels.

For a noise contaminated 1D signal, we simulated filtering performance as given in Fig.3.22-3.23. The first signal (Fig.3.22-a) is a 1D step shaped signal that is contaminated by additive normal noise. The second row is the Gaussian filtered result using a 5-tap long kernel. The third row corresponds 21-tap Gaussian filtering. As visible, the filtered signal is becoming smoother. The forth row is morphological derivative obtained by closing and opening operations. The fifth row is the median filtering. It is perceptible that sharp steps of input signal are more preserved with median and morphological filters rather than Gaussian. The last row is the suppression filter result. The sharp edges are very well preserved, although the signal not as smooth. Note that a second pass of suppression would preserve edges as well as smooth the signal easily. Fig.3.23 shows the derivatives of the original and filtered signals. It is obvious that suppression maintains the sharp edges best.

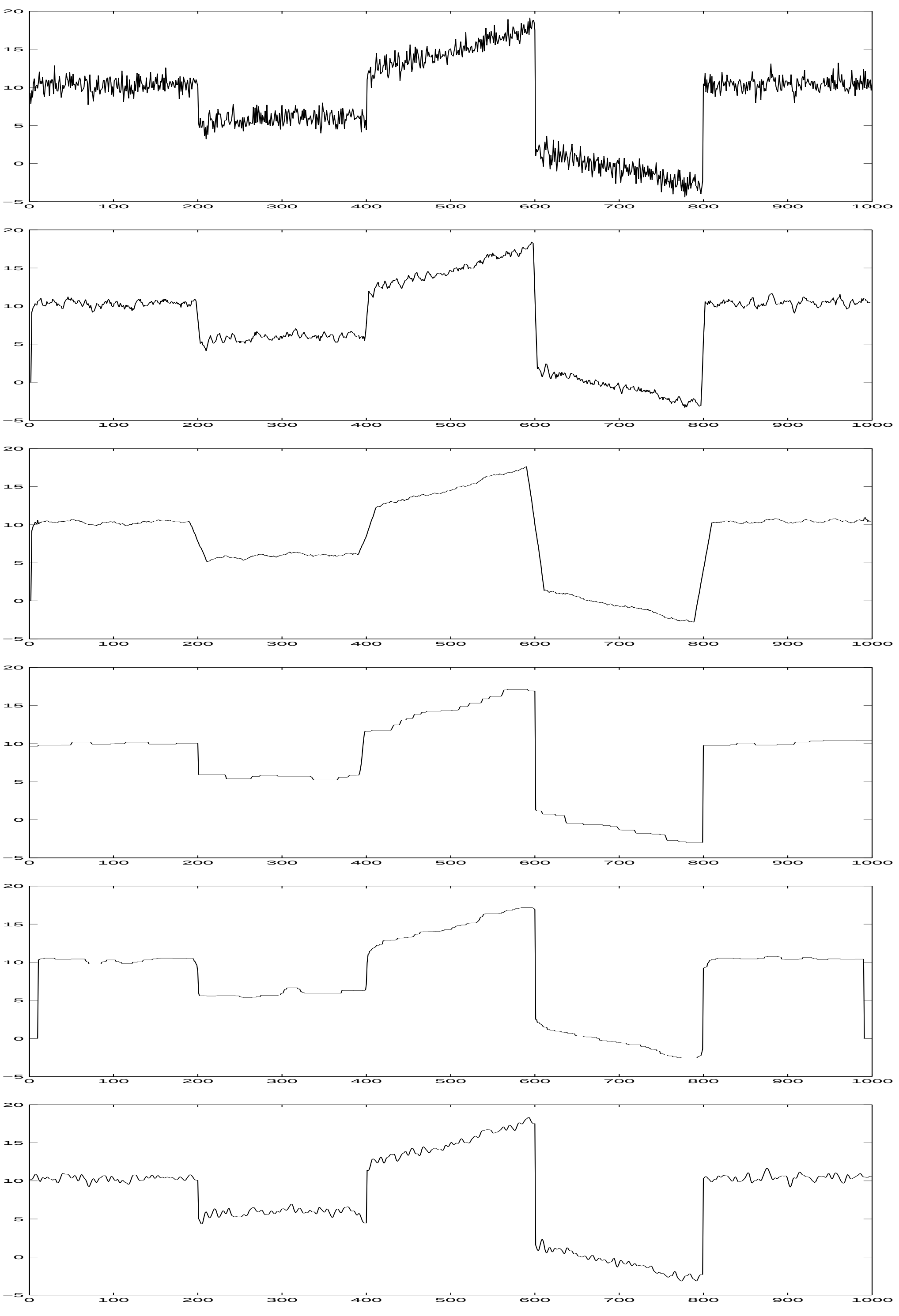


Figure 3.22: (a) original signal, (b) 5-tap Gaussian, (c) 21-tap Gaussian, (d) morphological smoothing, (e) median filtering, (f) suppression filter.

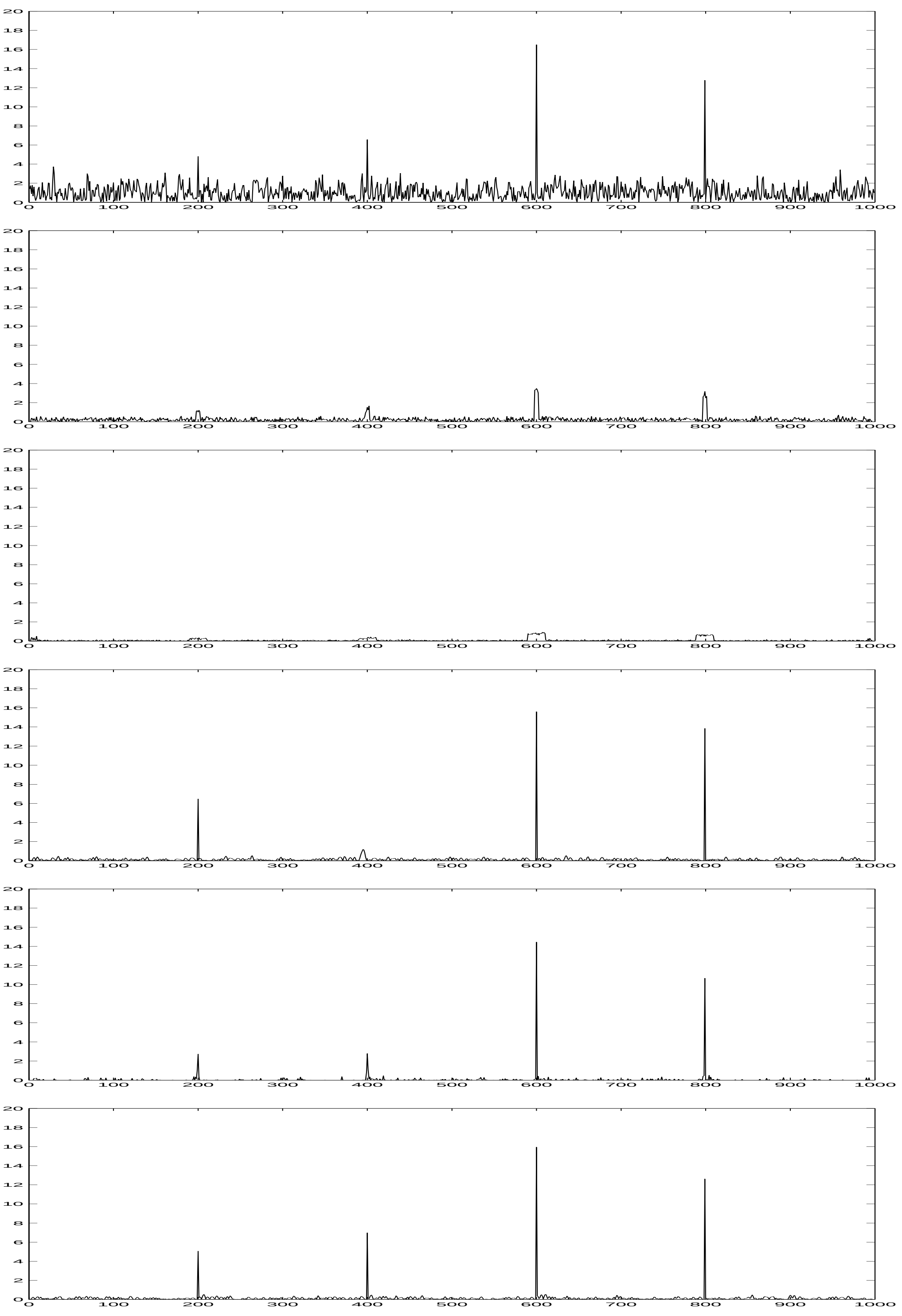


Figure 3.23: Derivatives of the (a) original image, (b) 5-tap Gaussian, (c) 21-tap Gaussian, (d) morphological smoothing, (e) median filtering, (f) suppression filter.

3.2.6 Comparison of Filters

We exhaustively tested some of the existent and also our own novel filtering techniques; median, morphological, low-pass filtering, image simplification, and band-suppression filters. These filters are applied to each video frame. It is possible to use a 3D filter kernel or apply a filter along the time axis of the video, e.g. on xt or yt-planes, however, such a filtering blurs image region where motion is present.

Sample sets of filtering results are demonstrated in figures 3.24 - 3.25. Here, image a's are the original images, b's are 3×3 fast median filtered outputs. Images c's are the Gaussian filtered results with a window size 5×5. The morphological smoothing results based on gray level opening/closing with a 3×3 structuring element are given in d's. The simplified images using Lorentzian based robust estimator are e's. The suppression filter results are f's.

The Gaussian filter is not computationally cheap, besides, it causes the most destroyed and blurred edge structure out of the filters we tested. Having a sharp edge structure, region growing segmentation do not cut across the object boundaries. On the other hand, morphological smoothing performance depends the structuring element. For structuring elements larger that 3×3 windows, it becomes significantly slower. Since it does not preserve the edges, iterative application destroys boundaries although it can smooth image well. Median filter is effective to remove salt-and-pepper noise and sparks, as well as it preserve edge structure. However, using a 3×3 kernel is not sufficient to remove the interfering texture. The fast implementation of the higher order kernels rather than 3×3 is practically very difficult. Thus, for larger window sizes it becomes computationally very demanding.

Our novel band suppression filter design is the most suitable in terms of computational simplicity. It can remove the low frequency texture effectively, too. It also preserves the edge structure best. One disadvantage of suppression is that it is unable to remove salt-and-pepper noise. The simplification by image reconstruction is a good compromise between texture smoothing, edge preservation, and computational constraint.

3.3 Change Detection Mask

The change detection mask (CDM) is defined as the color dissimilarity of two frames with respect to a given set of rules. A common operation before extracting the CDM is global motion compensation. The computational simplicity makes the CDM a good candidate for real-time applications [37]. Considering a stationary camera, consistent objects, and constant lighting conditions, the pixel-wise color difference of two consecutive frames is an indication of moving objects in the scene. However, not all the color change happen because of moving objects; camera motion, intensity changes and shadows due to the nonuniform lighting between video frames, and image noise also contribute frame difference. Thus, although the CDM gives clues about the object in the scene, it is mostly unreliable. A moving region that has smooth color texture might be missed, whereas a stationary background might be observed as in motion because of a slight shake of the camera imaging plane. Using the CDM alone to decide objects and their movements gives poor performance.

On the other hand, for the cases where the motion is very low, rigid body constrained models are ineffective, and region growing causes over-segmentation, the CDM may provide acceptable segmentation results. It can be blended into the other segmentation techniques as an additional feature to improve the accuracy.

There are various approaches to obtain change detection masks in the literature. Generally, the change detection mask between two successive frames is estimated by global thresholding the frame difference. Let $I(p,t)$ and $I(p,t{+}1)$ be the luminance values of pixel p at time t and $t{+}1$, respectively. A simple CDM $c(p,t)$ is computed by

$$c(p,t) = \begin{cases} 1 & |I(p,t) - I(p,t{+}1)| > \lambda \\ 0 & \quad otherwise \end{cases} \tag{3.39}$$

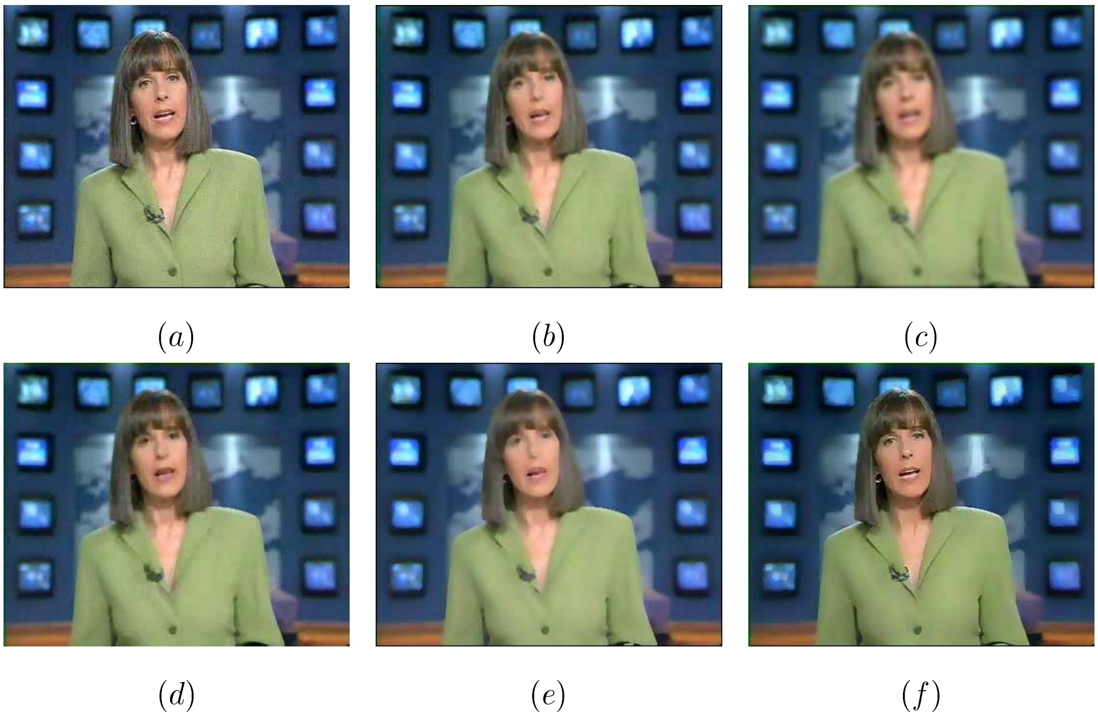


Figure 3.24: (a) The original image, (b) median filtered, (c) Gaussian filtered, (d) morphological smoothed, (e) simplified, and (f) suppression filtered results.

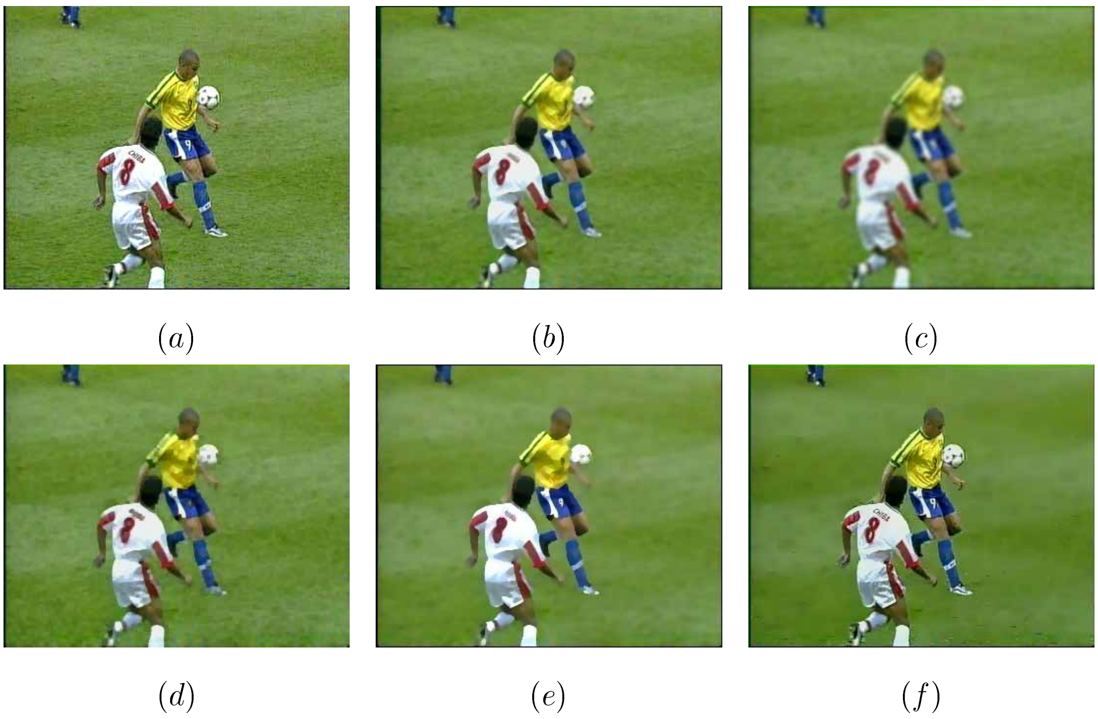


Figure 3.25: . (a) The original image, (b) median filtered, (c) Gaussian filtered, (d) morphological smoothed, (e) simplified, and (f) suppression filtered results.

Because the above definition of CDM is sensitive to the noise, a local window $\wp$ with size K is usually incorporated to compute block difference

$$c(p,t) = \begin{cases} 1 & \sum_i |I(p_i,t) - I(p_i,t{+}1)| > \lambda K, \quad p_i \in \wp \\ 0 & \qquad\qquad otherwise \end{cases} \tag{3.40}$$

Alternatively, a minimum difference score within a window in the next frame can be assigned as the CDM instead of taking pixel-wise differences

$$c(p,t) = \begin{cases} 1 & \min |I(p,t) - I(p_i,t{+}1)| > \lambda \;\; p_i \in \wp \\ 0 & \qquad\qquad otherwise \end{cases} \tag{3.41}$$

In [81], the boundaries of changed image areas are smoothed by a relaxation technique using local adaptive thresholds [1],

$$\begin{aligned} d^2 &= [I(p,t) - I(p,t{+}1)]^2 \\ c(p,t) &= \begin{cases} 1 & d^2 > \frac{2\sigma_n^2\sigma^2}{\sigma_n^2+\sigma^2}\,[k^+(v_c^+ - v_u^+) + k^x(v_c^x - v_u^x)] \\ 0 & \qquad\qquad otherwise \end{cases} \end{aligned} \tag{3.42}$$

Above, the parameters c, u correspond the pixels where the CDM is 1 and 0, rescpectively. The rule should read as follows: if d^2 exceeds the threshold term on the right hand side of the above equation, the point is set to changed, otherwise it is set to unchanged. In the threshold term, σ_n^2 is equal to twice the variance of the assumed Gaussian camera noise distribution. σ^2 is the variance of luminance differences within object regions. The terms v^+ and v^x, are a measure for the inhomogeneity of the neighborhood of pixel, which is separated into horizontal or vertical neighbors and diagonal neighbors. The term v^+ denotes the number of horizontal and vertical neighbors of the pixel with the opposite label. In the same way the term v^x denotes the number of diagonal neighbors of the point with the opposite label. The algorithm adapts frame-wise to the variances σ^2 and σ_n^2.

In [4], the mask after thresholding is connected with the previous. Specifically, the mask after thresholding is extended by pixels which are set to foreground in the of the previous frame. This is based on the assumption that all pixels which belonged to the previous should belong to the current change detection mask. In order to avoid infinite error propagation, however, a pixel from the previous is only labeled as changed in the change detection mask, if it was also labeled as changed in one of the last frames. The value denotes the depth of the memory, which adapts automatically to the sequence by evaluating the size and motion amplitudes of the moving objects in the previous frame. By the last step, the mask is simplified and small regions are eliminated.

We preferred to compute difference within a local window since taking pixel-wise frame difference is very noise sensitive. The larger the size of the matching window, the less sensitive the CDM becomes to the image noise, but also, more resistant to region movements. Thus, there is a trade-off between the size of the window and the sensitivity of the CDM. We observed that a window size between 3×3 to 5×5 is a good compromise. First, window-wise differences $\delta(p)$ is computed for a pixel p in the current frame t and a set of pixels q_n in the following frame $t{+}1$ within a matching window $\wp_1$ as

$$\delta(p, q_n) = \sum_i \sum_k |I_k(p_i, t) - I_k(q_{n,i}, t{+}1)| \tag{3.43}$$

where p_i's are the pixels in the matching window, and $q_{n,i}$'s are the pixels in the next frame around the center pixel q_n, and k represents the color channels. To compensate small motion, the center pixels q_n in the next frame are selected within a 3×3 block $\wp_2$ around p. The minimum of the computed distances $\delta(p, q_n)$'s then determines the CDM score of pixel p

$$c(p,t) = \begin{cases} 1 & \frac{\mu}{64} < \min_n \delta(p, q_n), \\ 0 & otherwise \end{cases} \tag{3.44}$$

where $q_n \in \wp_2$. Above, μ is the averaged dynamic color range of the three channels. To improve spatial connectivity of the binary CDM, an averaging operation may also be incorporated. Fig. 3.26 shows CDM results obtained with various window sizes

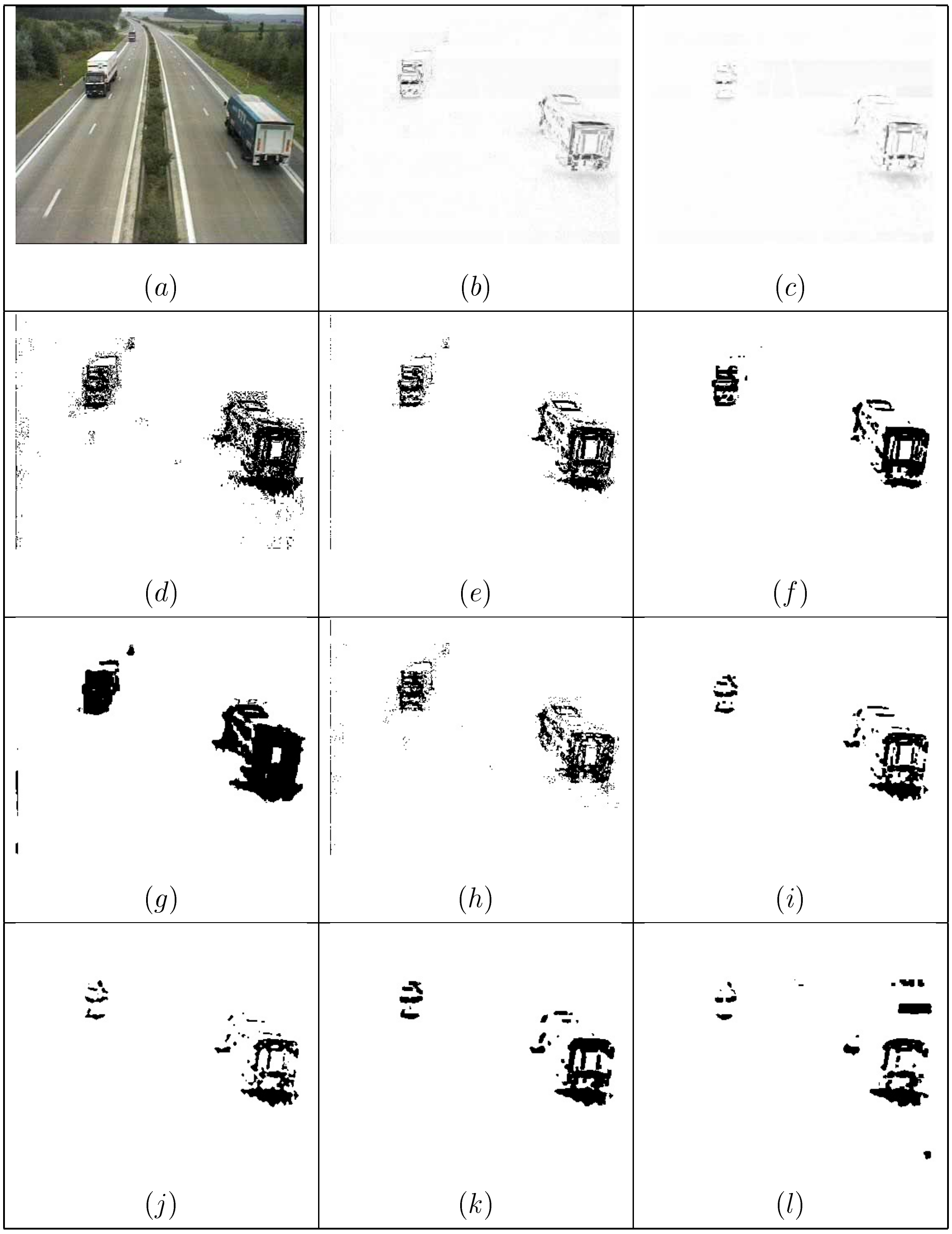


Figure 3.26: (a) A frame from video, (b) pixel-wise difference computed for that frame, (c) minimum distance in 3×3 window, (d) result *b* thresholded by 8, (e) result *b* thresholded by 16, (f) result of (*b*) averaged in 3×3, thresholded by 8, (g) result of (*b*) averaged in 5×5, thresholded by 8, (h) result of (*b*) scored in 3×3 by 8, thresholded by 4, (i) result of (*c*) averaged in 3×3, thresholded by 8, (j) result of (*c*) scored in 3×3 by 8, thresholded by 4, (k) result of (*c*) averaged in 5×5, thresholded by 8, (l) minimum distance in 5×5 window averaged in 5×5, thresholded by 4.

and techniques for a highway surveillance video. In the figure, (a) is a frame from video, (b) is the pixel-wise difference computed for that frame, (c) is the minimum distance in 3×3 windows, (d) is the result of (b) thresholded by 8, (e) is the result b thresholded by 16, (f) is the result (b) averaged in 3×3 and then thresholded by 8, (g) is the result (b) averaged in 5×5 and then thresholded by 8, (h) is the result (b) scored in 3×3 by 8 and thresholded by 4. Here, scoring refers to and thresholding the number of points in a window, i.e., as a local statistical evaluation. The Fig.3.26-(i) is the result (c) averaged in 3×3 and thresholded by 8, (j) is the result (c) scored in 3×3 by 8 and thresholded by 4, (k) is the result (c) averaged in 5×5 and thresholded by 8, l is the minimum distance in 5×5 window averaged in 5×5 and thresholded by 4. It is visible that instead of simple thresholding, an averaging based thresholding gives less noisy CDM masks. Increasing the size of the averaging window decreases the number of pixels marked as changed, and improves the CDM precision. Using the minimum in a window did not improve quality significantly, yet it is computationally demanding.

