DUBLIN CITY UNIVERSITY
SCHOOL OF ELECTRONIC ENGINEERING

Video Object Segmentation for Future Multimedia Applications

by

Noel Edward O'Connor B. Eng.

A thesis submitted in partial fulfilment of the requirements for the degree of PhD in Electronic Engineering

July 1998

Supervisor: Dr. S. Marlow

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of PhD in Electronic Engineering is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: Noel Edward O'Connor I.D. No.: 92700144

Date: 25/10/98

Acknowledgements

I would like to extend my heartfelt gratitude to Dr. Sean Marlow for his excellent and supportive supervision during the course of this work. Similar thanks are due to Dr. Noel Murphy, the entire Video Coding Group (past and present), and the other denizens of J119.

I would also like to thank everyone else who has put up with me over the last number of years. I promise I'll find something else to talk/whinge about in the pub (enter singing coins stage left). Special honourable mentions to my parents *("I take the cue from certain people I know")*, Emma *("we have been through hell and high tide, and yet I can surely rely on you")*, and of course Seamus and Donnacha (the last of the famous international brick-layers).

Thanks

"There's more to life than books you know, but not much more"

S. P. Morrissey

ABSTRACT

Video Object Segmentation for Future Multimedia Applications

by Noel Edward O'Connor B. Eng.

An efficient representation of two-dimensional visual objects is specified by an emerging audiovisual compression standard known as MPEG-4. It incorporates the advantages of segmentation-based video compression (whereby objects are encoded independently, facilitating content-based functionalities), and also the advantages of more traditional block-based approaches (such as low delay and compression efficiency). What is not specified, however, is the method of extracting semantic objects from a scene corresponding to a video segmentation task. An accurate, robust and flexible solution to this is essential to enable the future multimedia applications possible with MPEG-4.

Two categories of video segmentation approaches can be identified: supervised and unsupervised. A representative set of unsupervised approaches is discussed. These approaches are found to be suitable for real-time MPEG-4 applications. However, they are not suitable for off-line applications which require very accurate segmentations of entire semantic objects. This is because an automatic segmentation process cannot solve the ill-posed problem of extracting semantic meaning from a scene.

Supervised segmentation incorporates user interaction so that semantic objects in a scene can be defined. A representative set of supervised approaches with greater or lesser degrees of interaction is discussed. Three new approaches to the problem, each more sophisticated than the last, are presented by the author. The most sophisticated is an object-based approach in which an automatic segmentation and tracking algorithm is used to perform a segmentation of a scene in terms of the semantic objects defined by the user. The approach relies on maximum likelihood estimation of the parameters of mixtures of multimodal multivariate probability distribution functions. The approach is an enhanced and modified version of an existing approach yielding more sophisticated object modelling. The segmentation results obtained are comparable to those of existing approaches and in many cases better. It is concluded that the author's approach is ideal as a content extraction tool for future off-line MPEG-4 applications.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Context of this research

The proliferation of personal computers in recent years, in conjunction with the continuous evolution of digital networks, has ensured a worldwide trend towards a new digital age. Nowhere is this more prevalent than in the area of telecommunications and multimedia. The introduction of the telephone revolutionised the world by allowing instant voice communication. A similar revolution is already underway with the expansion of digital communication services to include audio, visual and data-based information. More and more in the business world, video conferencing is being used as a viable alternative to extensive travelling in order to meet with colleagues or clients around the globe. The videophone, whilst only a novelty at the moment, is likely to become more widely adopted by the general public in the future. After all, no science fiction movie is complete without the pre-requisite videophone in every day use! Similarly, the television broadcast, film production and graphic design communities, in their never-ending quest for an enhanced reflection of reality, are turning to digital multimedia technologies as a means of meeting their requirements.