3.4 Summary

There are two main contributions of this chapter. The first contribution is the development of the spatiotemporal data structure from the input video sequence to integrate motion attributes of moving regions and spatial segmentation results. The second contribution is to development of two image filtering mechanisms, which employ robust estimators and regressive band-suppression filters. This filters are adopted in the segmentation framework to remove noise as well as simplify spatial color variance.

With regard to spatiotemporal data structure, several pixel-wise attributes are considered, and also color spaces are evaluated to determine an optimal space for volume growing based techniques. The metric that is used for computing the inter-color distances is the major factor of selecting a color space. Since most of the processing time of a region growing algorithm is spent by computing the color distances between image pixels, a simple metric can accelerate segmentation considerably. Among the other considerations are robustness towards illumination changes, capability to separate chrominance from luminance, flexibility of identifying application specific colors, i.e. skin colors, and whether the performance conforms with human reception. Experimental results indicate that the YUV space provides better segmentation for region growing while keeping the computational complexity low.

In the filtering stage, the disadvantages of classical Gaussian, morphological, and median filters are addressed. The Gaussian filter is computationally expensive, besides it blurs edge structure. On the other hand, morphological smoothing performance depends the structuring element. For large structuring elements, it becomes considerably slow. It does not preserve the edges either especially when it is iteratively applied. Median filter is effective in removing salt-and-pepper noise and sparks and preserving edge structure. However, the fast implementation of the higher order kernels is practically very difficult, and filtering operation for larger window sizes becomes computationally very demanding. The introduced band suppression filter design is the most suitable in terms of computational simplicity. It can remove the low frequency texture effectively and also preserves the edge structure. One disadvantage of suppression is that it is unable to remove salt-and-pepper noise. The simplification by image reconstruction is a good compromise between texture smoothing, edge preservation, and computational constraint. The second filter design utilizes robust estimators to fit a planar surface to windows centered around image pixels. Although it has higher complexity than the suppression filters, this filter can effectively remove noise and spatial texture utilizing downhill simplex minimization techniques.

We also elaborated standard change detection mask approaches. Instead of using pixel-wise differences, the minimum of window-wise distances are considered to compensate for small motion as well as noise. This mask is then utilized to determine one of the object descriptor as explained in the next chapter.

Chapter 4

An Unsupervised Moving Object Segmentation Framework

If you will be observant and vigilant, you will see at every moment the response to your action. Be observant if you would have a pure heart, for something is born to you in consequence of every action.

Rumi

4.1 System Architecture

Video segmentation aims to extract objects, determine events, mine for specific information, and analyze characteristics of a video sequence. Each of the segmentation algorithms summarized in Chapter 2 has is own advantages. However, they are based solely on a single criterion to solve the many-sided segmentation problem, or designed for specific applications. It would be desirable to have a general segmentation framework that combines distinct qualities of separate methods without getting hampered into their pitfalls.

Semi-automatic segmentation methods have the power of correlating semantic information with extracted regions using human assistance. Such assistance often obligates training of user to understand the behaviour of the segmentation method. Besides, for real-time and large scale systems, entering object boundaries by hand is cumbersome. Real-time video systems require user independent processing tools to deal with multiple channels of streaming data. In addition, the vast amount of video data demands for automatic segmentation. Thus, developing a segmentation framework that operates with minimal human supervision is a main objective.

Color clustering based methods are computationally simple. Histogram analysis delivers satisfactory segmentation result especially for multi-modal histograms, and makes it possible to adapt certain thresholds for different input video. On the other hand, these methods fail to establish spatial connectivity although they attempt using morphological tools, which impair boundaries and slow down the process, or fuzzy estimators to improve connectivity. Obtaining accurate object boundaries is important for the object-oriented coding standards. In terms of connectivity and boundary accuracy, region growing based method performs better.

Although motion give the most discriminative information in video segmentation, the estimation of an accurate dense motion field is extremely slow, hence not suitable for processing of large volumes of video and real-time data. Motion models or block-wise motion vectors may be used instead of dense motion fields. Whereas, a chicken-egg problem exists in modeling motion; should the region of motion field that a model will be fitted be determined first, or should motion field be used to obtain the region of motion? Stochastic methods are capable to overcome the above problem by simultaneously modeling flow field and spatial connectivity, but they require the number of objects supplied as a priori information before the segmentation. Small and non-rigid motion gives rise to additional model fitting difficulties. Briefly, computational complexity, priority, and modeling issues are to be considered in utilizing motion for segmentation.

Existing tracking techniques do not employ a feedback of segmented regions which exploits segmentation results of a frame to improve the accuracy of the already segmented previous frames. The propagation of segmentation results is usually viewed as a tracking problem. Yet, for most of the cases, more than two video frames are

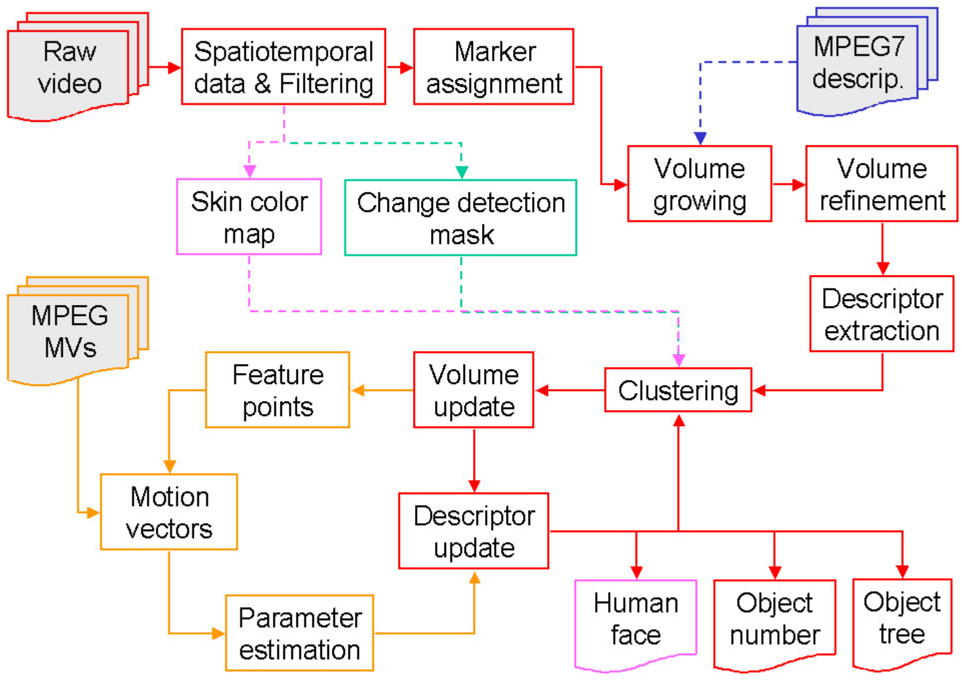


Figure 4.1: Flow diagram of the video segmentation algorithm showing all the major modular stages.

already available before segmentation. It will improve the segmentation performance if we can propagate object information forward and backward in time without saddling into the initial segmentation limitations.

Another constraint arises from the diversity of prospective segmentation applications that need different processing tools. A general purpose object segmentation system is expected to be made up by compatible processing modules that can be easily modified with respect to the application parameters. Even user assistance and any priori information should be easily embedded into the segmentation framework without reconstructing the system architecture.

In summary, we designed our segmentation framework to meet with the following targets

- Automaticity,
- Adaptability,
- Accuracy,
- Computational complexity,
- Information propagation.

A general flow diagram of the video object segmentation framework is given in Fig. 4.1. In the diagram, main algorithm and extension of the main algorithm are shown in different colors.

Before segmentation, the input video sequence is sliced into video shots that are defined as groups of consecutive frames having similar attributes between two scene-cuts. Emerging multimedia description standards define key frames representing each scene-cut; with MPEG-7 encoded sequences, the scene-cut information is already available. For raw data, key frames are extracted by using color histograms of the frames. Our algorithm takes certain number of consecutive frames within the same video shot, which we called as a video chunk, and process them at the same time. The length of the video chunk could be the length of the corresponding shot, or a size that exhibits discriminatory object motion within. Two limiting factors are the memory requirement due to large data size, and latency. For video chunks that contain 10$\sim$50 frames, both of these constraints can be satisfied using a decent system configuration.

We apply one of the filters explained in the previous chapter to the input video data. A simplification filter or a median filter is used if strong noise is present, otherwise a suppression filter is employed. The salt-and-pepper type of noise can be determined by evaluating the spatial frequency statistics, particularly at the high end of the frequency spectrum. If we assume that the noise is white then its spatial distribution over the image will be random. Measure of randomness can be tested using a relative variance score. It is calculated by first computing the mean and variance of the edge pixels in small image windows, from which a median variance score can be found. Although the test is sensitive to the window size and edge pixel density it works adequately as long as the number of the windows is sufficiently large. This evaluation is done for a sample image from the video data. Basic statistics are also extracted in the filtering stage to determine video adaptive parameters.

A spatiotemporal data structure is formed by indexing the point-wise features of frames to access video data effectively. These features include color values, edge magnitude, change detection, texture, etc. After filtering and building the spatiotemporal data structure, markers are selected. We acquired the smallest homogeneous parts of the spatiotemporal data by growing a volume around marker points, that are also called as seed in some other references. Markers are used for enlarging volumes by using color and texture based similarity criteria. The grown volumes are refined to remove small and shell-like volumes. Then, motion trajectories are determined. These trajectories serve as the estimations of translational motion. Without motion estimation, a functional approximation of motion is obtained. Self-descriptors for each volume, mutual-descriptors for a pair of volumes are computed from trajectories and also from other volume statistics. These descriptors are designed to capture motion, shape, color and other characteristics of the grown volumes. At this stage, we have the smallest homogeneous parts of a video sequence and their relations in terms of descriptors. Application specific constraints can be incorporated as separate descriptors.

In a following clustering stage, volumes are merged into objects by evaluating their descriptors. An iterative, hierarchical fine-to-coarse clustering is carried out until the motion similarity of merged objects becomes small. Alternative clustering techniques, i.e. adaptive k-means or fuzzy clustering, can also be used. After clustering, an object partition tree that gives the video object planes for every possible number of objects is generated. The object partition tree is appended to the input video for further recognition, data mining, event analysis purposes.

S	volumetric spatiotemporal data
x_M, y_M, t_M	dimensions of S
p	a point of S; $p = (x, y, t)$
m	a marker point and its vector
$w(p)$	a vector of S at point p
$w_y(p)$	color value (Y) at point p
$w_u(p)$	color value (U) at point p
$w_v(p)$	color value (V) at point p
$w_d(p)$	change detection mask at point p
$w_\theta(p)$	texture elements at point p
$w_e(p)$	edge magnitude at point p
$w_\phi(p)$	edge orientation at point p
$w_{tc}(p)$	target color mask at point p
$\nabla S(p)$	color gradient at point p
V_i	a volume within S
R_i^t	region of V_i at frame t
$\gamma(i)$	quantitative descriptor of volume V_i
$\Gamma(i, j)$	relational descriptor of volume pair V_i and V_j

Table 4.1: Notation of common parameter.

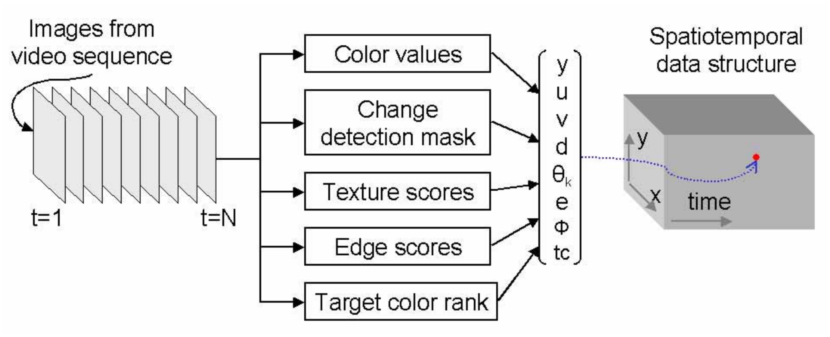


Figure 4.2: Construction of spatiotemporal data from the video.

The red colored boxes in Fig.4.1 represents the mentioned algorithm. In case MPEG-7 dominant color descriptors are available, as shown in blue, they are embedded in the volume growing stage. Change detection masks (green) are essential for specific applications such as surveillance that has stationary background. For human featuring video, skin color map (purple) is incorporated. An improved motion descriptor can be determined using available MPEG motion vectors in the clustering loop (orange). We will explain extensions in detail in the following sections. Table 4.1 summarizes the notation used in this chapter.

4.2 Formation of Spatiotemporal Data Structure

Frames of the input video are assembled into a spatiotemporal data structure S as illustrated in Fig. 4.2. Each element of this data structure $S(x, y, t)$ is a vector $w(x, y, t) = [y, u, v, d, \theta_k, e, \phi, tc]_{x,y,t}$ that consists of color values, change detection scores, texture and other statistics of the spatiotemporal point (x, y, t), which is also denoted as p. Here, (x, y) is the spatial coordinates and t is the frame number. The parameters y, u, and v stand for the luminance and chrominance values in the YUV color space respectively, d is the change detection mask, θ_k's are texture scores, e and ϕ are edge magnitude and orientation, and tc is the target color mask. We used a

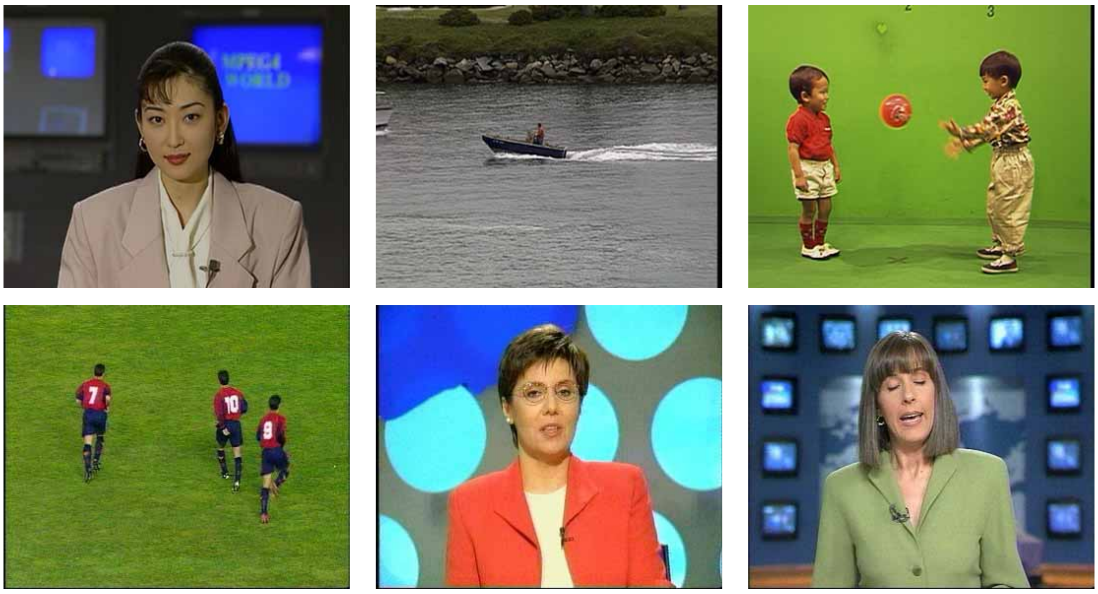


Figure 4.3: Original images from the input test sequences.

simplified notation for the components of the spatiotemporal data vector $w(x, y, t)$, e.g., $w_d(x, y, t)$ or $w_d(p)$ for the change detection mask value of the point p.

Color Elements

The first three elements of the vector $w(x, y, t)$ are the YUV color space values (w_y, w_u, w_v) of an image point (x, y) at frame t. We preferred the YUV color space basically since it has illuminance independent component, it performs in accordance with the human perception, and color distance can be computed using the magnitude or magnitude-square norms rather than complicated divisions and inversions. More discussion is given in the previous chapter. The metric used for computing color distances is a major factor of selecting a color space. A simple metric accelerates segmentation significantly because most of the processing time of a volume growing algorithm is spent while computing the color distances between the points. In Figures 4.3, 4.4, 4.5, the original frames from our test sequences and their Y, U, V channels

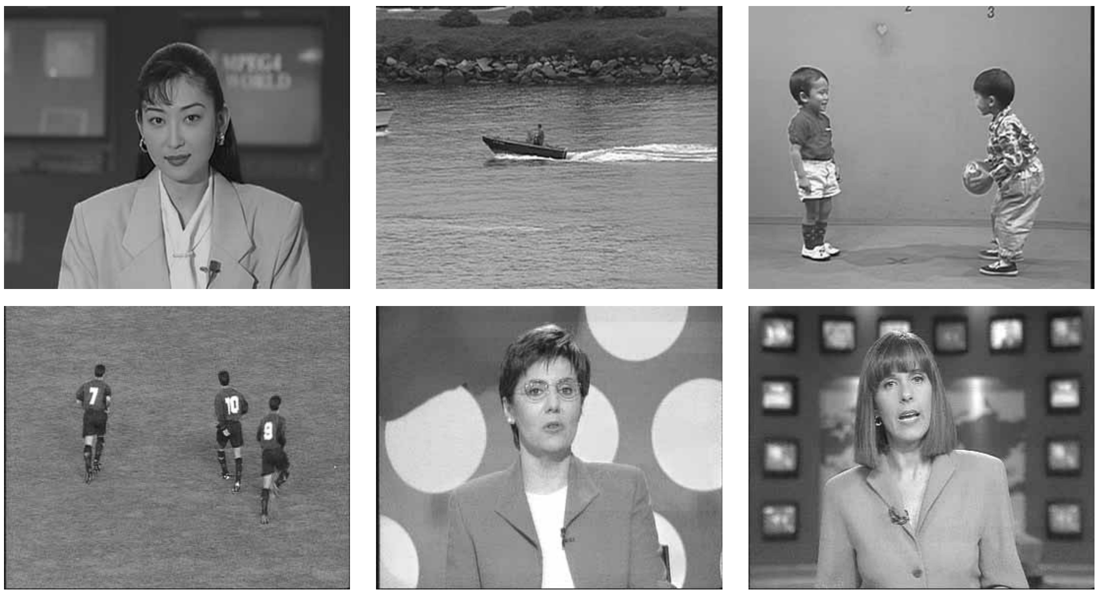


Figure 4.4: Y channel of the original images from the input test sequences.

are shown.

Texture Elements

The texture components $w_{\theta_1}, \ldots, w_{\theta_K}$ are computed by convolving the luminance channel w_y with the Gabor filter kernels as

$$w_{\theta,k}(x, y, t) = |h_k(x, y) \otimes w_y(x, y, t)|. \tag{4.1}$$

It is sufficient to employ the values for the spatial frequency $f = 2, 4, 8$ and the direction $\theta = 0, \pi/4, \pi/2, 3\pi/4$, which leads to a total of 12 texture features. The computation of the texture elements is computationally very expensive to be real-time for the existing state-of-art technology. Moreover, blending texture and color components into a similarity measure demands weighting parameters for each texture component. To accelerate segmentation, texture scores may be disregarded.

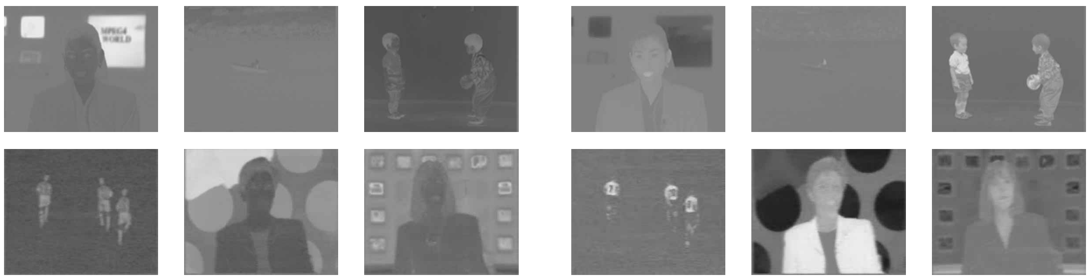


Figure 4.5: U and V channels of the original images. These channels are coded in lower bit rates in MPEG video.

Edge Elements

Although they are not utilized in the current implementation, edge attributes are proposed as a part of the spatiotemporal data for the future applications. Edge magnitude w_e and orientation w_ϕ are among the elements of spatiotemporal data vector w. These attributes are used for controlling linkage methods of volume growing, and rendering local image statistics for the following stages. As explained before, the Canny edge operator [23] is applied to compute the edge magnitude, and vector transform based method is utilized to determine the edge orientation for color data.

4.2.1 Filtering

Two main objectives of filtering are noise removal and simplification of color components to prevent volume growing algorithm from over-segmentation. The excessive number of small volumes not only slows down the algorithm, but also increases the array sizes and memory requirements. Small volumes require removal, and the unmarked points after removal need to be grouped within an existing volume which is a demanding work. Therefore, filtering improves the speed and stability of segmentation. However, most noise filtering techniques are also computationaly expensive especially considering the number of frames even a short video sequence may include.

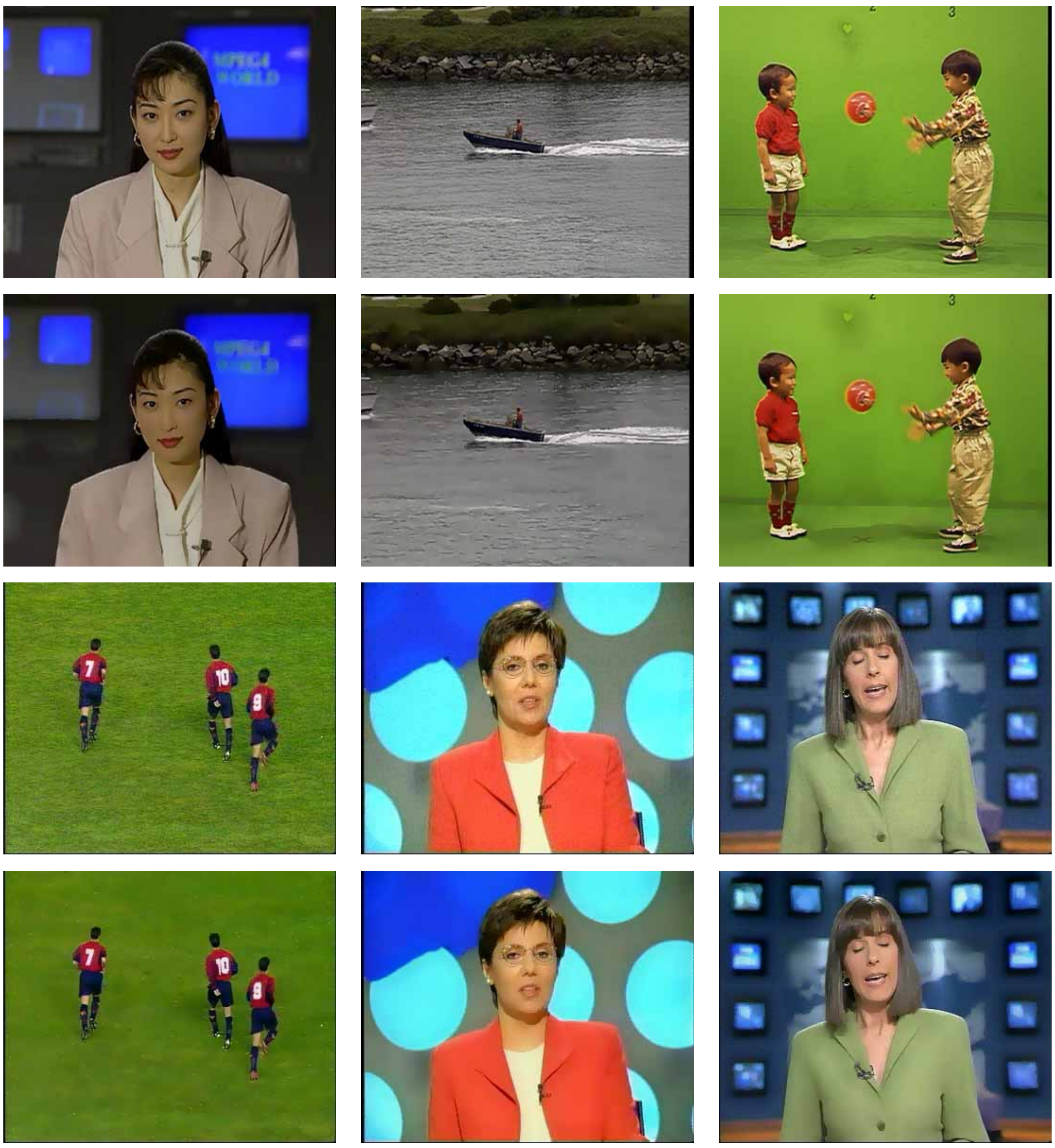


Figure 4.6: Originals ($1^{st}and3^{rd}$ rows) and filtered ($2^{nd}and4^{th}$ rows) images using the simplification filter.

The trade-off between the speed and quality is a user preference. We favored the simplification filter because it keeps the edge structure, smoothens texture within edges. It is also computationally feasible. The filtered input images are given in Fig. 4.6.

4.2.2 Color Quantization & MPEG-7

An alternative simplification method is color quantization. From the video sequence the most dominant colors up to a certain number are estimated using the Generalized Lloyd Algorithm (GLA) [102]. The dominant colors are determined by successive divisions of color clusters with the GLA algorithm in between, and then merging of the color clusters.

First, all color vectors are assumed to be in the same cluster, i.e., the number of clusters is 1. These color vectors are made up from the YUV color components of image points. The GLA measures the distances of color vectors to the cluster centers. For each cluster, a color cluster center is computed by means of averaging. The color vectors are grouped to the cluster center that has the smallest distance. The cluster centers are updated accordingly. After grouping the color vectors into clusters, a distortion score is computed. The distortion score is the sum of the distances of the color vectors to the cluster centers. The grouping is repeated until the distortion difference becomes negligible.

Then, each color cluster is divided into two new clusters by perturbing the color center if the number of total clusters are less than a maximum, which is 2^n. The GLA is repeated as described above starting by the new cluster centers. We observed that using 16 to 32 clusters gives acceptable segmentation performance. At these quantization levels, under-segmentation due to low color precision is prevented. As a final stage, the clusters that have close color centers are grouped to decide a final number of dominant colors. To increase the speed of dominant color extraction stage, only a single, subsampled input image in a shot with a high spatial sampling rate is used.

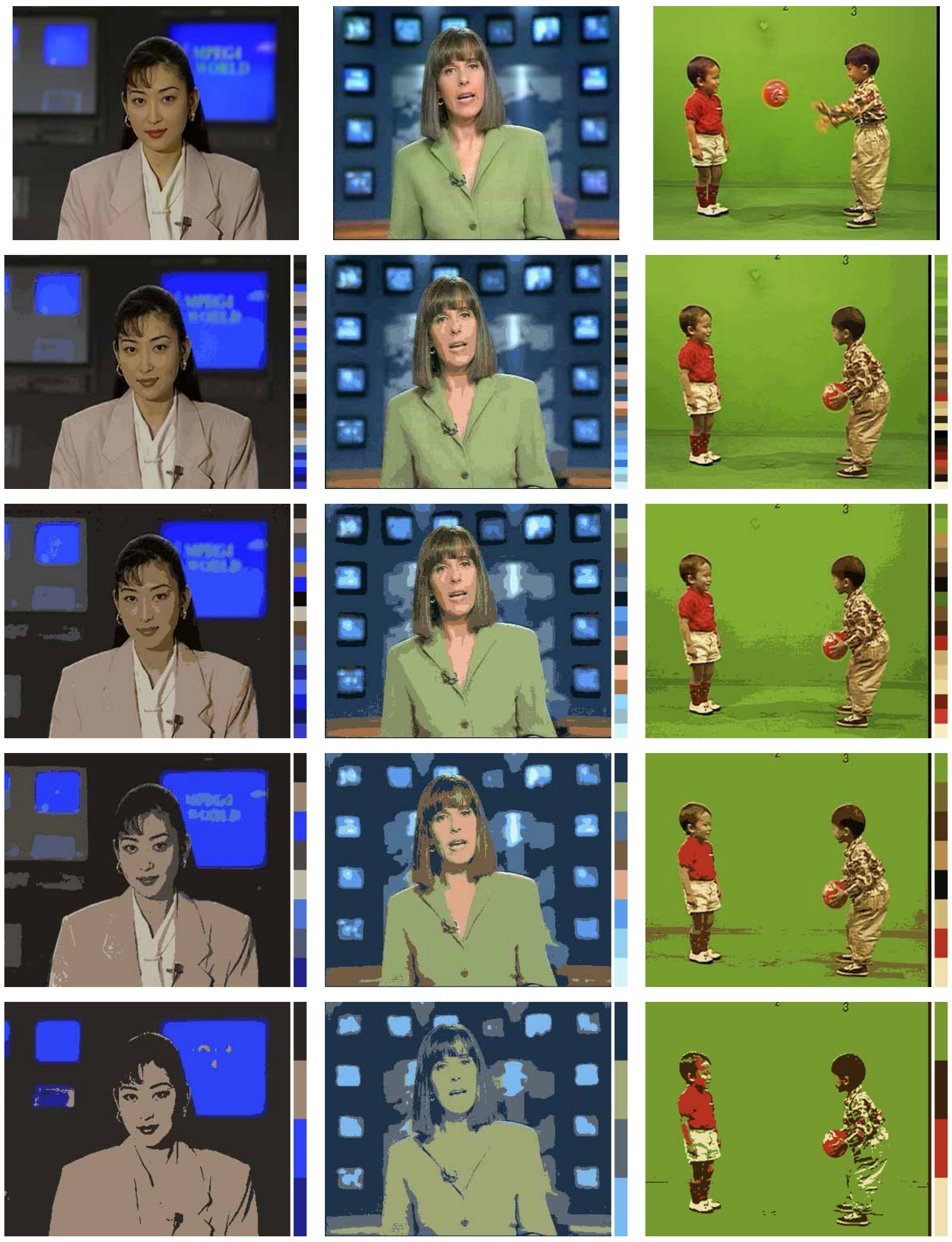


Figure 4.7: The first row is the original images, and the following rows are the quantized images using the dominant colors. The number of dominant colors is set to 32, 16, 8, 4 respectively. The dominant colors are shown next to each image.

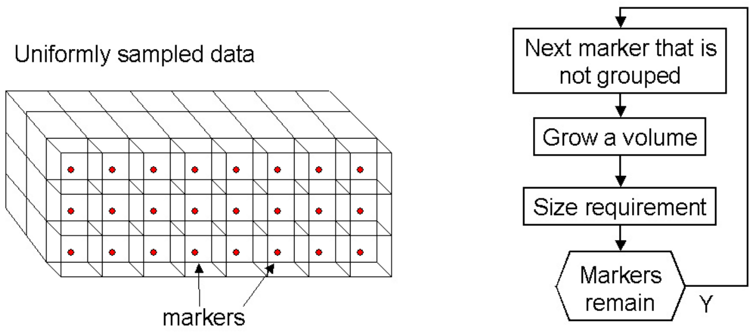


Figure 4.8: Using uniformly distributed markers with one-at-a-time volume growing.

The dominant color descriptor is a part of MPEG-7 with minor differences such as MPEG-7 has a smaller number of color bins, and it is based on *Lab* color space. In case MPEG-7 descriptors available with the input video, the dominant color descriptor is directly used to quantize the input video. In Fig. 4.7, the quantized images are given.

4.3 Marker Assignment

A marker is the seed of a volume around it. After initial processing, the smallest components of the spatiotemporal data structure are expanded from markers. We will denote markers as m_i, and call these small components as volumes V_i's. For each marker m_i, a volume and an index i is assigned. A marker is selected such that it represent its local neighborhood as relevant as possible. Points that have small color gradient magnitude are good candidates to represent their local neighborhood.

4.3.1 Uniformly Distributed Markers

The easiest way to assign markers is to distribute them uniformly in the spatiotemporal data S disregarding the color distribution. The centers of identical polyhedrons, i.e. cubes or prisms, are set as marker points. These polyhedrons constitute the spatiotemporal data S all together. Uniformly distributed marker assignment does not require any computation. At first, all points are marked as available. The first available marker is selected along the scanning direction, i.e., first frame to last, top to bottom, left to right. A volume is initiated and expanded according to color distance criteria. All the points included inside the grown volume is assigned as unavailable. Next marker is the first available marker in the rest of the S as demonstrated in Fig. 4.8. The simplest interpretation of uniformly distributed markers is all the points of the S are marker points. A more intricate marker assignment method first divides the spatiotemporal data uniformly into polyhedrons, then finds the point that has the minimum color gradient inside a polyhedron. These minimum points are the only available points in this case.

4.3.2 Minimum Gradient Points as Markers

Markers are selected among the low color gradient points. Let Q be the set of all available points, i.e., it is all the points of S initially. The color gradient magnitude $|\nabla S(x,y,t)|$ is computed for all spatiotemporal points, and the points having local minimum gradient magnitude are chosen as markers m_i. The color gradient magnitude is defined as

$$\begin{aligned} |\nabla S(x,y,t)| &= \sum_{k=y,u,v} [|w_k(x^-,y,t) - w_k(x^+,y,t)| \\ &\quad +|w_k(x,y^-,t) - w_k(x,y^+,t)| \\ &\quad +|w_k(x,y,t^-) - w_k(x,y,t^+)|] \end{aligned} \tag{4.2}$$

where $()^-$ and $()^+$ represent equal distances from the center point, i.e., $x-1, x+1$, etc. A volume V_i is grown as will be explained in the following section, and all the points of the volume is removed from the set Q

$$m_i = \arg\min_{Q} |\nabla S(x,y,t)| \quad ; \quad Q = S - \bigcup_{j=1}^{i} V_j. \tag{4.3}$$

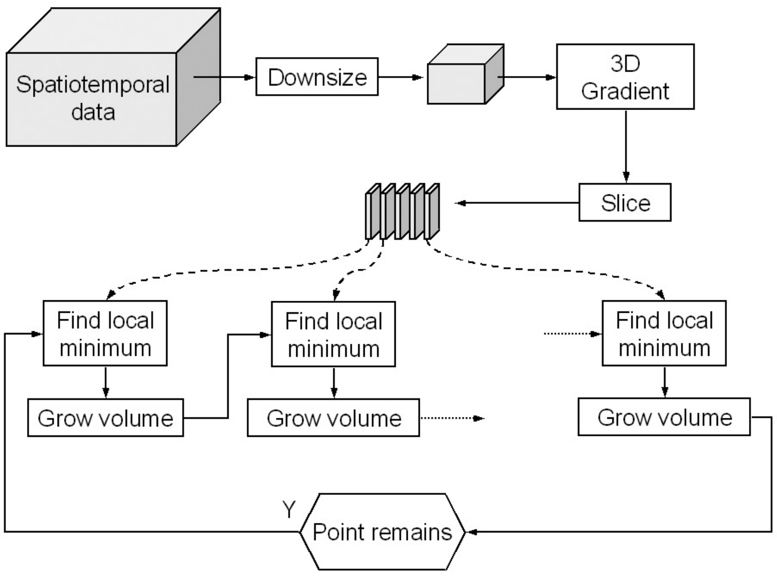


Figure 4.9: A fast marker selection method that finds minimum gradient magnitude points in the downsampled and sliced data using one-at-a-time volume growing.

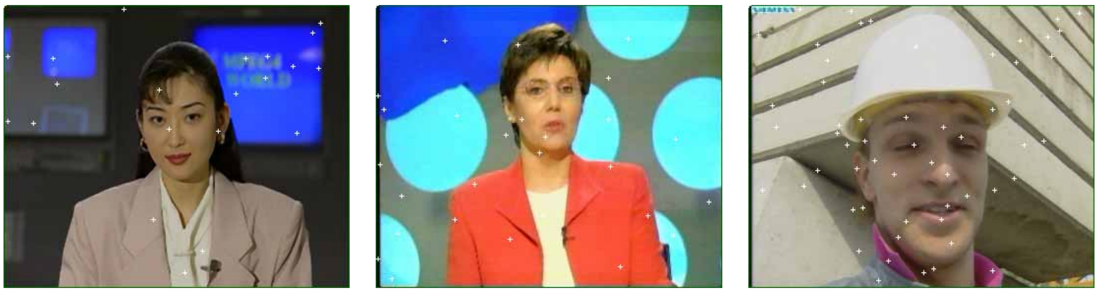


Figure 4.10: The markers corresponding to the volumes that are bigger than 0.005% of the total size of the S.

The next minimum in the remaining set is chosen, and selection process repeated until no more available point remains in the S. Finding the minimum is a computationally expensive process. Rather than searching the full-resolution spatiotemporal data S, a subsampled version is used. More computational reduction is achieved by dividing subsampled S into slices. The minimum is found for the first slice, and a volume is grown, then the next minimum is searched in the next slice as illustrated in Fig. 4.9. The extracted markers for the input video is presented in the Fig. 4.10.

4.4 Volume Growing

As the union of regions assembles 2D objects in an image, the union of volumes constructs objects in the spatiotemporal data. In other words, a volume V_i is a collection of the same object's projections, R_i^t, onto the frames of the video sequence. Volumes are the smallest components of the spatiotemporal data S and they compromise homogeneous color and texture distributed within. Using markers and evaluating various distance criteria, volumes are grown iteratively by grouping the neighboring points of similar characteristics. By volume growing, all the frames of an input video are segmented simultaneously. Moreover, no account of the quantitative information about the regions and region boundaries need to be kept to track regions. From a theoretical viewpoint, volume growing is a superior set of region growing methods.

There are several advantages of volume growing:

1. It solves the problem of tracking objects and correlating the segmented regions between the consecutive frames. It enables information propagation.

2. It handles effectively the appearing and disappearing regions without registration difficulties which require more computation and information gathering.

3. Particularly, it supplies an approximation of the translational motion of the regions without going into exhaustive motion computations. By simply growing volumes, important motion information is obtained.

4. Volume growing satisfies the spatial connectivity constraint of segmented points without further processing.

5. It enables segmenting multiple frames at once.

6. It is computationally simple.

Suppose that we start with a single point p and wish to expand from that seed point to fill a coherent volume. Let's define a distance measure $\Psi(p, q)$ such that it produces a low value if points p and q are similar and a high value otherwise. Now, consider a point p that is adjacent to another point q. We can include point q into point p's volume if distance $\Psi(p, q) < \epsilon$ for some threshold ϵ. We can then proceed to the other neighbors of p and do likewise. Suppose that $\Psi(p, q) < \epsilon$ and we added pixel q to pixel p's volume. We can now similarly consider the neighbors of q and add them likewise if they are similar enough. Of course, we now have unanswered questions to address:

- How do we define distance measure Ψ?

- What threshold ϵ do we use? Does it change or stay constant?
- How do we update volume attributes?

One obvious distance measure is to compare individual points intensities. In the following sections, we discuss the above questions.

4.4.1 Linkage Methods of Volume Growing

In principle, volume growing methods are applicable whenever a distance measure and linkage strategy can be defined. Several linkage methods were developed, they distinguished in the spatial relation of the points for which the distance measure will be computed:

- Single-linkage volume growing: A point is joined to its neighboring points whose properties are similar enough.
- Hybrid-linkage volume growing: Similarity among the points is established based on the properties within a small neighborhood of the point itself instead using the immediate neighbors only.
- Centroid-linkage volume growing: A point is joined to a volume by evaluating the distance between the centroid of the target volume and the current point.
- Counterexamples: Yet another approach is to provide not only a point that is in the desired volume but also counterexamples that are not in the volume. This method allows us to use not only the similarity to the volume but the dissimilarity to the exterior (the counterexample). It has the advantage of not requiring a predetermined threshold; the threshold is simply the value at which the candidate point becomes more similar to the outsider volume than to the target volume. It does, however, require some prior knowledge to “train” the system.

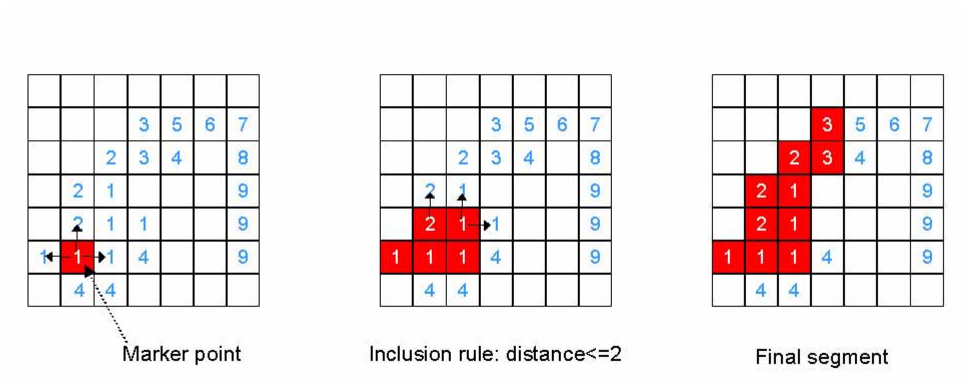


Figure 4.11: The simplest implementation of single linkage algorithm compares each candidate points with the initial marker.

4.4.2 Single-linkage Algorithm

When considering inclusion of a point p^- into the growing volume V, the static single-linkage technique compares back to the marker point m by using $\Psi(m, p^-)$. We now have the advantage of using a single basis for comparison across all points in the volume. However, it means that the volume produced is very sensitive to the choice of marker point.

One way of removing the above effect is to compare point p^- to the neighboring point p^+ already in the volume using a distance function $\Psi(p^-, p^+)$. A point is joined to its neighboring points whose properties are similar enough. In this way, each point that is already in the volume can bring in neighbors who are like it. One advantage of this method is that it produces transitive closures of similarity. If p is similar to q, and if q is similar to r, p and r end up in the same volume. Of course, this method can cause significant drift as one grows farther away from the original marker point. Infact, the original seed is of no significance once one grows out more than one point as illustrated in Fig.4.12. It is possible to normalize the similarity with the standard deviation of the color difference over a local neighborhood. Despite its

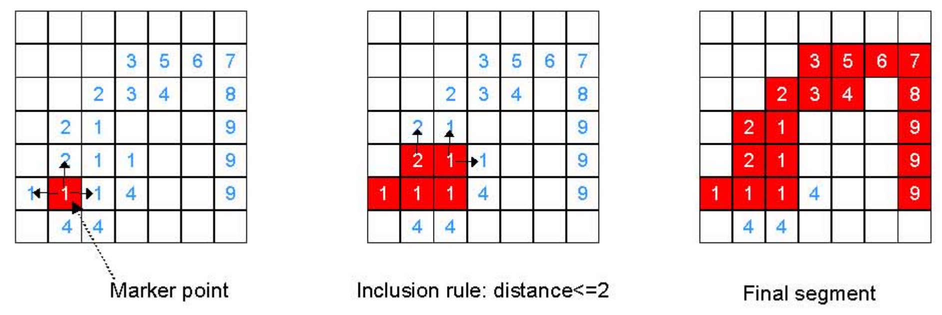


Figure 4.12: Transitive single linkage method may cause leakage.

simplicity, single-linkage method is very sensitive to the noise. Although a smoothing filter improve the noise sensitivity, it would lead under-segmentation which is a more severe problem.

The grown volumes using the single linkage algorithm are presented in Fig. 4.13. The output images are color coded for illustration purposes, each color corresponds to a different region thus a different volume.

4.4.3 Dual-linkage Algorithm

Instead using immediate neighbors only, similarity among the points is established based on the properties within a neighborhood of the point. Volumes are enlarged by evaluating dual distance metrics.

First, a representative vector, called as centroid, is assigned to the new volume that will be grown. This vector is composed of the color and other statistics of the volume, and initially it is equal to the marker's feature vector. Two distances, $\Psi_\bullet$ and $\Psi_\circ$, are measured for every candidate point. The $\Psi_\bullet$ gives the distance between the centroid and the candidate point. It serves as a "volume-wise" measure. The second distance $\Psi_\circ$ determines the difference between the candidate point and an

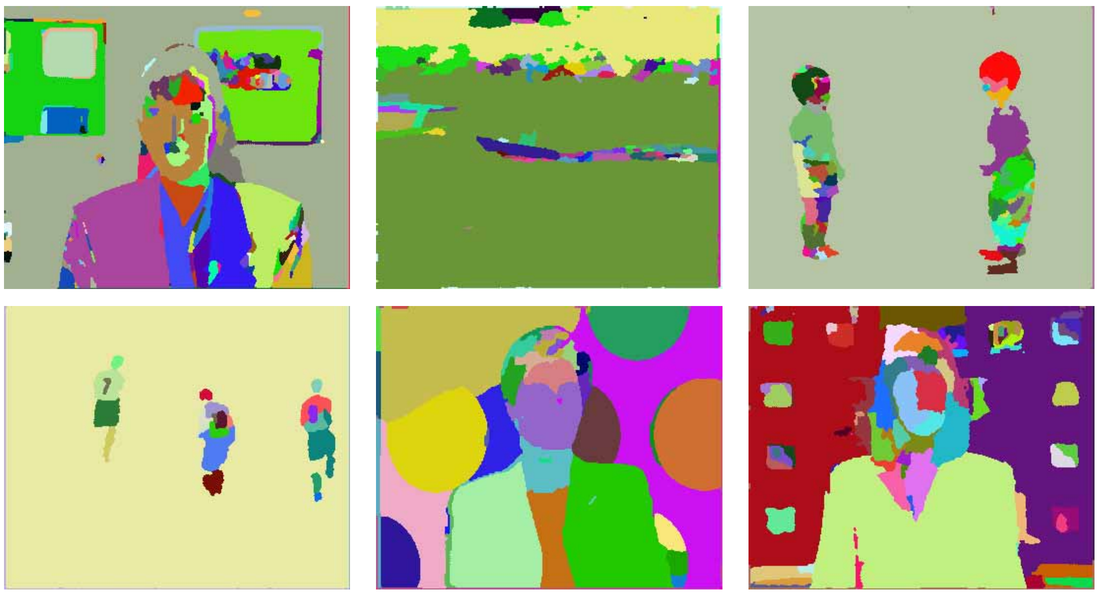


Figure 4.13: The segmented regions by single linkage volume growing method. Markers are selected from the minimum gradient points.

adjacent point to the candidate that is already included in the current volume. In case there are more than one such adjacent points exit, the closest one is chosen in the scanning direction. The second measure functions as a “point-wise” measure between the outer shell of the volume and the candidate points, which explains why we denoted the subscript as a shell.

Let p^- be an unmarked candidate point that is adjoint to the current volume V's boundary, which is called as active shell. Let p^+ be a point adjoint to p^- and in the active shell. Then, the volume-wise distance $\Psi_\bullet$ is defined as

$$\Psi_\bullet(m, p^-) \quad = \quad \sum_k |\omega_k(m) - w_k(p^-)| \qquad k : y, u, v \tag{4.4}$$

where $w(m)$ is the centroid feature vector, and $w(p^-)$ is the feature vector of p^-. Similarly, the second local distance $\Psi_\circ$ is defined as

$$\Psi_\circ(p^+, p^-) \quad = \quad \sum_k |w_k(p^+) - w_k(p^-)| \qquad k : y, u, v. \tag{4.5}$$

where $w(p^+)$ is the feature vector of the point p^+ that is already included in the volume and adjacent to p^-. Magnitude operator is chosen, yet the Euclidian distance between the color elements can be utilized. If the distances $\Psi_\bullet$ and $\Psi_\circ$ are smaller then $\epsilon_\bullet$ and $\epsilon_\circ$, the point p^- is included in the volume. The neighboring point p^- is set as an active shell point, and the feature vector for the marker is updated by averaging rule. In the next cycle, the neighboring pixels of the active shell are probed. Volume growing is repeated either until no point remains in the spatiotemporal data, or no more expansion occurs.

A similar approach to distance measure involves not only comparison to a single point or to a volume statistic, but by calculating cumulative differences as one follows a path from the marker to the candidate. In other words, if point p^+ is a neighbor of marker m, and candidate point p^- is a neighbor of p^-, instead of using $\Psi_\bullet(m, p^-)$ or $\Psi_\circ(p^+, p^-)$, we can use $\Psi_\circ(m, p^+) + \Psi_\circ(p^+, p^-)$. This is equivalent to finding the minimum-cost path from m to p^- and using this as the basis for the addition or rejection of point p^-. However, such an evaluation increases computational load.

The computed distances are compared with the thresholds. A volume-wise threshold $\epsilon_\bullet$ limits the variation of the features, i.e., color variation in the volume. The point-wise threshold $\epsilon_\circ$ prevents crossing over the edges. The global and local thresholds are adaptable to the input video using the color histograms. These histograms, $h_y(l)$, $h_u(l)$, and $h_v(l)$ are obtained in the filtering stage. The mean and variance in each histogram are computed by

$$\eta_k = \frac{1}{L}\sum_l h_k(l) \qquad \sigma_k^2 = \frac{1}{L}\sum_l (h_k(l) - \eta_k)^2, \tag{4.6}$$

where color value is in the range $0 \leq l < L$, and L is equal to 256 for an 8-bit coded channel. Since, the variance indicates modalities of color histogram, a high variance value represents multi-modal color distribution, likewise high spatial texture and multiple regions. Small values suggest a smooth color distribution and fewer

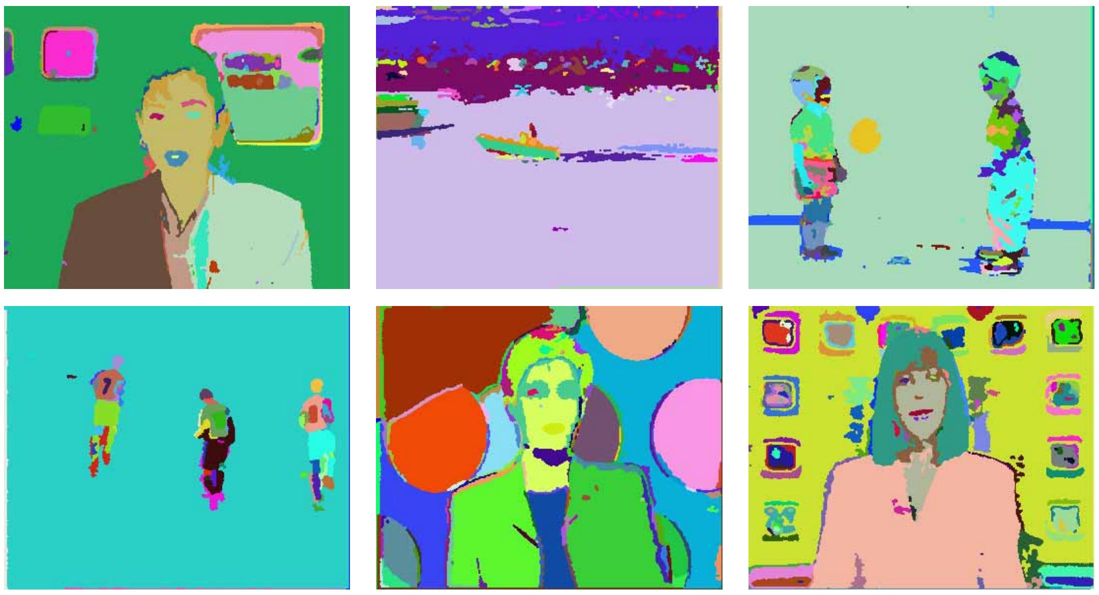


Figure 4.14: The regions obtained using dual-threshold volume growing.

regions. In general, with the increasing number of regions in an image, the color variance becomes larger. If the value of the variance is higher then there should be more regions, thus, a smaller threshold should be chosen. Thus, the variance and threshold are reciprocal. Dynamic color range μ is another parameter that has an effect on the thresholds. It is defined for a channel as

$$\mu = c_2 - c_1 \tag{4.7}$$

$$\int_0^{c_1} h_k(l)\, dl = \int_{c_2}^{1} h_k(l)\, dl = 0.05 \tag{4.8}$$

where c_1 and c_2 are constant numbers. As the dynamic color range changes, the distance between color clusters of the histogram varies assuming the number of clusters remains same. Therefore, the thresholds should be scaled with the dynamic range. There is a perceptive minimum for the thresholds since human vision is not very sensitive to small color variations. In conclusion, we designed the following formula for the volume-wise threshold

$$\epsilon_k = \max\left(\frac{\mu_k}{4\sigma_k^2}, 8\right) \quad \rightarrow \quad \epsilon_\bullet = \sum_k \epsilon_k \tag{4.9}$$

We aggregate three thresholds of color channels in the same summation term on account of using a single distance function $\Psi_\bullet$ as in Eq.4.4,

The point-wise threshold $\epsilon_\circ$ limits the inclusion of a point even it satisfies the volume-wise distance constraint. It is proportional to the average color discontinuity between the neighboring points. Specifically, the average edge magnitude determines the point-wise threshold

$$\epsilon_\circ \;=\; \kappa \frac{1}{M} \sum_k \sum_p w_k(p, e) \tag{4.10}$$

where M is the total number of points in a frame, k is the color channel, and $w_k(p, e)$ is the edge magnitude at point p in channel k. The scaling constant κ is set to 0.95 since not all the points correspond to an edge. The above statistics are derived from a single frame of the video.