The arrival of this new digital age has brought with it many technological challenges. Multimedia data must be represented in a manner suitable for meeting the requirements placed upon it. It must be compressed to minimise storage costs and to ensure that it can be transmitted at the data rates possible on existing networks. It needs to be protected against corruption. It should be processed in such a way as to facilitate reuse. In certain cases it must be encrypted. Sometimes, in order to protect ownership rights and for other legal considerations, it is necessary to incorporate non-removable qualities in the data. Digital visual information (i.e. still images and video sequences) in particular presents significant challenges, consisting as it does of large volumes of information to be processed in order to meet these requirements.

Aspects of video compression are considered in this thesis. Technological advances in this field in recent years have presented new challenges. After addressing issues of compression of digital video for user requirements such as efficient storage and transmission, it has become apparent to the research community that given these possibilities, further enhanced functionalities are required. It is no longer enough for the average person, either in the home or in the workplace, to be a passive observer of visual information: the visual data must be efficiently represented in such a way as to facilitate interaction. Similarly, it is no longer enough for people involved in producing the visual information (from film producers to graphic artists) to be capable of editing and manipulating content on the traditional (unwieldy) image by image basis. Rather they require access to, and interaction with, the *content* of the digitised scene in order to improve it, modify it, or construct a completely new scene. For these reasons, the envisaged enhanced functionalities are usually termed *content-based functionalities*.

In order to allow access to the content of visual data, the research community has turned its attention to the analysis of a digitised scene (either still image or moving video) into its component parts, how ever these are defined. This process is usually termed *segmentation* as the visual data is segmented into units smaller than images, which are not geometrically regular partitions of an image, and which reflect image content. Compression of the image or sequence in terms of these units offers a means of providing interactive content-based functionalities. The discussion in this thesis focuses on this important issue of segmentation.

1.2 Objectives of this research

The first objective of this thesis is to place in context the further investigations described here. To achieve this objective, various methods of video compression for existing user applications are first described. The content-based functionalities required by future multimedia applications are then introduced. The actual compression approach converged upon for potentially providing these functionalities is examined. This approach does not completely solve the problem of providing the required functionalities. Rather, it solves part of the problem and leaves the other part, a video segmentation problem, as an open issue. This serves to emphasise the importance of

video segmentation as an enabling technology for future multimedia applications and justifies the research carried out by the author in this area.

The second objective of this thesis is to describe the current state of the art technology being used to address the issue of segmentation. Segmentation may be addressed in different ways depending on the nature of the user application which requires content-based functionalities. In this work, the approaches are divided into two categories corresponding to on-line and off-line segmentation. Existing segmentation approaches which can obtain a measure of success for both scenarios are described. Some of these approaches help form the basis of a new approach to segmentation (see below) whilst others are used for comparison and evaluation of this new approach.

The third objective of this thesis is to investigate a new approach to solving the segmentation problem considering the off-line scenario. Three schemes, each more sophisticated than the last, are developed by the author in order to address this issue. The mathematical justification of the technologies investigated in these approaches is presented, as well as results to illustrate the nature of the approaches and their performance. In each case, the performance of the approach is analysed.

An implicit objective of any research work is to indicate a direction for further research. The final objective of this thesis is to consider the proposed solution to the segmentation problem and to present possibilities for improvement. The possibility of using the solution as a basis for addressing related (future) challenges in the field is also presented.

1.3 Structure of this thesis

The first half of this document is a review of existing technologies. In chapter two, the block-based approach to video compression is described. The underlying philosophy of redundancy reduction for compression is first explained. The compression tools developed based on this philosophy are then described. Finally, the organisation of these tools into complete block-based encoding/decoding schemes is described. The schemes described are H.261, H.263, MPEG-1 and MPEG-2, which are all international standards for video compression. It is also explained in this chapter how a block-based

approach can lead to visually disturbing compression artefacts, particularly at lower bit rates.

In the third chapter, an alternative approach to compression based on video segmentation is described. It is explained how this approach was initially investigated as a means of avoiding block-based artefacts. It is further explained how content-based functionalities are feasible when segmentation is included in a compression scheme. Two of the best known (in Europe at least) compression schemes embodying this approach, known as SIMOC and MORPHECO, are described and discussed. It is explained that neither approach is capable of supporting the envisaged content-based functionalities, since they are both tightly-coupled analysis-coding systems, which do not produce real semantic object segmentations.

The new audiovisual (AV) compression standard, known as MPEG-4, which supports content-based functionalities by block-wise encoding arbitrarily-shaped image segments corresponding to semantic objects, is presented in chapter four. This completes the description of the evolution of video compression technology from purely block-based approaches, through purely segmentation-based approaches, to finally, a hybrid segmentation-based and block-based approach. The content-based functionalities which the MPEG-4 standard supports are outlined and possible applications are discussed. The decision that the MPEG-4 standard should not standardise segmentation is justified. The consequence of this, namely the importance of video segmentation for future MPEG-4 applications, is high-lighted. The future work of the MPEG standards group, known as MPEG-7, is also briefly described and the potential importance of segmentation as a feature extraction tool in this context is proposed.

Current state of the art solutions to the segmentation problem in the context of MPEG-4 applications are described in chapters five and six. Chapter five describes three existing approaches to *unsupervised* segmentation. The performance of these techniques in terms of potential MPEG-4 applications is analysed. It is explained how all three approaches are restricted by the fundamental inability of automatic segmentation techniques to extract semantic meaning from a scene. It is concluded that these approaches are only

suitable for a subset of possible MPEG-4 applications, corresponding to on-line real-time MPEG-4 applications which can tolerate inaccurate object segmentations.

Chapter six describes three existing approaches to *supervised* segmentation which are suitable for off-line, non real-time MPEG-4 applications. Examples of applications possible with this approach to segmentation are presented. Again, the performance (amongst other considerations) of these approaches is considered. Whilst the segmentation results obtainable with each scheme are comparable, it is proposed that they can be distinguished on the basis of user interaction. One of the approaches in particular can obtain accurate segmentation results with a minimum amount of interaction, which is easily performed. This approach forms the basis of the author's further investigations which attempt to address certain limitations associated with the scheme.

The second half of the document details the further investigations carried out by the author. The mathematical background necessary for the author's solution to the supervised segmentation challenge is presented in chapter seven. The chapter introduces multidimensional statistical signal processing and outlines key techniques in this field such as Maximum Likelihood (ML) estimation and the Expectation-Maximisation (EM) algorithm. Existing applications of these techniques to the segmentation problem (and the promising segmentation results obtained) are discussed. It is concluded that these techniques can be very useful tools for segmentation.

The method of user interaction found to be most suitable in chapter six, is incorporated into two supervised segmentation schemes developed by the author, which are presented in chapter eight. The first is a rudimentary region-based scheme based on clustering. The second is an enhanced region-based scheme based on the statistical processing techniques of chapter seven. It is explained how these schemes can be used to perform object segmentation for off-line MPEG-4 applications. However, these schemes are not investigated in full as they include a number of drawbacks, such as inaccurate object segmentation and problematic object tracking, which hamper their development. Segmentation results obtained with these schemes are presented to illustrate these claims.

The method of object modelling which proved to be successful in one of the approaches of chapter six, is incorporated into a supervised segmentation scheme developed by the author, which is presented in chapter nine. This object model is used to enhance the second approach presented in chapter eight. The result is a modified enhanced version of the scheme described in chapter six which yields more appropriate object models. The new approach addresses the limitations of the schemes presented in chapter eight by automatically segmenting and tracking actual semantic objects. It also addresses the limitations of the similar approach described in chapter six. The performance of the scheme is discussed, and the limitations of the approach in terms of tracking objects are explained. Possible ways of addressing these limitations are then suggested. It is concluded that the approach is an ideal candidate for segmentation in future off-line MPEG-4 applications.

The final chapter reviews the conclusions drawn in this thesis and suggests directions for future research considering the approach to segmentation described in chapter nine. Possible high level enhancements of the scheme when considered in a complete MPEG-4 video object segmentation application are proposed. The potential use of the object models used in the author's approach, in order to extract descriptions of MPEG-7 features, is also discussed.