For a marker point m, the growing is accomplished by evaluating the adjoint points distances $\Psi_\bullet(m, p^-)$ and $\Psi_\circ(p^+, p^-)$ in a three dimensional 6-points neighborhood. Those points are $(x+1, y, t)$, $(x-1, y, t)$, $(x, y+1, t)$, $(x, y-1, t)$, $(x, y, t+1)$, $(x, y, t-1)$ where the center point is (x, y, t). If the distances are smaller than their corresponding thresholds $\Psi_\bullet(m, p^-) < \epsilon_\bullet$ and $\Psi_\circ(p^+, p^-) < \epsilon_\circ$, the new point p^- is included in the volume V and set as an active shell point. The feature vector of the volume is updated by the new average. The volume growing is carried on until no more adjoint point satisfies the distance criteria. Then, a new marker is selected from the remaining available points, and growing is repeated. The segmented images using the dual-linkage method are presented in Fig. 4.14.

4.4.4 Centroid-linkage Algorithm

Another approach is comparing candidate point p^- to the volume features so called as "centroid". Initially, a volume consists of pixel m alone, so pixel m dominates

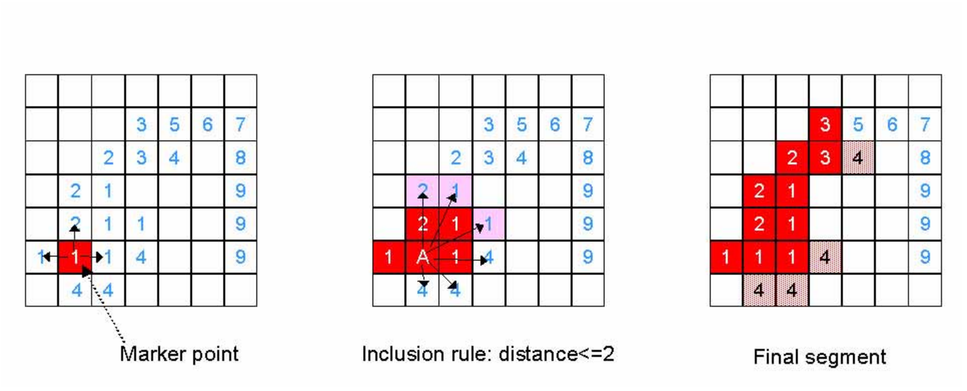


Figure 4.15: Centroid-linkage algorithm compares a candidate point with the centroid that is equal to marker initially. The pink points in the last figure are included at the final step.

the volume feature. As the volume grows, aggregated statistics are computed, and any new point p^- which may be added to the volume is compared not to point p^+ or but to these aggregated statistics. One simple such statistic is to keep an updated mean of the volume points. As each new point is added, the mean is updated. Although gradual drift is still possible, the weight of all previous points in the volume act as a damper on such drift as illustrated in Fig.4.15. A similar technique is to initialize the volume with not only a single point but a small set of points to better describe the volumes statistics. With such initialization, not only a volume mean is suggested but the variance as well. Candidate points can be compared to the volume mean with respect to the volume variance. The variance can be computed by sampling a small area around the initial marker point.

We implemented centroid-linkage volume growing approach by one-at-a-time marker selection. After a marker point m is selected, a volume centroid mean vector is initialized using markers features. Then in a 6-points neighborhood, the adjoint

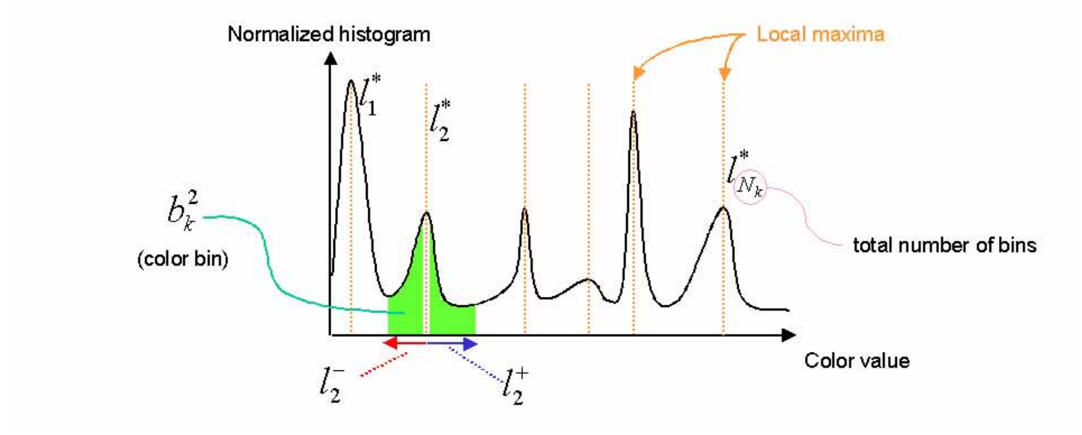


Figure 4.16: The histograms are used to determine color modalities of video to decide thresholds.

points p^- are evaluated. Color distances $\Psi(m,p^-)$'s are computed as in Eq.4.4,

$$\Psi(m,p^-) \;=\; \sum_k |\omega_k(m) - w_k(p^-)| \qquad k : y,u,v. \tag{4.11}$$

If the color distance $\Psi(m,p^-)$ is less than a threshold ϵ, the point p^- is included in the volume, assigned as an active shell point, and the centroid vector is updated accordingly. In the color distance equation, a magnitude norm is preferred rather than the difference square L^2 norm. Our experiments showed that using L^2 norm does not improve the segmentation performance significantly, although it takes more processing time.

As an alternative to the previous distance formulation, we also developed a logarithmic scaling based distance measure. The dominant colors are used if they are available. In case quantization of the color spectrum is not feasible due to very low color dynamic range of the input, we estimate the number of possible color bins for each color channel. This is accomplished by using color histograms as mentioned before.

The color histograms are smoothed within 5-tap filters, and the dynamic ranges

μ_k, $k = y, u, v$ are computed. The local maxima of the histograms are obtained for each channel. We denote the local maxima as $h_k(l_i^*)$. Here, $h_k()$ is the color histogram of the channel k, and l^* is the i^{th} local maximum. Note that there are multiple local maxima for a channel such that $l_i < l_{i+1}$ and $i = 1..N_k$ for color channel k where N_k is the number of maxima in this channel. Each of these local maxima is assigned to a color bin b_k^i. For a bin b^i, two numbers l_i^- and l_i^+ that correspond to the distances to adjoint bins $i+1$ and $i-1$ are computed

$$l_i^- = \frac{1}{2}(l_i - l_{i-1}), \qquad l_i^+ = \frac{1}{2}(l_{i+1} - l_i) \tag{4.12}$$

After a marker m is chosen, the centroid for the current volume is assigned using the markers feature vector. Using marker point, three lengths l_y, l_u, l_v are selected

$$l_k = \begin{cases} l_i^- & l_i^* - l_i^- < w_k(m) \leq l_i^* \\ l_i^+ & l_i^* < w_k(m) \leq l_i^* + l_i^+ \end{cases} \tag{4.13}$$

We devised the following Lorentzian-based measure as the distance between the centroid and the candidate point p^-

$$\Psi(m, p^-) \;=\; \sum_k N_k \log(1 + \frac{|w_k(m) - w_k(p^-)|}{l_k}) \;; \quad k = y, u, v. \tag{4.14}$$

We scaled the channel differences $|w_k(m) - w_k(p^-)|$ with the corresponding lengths l_k's for normalization. The addition term keeps the logarithmic distance measure positive. The Lorentzian term is sensitive enough towards the small color differences while it prevents the computed distance from inflating for a slightly large color difference in a single channel although the color differences in the other channels are very small. Considering a channel that has more distinctive dominant colors provides more information for segmentation, the channel distances are weighted by the corresponding N_k's. In the implementation, the divider lengths l_k's are integrated with the weight terms for computational simplicity. Then, the distance threshold is set as

$$\epsilon = (N_r + N_g + N_b) \tag{4.15}$$

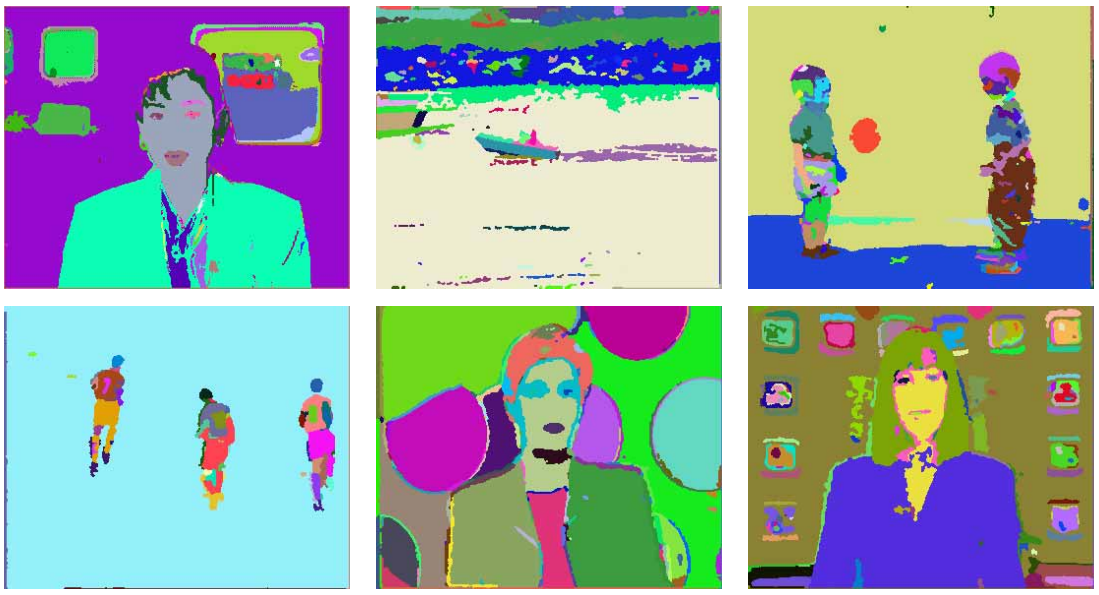


Figure 4.17: The segmented regions using the centroid-linkage method.

which means that the $w_y(p^-)$, $w_u(p^-)$, $w_v(p^-)$ are all within the same color bins with the centroid. Thresholds are calculated for each seed pixels after they are selected. The results for the centroid-linkage algorithm are shown in Fig. 4.17.

4.4.5 Modes of Volume Growing

Volume growing can be carried out either by growing multiple volumes simultaneously, or expanding only one single volume at a time. Furthermore, the expansion itself can be done either in an intraframe - interframe switching fashion, or a recursive outward growing style.

- Simultaneous growing: After all marker points are determined, volumes are grown simultaneously from each marker. At a growing cycle, all the existing volumes are updated by examining the neighboring points to the active shell of the current volume. In case a volume is stopped growing, an additional marker is selected. This marker is an adjoint point to the boundary of the stopped

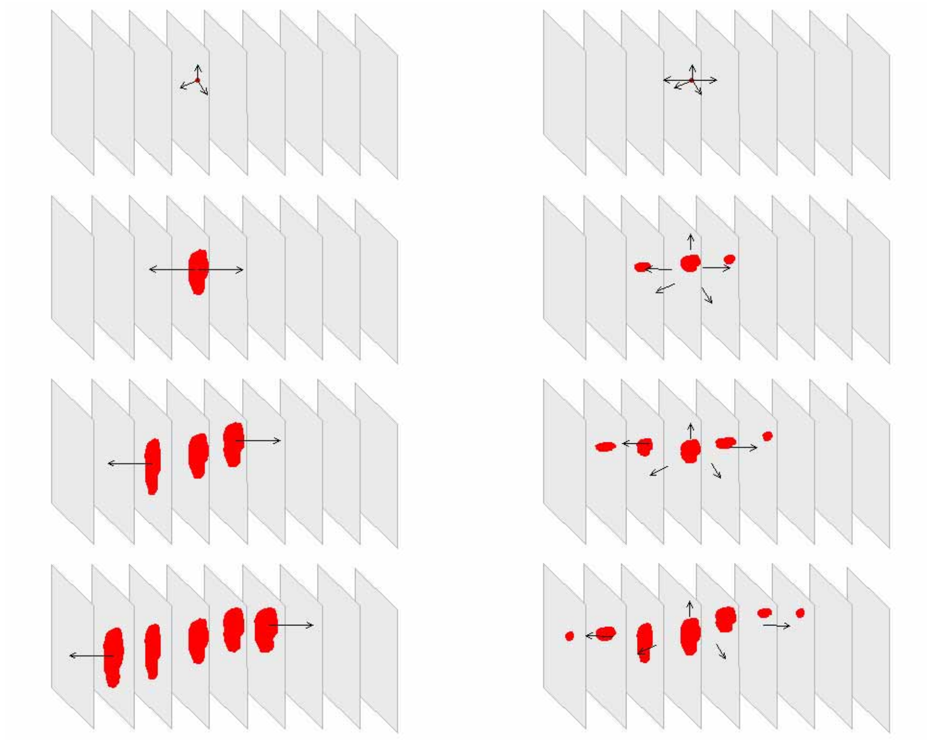


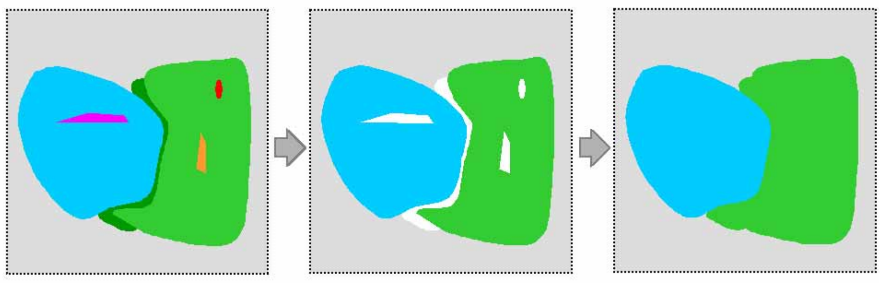
Figure 4.18: On the left: volume growing by intraframe-interframe switching, on the right: recursive diffusion. As visible, recursive diffusion grows volumes as an inflating balloon, whereas switching method first enlarges a region in a frame than spreads this region to the adjoint frames.

volume. A volume is initialized for the new marker point. Simultaneous growing is done until no point remains in the S. Although simultaneous growing is very fast and straightforward, it divides homogeneous volumes into multiple smaller volumes, thus a volume merging stage is necessary after volume growing.

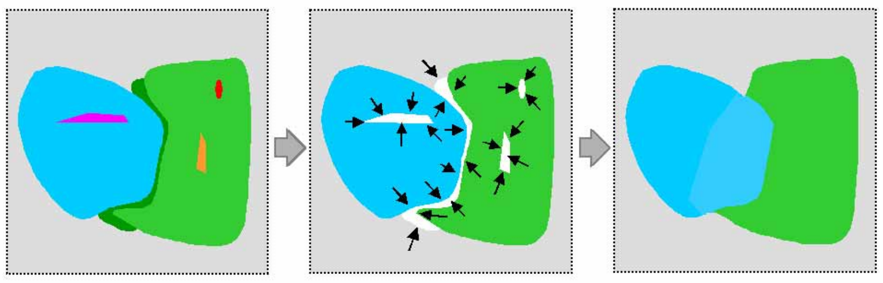
- One-at-a-time growing: At each cycle, only a single marker point is chosen, and a volume is grown around this marker. After the volume stopped growing, another marker in the remaining portion of the spatiotemporal data is selected. Selecting new markers continues until no more point remains in the S. An advantage of one-at-a-time growing is that it can be implemented by recursive programming. However, it demands more memory to keep all the pointers. It generates homogeneous volumes, thus do not need a merging stage.

- Recursive diffusion: Volume growing links the neighboring points to the volume's current boundary points, which constitutes an "active shell". For the spatiotemporal data, those points are defined as either 6 or 26 adjoint points (orthogonal neighbors and orhtogonal/diagonal neighbors). In recursive diffusion volume growing mode, the neighboring points to the active shell are evaluated disregarding whether they are in the same frame with the active shell point or not as illustrated in Fig. 4.18. After a point is included within a volume, the point becomes a point of the active shell as long as it has a neighbor that is not included in the same volume. By updating the active shell as described, the volume is "diffused" outward from the marker. Instead of using only adjoint points, other points within a local window around the active shell point can be used in diffusion as well. However, in this case the computational complexity increases, moreover, overall compactness of the volume shape may deteriorate.

- Intraframe-interframe switching: Volumes grown using recursive diffusion tends to be topologically non-compact by having several holes and ridges within. Such volumes usually generate unconnected regions when it is sliced frame-wise, i.e., a volume could have multiple separate regions in a frame taken from the spatiotemporal data. Intraframe-interframe switching proposes a better solution in that sense. The connected shapes are obtained by applying the diffusion mechanism first within the same frame to grow a region, then propagating it to the previous and next frames. The grown region is assigned as the active shell for the neighboring frames. As a result, each frame-wise projection of a volume will be a single connected region, and the volume will have a more compact shape overall.



(*a*)



(*b*)

Figure 4.19: (a) Small volumes are removed and the other volumes are inflated to fill empty areas. (b) Volume-wise similarity is evaluated to merge a small region into one of its neighbors.

4.4.6 Volume Refinement

After the growing stage, the spatiotemporal data S is divided into multiple volumes. Some of these volumes are negligible in size or very elongated due to the fine texture and edges. For much the same reason, they effect the computational load of the clustering algorithm. Moreover, some edge points and singularities may not be grouped into any of the volumes after the growing stage.

A simple way of clearing small and elongated volumes is marking their points as ungrouped (Fig.4.19-a). The ungrouped points that are attached to a valid volume boundary are transferred into an active shell. An active shell point is then included in the most color similar volume next to it. A relaxed color threshold can be utilized, or inclusion can be done without using any thresholds. The active shell is updated, and inclusion is iterated until no more ungrouped point remains.

Alternatively, a small volume can be merged into its neighboring most similar volume as a whole (Fig.4.19-b). In this case, similarity is defined as a combination of the mutual surface, color distance, mutual volume, and compactness ratio. Our arguments on the small volume similarity are summarized as

- The resultant volume should have a more compact shape rather than having more elongated to avoid leakage problems. In general, unifying a small volume into its neighbor that has the largest mutual boundary prevents from elongation. Compactness scores are computed to verify each possible merge satisfies the shape constraint.

- Image irregularities and relatively high motion often cause similar points to be grouped into separate small volumes. Unifying a small region with its neighbor that has similar color attributes to improve color consistency rather than unifying with a dissimilar one. Thus, color distance is incorporated in the evaluation.

- Around a small volume, the probability of that another small volume exists is relatively high. Along the edges, multiple tiny volumes may grow easily. If we unite a small volume with one of its neighbor that is also small, we would increase both of their sizes, and eliminate two small volumes at the same time.

The combination of the above properties is achieved by using an ordered list approach. First, all the small volumes are determined. This information is already available after the volume growing stage. The neighboring volumes V_j's of a small volume V_i are found, and mutual boundary ratios $\Gamma_{br}(i,j)$, color distances $\Gamma_{cd}(i,j)$, mutual sizes $\Gamma_{si}(i,j)$, and compactness ratios $\Gamma_{cr}(i,j)$ are computed as

$$\Gamma_{si}(i,j) \quad = \quad \gamma_{si}(i) + \gamma_{si}(j) \tag{4.16}$$

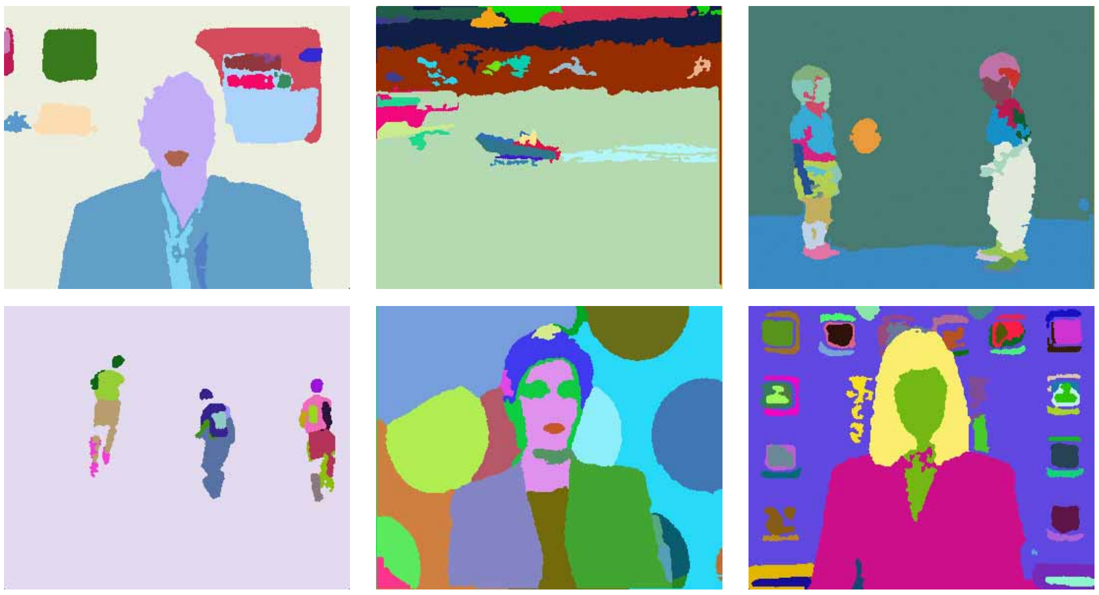


Figure 4.20: The regions obtained using centroid-linkage volume growing after volume refinement.

$$\Gamma_{cd}(i,j) = \sum_{k\in(y,u,v)} |\gamma_{mk}(i) - \gamma_{mk}(j)| \tag{4.17}$$

$$\Gamma_{br}(i,j) = \frac{\gamma_{bo}(i) + \gamma_{bo}(j) - \gamma_{bo}(i\cap j)}{\gamma_{bo}(i)} \tag{4.18}$$

$$\Gamma_{cr}(i,j) = \frac{\gamma_{co}(i\cap j)}{\gamma_{co}(i) + \gamma_{co}(j)} \tag{4.19}$$

where γ_{si} is the size of the volume, γ_{bo} is the boundary, γ_{mk} is the color mean corresponding to channel k, and γ_{co} is the compactness score. More explanation about the above parameters can be found in the following descriptors section. These attributes are ordered in separate lists. The lists are ordered as

$$\Gamma_{si}(i,j) > \Gamma_{si}(i',j') \implies r_{si}(i,j) > r_{si}(i',j') \tag{4.20}$$

$$\Gamma_{cd}(i,j) > \Gamma_{cd}(i',j') \implies r_{cd}(i,j) < r_{cd}(i',j') \tag{4.21}$$

$$\Gamma_{br}(i,j) > \Gamma_{br}(i',j') \implies r_{br}(i,j) < r_{br}(i',j') \tag{4.22}$$

$$\Gamma_{cr}(i,j) > \Gamma_{cr}(i',j') \implies r_{cr}(i,j) > r_{cr}(i',j') \tag{4.23}$$

(4.24)

where r is the ranking function, i.e. the rank of the volume combination V_i, V_j in the corresponding attribute list. From these lists, a score $\tau(i, j)$ is computed for possible volume merges with respect to their ranks. The score $\tau(i, j)$ is the summation of the ranks corresponding to the pair (i, j) such that

$$\tau(i,j) = r_{si}(i,j) + r_{cd}(i,j) + r_{br}(i,j) + r_{cr}(i,j). \tag{4.25}$$

The neighboring volume V^* that gives the maximum score is chosen, i.e.,

$$V_i^* = \arg\max_{V_j} \tau(i,j) \tag{4.26}$$

the small volume V_i is merged into V_i^*, and the attributes are updated. Although this technique has more formulation, it is faster than the point-wise expansion, since it uses a few statistics instead of dealing with thousands of points. Yet, it requires accurate initial volumes. We determined that a minimum size 0.5% of the total size of S is adequate as a minimum volume size threshold for most applications. Fig. 4.20 shows the volumes after the refining stage.

4.5 Analysis of Volumes

Volume growing provides color homogeneous parts of the video. On the other hand, most video objects, i.e. cars, contain multiple parts that may have different color statistics although those parts have consistent motion. Thus, our next goal is to extract motion information of volumes to construct motion consistent objects.

We employ descriptors to capture various aspects of the volumes. These descriptors characterize motion, shape, and color characteristics, as well as they evaluate mutual relations of the volumes. The volumes are grouped with respect to the descriptors at the clustering stage in order to assemble the objects.

The first motion related attribute of a volume is its trajectory.

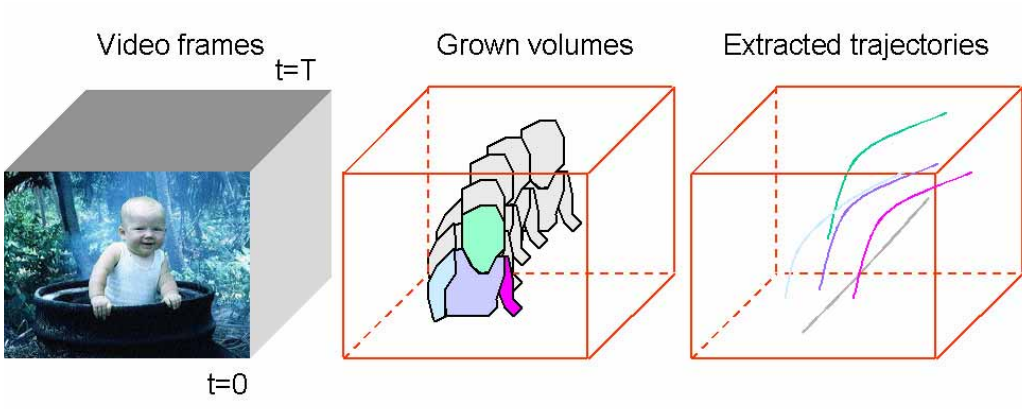


Figure 4.21: A trajectory is the connected frame-wise center of masses of a volume.