All segmentation results in chapters eight and nine are obtained using MPEG-4 test sequences in QCIF or QSIF format. The presented images, however, are scaled either up or down in order that the results may be arranged appropriately in the document. A glossary of commonly used abbreviations and terms is provided at the end of this document for the reader's convenience. A list of related publications which are referenced in the main text of the document is also provided. The appendices contain information useful in understanding the content of the document, including ancillary related information not contained in the main text. Appendix A includes a derivation necessary for deriving key equations in chapter seven. Also included are two examples of the derivations in chapter seven in their simplest form. Appendix B contains segmentation results to illustrate that the segmentation process of chapter nine can perform equally well for different image resolutions or formats.

2. BLOCK-BASED VIDEO COMPRESSION

2.1 Introduction

In this chapter, the traditional approaches to digital image/video compression are briefly reviewed. At the highest level, image compression can be considered to take place in two steps. In the first step the spatial redundancy in the image is reduced. Redundancy in this context refers to those image components to which the human visual system is least sensitive, and which may be successfully removed without causing a significant deterioration in subjective image quality. The second step is to efficiently represent in a bitstream the entropy of the components not removed. When considering video, and the significant amount of temporal redundancy present in most sequences, a prior temporal redundancy reduction step is required. The high level steps to be followed in this case are (i) to reduce temporal redundancy between images, (ii) to reduce the spatial redundancy of the result, and (iii) to form an efficient bitstream representation of the entropy of the remaining image data.

Most image and video compression schemes operate on blocks of pixels as opposed to the entire image. In this way, each block can be treated independently. This is important if the scheme is to be used in a real-time application (i.e. a decoder can start to decode blocks as soon as they become available). Also, the transmission error characteristics can be improved by treating blocks independently (i.e. an error in decoding one block will be localised as it cannot propagate into other blocks)[1].

The following sections outline the main redundancy reduction tools used in image/video compression. The chapter focuses on the application of these tools in block-based compression schemes. These tools have been successfully integrated into a number of international standards and these standards and their target applications are described. These descriptions (and indeed the descriptions of other compression schemes

[1] Error robustness and the possibility of error detection and concealment are important considerations for nearly all applications involving compressed visual data.

throughout this thesis) focus on redundancy reduction and not the method of entropy encoding.

2.2 Transform-based Coding

In the transform-based approach to compression, the image signal is mapped from one domain (normally spatial or temporal) into another domain, referred to as the transform domain. In the case of an orthogonal transform, the transform operation is unique and reversible [1]. Furthermore, the energy of the signal is preserved in the mapping between domains and thus, the original signal can be completely recovered by the inverse transformation. It is not the transformation itself which results in compression. Rather, compression is achieved by processing the signal in the transform domain. In the case of a digital image, the data to be transformed consists of a set of pixels. The transformed pixels in the new domain are referred to as coefficients as the transform can be considered to be a series expansion of the original signal using a new set of basis vectors [1].

An important property of orthogonal transforms is decorrelation. If an input signal is highly correlated, a properly chosen orthogonal transform can reduce (even eliminate) correlation [1]. This is an important property of orthogonal transforms when they are considered for image/video compression purposes as it means that each coefficient in the transform domain can be treated independently.

Another very important property of orthogonal transforms when applied to correlated signals is energy compaction [1]. This means that, in the transform domain, a large percentage of the overall signal energy is concentrated in a small number of the transform coefficients. Compression can be achieved by filtering these coefficients in the transform domain to remove those which can be considered to be less important to the overall signal representation. In the following section, the Discrete Cosine Transform (DCT), which is the transform most often used in image/video compression, is described. As illustrated in the remainder of the chapter, it is at the heart of the most popular image/video compression schemes.