4.5.1 Extraction of Trajectories

Motion trajectory is a high level feature associated to a moving region, defined as the localization of one of its representative points such as its centroid as simulated in Fig. 4.21. The centroid can be chosen as the center of mass of a volume's frame-wise projection. Another centroid definition is the intersection of the longest line within the region and another line that is longest in the perpendicular direction. We used the center of mass as the centroid since that can be extracted as a volume being grown, besides, the second definition is hard to compute. For each volume V_i, a trajectory $T_i(t) = [X_i(t), Y_i(t)]^T$ is extracted by computing the frame-wise averages of volume i's coordinates

$$T_i(t) = \begin{bmatrix} X_i(t) \\ Y_i(t) \end{bmatrix} = \begin{bmatrix} \frac{1}{R(i,t)} \sum x \\ \frac{1}{R(i,t)} \sum y \end{bmatrix} \quad ; \quad (x, y) \in R_i^t. \tag{4.27}$$

Above, R_i^t is the corresponding region to the volume V_i at frame t. The $R(i,t)$ is the area of the region R_i^t. The sum of the areas time-wise gives the volumes size

$$\gamma_{si}(i) = \sum_t R(i,t), \tag{4.28}$$

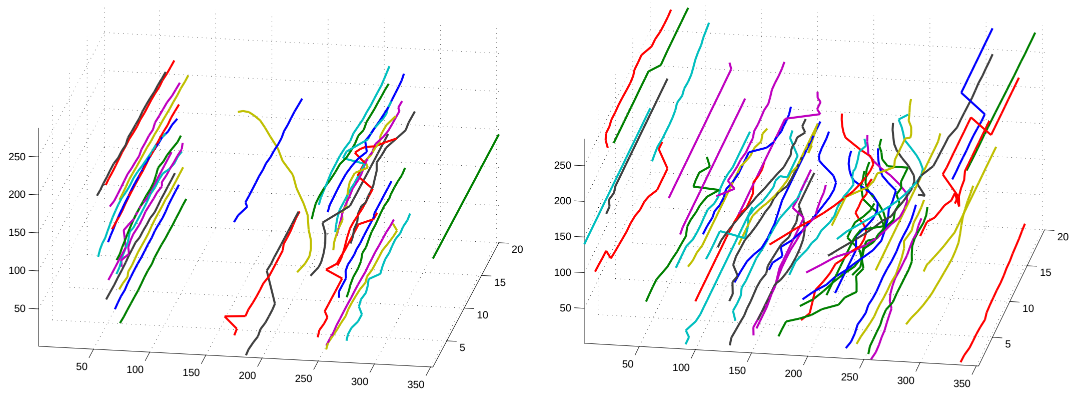


Figure 4.22: Sample trajectories of the *child* and *foreman* video for 20 frames.

where $\gamma_{si}(i)$ is the size of the volume i. Each region R_i^t has a boundary $B(i,t)$. Sample trajectories are shown in Fig. 4.22.

An important property of the motion trajectory is that it approximates the translational motion of the volume it belongs in most of the cases. This motion is the easiest to be perceived by the human visual system, for much the same reason it is the most discriminative in object recognition. In addition, it is the part of the parameterized motion that can be estimated more accurately. A motion analyzer performs more robust by utilizing translational motion information rather than using rotational motion only. Motion trajectory enables to comprehend the motion of a region between the frames without involving in dense motion vector computation. Such an information can be used to initialize motion parameters in motion estimation or parameterized model fitting processes to accelerate computation.

Motion Trajectories as Defined in MPEG-7

Motion trajectory is also described as a part of MPEG-7. However, MPEG standards do not propose any method to obtain such trajectories. The SpatioTemporalLocator describes spatiotemporal regions (Fig.4.23) in a video sequence and provides localization functionality especially for hypermedia applications. It consists of

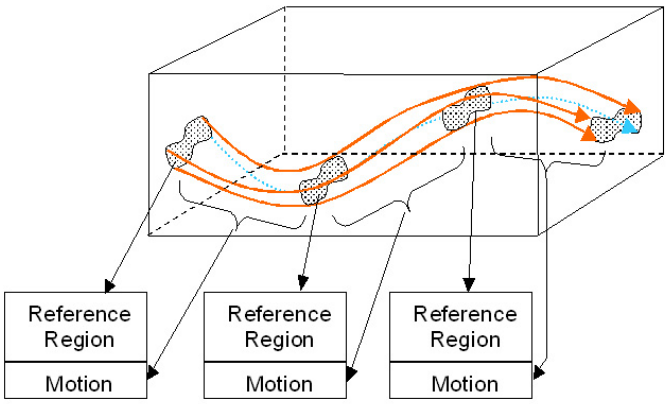


Figure 4.23: Spatiotemporal regions defined as in MPEG-7.

FigureTrajectory and ParameterTrajectory. Moving regions in multiple frames are described by one or several sets of reference region and motion. FigureTrajectory and ParameterTrajectory describe each set of a reference region and a motion. The two description schemes are selected according to moving object conditions. If a moving object region is rigid and the motion model is known, ParameterTrajectory is appropriate. For non-rigid moving object region, FigureTrajectory is appropriate. The usage depends on applications.

FigureTrajectory describes a spatiotemporal region by trajectories of the representative points of a reference region. Reference regions are represented by three kinds of figures: rectangles, ellipses and polygons. For rectangles and polygons, the representative points are their vertices. Although there are four vertices for a rectangle, only three of the four are described because the rest can be easily calculated. For ellipses, three vertices of their circumscribing rectangles are selected as the representative points. The trajectories are interpolated using the TemporalInterpolation descriptor. For this descriptor, the reference region description is omitted because the TemporalInterpolation descriptor can directly express the representative points of figures.

mean of color k	$\gamma_{my}(i)$	$\frac{1}{\gamma_{si}(i)}\sum w_k(p),\ \ p \in V_i$
size	$\gamma_{si}(i)$	$\sum_t R(i,t)$
boundary	$\gamma_{bo}(i)$	$\sum_t B(i,t)$
compactness	$\gamma_{co}(i)$	$\frac{1}{\gamma_{ex}(i)}\sum_t \frac{R(i,t)}{(B(i,t))^2}$
vertical translation	$\gamma_{vt}(i)$	$Y_i(1)-Y_i(t_T)$
horizontal translation	$\gamma_{ht}(i)$	$X_i(1)-X_i(t_T)$
trajectory length	$\gamma_{tl}(i)$	$\sum_t \lvert T_i(t)-T_i(t-1)\rvert$
existence	$\gamma_{ex}(i)$	$\sum n_t,\ \ n_t = 1 \leftarrow R(i,t) \neq 0$
change detection	$\gamma_c(i)$	$\frac{1}{\gamma_{si}(i)}\sum_p w_c(p),\ \ p \in V_i$
target color	$\gamma_{tc}(i)$	$\frac{1}{\gamma_{si}(i)}\sum_p w_{tc}(p),\ \ p \in V_i$
motion parameters	$\gamma_{pm}(i)$	$\{a_1, a_2, ..., a_6\}$

Table 4.2: Volume's quantitative descriptors

ParameterTrajectory describes a spatiotemporal region by a reference region and trajectories of motion parameters. Reference regions are described using the Region Locator Descriptors. Motion parameters and parametric motion model specify a mapping from the reference region to a region of an arbitrary frame. The trajectories of the motion parameters are interpolated and described using the TemporalInterpolation descriptor.

4.5.2 Quantitative Descriptors

Descriptors are the numerical scores that interpret various attributes of a volume. These attributes include motion, shape, color (and texture if it is utilized), change detection, and application specific features such as skin color, etc. We denoted these scores as $\gamma(i)$ and $\Gamma(i,j)$ for quantitative and relative descriptors respectively. We introduced and tested several descriptors.

Quantitative descriptors $\gamma(i)$'s evaluate a volumes self properties such as its size, surface, compactness, motion as summarized in the Table 4.2. Some of the quantitative descriptors are basic. The descriptor $\gamma_{si}(i)$ is the size of the volume V_i. The $\gamma_{my}(i)$, $\gamma_{mu}(i)$, and $\gamma_{mv}(i)$ are the averaged values of the corresponding color channels of a volume. The boundary $\gamma_{bo}(i)$ is the number of points on the surface of a volume. The descriptor $\gamma_{ex}(i)$ counts how many frames of the sequence the volume V_i appears. We define compactness $\gamma_{co}(i)$ as the ratio of its volume to its boundary square

$$\gamma_{co}(i) = \frac{1}{\gamma_{ex}(i)} \sum_t \frac{R(i,t)}{(B(i,t))^2} \tag{4.29}$$

In 3D, a sphere has the largest compactness score. On the other hand, in the spatiotemporal data the most compact shape is a cylinder along the time axis since we want the frame-wise projections of a volume to be compact but not its overall shape. To compensate for the radius, the boundary value is squared as above. The more a volume elongates in its frame-wise projections, the lower its compactness score becomes. The compactness score is sensitive to the boundary ripples, i.e., a rough surface.

Some motion descriptors of a volume are based on the volume trajectory. The descriptor γ_{tl} measures the length of the trajectory

$$\gamma_{tl}(i) = \frac{1}{\gamma_{ex}(i)} \sum_t |T_i(t) - T_i(t-1)|. \tag{4.30}$$

Usually, stationary objects have shorter trajectory lengths. Two other descriptors $\Gamma_{vt}(i)$ and $\Gamma_{ht}(i)$ compute the vertical and horizontal displacement between the first and the last frames that a volume exists between. One motion related descriptor set is the affine motion parameters of the volume instead of its trajectory

$$\gamma_{pm}(i) = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_3 \\ a_6 \end{bmatrix} \tag{4.31}$$

These parameters are computed for a volume at each frame. Other quantitative descriptors are the change detection score γ_{cd} and the skin color feature score γ_{sc}. These are defined in the following sections.

mean of T. distance	$\Gamma_{\mu}(i,j)$	$\frac{1}{\Gamma_{ex}(i,j)}\sum\Delta_t(i,j)$
variance of T. distance	$\Gamma_{\sigma}(i,j)$	$\frac{1}{\Gamma_{ex}(i,j)}\sum(\Delta_t(i,j)-\Gamma_{\mu}(i,j))^2$
maximum T. distance	$\Gamma_{xt}(i,j)$	$\max\Delta_t(i,j)$
directional difference	$\Gamma_{dd}(i,j)$	$\sum\lvert T_i(t)-T_i(t-1)-T_j(t)+T_j(t-1)\rvert$
parameterized distance	$\Gamma_{pm}(i,j)$	$< A_i - A_j, A_i - A_j >$
compactness ratio	$\Gamma_{cr}(i,j)$	$[\gamma_{co}(i\cap j)][\gamma_{co}(i)+\gamma_{co}(j)]^{-1}$
mutual boundary ratio	$\Gamma_{br}(i,j)$	$[\gamma_{bo}(i\cup j)][\gamma_{bo}(i)]^{-1}$
color difference	$\Gamma_{cd}(i,j)$	$\sum_k\lvert\gamma_{mk}(i)-\gamma_{mk}(j)\rvert$
coexistence	$\Gamma_{ex}(i,j)$	$\sum_t n_t,\ \ n_t=1\leftarrow(R(i,t)\neq0)\cap(R(j,t)\neq0)$
neighborhood	$\Gamma_{ne}(i,j)$	$1\leftarrow\gamma_{co}(i\cap j)>0$

Table 4.3: Relational descriptors of a volume pair V_i, V_j

4.5.3 Relational Descriptors

The relational descriptors evaluate "relative" similarity between a pair of volumes. As quantitative descriptors, they characterize motion, shape, and color aspects. The motion related relative descriptors utilize the mutual trajectory distance. This distance $\Delta_t(i,j)$ is calculated between the trajectories $T_i(t)$ and $T_j(t)$ for frame t by the equation

$$\Delta_t(i,j) = \lvert T_i(t)-T_j(t)\rvert. \tag{4.32}$$

The mutual trajectory distance is used to obtain the motion relative descriptors. The mean of the trajectory distance $\Gamma_{mt}(i,j)$ measures average distance between the volumes centroids

$$\Gamma_{\mu}(i,j) = \frac{1}{\Gamma_{ex}(i,j)}\sum\Delta_t(i,j) \tag{4.33}$$

in which the summation term is applied to frames in which both of the volumes exist. In the above equation, $\Gamma_{ex}(i,j)$ is the number of the frames that both volumes V_i and V_j exist. The variance of the trajectory distance $\Gamma_{\sigma}(i,j)$ indicates consistency between the motions of two volumes

$$\Gamma_{\sigma}(i,j) = \frac{1}{\Gamma_{ex}(i,j)}\sum[\Delta_t(i,j)-\Gamma_{\mu}(i,j)]^2. \tag{4.34}$$

A small variance score means two volumes have almost similar translational motion, and a big variance reveals volumes having different motion, i.e., getting away from each other, moving in the opposite directions, etc. The maximum distance $\Gamma_{xt}(i,j)$ quantifies how far two volumes gets away from each other. Some exceptions of the motion descriptors are a large background since its trajectory is located on the center of the frames mostly, and a highly concave volume that has a trajectory falls outside its boundary. To distinguish volumes that have small motion variances but opposite motion directions, e.g., two volumes turning around an axis, the directional difference $\Gamma_{dd}(i,j)$ is defined.

Motion of a volume can also be characterized by the model parameters $\gamma_{pm}(i)$. The parameters of two volumes are utilized to derive a relational motion similarity descriptor $\Gamma_{pm}(i,j)$. Motion parameters are arranged in a vector to calculate their difference

$$\Gamma_{pm}(i,j) = \sum_t |A_i^t - A_j^t| \tag{4.35}$$

where A^t is the vector of motion parameters of frame t

$$A^t = [a_1 \; a_2 \; a_3 \; a_4 \; a_5 \; a_6]_t. \tag{4.36}$$

The parameter distance is defined as

$$|A_i - A_j| = \left[c_R \sum_{n=1,2,4,5} (a_{i,n} - a_{j,n})^2 + c_T \sum_{n=3,6} (a_{i,n} - a_{j,n})^2 \right]^{1/2} \tag{4.37}$$

where the multipliers are selected as $c_R \gg c_T$ to weight the rotation/zoom difference proportional with the translation difference.

Shape information is captured as relational descriptors too. The compactness ratio $\Gamma_{cr}(i,j)$ of a pair of volumes is the amount of the change on the total compactness before and after the two volumes are merged

$$\Gamma_{cr}(i,j) = \frac{\gamma_{co}(i \cap j)}{\gamma_{co}(i) + \gamma_{co}(j)}. \tag{4.38}$$

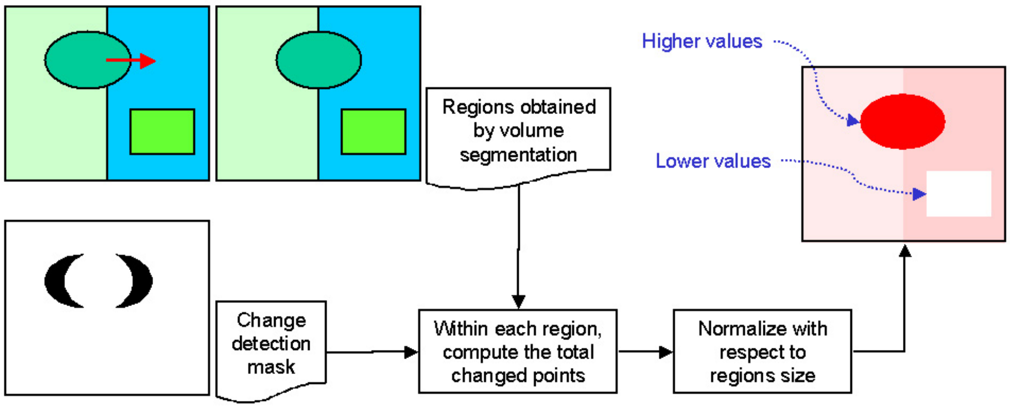

Regions obtained by volume segmentation
Higher values
Lower values
Change detection mask
Within each region, compute the total changed points
Normalize with respect to regions size

Figure 4.24: Utilizing $\gamma_c(i)$ empowers moving object detection.

A small $\Gamma_{cr}(i,j)$ means the merging generates a less compact object which is not good. By merging, we aim to obtain more compact objects since natural objects tends to be compact. Another shape related descriptor $\Gamma_{br}(i,j)$ calculates the ratio of mutual boundary of two volumes V_i and V_j to the total boundary of volume V_i

$$\Gamma_{br}(i,j) = \frac{\gamma_{bo}(i \cap j)}{\gamma_{bo}(i)}. \tag{4.39}$$

In clustering, a volume pair with high $\Gamma_{br}(i,j)$ is preferred. Except an enclosing background, the higher values of mutual boundary ratio is a sign of higher compatibility. The color difference descriptor Γ_{cd} is the sum of the difference between the color means. Other descriptors are Γ_{ex} which is the number of frames that both of the volumes exist, and $\Gamma_{ne}(i,j)$ the neighborhood descriptor that is equal to 1 if volumes have a mutual boundary. The relational descriptors are summarized in Table 4.3.

4.5.4 Change Detection Mask in Segmentation

The frame difference and change detection mask are among the widely used features in computer vision. Blending motion compensated change detection masks with additional video features improves the performance of object segmentation for

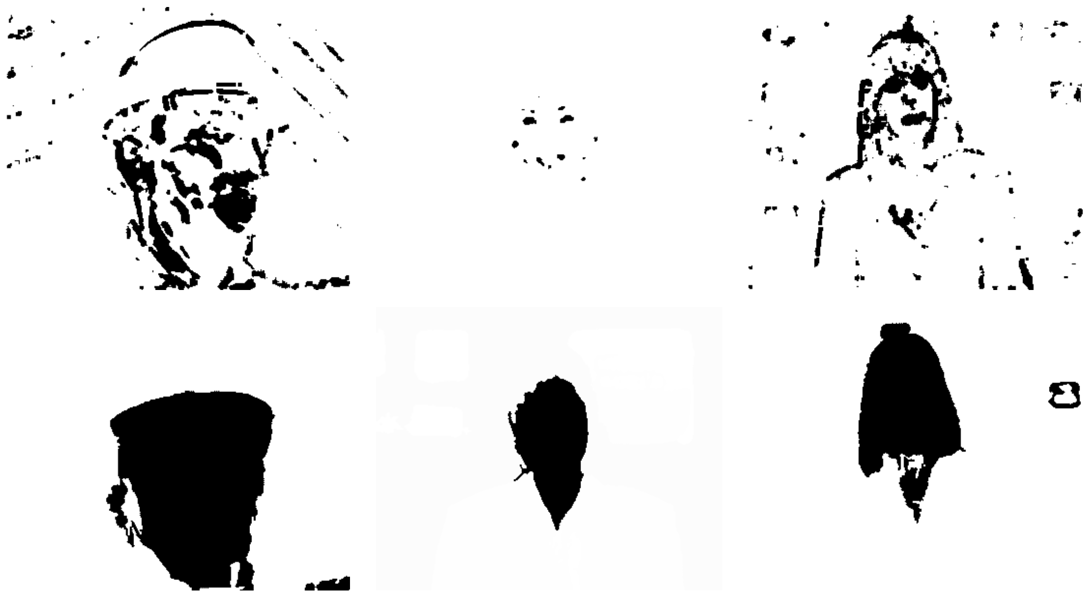


Figure 4.25: Upper row: the point-wise change detection masks $w_c(p)$ for frames from *foreman*, *akiyo*, and *newsroom* video, black means a changed point. Lower row: the thresholded descriptor.

certain applications. After the CDM scores $w_c(p)$ are found as discussed in the previous chapter and the volumes V_i's are obtained, the CDM based descriptor $\gamma_c(i)$ is computed as illustrated in Fig.4.24. For each volume, the number of the changed points is determined while compensating for trajectory motion of the volume

$$\gamma_c(i) = \frac{1}{\gamma_{si}(i)} \sum_t \sum_{x,y \in R_i^t} w_c\left(x - X_i(t), y - Y_i(t), t\right). \tag{4.40}$$

The change detection scores of the volumes, $\gamma_c(i)$, are uniformly normalized to $[0,1]$ range. Our tests show that volumes having $\gamma_c(i)$ values higher than $0.2 - 0.3$ often correspond moving regions. We present volumes with thresholded change detection scores in Fig.4.25.

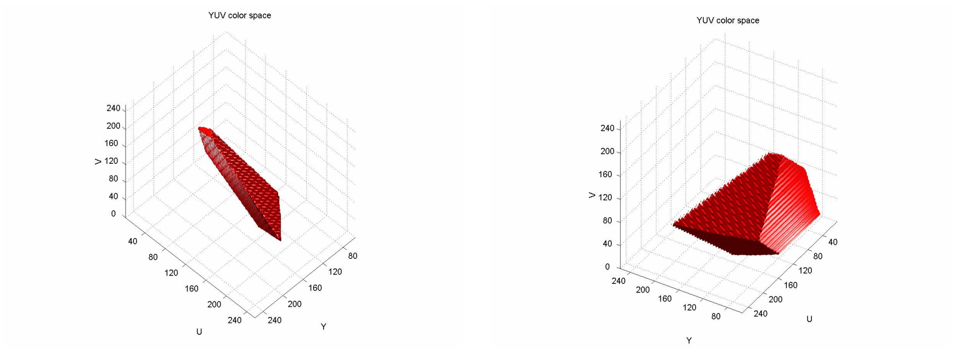


Figure 4.26: Skin colors in the YUV color space from different viewpoints.

4.5.5 Color Detection Mask

Color is a powerful descriptor that has practical use in the detection of objects exhibiting certain color attributes. The aim of the color detection mask is to classify points of the input video as target color and non-target color points, i.e., in human face locating skin and non-skin point.

The use of color information has gained increasing attention first in the face locating problem. Some recent publications that have reported this study include [75], [25]. We generalized face locating approaches to evaluate whether a point is in a given target color set or not. We compute the target color concentration of each volume after growing volumes. For a volume V_i, the number of points in the target color set is found

$$\gamma_{tc}(i) = \frac{1}{\gamma_{si}(i)} \sum_{t} \sum_{x,y \in R_i^t} w_{tc}(x,y,t)\,. \tag{4.41}$$

Human skin has certain color characteristics. As a special case, we adapted a human skin color map. We have found that a skin-color can be identified by the presence of a certain set of color values that is illustrated in the YUV space as shown in Fig. 4.26. The color ranges are sculptures as

$$w_{tc}(p) = \begin{cases} 1 & \begin{array}{c} |\tan^{-1}(\frac{y+1.77u}{y+1.40v}) - \frac{\pi}{4}| < \frac{\pi}{8} \ \wedge \\ |\tan^{-1}(\frac{y-0.34u-0.72v}{y+1.40v}) - \frac{\pi}{6}| < \frac{\pi}{18} \ \wedge \\ |\tan^{-1}(\frac{y+1.77u}{y-0.34u-0.72v}) - \frac{\pi}{5}| < \frac{\pi}{15} \ \wedge \\ y + 0.48u + 0.23v > 20 \end{array} \\ 0 & else \end{cases} \tag{4.42}$$

where y, u, v are the color values of the point p. The $w_{tc}(p)$ is assigned as a skin color point if the color values at p fall inside their respective ranges. These ranges are adapted from the RGB color space to the YUV space by exhaustive tests. The above conditional formula do not require any computation; it is implemented as a look-up table.

4.5.6 Feature-based Motion Estimation

A crucial egg-or-chicken problem of motion based segmentation is “should the region of support be obtained first by color segmentation then motion field is estimated, or first motion field is obtained then region of support is determined?” Volume growing provides all the region of supports, i.e. frame-wise projections of volumes, and approximation of the translational motion. Therefore, volume growing improves the above problem by supplying the region of supports and an initial estimation of motion at the same time.

The motion of volumes can be represented by parameterized models. These parameters are found by fitting a motion model, preferably affine or higher, to motion vectors. Motion vectors are estimated for a limited number of confidence points in each region of support. In comparison, an optical flow method would be restricted to the small motions, moreover it is intensity sensitive, and also computationally expensive. A dense block-matching method would be infeasible. On the other hand, using selected feature points has advantage of preventing from the aperture problem, and it is not computationally intensive.

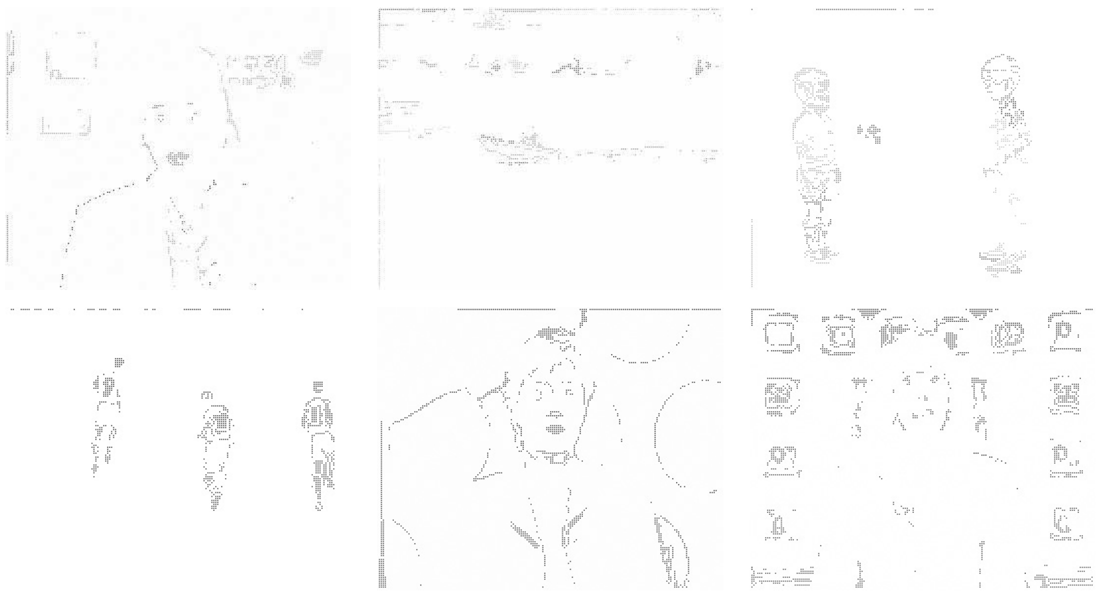


Figure 4.27: Extracted feature points with respect to Eq. 4.44.