2.2.1 DCT-based Coding

The DCT has several properties which make it very attractive for use in the field of image/video compression. In particular, the DCT compacts the energy of the signal into a few coefficients, it has a fast implementation (both forward and inverse transformations), there is minimum residual correlation, and it is real-valued and separable. In addition, the DCT is a close approximation to the statistically optimal Karhunen-Loève Transform (KLT) [1].

The coefficients of a 2-D DCT of a block of pixels have a special interpretation which is exploited in image/video compression. The basis functions of the DCT are arranged in order of increasing vertical and horizontal spatial frequency. The top left region in a transformed block represents the DC and low frequency coefficients. Due to energy compaction these contain most of the signal information and are the most visually important. The coefficients located in horizontal and vertical positions represent increasing horizontal and vertical frequencies (referred to as AC coefficients).

In order to encode an image using the DCT, the normal procedure is divide the image to be encoded into 8×8 non-overlapping blocks of pixels and to encode each block independently [2]. Each block undergoes a 2-D DCT. The resultant coefficients are then quantized in order to reduce entropy and to aid in the elimination of any small and unimportant coefficients [2]. In order to exploit the characteristics of the resultant coefficients, they are treated in a particular order. The most popular order is the zig-zag scan [2]. This results in quantized coefficients in order of increasing frequency, starting with the DC coefficient. This is desirable as large numbers of the higher order coefficients along this scan will be zero after quantization.

The organisation of coefficients described above is suitable for Run Length Encoding (RLE) where the length of each run (i.e. a number of similar values in a row in the scan) of a particular quantization level are the coding events. These are represented using pre-determined variable length code-words (VLCs). Shorter VLCs are used for more frequently occurring runs (such as runs of zero) thereby achieving compression. For further compression efficiency, a special end of block VLC can be transmitted at any point indicating that all the remaining coefficients in the block are zero.

The DCT-based approach to image compression has proven to be very robust and efficient. In fact the Joint Photographic Experts Group (JPEG) of the International Standards Organisation (ISO) has specified a DCT-based still image compression standard (normally referred to simply as JPEG). This standard utilises the above approach in order to specify a number of coding modes for still images [2].

2.3 Hybrid (Transform-DPCM) Coding

Redundancy can also be removed by a technique known as predictive coding. In this approach, each pixel in an image is predicted by pixels in its local neighbourhood and the prediction error is transmitted. The prediction error has lower entropy than the original pixel sample and can thus be quantized with fewer quantization levels, giving data compression.

The most popular form of predictive coding is known as differential pulse code modulation (DPCM). This approach utilises a feed-back loop in the encoding process (referred to as the coding loop). The prediction of each pixel in the encoder is made with respect to a previously encoded and reconstructed pixel, thus avoiding error propagation and allowing the decoder to mimic exactly the operation of the encoder. It is therefore necessary for the encoder to perform decoding and to store reconstructed pixels in order to make the prediction.

DPCM can be used to reduce the temporal redundancy present between successive images in most video sequences. As in DCT-based coding, DPCM operates in a block-wise manner. In this case, the prediction of a block is extracted from the previous reconstructed image. The prediction error between the original block and its prediction can then undergo spatial redundancy techniques in order to achieve further compression. DCT-based encoding (with VLCs tailored to prediction error data) is normally used and the overall approach is a hybrid transform-DPCM compression scheme.

2.3.1 Motion Estimation and Compensation

The method of obtaining the prediction for a block of pixels is known as *motion estimation* (ME) as the underlying objective is to calculate the motion of a block from one image to the next. This proceeds by moving a block around its co-ordinates in the previous reconstructed image in order to determine how the block has moved from one image to the next[2]. What is actually sought is the location which produces the best match for the block. A number of methods exist for determining the best match [3], but the one most often used is the mean absolute distance (MAD). A number of algorithms exist for fast searches which attempt to converge quickly on the best match [3]. Having found the best match, the result is a *motion vector* for the block which specifies the displacement in both vertical and horizontal components.