Feature Points

A feature point should have higher spatial energy than the other points since it will be used in block matching, and higher spatial energy point give more accurate estimate in matching. A simple spatial energy function $\Omega(p)$ is defined as a variant of the local texture

$$\Omega(p) = \sum_k [\max w_k(q) - \min w_k(q)], \qquad q \in \wp(p) \tag{4.43}$$

where $\wp(p)$ is a local window around the point p. However, the above definition does not quantify the energy accurately. We utilized the following variance based energy function

$$\Omega(p) = \sum_k \sum_q (w_k(q) - \bar{\mu}_k)^2, \qquad q \in \wp(p) \tag{4.44}$$

Here, $\bar{\mu}_k$ is the mean of the color channel k in the window $\wp(p)$. After the spatial energy is computed in the region R_i^t, the points of R_i^t are ordered in a list. The first in the list is assigned as a feature point $p_f(i,t)$, and a spatial window around it is cleared. Then, the next available point in the list is chosen. In our simulations, we used 5×5 local windows. A certain number of such feature points $p_f(i,t)$, around 50, are determined for each region R_i^t of the volume V_i. Fig. 4.27 shows the feature points selected for the corresponding images.

Cross Search Block-Matching

Cross search is a suboptimal implementation of exhaustive search block matching technique. This algorithm was introduced by Ghanbari [88]. The basic idea is still a logarithmic step search, however, the main difference between this and the logarithmic search method presented before is that the search locations are the end points of a diagonal cross "$\times$" rather than a regular "+". The algorithm may be described as follows:

- Step 1 : The center block is compared with the current block and if the distortion is less than a threshold, the algorithm stops. Distortion is the color difference between the blocks.
- Step 2 : Pick the first set of points in the shape of a "$\times$" around the center. Move the center to the point of minimum distortion.
- Step 3 : If the step size is bigger than 1 halve it and repeat step 2, otherwise go to step 4.
- Step 4 : If in the final stage the point of minimum distortion is the bottom left or the top right point, then evaluate distortion at 4 more points around it with a search area of a "+". If, however, the point of minimum distortion is the top left or bottom right point, evaluate the distortion at 4 more points around it in the shape of a "$\times$".

The above algorithm is demonstrated in Fig. 4.28. The cross search algorithm requires $5 + 4\log(2h)$ comparisons, where h is the largest allowed displacement. The

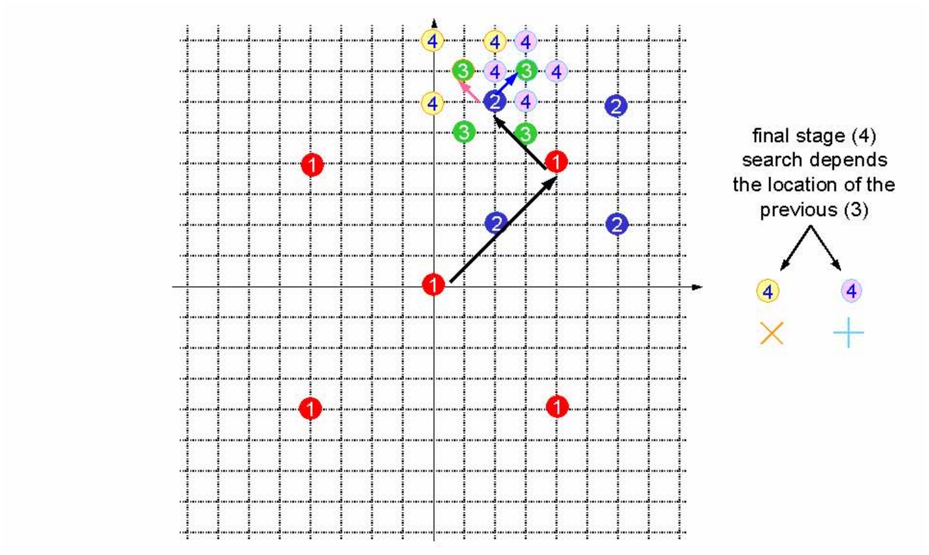

final stage (4) search depends the location of the previous (3)

Figure 4.28: The cross search algorithm finds the suboptimal motion vector.

algorithm has a low computational complexity. It is, however, not the best in terms of compensation. By cross block matching, we found the motion vectors $m(p_f)$ of the feature points p_f determined in the previous section. The search range is chosen as ± 8 points, and the search center shifted with respect to the trajectory motion.

Model Fitting to the Motion Vectors

Using the above technique, motion vectors are estimated for the feature points. These vectors are fitted to an affine motion model by minimizing a Lorentzian based error term. The Lorentzian term converts minimization problem to a robust estimator. Robust estimation enables to detect and reject the measurement outliers that violate the motion model.

Given the motion vectors $m(p_f)$ for a region R_i^t, we want to recover an affine model $u(p_f, a_1, .., a_6)$ that minimizes

$$J(m,u) = \sum_{p_f} \log(1 + [m(p_f) - u(p_f)]^2) \tag{4.45}$$

$$= \sum_{p_f} \log(1 + [m_x(p_f) - a_1 x_f - a_2 y_f - a_3]^2 \\ +[m_y(p_f) - a_4 x_f - a_5 y_f - a_6]^2) \quad (4.46)$$

where $m_x(p_f)$ and $m_y(p_f)$ are the vertical and horizontal components of the motion vector, and p_f is (x_f, y_f, t). We build robust estimator by using downhill simplex minimization. An iterative continuation method is used in which the previously estimated motion models $u(p_f)$ is used as the observation of the next iteration.

4.6 Clustering Volumes into Objects

Clustering is the unsupervised classification of volumes into objects. The volumes V_i's are clustered into objects using their descriptors. The clustering problem in general has been addressed in many contexts and by researchers in many disciplines. Different approaches to clustering data can be categorized as hierarchical and partitional approaches. Hierarchical methods produce a nested series of partitions while a partitional clustering algorithm obtains a single partition of the data. Merging the volumes in a fine-to-coarse manner is an example to hierarchical approaches. Grouping volumes using adaptive k-means method in an coarse-to-fine manner is an example to the partitional approaches as illustrated in Fig. 4.30. We implemented both of these methods. The frames from the original videos are given in Fig. 4.29 to give an idea about the underlying motion in each sequence.

4.6.1 Fine-to-Coarse Hierarchy

Now, we have the smallest components of the video and their attributes. The next task of video segmentation is finding the most similar volumes that forms the same object using their descriptors. A fine-to-coarse hierarchical merging method is appropriate for this purpose. In this approach, determination of most similar volumes is done iteratively. At each iteration, all the possible volume combinations

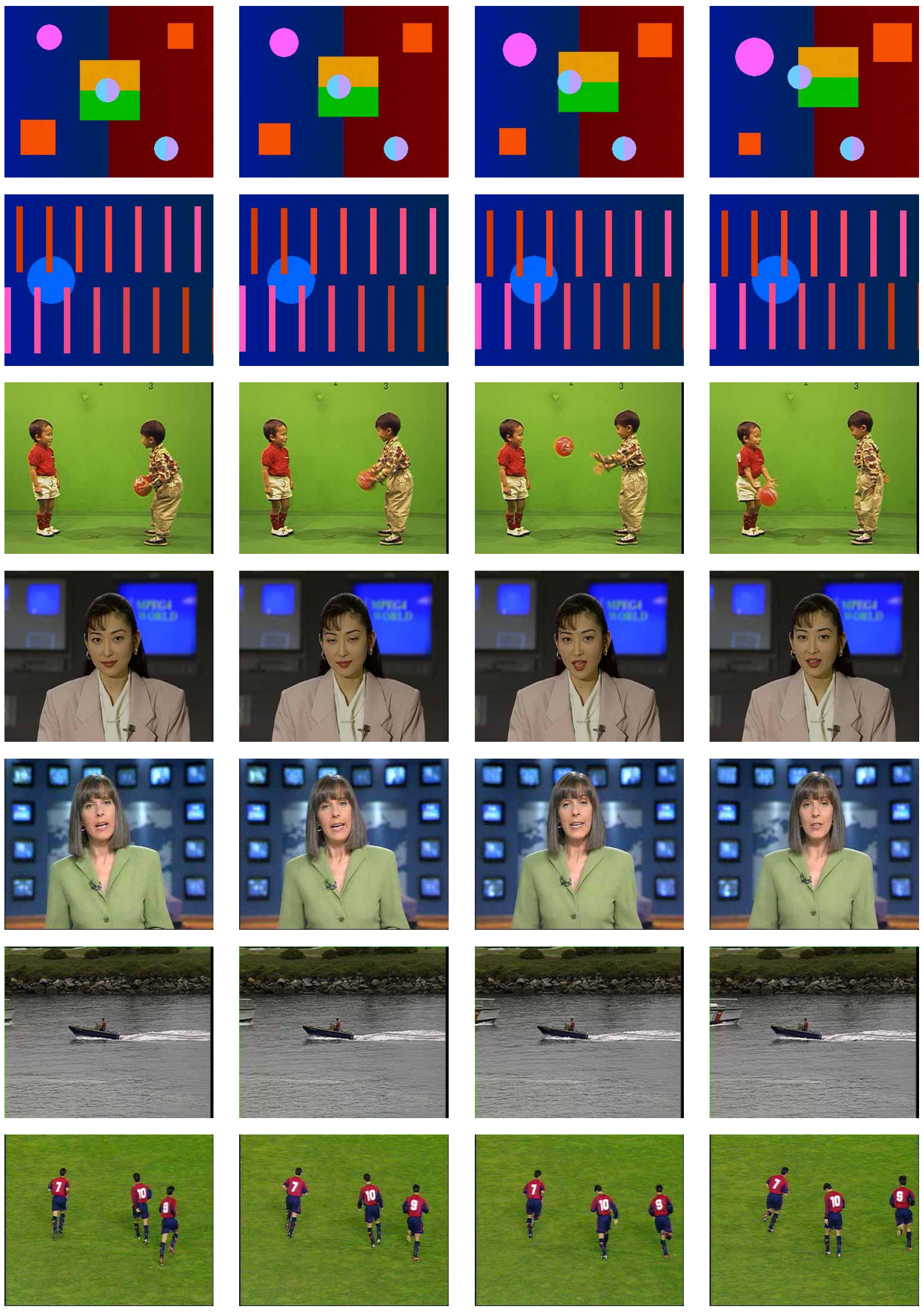


Figure 4.29: The 1st, 8th, 16th and 24th frames from the test sequences.

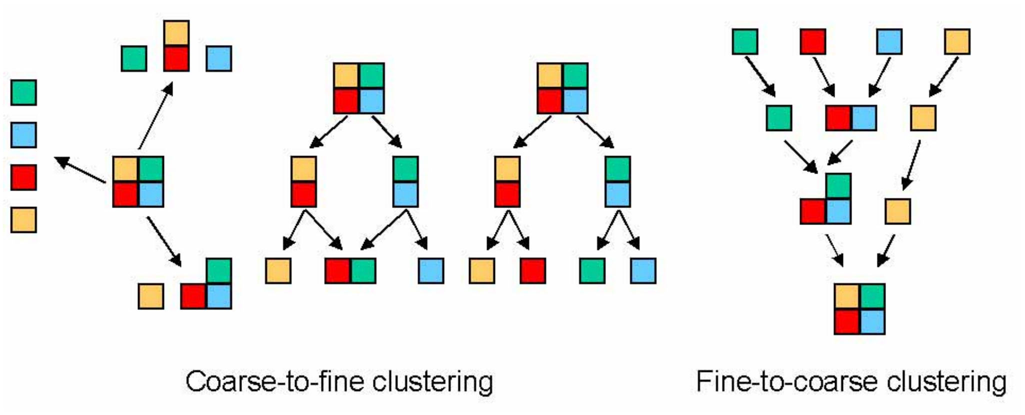


Figure 4.30: Coarse-to-fine (adaptive-k, GLA, quad-tree) and fine-to-coarse clustering. The first approach divides the volumes into certain number of clusters at each time, the second merges a pair of volumes at each level.

are evaluated. The pair having the highest similarity score are merged and affected descriptors are updated. The iterative nature of the algorithm enables an hierarchical clustering.

From another point of view, the semantic information is being hunted at this stage of the segmentation. Therefore, we have to decide which criteria dictate the similarity of volumes, and which volumes of the video form a semantic object. Our decision should include only the available information, i.e. volume descriptors. The following observations are made on the similarity of two given volumes:

1. Two volumes are similar if their motion is similar. In other words, volumes having similar motion construct the same object. A stationary region has high probability of being in the same object with another region that is stationary, i.e., a tree and a house in the same scene. We already measured the motion similarity of two volumes in terms of motion based relational descriptors $\Gamma_{\sigma}(i,j)$, $\Gamma_{dd}(i,j)$, $\Gamma_{pm}(i,j)$, $\Gamma_{\mu}(i,j)$, and $\Gamma_{xt}(i,j)$. These descriptors should be incorporated in the similarity definition. However, without using further intelligent

models, it is not straightforward to distinguish motion similar objects. Such intelligence includes but not limited to neural networks, Markov models as in human recognition.

2. Objects tend to be compact. A human face, a car, a flag, a soccer ball are compact objects made by smaller volumes. For instance, a car in a surveillance video is formed by separate elongated smaller regions. Shape of a volume gives clues about its identity. We captured shape information in the descriptors $\Gamma_{cr}(i,j)$ and $\Gamma_{br}(i,j)$ and also volume boundary itself. Whereas, compactness needs deliberate employment. If a volume is enclosing another volume, their merge will increase compactness whether these two volumes correspond to same the object or not. As a counter example to compactness, we can consider cloud formations, walking person, etc. To improve the success of shape related descriptors, application specific criteria should be used, i.e., a human model for indoors surveillance and videoconferencing.

3. Objects have connected parts. This is obvious for most of the cases, an animal, a car, a plane, a human, etc., unless an object is visible only partially. We begin evaluation of similarity with the volumes that are neighbors to each other. Neighborhood constraint is useful, and yet, can easily deteriorate the segmentation accuracy in case of an under segmentation, i.e., background enclose most of the volumes. The descriptor $\Gamma_{ne}(i,j)$ is used to keep neighborhood data.

4. An object moves as a whole. Although this statement is not true for human objects, for rigid bodies, it is useful. A change detection mask becomes very functional in constructing objects that are moving in front of a stationary background.

5. Color similarity of volumes is potential but not sufficient. Volumes are already color consistent, therefore there is little room for utilization of color information

to determine a neighbor to merge in. In fact, most objects are made from small volumes that have different colors, i.e., human body consists face, lips, hair, dress, etc. When forming the similarity measure, color should not be a key player. However, for specific video sequences features people, human skin color is an important descriptor.

6. Important object is at the center. For this reason, we introduced a focus filter. We can find good examples as in head-and-shoulder sequences, sports, etc.

In terms of the descriptors, the similarity measure is

- inversely proportional to the distance variance Γ_{mv}, the directional difference Γ_{dd}, the parameterized motion difference Γ_{pm},
- proportional to the compactness ratio Γ_{cr}, to the mutual surface ratio Γ_{sr}, the change detection mask Γ_{cd}, skin color feature Γ_{sk}.

To blend all the above observations and statements, we developed a ranking scheme based similarity measure. For all possible neighboring volume pairs (V_i, V_j), the descriptors are ordered in separate lists $r(i,j)$. Using the ranks in the corresponding lists, a similarity measure $\Lambda_o(i,j)$ is computed at the object level o for the pair i, j as

$$\begin{aligned}\Lambda_o(i,j) &= c_{mv} r^{\uparrow}_{mv}(i,j) + c_{dd} r^{\uparrow}_{dd}(i,j) + c_{pm} r^{\uparrow}_{pm}(i,j) \\ &\quad + c_{cr} r^{\downarrow}_{cr}(i,j) + c_{sr} r^{\downarrow}_{sr}(i,j) + c_{cd} r^{\uparrow}_{dd}(i,j) + c_{sk} r^{\downarrow}_{sk}(i,j) \qquad (4.47)\end{aligned}$$

where $r^{\downarrow}$ is the rank for the pair V_i, V_j in the list that is ordered from the larger value to smaller, and $r^{\uparrow}$ is from smaller value to larger. There are constant weights c's to adjust the contribution of each descriptors. The pair (V_i^*, V_j^*) having the maximum score are merged, and the descriptors are updated accordingly. Clustering is performed until there are only two volumes remain, i.e. $o = 2$. Results of above method are presented in Fig. 4.31.

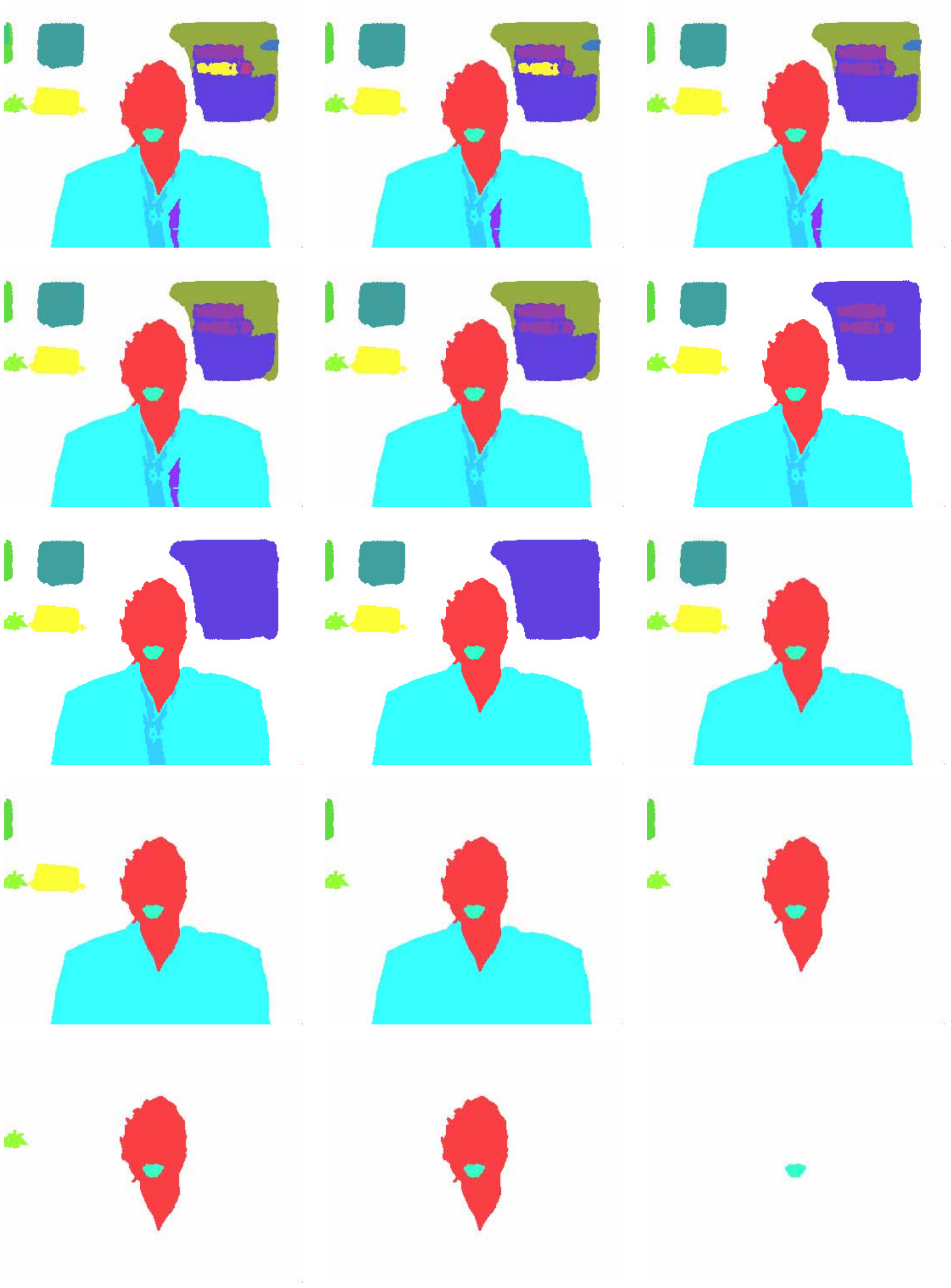


Figure 4.31: Fine-to-coarse clustering results at object levels 19, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 4, and 3.

At a level of the clustering algorithm, we can analyze whether the chosen volume pair at that level fulfill the motion consistency. This can be done by observing the behaviour of the similarity score. If this score gets small or shows a sudden drop, the merge is likely to be not a valid merge although it is the best merge. The derivative of the similarity score with respect to object level gives this information

$$\frac{\partial \Lambda_o}{\partial o} = \Lambda_{o-1} - \Lambda_o. \tag{4.48}$$

4.6.2 Coarse-to-Fine Hierarchy

Alternatively, one can prefer to use a partitional clustering method that divides the volumes into certain number of groups, i.e., into 2, 3 partitions. To achieve a partitional clustering, a common quantifier should be decided upon. Since motion is a distinctive metric of the volumes, motion trajectories are suitable for this purpose. Parameterized motion models would be used as well. Another quantifier is the change detection descriptor.

In coarse-to-fine clustering, all the volumes are in the same group initially. Then, the following algorithm is applied to volumes:

1. Partition all volumes into K initial clusters. At the beginning of the algorithm, there are two initial clusters, i.e., $K = 2$.

2. Assign each volume to the cluster k whose centroid $\widehat{C}_k = [\widehat{C}_{k1}\ \widehat{C}_{k2}\ ..\ ]^T$ is nearest.

3. Recalculate the centroid $\widehat{C}_k$ for the cluster receiving in a new volume and for the cluster losing a volume.

4. Repeat step 2 until no more assignment takes place.

Initial cluster centroids can be chosen either randomly, or using the perturbed previous clustering levels $K-1$ centroids. This method which is also referenced as k-means,

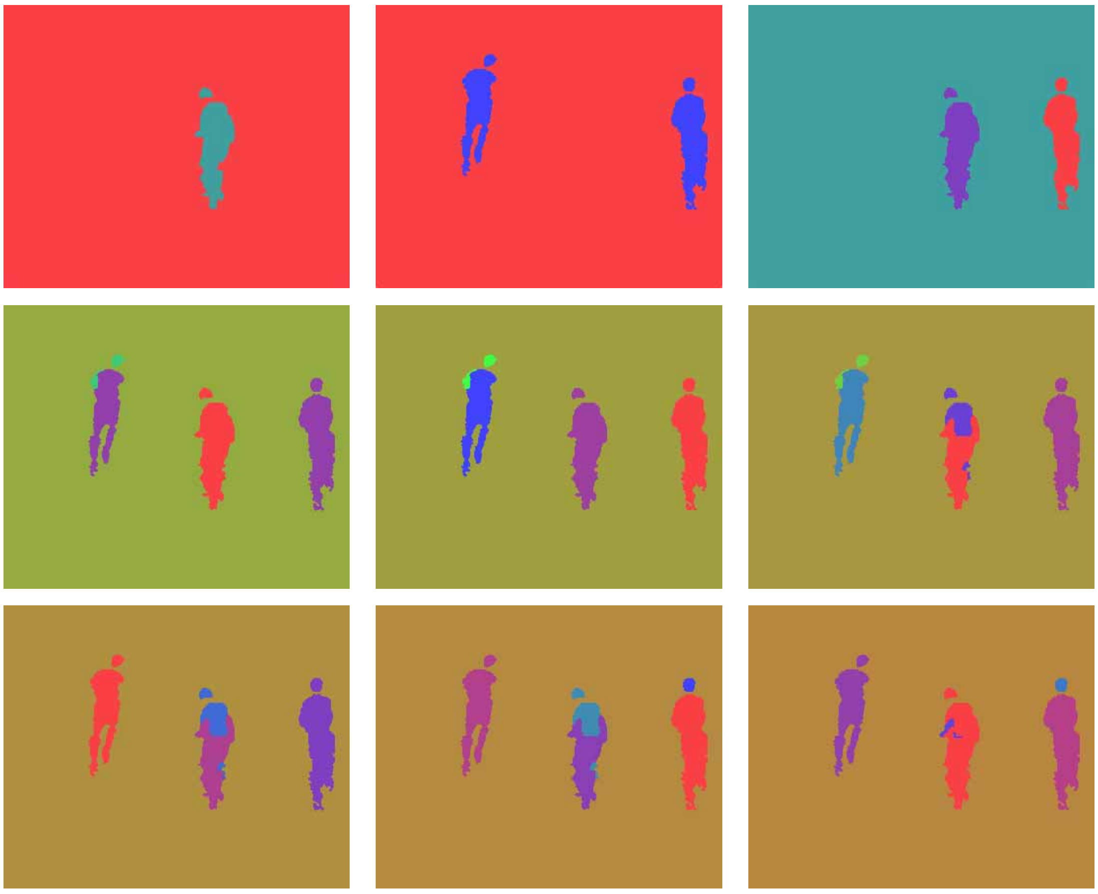


Figure 4.32: Coarse-to-fine clustering results at object levels 2, 3, 4, 5, 6, 7, 8, 9, and 10.

tries to minimize the sum of the within cluster variances

$$\sum_{k=1}^{K} \sum_{i} \delta_{ik} |C_i - \widehat{C}_k|. \tag{4.49}$$

The indicator function δ_{ik} equals to 1 if the observation C_i comes from cluster k, or 0 otherwise based on the previous iteration. The centroid $\widehat{C_k}$ is made of the mean values of the elements C_{kj} in the k^{th} cluster

$$\widehat{C}_{kj} = \frac{1}{n_k} \sum_{l} C_{lj} \tag{4.50}$$

We denote n_k as the number of volumes belonging to the cluster k. In order to increase

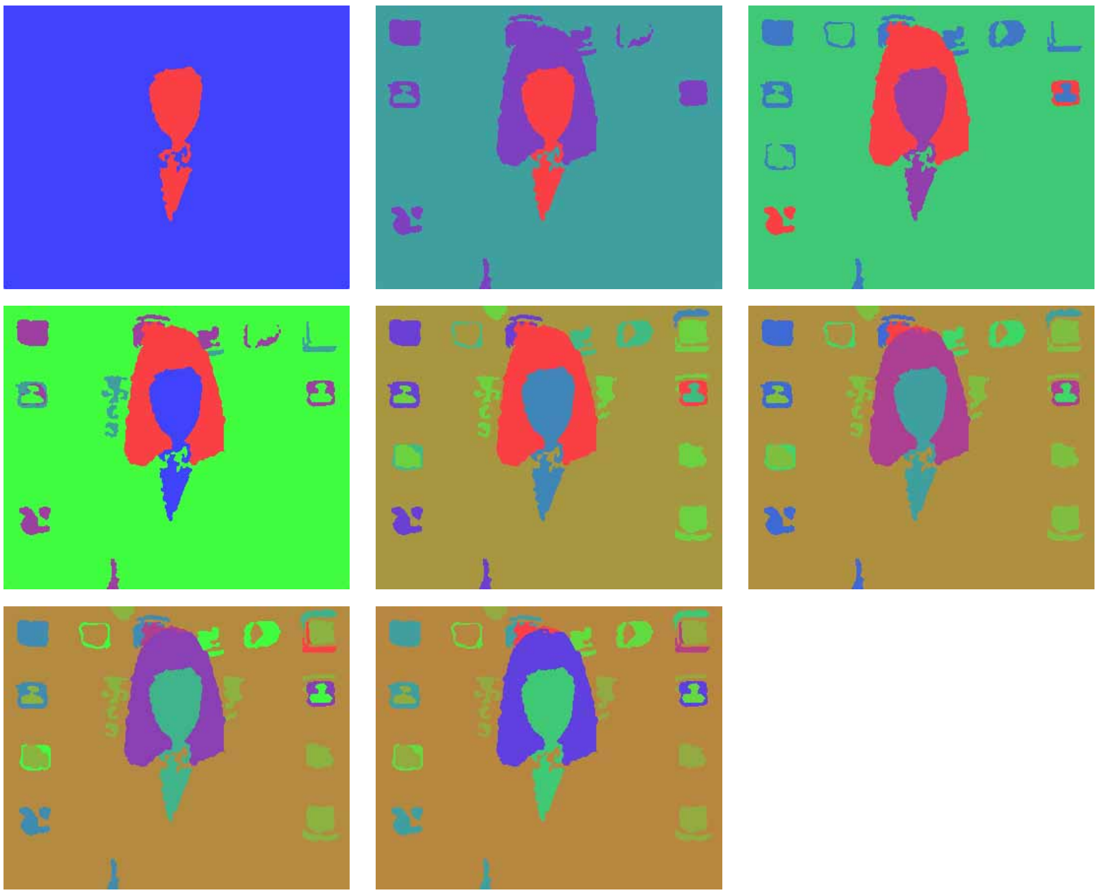


Figure 4.33: Coarse-to-fine clustering results at object levels 3, 4, 5, 6, 7, 8, 9, and 10.

the stability in cluster analysis, specific weights or adaptive weights in the distance formula could be applied rather than an ordinary weight. The following weights

$$q_j = \frac{1}{\sum_k^K \sum_i^{n_k} \delta_{ik} |C_{ij} - \widehat{C}_{kj}|} \tag{4.51}$$

can be used in the distance term

$$|C_i - \widehat{C_k}| = \sum_j q_j |C_{ij} - \widehat{C}_{kj}|. \tag{4.52}$$

In coarse-to-fine clustering we used the trajectories T_k's as the observations C_k's,

$$|C_i - \widehat{C_k}| = \sum_j q_j |T_{ij} - \widehat{T}_{kj}|. \tag{4.53}$$

However, since no neighborhood constraint is involved in the adaptive k-means clustering, the clustered volumes are not required to be connected. This is the main deficiency of partitional clustering using adaptive k-means as implemented above. The spatial position can be incorporated, but weighting the trajectory elements and spatial location elements becomes another problem. The segmented volumes by adaptive k-means clustering are presented in Fig. 4.32.