Given the motion vector, the prediction is obtained by simply copying the contents of the displaced block in the previous reconstructed image into the current block location. This process is known as *motion compensation* (MC). The original block and the MC block are subtracted and the result is a prediction error block which can undergo DCT-based encoding. Both ME and MC are required in an encoder, however the decoder only requires a MC module as the motion vectors are transmitted. Both encoder and decoder require a memory for the previous encoded image (referred to as a *frame memory*).

2.4 Block-based Video Compression Standards

The video compression tools described in the previous sections have undergone substantial research and subsequent development to international standardisation status. The way the tools are employed in a standardised compression scheme is very much dependent on the nature of the applications which motivated the development of the standard in the first place. In the following sections the main video compression standards are briefly described and compared.

---

[2] This assumes that all pixels in the block move with coherent translational motion. The effects of other more complicated motions such as rotation are ignored.

2.4.1 ITU-T H.261 Video Compression

In response to the growing demand for visual telephony and video conferencing services, the International Telecommunications Union (ITU-T) developed video compression standards targeted at this narrow bandwidth real-time application. Recommendation H.261 is a video codec specification suitable for real-time AV services at $p \times 64$ Kb/sec, where p is the range of 1 to 30 [4]. This choice of bandwidth is motivated by the bit-rate of baseline ISDN (Integrated Services Digital Network). The overall codec has a hybrid structure incorporating both transform and predictive coding. The algorithm creates two types of frames. The first, known as an INTRA frame (I-frame), is encoded independently using the DCT. The second type of frame, known as an INTER frame (P-frame), is encoded with reference to a previous I or P-frame. A simplified diagram of the codec structure is illustrated in Figure 2.1 [4].

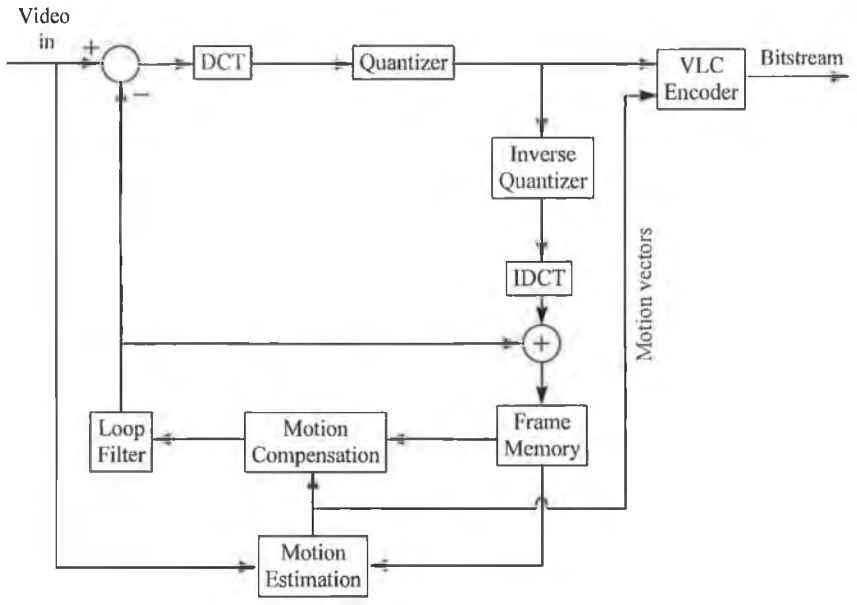


Figure 2.1 - Structure of a H.261/H.263 encoder

An image to be encoded is divided into 16×16 non-overlapping blocks of pixels. The coding unit is referred to as a *macroblock*. Each macroblock is encoded independently. A macroblock has four coding mode decisions associated with it corresponding to INTRA or INTER, MC or no MC, loop filtering or not, and coded or not-coded. Information on these four modes must be transmitted for each macroblock as overhead. Each macroblock is decoded and reconstructed in the encoder. In INTER mode, the output of the inverse transform is the prediction error which is added to the prediction

obtained by motion compensation. In INTRA mode the output is the reconstructed macroblock. The decoded macroblocks are aggregated to form the previous reconstructed image which is stored in the frame memory for encoding the next image.