4.7 Multi-Resolution Object Tree

A computer can help human to overcome tedious tasks involved in segmentation easily, however, no existing object segmentation and recognition system is as complete as a human being. Therefore, human becomes the ultimate decision maker in analyzing the results of video segmentation. It is necessary to provide the segmentation results in an appropriate format to user or other decision mechanism for further analysis. An object-based tree representation that captures the segmentation results of a video enables effective human access.

The segmentation algorithm supplies volumes, their attributes, and information about how these volumes should be merged. Putting this information in a tree

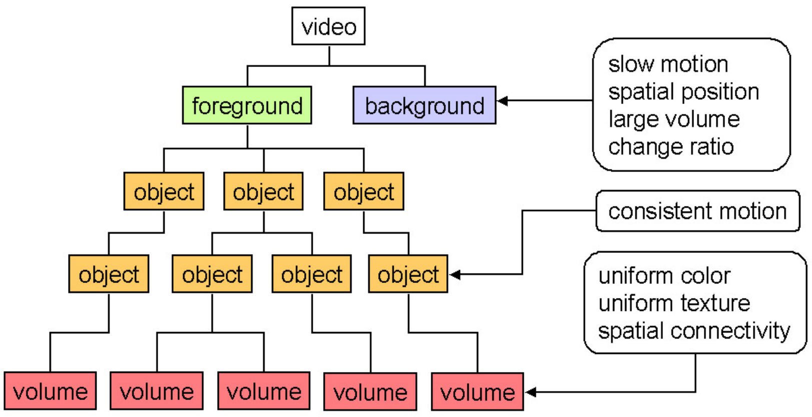


Figure 4.34: Multi-resolution partition of objects in a hierarchical tree representation.

representation, we obtain an object-based structure as demonstrated in Fig. 4.34. In this tree representation, the video is divided into objects, and objects into volumes. At the lowest volume level, the descriptors and boundaries are available. Volumes are homogeneous in color, texture, and they are connected within. The clustering generates higher levels that are consistent in motion. A background can be identified using shape, size, motion attributes. In case a user wants to change the clustering method or reassigns the number of objects, the video is not required to be segmented again. Only the clustering stage is executed, which takes a negligible computation, with the already derived descriptors. The object tree can be appended to the video as a header after the segmentation. If user prefers to outline a boundary around or on the objects of interest, those objects are instantly detected using the object tree.

4.8 Test Version

Out of several possible configurations as proposed in this chapter, we specified a version to be used as a reference. This version is designed to be computationally as simple as possible, and it uses suboptimal versions of the procedures. A flow diagram

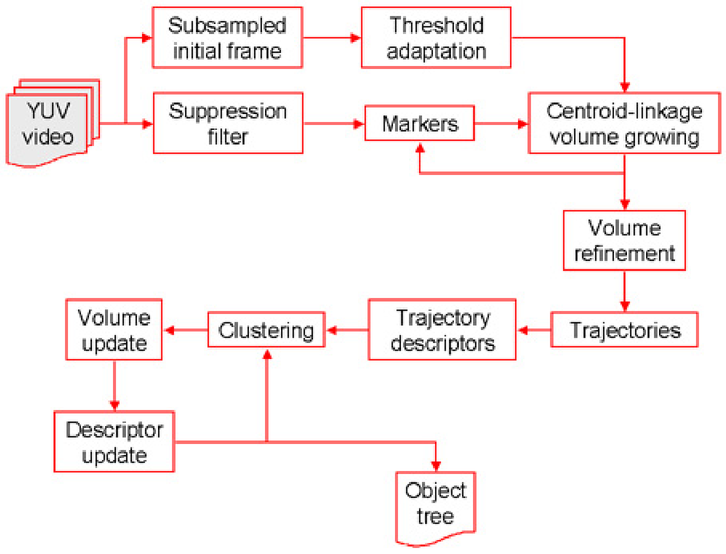


Figure 4.35: The version optimized for speed.

is given in Fig. 4.35.

First, the color distance thresholds are adapted by using a similar method as explained in the centroid-linkage. The first frame of the video sequence is subsampled in both vertical and horizontal spatial directions, and the color histograms are computed for each channel. These histograms are smoothed with recursive application of a moving average filter. The first and second order derivatives are computed to determine the local maxima. The numbers of local maxima at each color channel are used in the distance metric and threshold as presented before.

The minimum gradient method is used to determine the markers. The marker pixels are selected from the subsampled data instead of the full resolution data. In a single subsampled frame, the pixel that has the minimum color gradient is selected as a marker and a volume is initiated. After the current volume is grown, the next marker is chosen among the pixels of the next frame. The frames are switched in a cyclic manner to cover all the frames. If the same video frame is used for selecting the

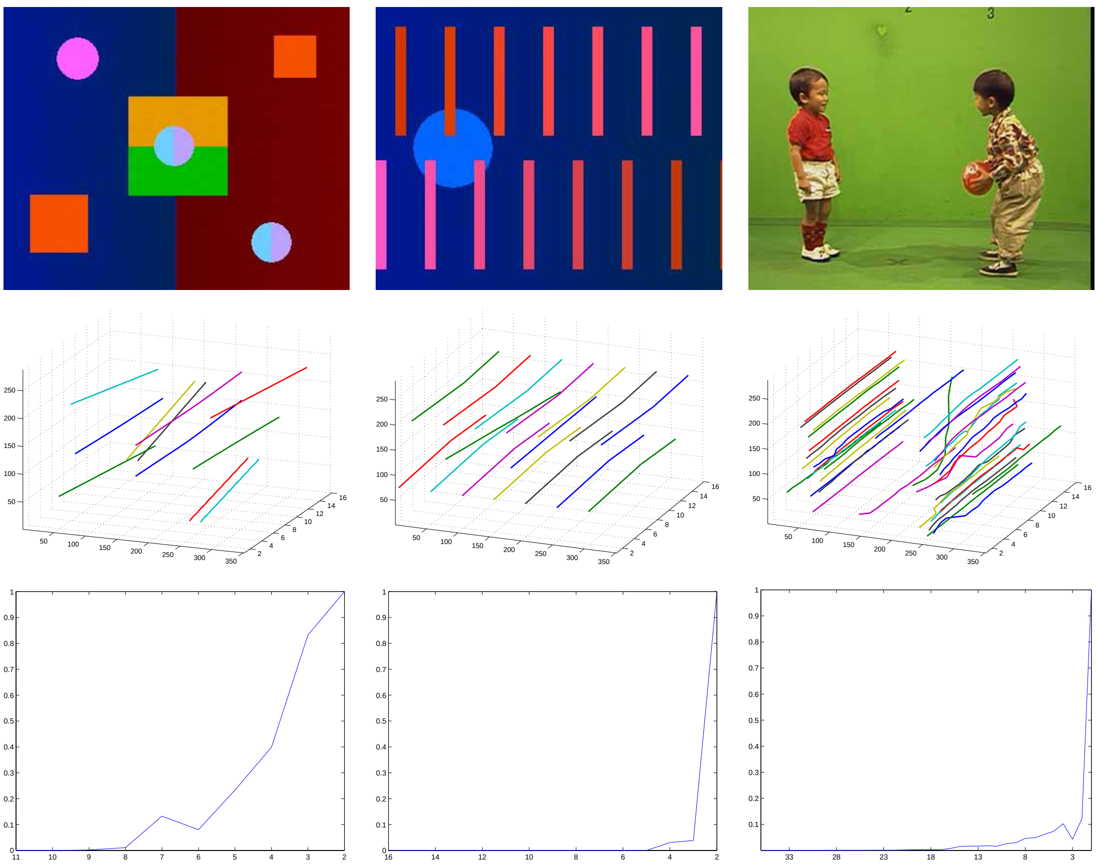


Figure 4.36: The initial trajectories and minimum variance vs. object level graphs.

markers, a region that is visible only in the remaining frames may not be segmented. Optimally, all the frames of the input video are required to be searched for the markers. However, such an exhaustive search would be computationally very expensive. The frame-wise minimum gradient selection is a sub-optimal solution that accelerates the search problem significantly.

Centroid-linkage technique is employed when volumes are grown. A color distance is computed between the candidate pixel and the centroid vector of the current volume. Intra-frame, inter-frame switching method is utilized to prevent a volume from having disconnected regions. After the spatiotemporal data are segmented into volumes, the pixels belong to small and elongated volumes are unmarked and the remaining volumes are inflated to fill up the empty pixels by using color distance.

While growing volumes, the corresponding volume trajectories are extracted simultaneously. In the clustering stage, fine-to-coarse merging algorithm is employed to choose the best merge at each object level. The variance of the trajectory distance descriptor $\Gamma_\sigma(i,j)$ is used to compute the similarity score of the clustering algorithm. The initial trajectories in the spatiotemporal data and the trajectory variances of the merged volumes at each object level are given in Fig. 4.36. The segmentation results at the various object levels are presented in Figs. 4.37-4.38.

4.9 Summary

We introduced an automatic segmentation framework. Our goal was to detect accurate boundaries of moving objects. The framework takes multiple video frames at once, constructs a spatiotemporal data structure, and by using a three dimensional volume growing technique it finds the smallest components of the video. These volumes are expanded from the marker points using several linkage methods. Quantitative descriptors that represents each volume, and relational descriptors that capture the mutual properties of a pair of volumes are determined by evaluating the shape, trajectory and parameterized motion. The descriptors are utilized in clustering methods to construct objects. The main stages of the presented automatic segmentation framework are itemized as

1. Constructing the spatiotemporal data
2. Filtering and simplifying color distributions
3. Assigning markers as seeds of volumes
4. Volume growing
5. Removal of volume irregularities

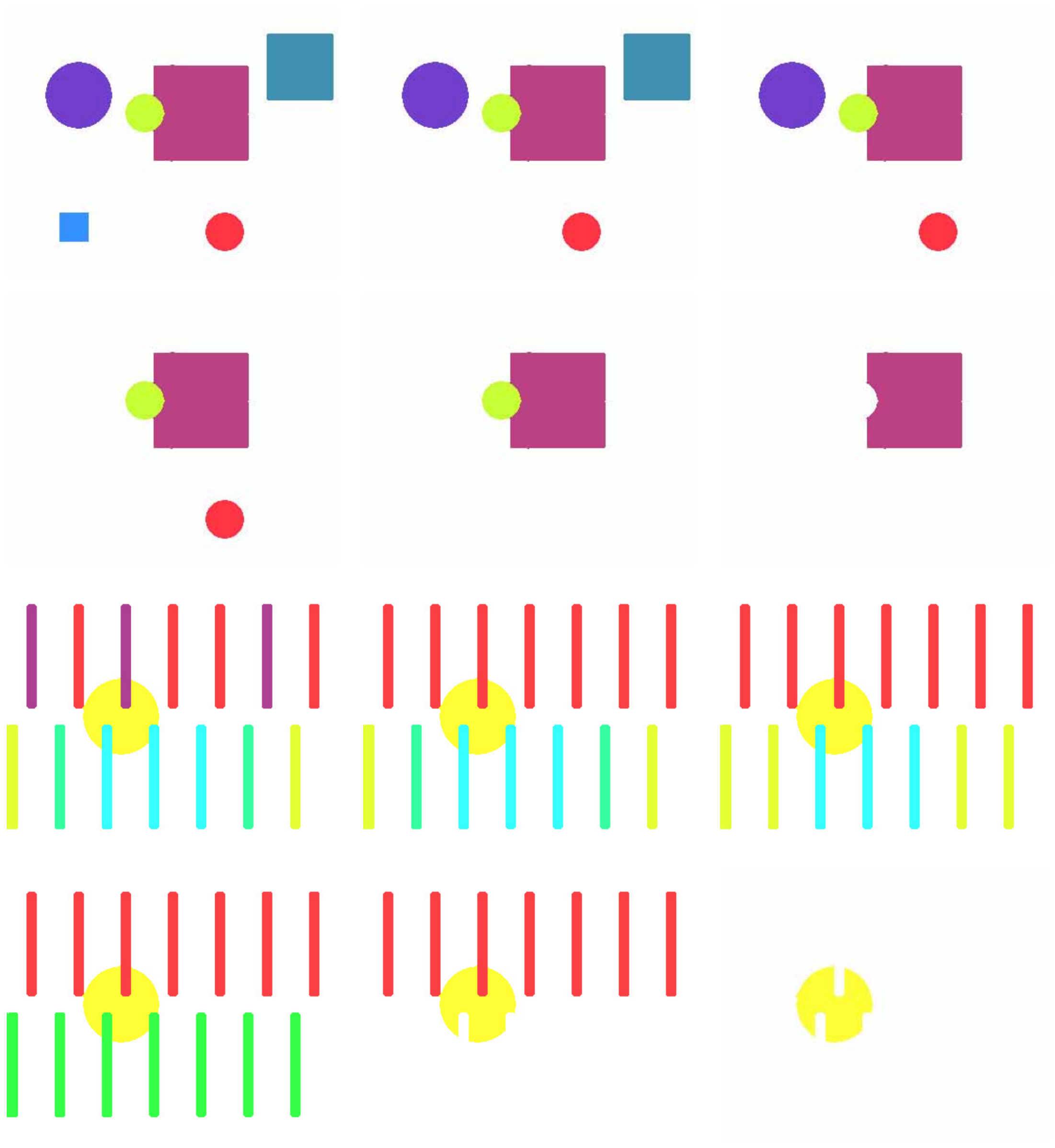


Figure 4.37: Test results at object levels 7, 6, 5, 4, 3, 2 for the synthetic sequences.

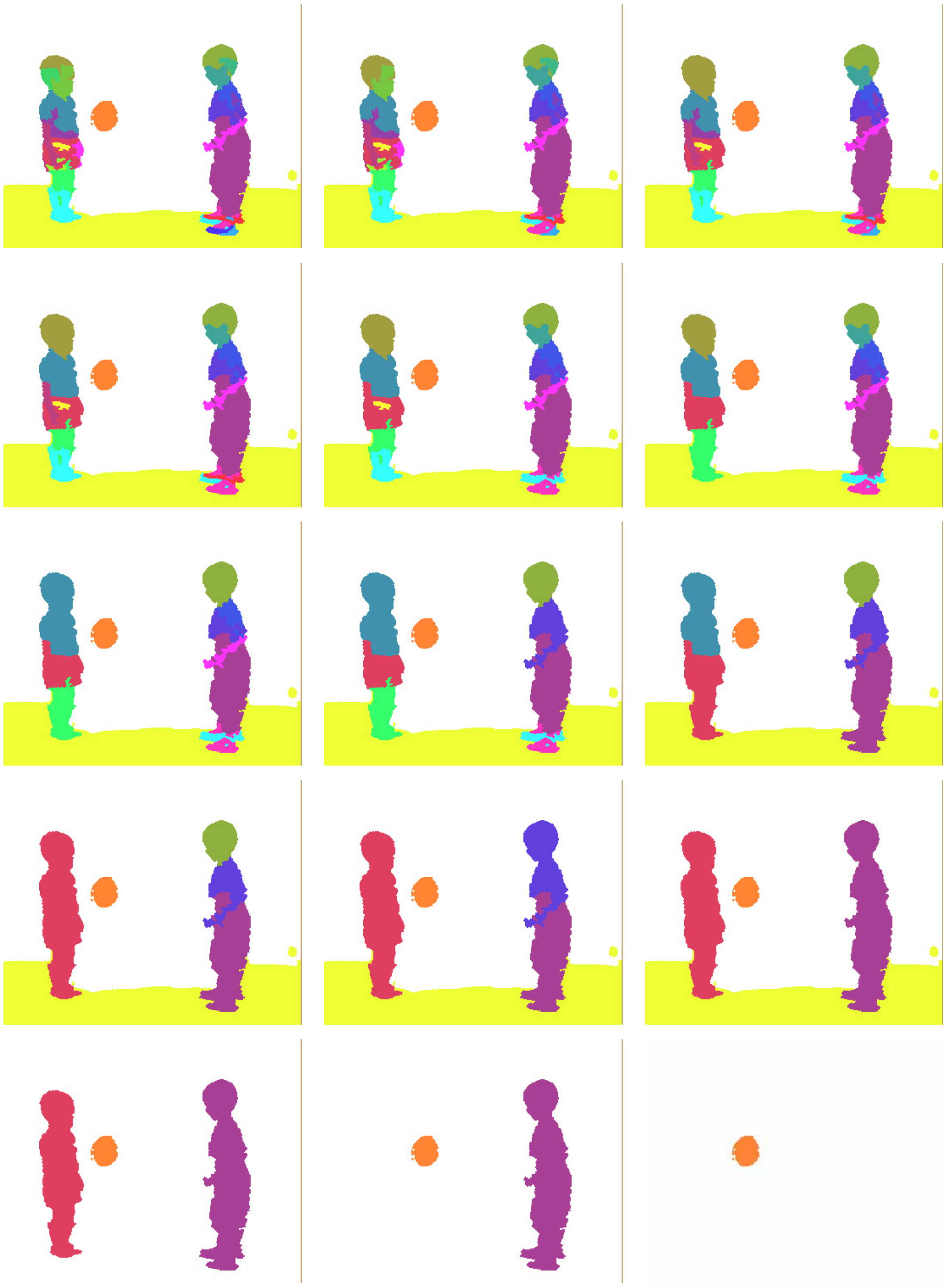


Figure 4.38: Test results at object levels 36, 33, 27, 24, 21, 18, 15, 12, 9, 8, 7, 6, 5, 4, and 2.

6. Extracting trajectories

7. Deriving quantitative and relational descriptors

8. Clustering volumes into multi-resolution object tree

We developed a spatiotemporal data structure which embodies the color, texture, edge, change detection, and other features of the video frames in terms of vectors corresponding to image points.

We implemented two marker selection approaches within the spatiotemporal data. One approach uses uniformly distributed markers, the second approach assigns markers by choosing the minimum gradient points.

We discussed several linkage methods; single-linkage, dual-linkage, and centroid-linkage volume growing. For each method, we proposed threshold adaptation techniques. We proposed to utilize MPEG-7 dominant color descriptors in threshold determination.

We designed variants of the volume growing. Out of these, the simultaneous growing and one-at-a-time growing methods basically differ in the number of markers that are active at each iteration. The recursive diffusion and intraframe/interframe switching methods offer different expansion mechanisms.

We extracted motion trajectory of each volume using the frame-wise center of masses. We utilized several novel descriptors to quantify volumes. We introduced the relational descriptor concept which evaluates the similarity of a pair of volumes. The descriptors are designed to capture the characteristics of volumes as much as to incorporate priori information and application specific constrains. We discussed to integrate skin color map and change detection map into segmentation framework.

Hierarchical and partitional clustering approaches were adapted to generate objects. We proposed a rank based similarity measure of volumes. We developed a multi-resolution object tree representation as an output of the segmentation.

The proposed framework blends the advantages of color, texture, shape, motion based segmentation methods in an automatic and computationally feasible way.

Chapter 5

Concluding Remarks

"Verily, with every difficulty there is relief."

Automatic object segmentation has potential to improve recognition, identification and event analysis of video sequences, it is vital in object-based compression and coding standards, and also provide a better level of video editing, manipulation and animation. Towards this goal, several techniques to detect and segment video objects have been presented and discussed. To conclude this thesis, we first summarize the major contributions of this work, then outline several directions for future work.

5.1 Summary of Main Contributions

The developed segmentation framework combines various disciplines of image and video processing from filtering to data clustering. This thesis has addressed a number of challenging issues associated with video segmentation and image processing. Out of several achievements of the thesis work, the most prominent ones are emphasized below:

1. A solution to the problem of fusing color, shape, motion, and also texture, target color map, change detection map based techniques into a unified segmentation approach is proposed. An automatic moving object segmentation framework is developed. This framework is designed to be adaptable to the input video and compatible with the specific applications. It generates accurate object boundaries from the color video sequences with a feasible computational complexity.

   We believe that extraction of the smallest homogeneous components of a video scene, determination of region boundaries in every frame, obtaining video object properties, mining for any useful information, and presenting all these video analysis in a comprehensible setting are among the accomplishments. The interpretation of the segmentation results depends on the user interest, specific application, and the type of the information that is being mined for.

2. A novel 3D volume growing method for video data is introduced. The method is inspired by region growing, and empowered by expanding the scope of the 2D methods into a 3D spatiotemporal data that is constructed from the video. Instead of regions, 3D volumes are employed as the smallest constituents of objects. Adding another dimension enables better analysis of video for segmentation purposes. Within the volume 3D growing method, the following improvements are also identified:

   - A fast marker selection method is designed. To accelerate selection, a subsampled version of spatiotemporal data is searched for markers. The search is done in sliced portions of the subsampled data to find the local minima. This provides a suboptimal but significantly fast solution.
   - MPEG-7 dominant color descriptor is utilized in adaptive threshold determination. Segmentation is made adaptable to the input video by analyzing the color histograms. MPEG-7 dominant color descriptor provides useful

histogram information, and this is used to decide the thresholds. A logarithmic color distance function is formulated using the adaptive thresholds. This distance function compensates the color difference of separate channels with respect to their dynamic ranges and variances.

- Diffusion and intraframe-interframe switching segmentation approaches are developed. Diffusion strategy addresses volume expansion as an inflating balloon. On the other side, intraframe-interframe switching approach accomplishes frame-wise connected projections of a volume by growing it first into a single frame then expanding to the adjoin frames.
- Rank based volume refinement strategy is proposed. A similarity score for each possible volume merge is found by evaluating the ranks of several descriptors in the corresponding ordered lists to determine the volume pair that should be merged. Utilizing the ranks in the similarity function instead of variants of the distance magnitude is a novel method.

3. Motion trajectories are acquired without computing any dense motion. As an advantage, motion trajectories enable to approximate translational motion of a region without computing motion vectors. The idea of obtaining translational motion without a block-matching or optical flow method is novel. In the segmentation framework, these trajectories are used to determine the motion based object descriptors. In addition, they can also initialize dense motion estimation algorithms.

4. Quantitative descriptors of each volume are defined. We formulated the color, shape, and motion of a volume in terms of quantitative descriptors. Such descriptors significantly simplify the evaluation of volumes and objects attributes. Relational descriptors of volume pairs are defined. These descriptors measure the relative similarity of the respective volume attributes such as motion variance, compactness, existence, etc. They are essential in the fine-to-coarse hier-

archical clustering algorithm.

5. Fine-to-coarse hierarchical clustering of volumes is proposed. Volumes are merged into each other using their descriptors. This methodology can also be adopted to the region segmentation problems. Multi resolution object tree representation of video is conceptualized. The object tree can be appended to the video as a header after the segmentation. If user prefers to utilize a different metric in the clustering part or specifies the number of objects, the segmentation process is not repeated, but only the multi-resolution tree is restructured.

In addition, effects of color space selection on color based segmentation are investigated. Three subjective tests are designed to evaluate performances of several color spaces for region growing.

Also, a human skin color cluster in the YUV color space is parameterized. The parameters are obtained by training cluster model with the test images. This cluster is implemented as a look-up table to minimize the computation time.

To improve on video filtering, a novel Lorentzian-based image simplification method is developed. The simplification filter is robust towards the deviations such as the outlier points. This simplification method is employed to filter the noise, as well as to remove fine texture in terms of the high frequency spatial color distribution.

In summary, the future and success of automatic video segmentation applications will depend on a variety of key components. These components will be responsible for everything from object-based coding, to the content analysis, indexing, retrieval, identification, recognition, editing, manipulation and animation. In this thesis, we focus on the components of this object extraction system that deal with traditional video processing and data understanding issues. More specifically, the issues related to homogeneous region growing theory and the clustering of video objects, as well as the classical problem of filtering. The results that we have obtained are promising and the contributions of this work have already had a noticeable impact.

5.2 Future Directions

This research will serve as a stepping stone for the further developments of the automatic segmentation concept. Among such extensions, application specific modifications, hierarchical volume growing methodologies, adaptation for streaming video could be considered in future work.

Improvements on the Algorithm

Application Specific Constraints

The modular design of the segmentation framework enables embedding application specific constraints. One such example is road surveillance video in which cameras are often stationary and regularly moving objects motion can be limited within the road boundaries. In addition, the motion of the vehicles can be analyzed to detect congestion, accident, or suspicious behaviour, e.g., driving in the opposite lane. Similarly, for human objects, a skin color feature and length-height aspect ratio can be incorporated. The volume descriptors are suitable to integrate such constraints.

Hierarchical Volume Growing Methods

One possible improvement on the current implementation is developing hierarchical volume growing methods to improve the speed of the segmentation processes. Hierarchical growing manages fine spatial texture effectively, however the blocking artifacts may occur on the boundaries. Smaller volumes may not be detected either. The propagation of the segmentation information between the layers, fast generation and indexing of low resolution data structures are some of the issues to be considered.

Adaptation for Memory Constraints

Due to the memory considerations, the spatiotemporal data can be constructed up to a certain number of video frames each time, then the object information is propagated to the following data structure. For this purpose, the object descriptors and boundaries can be used to initialize the new markers and volumes as it is done in the standard tracking approaches. In this way, video adaptive parameters, such as color distance thresholds can be reused to reduce the computational load. On the other hand, object correspondence problem between the two spatiotemporal data becomes a main issue. Correspondence problem has to address some other irregularities such as occlusion, newly appearing, and disappearing objects as well.

Extensions on Multi-Resolution Object Tree

In some applications, it may be useful to have a clustering that is not a partition. This means clusters are overlapping. Fuzzy clustering and functional clustering are ideally suited for this purpose. Also, fuzzy clustering algorithms can handle mixed data types. However, a major problem with fuzzy clustering is that it is difficult to obtain the membership values. It is required to represent clusters obtained in a suitable form to help the decision mechanism specific to the application.

Integration with Other Systems

Adaptation for semi-automatic object extraction methods is a possible extension for the proposed segmentation framework. For semi-automatic algorithms, the user is required to identify the semantic objects of interest initially, and contours of regions of interest are passed to the computer. These regions are tracked temporally from the previous frame. Since temporal tracking tends to introduce boundary errors, the region boundary needs to be modified and updated according to homogeneity criteria in the current frame. Parametric motion models are utilized for temporal tracking while the active snake contour, the watershed algorithm and other techniques are employed for spatial region boundary updates. The use of volume growing approach can provide region boundaries while keeping the computational cost low.

The multi-resolution object-tree is a structured representation of the objects that can be obtained from an initial partition. The leaves of the tree represent objects that belong to the initial partition. The remaining nodes of the tree represent objects that are obtained by merging the other objects. The root node represents the entire video. The tree represents a fairly large set of objects at different scales. Large objects or objects having distinctive motion appear close to the root whereas small details can be found at lower levels. This representation should be considered as a compromise between representation accuracy and processing efficiency. The main advantage of the tree representation is that it allows the fast implementation of sophisticated processing techniques. These properties can be used towards developing efficient video representation, analysis, search, and browsing tools.

List of Publications

Conference Papers

1. F. Porikli and Y. Wang, "An unsupervised multi-resolution object extraction algorithm using video-cubes," *IEEE Int'l Conf. on Image Processing*, Thessaloniki, Greece, Oct. 2001.

2. F. Porikli, "Accurate detection of edge orientation for color and multi spectral imagery," *Proc. IEEE Int'l Conf. on Image Processing*, Thessaloniki, Greece Oct. 2001.

3. F. Porikli, "Video object segmentation by volume growing using feature-based motion estimator," *Proc. 16th International Symposium On Computer and Information Sciences*, Antalya, Turkey, Nov. 2001.

4. F. Porikli and Z. Sahinoglu, "An online renegotiation-based bandwidth management with circuit assignment for VBR traffic in communication networks", *6th World Multiconference on Systemics, Cybernetics and Informatics*, Orlando, Florida, July 2002.

5. F. Porikli and Z. Sahinoglu, "Dynamic bandwidth allocation with optimal number of renegotiations in ATM networks", *Proc. 10th Int'l Conf. Communication and Computer Networks*, Scottsdale, Arizona, Oct. 2001.

6. F. Porikli, "Image simplification by robust estimator based reconstruction filter," *Proc. 16th Int'l Symposium On Computer and Information Sciences*, Antalya, Turkey, Nov. 2001.

7. F. Porikli, "Object segmentation of color video sequences," *Proc. Int'l Conf. On Computer Analysis of Images and Patterns*, Warsaw, Poland, Sept. 2001.
8. F. Porikli and T. Keaton, "An unsupervised road extraction algorithm for very low-resolution satellite imagery", *Int'l Conf. Pattern Recognition and Remote Sensing* Andorra, June 2000.
9. F. Porikli, "Stripe mesh based disparity estimation by using 3-D Hough transform," *Proc. IEEE Int'l Conf. on Image Processing*, Santa Barbara, CA, Oct. 1997.
10. F. Porikli, Y. Wang and C. Swain, "Adaptive stripe based patch matching for depth estimation", *IEEE Int'l Conf. on Acoustics Speech and Signal Processing*, Munich, Germany, April 1997.
11. F. Porikli, Y. Wang, "Disparity estimation by patch matching", *Proc. Picture Coding Symposium*, Berlin, Germany, Sept. 1997

Papers in Review

1. F. Porikli and Y. Wang "Automatic video object segmentation by 3D volume growing," *IEEE Trans. Circuits Syst. Video Technology*, submitted March 2002.
2. F. Porikli and Y. Wang, "Constrained region extraction of video objects by color masks and MPEG-7 descriptors,", *IEEE International Conference on Multimedia and Expo*, Lausanne, Switzerland, August 2002.
3. F. Porikli, "Automatic threshold determination of centroid-linkage region growing by MPEG-7 dominant color descriptors," *Proc. IEEE Int'l Conf. on Image Processing*, Rochester, New York, Oct. 2002.

Bibliography

[1] T. Aach, A. Kaup, and R. Mester. Change detection in image sequences using gibbs random fields: A bayesian approach. *Proc. Int. Workshop on Intelligent Signal Processing*, 1993.

[2] G. Adiv. Determining the three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI-7:384–401, 1985.

[3] J. K. Aggarwal, L. S. Davis, and W. N. Martin. Corresponding processes in dynamic scene analysis. *Proc. IEEE*, (69):562572, May 1981.

[4] A. Alatan, L. Onural, M. Wollborn, R. Mech, E. Tuncel, and T. Sikora. Image sequence analysis for emerging interactive multimedia services-the european cost 211 framework. *IEEE Trans. Circuits Syst. Video Technol.*, 8:19–31, 1998.

[5] M. Amadasun and R. King. Textural features corresponding to textural properties. *IEEE Trans. Systems, Man, and Cybernetics*, (19):1264–1276, 1989.

[6] H. An and B. Cheng. A kolmogorov-smirnov type statistic with applications to test for normality in time series. *International Statistics Review*, 59:287–307, 1991.

[7] E. Ardizzone, G. Gioiello, M. LaCascia, and D. Molinelli. A real-time neural approach to scene cut detection. *Proc. of IS-T/SPIE - Storage and Retrieval for Image and Video Databases IV*, 1996.

[8] F. Arman, A. Hsu, and M. Chiu. Image processing on compressed data for large video databases. *ACM Multimedia*, pages 267–272, 1993.

[9] A. Azarbayejani, C. Waren, and A. Pentland. Real-time 3d tracking of the human body. *in Proc. IMAGECOM 96, Bordeaux*, May 1996.

[10] B. Bascle, P. Bouthemy, R. Deriche, and F. Meyer. Tracking complex primitives in an image sequence. *in Proc. ICCV94, Jerusalem*, 1994.

[11] A. Baumberg and D. Hogg. An adaptive eigenshape model. *in Proc. British Vision Conf. (BMVC), Birmingham*, Sept. 1995.

[12] J. Bergen and E. Adelson. Visual texture segmentation based on energy measures. *J. Opt. Soc. Am. A*, 3, 1986.

[13] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *J. Roy Statist. Soc. B*, 36(2):192–236, 1974.

[14] L.S. Davis B.J. Schachter and A. Rosenfeld. Some experiments in image segmentation by clustering of local feature values. *Pattern Recognition*, (11):19–28, 1979.

[15] M. J. Black. Combining intensity and motion for incremental segmentation and tracking over long image sequences. *in ECCV92*, page 485493, 1992.

[16] L. Bonsiepen and W. Coy. Stable segmentation using color information. 1991.

[17] A. Bors and I. Pitas. Motion and segmentation prediction in image sequences based on moving object tracking. *IEEE*, pages 663–667, 1998.

[18] P. Bouthemy and E. Francois. Motion segmentation and qualitative dynamic scene analysis from an image sequence. *Int. J. Comput. Vision*, (10):157–187, 1993.

[19] A.C. Bovik, M. Clark, and S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12(1):55–73, 1990.

[20] F. Bremond and M Thonnat. Tracking multiple nonrigid objects in video sequences. *IEEE Trans. Circuit, Syst. Video Technol.*, (8):585–591, Sept. 1998.

[21] P. J. Burt, T. H. Hong, and A. Rosenfeld. Segmentation and estimation of image region properties through cooperative hierarchical computation. *IEEE Trans. Syst., Man, Cybern.*, (SMC-11):802809, 1981.

[22] T. Caelli and D. Reye. On the classification of image regions by colour, texture, and shape. *PR*, 26:461–470, 1993.

[23] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 8:679–698, 1986.

[24] M. Celenk. Colour image segmentation by clustering, 1991.

[25] D. Chai and K. N. Ngan. Face segmentation using skin-color map in videophone applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(4):551–561, 1999.

[26] M. Chang, M. I. Sezan, and A. M. Tekalp. An algorithm for simultaneous motion estimation and scene segmentation. *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'94*, (5):221–224, 1994. Adelaide, Australia.

[27] M. Chang, A. M. Tekalp, and M. I. Sezan. Motion field segmentation using an adaptive map criterion. *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'93*, (5):33–36, 1993. Minneapolis, MN.

[28] P. C. Chen and T. Pavlidis. Image segmentation as an estimation problem. *Comput. Graphics Image Processing*, (13):153172, 1980.

[29] Y. Chen, M. Nixon, and D. Thomas. Statistical geometrical features for texture classiffication. *Pattern Recognition*, (28):537–552, 1995.

[30] J. G. Choi, S. W. Lee, and S. D. Kim. Spatio-temporal video segmentation using a joint similarity measure. *IEEE Trans. Circuits Syst. Video Technol.*, (7):276–286, 1997.

[31] S. Choi, Y. Seo, H. Kim, , and K. Hong. Where are the ball and players? soccer game analysis with color-based tracking and image mosaic. *in Proc. ICIAP97*, 1997.

[32] A.P. Choo, A.J. Maeder, and B. Pham. Image segmentation for complex natural scenes. *IVC*, 8:155–163, 1990.

[33] C.C. Chu and J.K. Aggarwal. Integration of image segmentation maps using region and edge information. *PAMI*, 15(12):1241–1252, December 1993.

[34] J. M. Corridoni and A. Del Bimbo. Automatic video segmentation through editing analysis. *Lecture Notes in Computer Science*, 974:179–190, 1995.

[35] T. Cover and P. Hart. Nearest neighbor pattern classification. 1967.

[36] I. Cox and S. Hingorani. An efficient implementation of reids multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Trans. Pattern Anal. Machine Intell.*, (18), Sept. 1996.

[37] G.Fabri C.S.Regazzoni and G.Vernazza. Advanced video-based surveillance system. *Kluwer Academic Pub.*, 1999.

[38] R. de Queiroz, Z. Fan, and T. Tran. Optimizing blockthresholding segmentation for multilayer compression of compound images. *IEEE Trans. Image Proc*, 2000.

[39] Y. Deng and B. S. Manjunant. Netra-v: Toward an object-based video representation. *IEEE Trans. Circuit, Syst. Video Technol*, (8):616626, 1998.

[40] R. Deriche and O. Faugeras. Tracking line segments. *in ECCV88*, page 259268, 1990.

[41] N. Diehl. Object-oriented motion estimation and segmentation in image sequences. *Signal Processing: Image Commun.*, (3):23–56, 1991.

[42] B. Duc, P. Schtoeter, and J. Bigun. Spatio-temporal robust motion estimation and segmentation. *Proc. 6th Int. Conf. Comput. Anall. Images and Patterns*, pages 238–245, 1995.

[43] F. Dufaux, F. Moscheni, and A. Lippman. Spatio-temporal segmentation based on motion and static segmentation. *Proc. IEEE Int. Conf. Image Processing*, page 306309, 1995.

[44] P. Eichel, E. Delp, K. Koral, and A. Buda. A method for a fully automatic denition of coronary arterial edges from cineangiograms. *IEEE Transactions on Medical Imaging*, 7:313–320, 1988.

[45] F. Ferri and E. Vidal. Colour image segmentation and labeling through multiedit-condensing. *Pattern Recognition Letters*, 13(8):561–568, 1992.

[46] M. Fesharaki and G. Hellestrand. Real-time colour image segmentation. *Technical Report SCSE 9316, School of Computer Science and Engineering, University of New South Wales, Australia*, 1993.

[47] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content. *IEEE Computer*, pages 23–31, 1995.

[48] I. Fogel and D. Sagi. Gabor filters as texture discriminator. *Biological Cybernetics*, (1):103–113, 1989.

[49] A. Laineand J. Fun. Texture classiffication by wavelet packet signatures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (15):1186–1191, 1993.

[50] M. Galloway. Texture analysis using gray level run lengths. *Computer Graphics and Image Processing*, (4):172–179, 1975.

[51] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Machine Intell.*, (PAMI-6):721–741, 1984.

[52] C. Gu and M. C. Lee. Semiautomatic segmentation and tracking of semantic video objects. *IEEE Trans. Circuit, Syst. Video Technol.*, (8):572–583, 1998.

[53] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classiffication. *IEEE Transactions on Systems, Man and Cybernetics*, (SMC-3):610–621, 1973.

[54] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*, pages 453–507. Addison-Wesley, 1992.

[55] B. Heisele, U. Kresel, and W. Ritter. Tracking non-rigid, moving objects based on color cluster flow. *IEEE*, pages 257–260, 1997.

[56] M. Hild, Y. Shirai, and M. Asada. Initial segmentation for knowledge, 1992.

[57] K. Holla. Opponent colors as a 2-dimensional feature within a model of the first stages of the human visual system. *Proc. 6th Int. Conf. on Pattern Recognition, Munich, Germany*, pages 561–563, 1982.

[58] B. Horn and B. Schunck. Determining optical flow. *Artificial Intell.*, page 185203, 1981.

[59] M. Hotter and R. Thoma. Image segmentation based on object oriented mapping parameter estimation. *Signal Processing*, (15-3):315–334, 1988.

[60] P. Huttenlocker and W. Rucklidge. Tracking nonrigid objects in complex scenes. *in Proc. Int. Conf. Computer Vision (ICCV), Berlin*, Sept. 1992.

[61] E. Iwanari and Y. Ariki. Scene clustering and cut detection in moving images by dct components. *Technical Report of IEICE*, (PRU93119):23–30, 1994.

[62] W. B. Thompson J. K. Kearney and D. L. Boley. Optical flow estimation. *IEEE Trans. Pattern Anal. Machine Intell.*, (PAMI9):229–244, Mar. 1987.

[63] A.K. Jain and F. Farrokhnia. Unsupervised texture segmentation using gabor filters. *PR*, 24:1167–1186.

[64] A.K. Jain, M. Murty, and P. Flynn. Data clustering: a review. *ACM Computing Surveys*, (31):264–323, 1999.

[65] D. Jang and H. Choi. Moving object tracking using active models. *IEEE*, pages 648–652, 1998.

[66] S. Ji and H. Park. Image segmentation of color image based on region coherency. *Proc. of ICIP*, 1:80–83, 1998.

[67] S. Ju, M. Black, and A. Jepson. Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, page 307314, 1996.

[68] R. L. Kashyap, R. Chellappa, and A. Khotanzad. Texture classiffication using features derived from random field models. *Pattern Recognition Letters*, (1):43–50, 1982.

[69] D. Koller, K. Danilidis, and H. Nagel. Model-based object tracking in monocular image sequences of road traffic scenes. *Int. J. Computer Vision, vol*, (10):257281, 1993.

[70] M. Kopp and W. Purgathofer. Efficient 3x3 median filter computations. *Technical University, Vienna*, 1994.

[71] M. Kunt, A. Ikonomopoulos, and M. Kocher. Second generation image coding techniques. *Proc. IEEE*, (73):549574, Apr. 1985.

[72] K. I. Laws. *Textured Image Segmentation.* Phd thesis, Faculty of the Graduate School, University of Southern California, 1980.

[73] Y. Lim and K. Park. Image segmentation and approximation through surface type labelling an region merging. *Electronics Letters*, 24(22):1380–1381, 1988.

[74] X. Lin and S. Chen. Color image segmentation using modified hsi system for road following. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1998–2003, 1991.

[75] E. Littmann and H. Ritter. Adaptive color segmentationa comparison of neural and statistical methods. *IEEE Transactions on Neural Networks*, 8(1):175–183, 1997.

[76] L. Lucchese and S. Mitra. Unsupervised segmentation of color mages based on k-means clustering in the chromaticity plane. *Proc. of Content-based access of image and video libraries*, pages 74–78, 1999.

[77] J. Malik and P. Perona. A computational model of texture segmentation. pages 326–332, 1989.

[78] F. Marques and C. Molina. Object tracking for content-based functionalities. *SPIE Visual Commun. Image Processing, VCIP'97*, (3024):190–199, 1997.

[79] B.A. Maxwell and S.A. Shafer. A framework for segmentation using physical models of image formation. *IEEE Computer Vision and Pattern Recognition*, 1994.

[80] R. Mech and M. Wollborn. A noise robust method for segmentation of moving objects. *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'97*, (4):2657–2660, 1997.

[81] R. Mech and M. Wollborn. A noise robust method for 2d shape estimation of moving objects in video sequences considering a moving camera. *Signal Processing*, 66:203–217, 1998.

[82] T. Meier and K. N. Ngan. Automatic segmentation based on hausdorff object tracking. *ISO/EIC JTC1/SC29/WG11 MPEG97/m2238- Stockholm*, 1997.

[83] T. Meier and K. N. Ngan. Automatic segmentation of moving objects for video object plane generation. *IEEE Trans. Circuit, Syst. Video Technol*, (8):525–538, 1998.

[84] J. Meng and S. Chang. Tools for compressed-domain video indexing and editing. *SPIE Proceedings*, 2670:180–191, 1996.

[85] F. Meyer. Color image segmentation. *4th International Conference on Image Processing and its Applications, Maastricht, The Netherlands*, pages 303–304, 1992.

[86] F. Meyer and S. Beucher. Morphological segmentation. *J. Visual Commun. Image Representation*, (1):21–46, 1990.

[87] F. Meyer and P. Bouthemy. Region-based tracking using affine motion models in long image sequences. *CVGIP: Image Understanding*, (60):119140, 1994.

[88] M.Ghanbari. The cross search algorithm for motion estimation. *IEEE Transactions on Communications*, (38):950–953, 1990.

[89] F. Moscheni, S. Bhattacharjee, and M. Kunt. Spatiotemporal segmentation based on region merging. *IEEE Trans. on PAMI*, 20(9):897–915, 1998.

[90] D. W. Murray and B. F. Buxton. Scene segmentation from visual motion using global optimization. *IEEE Trans. Pattern Anal. Machine Intell.*, (PAMI-9):220–228, 1987.

[91] H. G. Musmann, M. Hotter, and J. Ostermann. Object-oriented analysis-synthesis coding of moving images. *Signal Processing: Image Commun.*, (1):117–138, 1989.

[92] A. Nagasaka and Y. Tanaka. Automatic scene-change detection method for video works. *Proc. 40th National Con. Information Processing Society of Japan*, 1990.

[93] A. Neri, S. Colonnese, G. Russo, and P. Talone. Automatic moving object and background separation. *Signal Processing*, (66):219–232, 1998.

[94] R. Ohlander, K. Price, and D.R. Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, (8):313–333, 1978.

[95] Y. Ohta. A region-oriented image-analysis system by computer. *Doctoral Dissertation*, (Kyoto University, Japan), 1980.

[96] Y. Ohta, T. Kanade, and T. Sakai. Color information for region segmentation. *Computer Graphics and Image Processing*, (13):222–241, 1980.

[97] O. Pichler, A. Teuner, and B. Hosticka. A comparison of texture feature extraction using adaptive gabor filtering. *Pattern Recognition*, 29(5):733–742, 1996.

[98] A. Pikaz and A. Averbuch. An effcient topological characterization of gray-level textures, using a multiresolution representation. *Graphical Models and Image Processing*, (59):1–17, 1997.

[99] L. Priese and V. Rehrmann. A fast hybrid color segmentation method. *Proceedings 15th DAGM Symposium*, pages 297–304, 1993.

[100] L. Qui and L. Li. Contour extraction of moving objects. *IEEE*, pages 1427–1431, 1998.

[101] T. Reed and J. M. Buf. A review of recent texture segmentation and feature extraction techniques. *CVGIP: Image Understanding*, (57):359–372, 1993.

[102] M. Sabin. Global convergence and empirical consistency of the generalized lloyd algorithm. *PhD thesis, Stanford University*, 1984.

[103] D. Sagi and S. Hochstein. Lateral inhibition between spatially adjacent spatial-frequency channels. *Perception and Psychophysics*, 37(4):315–322, 1985.

[104] P. Salembier, P. Brigger, J. R. Casas, and M. Pardas. Morphological operators for image and video compression. *IEEE. Trans. Image Processing*, (5):881–898, 1996.

[105] P. Salembier and M. Pardas. Hierarchical morphological segmentation for image sequence coding. *IEEE Trans. Image Processing*, (3):639–651, 1994.

[106] R. Schettini. A segmentation algorithm for color images. *PRL*, 14:499–506, 1993.

[107] I. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Trans. Pattern Anal. Machine Intell.*, (PAMI-9):5673, 1987.

[108] S. Siggelkow, R. Grigat, and A. Ibenthal. Segmentation of image sequences for object oriented coding. *IEEE*, pages 477–480, 1996.

[109] Wladyslaw Skarbek and Andreas Koschan. Colour image segmentation: A survey. Technical report, Institute for Technical Informatics, Technical University of Berlin, October 1994.

[110] G. Sperling and B. A. Dosher. *Handbook of perception and human performance. Vol I, K. Boff, L. Kaufman, J. Thomas*, pages 2–65. Wiley Interscience Publication Wiley, NY, 1986.

[111] H.D. Stein and W. Reimers. *Signal processing II: theories and applications*, pages 271–273. Elsevier Science Publ., North-Holland, 1983.

[112] C. Stiller. A statistical image model for motion estimation. *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'93*, (5):193–196, 1993. Minneapolis, MN.

[113] C. Stiller. Object-based estimation of dense motion fields. *IEEE Trans. on Image Processing*, (6):234–250, 1997.

[114] O. Sukmarg and K. Rao. Fast algorithm to detect and segmentation in mpeg compressed domain. *IEEE TENCON 2000, Kuala Lumpur, Malaysia,*, Sept. 2000.

[115] C. Sun and W. G. Wee. Neighboring gray level dependence matrix for texture classiffication. *Computer Vision, Graphics, and Image Processing*, (23):341–352, 1983.

[116] R. Taylor and P. Lewis. Color image segmentation using boundary relaxation. *Proceedings. 11th International Conference on Pattern Recognition*, III:721–724, 1992.

[117] W. B. Thompson and T. G. Pong. Detecting moving objects. *Int. J. Comput. Vision*, (4):3957, 1990.

[118] S. Tominaga. A colour classification method for color images using a uniform color space. *Proc. 10th. Int. Conf. on Pattern Recognition, Atlantic City, New Jersey*, I:803–807, 1990.

[119] L. Torres, D. Garc'ia, and A. Mates. A robust motion estimation and segmentation approach to represent moving images with layers. *In IEEE Intl. Conf. Acoust. Speech Sig. Proc.*, pages 2981–2984, 1997.

[120] D. Tsen and C. Chang. Color segmentation using perceptual attributes. *Proc. 11th IAPR Int. Conf. on Pattern Recognition*, pages 228–231, 1992.

[121] S.E. Umbaugh, R.H. Moss, W.V. Stoecker, and G.A. Hance. Automatic colour segmentation algorithms with application to skin tumor feature identification. *IEEE Engineering in Medicine and Biology*, 12(3):75–82, 1993.

[122] M. Unser. Sum and difference histograms for texture analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, (8):118–125, 1986.

[123] L. Vincent and P. Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *Pattern Analysis and Machine Intelligence*, 13(6), 1991.

[124] T. Vlachos and A. Constantinidies. Graph-theoretical approach to colour picture segmentation and contour classification. *IEE Proceedings*, I(140):36–45, 1993.

[125] H. Voorhees and T. Poggio. Detecting blobs and textons in natural images. *Proc. DARPA Image Understanding Workshop, Los Angeles, CA*, pages 892–899, 1994.

[126] H. Wang and M. Brady. Real-time corner detection algorithm for motion estimation. *Image Vision Comput.*, (13):695703, 1995.

[127] H. Wang and S. Chang. Automatic face region detection in mpeg video sequences. *Electronic Imaging and Multimedia Systems, SPIE Photonics China*, 1996.

[128] J. Wang and E. Adelson. Representing moving images with layers. *IEEE Trans. Image Processing*, (3):625–638, 1994.

[129] J. Wang and E. Adelson. Spatio-temporal segmentation of video data. *Proc. SPIE*, (2182):120131, 1996.

[130] T. Westman, D.A. Harwood, T. Laitinen, and M. Pietikainen. Color segmentation by hierarchical connected components analysis with image enhancements by symmetric neighborhood filters. In *ICPR90*, pages Vol–I 796–802, 1990.

[131] G. Wu and T. Reed. Image sequence processing using spatiotemporal segmentation. *IEEE Trans. on circuits and systems for video technology*, 9:798–807, 1999.

[132] B. Yeo and B. Liu. Rapid scene change detection on compressed video. *IEEE Transactions on Circuits and Systems for Video Technology*, 5:533–544, 1995.

[133] H. Zhang, C. Low, and S. Smoliar. Video parsing and browsing using compressed data. *Multimedia Tools and Applications*, 1(1):89–111, 1995.

[134] Z. Zhang. Token tracking in a cluttered scene. *INRIA, Sophia Antipolis*, (Res. Rep. 2072), Oct. 1993.