A two-dimensional 8×8 DCT is used for spatial data compression. In the case of an INTER mode decision, the input data to this DCT is the prediction error. In the case of INTRA mode, the input data are the pixel values themselves. DCT coefficients are quantized, zig-zag scanned and entropy encoded. The INTRA DC coefficient is encoded using a fixed length code-word (FLC). There are 31 quantizers available for the AC and INTER DC coefficients. The quantized coefficients are encoded by VLCs using RLC [2].

H.261 specifies one motion vector per macroblock. The motion vector data is predictively encoded using VLCs for the differences between the horizontal and vertical motion vector components [2]. Motion compensation in the encoder is optional in H.261 (although the decoder must accept one motion vector per macroblock).

An optional low pass filter may be introduced into the codec after MC. The motivation for this is to reduce the quantization noise in the feedback loop and/or to reduce the effects of high frequency artefacts introduced by MC [2]. No filter coefficients are defined but a separable filter, similar to that used in JPEG, is recommended.

Selected processing steps and parameters of the H.261 compression scheme are not specified. These are normally parts of the compression algorithm which are non-normative and yet have a significant impact on overall image quality and compression efficiency. This provides a suitable forum for competition of H.261 products in the market-place. For example, the method of motion estimation in the encoder, the use of a loop filter, the arithmetic process for computing the DCT, the control of the data rate, and the method of mode decision for each macroblock can be defined by the encoder designer.

2.4.2 ITU-T H.263 Video Compression

In addition to H.261, the ITU-T has produced another standard for video compression, developed for very low bit-rate applications such as video telephony over the public services telephone network (PSTN) and mobile telephony. This recommendation, known as H.263, is very similar in concept to H.261, although with significant improvements which allow the video codec to operate efficiently at bit-rates lower than 64 Kb/sec [2]. Like H.261, the overall codec structure is a hybrid DPCM/DCT encoding system. The simplified diagram of the H.261 encoder in Figure 2.1 translates directly to H.263. In the following, the main differences between the two are outlined.

In H.263, motion compensation with half-pixel accurate motion vectors is allowed. Motion compensation is performed using bilinear interpolation. Also, motion vectors are allowed to point outside the current image. This latter is called *unrestricted motion vector mode*. In this case, pixels at the edge of the image are used to form the prediction [2]. Another motion compensation method known as *advanced prediction mode* is also available. In this case, each macroblock can have four motion vectors associated with it. The prediction is formed using a technique known as overlapped block motion compensation (OBMC) [2].

An optional encoding mode known as syntax-based arithmetic coding (SAC) mode is available in H.263. In this mode, all VLC operations of H.263 are replaced with more efficient arithmetic encoding/decoding operations [2]. An optional extra type of encoded frame, known as a PB-frame is also available. This is actually two separate frames. The first is a normal P-frame, predicted from the previous P-frame. The second is a B-frame (where B stands for "bi-directional") which is predicted from both the previous P-frame and current P-frame being decoded [2].

2.4.3 MPEG-1: Audiovisual compression for digital storage media

The Motion Picture Experts Group (MPEG) was established by ISO in order to specify standards for AV compression. MPEG-1 is a standard supporting compression of AV data at rates of around 1.5 Mb/sec [2]. This data rate is motivated by the transfer rate of a CD-ROM which, due its low price and ease of production, has become widely

accepted as a digital storage medium for AV information. The objective of the MPEG-1 video compression standard is to specify a highly flexible source coding algorithm which is applicable to a wide range of digital storage media (DSM) applications. A very large number of compression options are specified in the encoding process. MPEG-1 is specified so that the decoder is much simpler than the encoder. A simplified decoder ensures that low cost, mass market implementations can be produced - a necessary requirement for the type of digital storage applications at which the standard is targeted.

The structure of MPEG-1 is based very much on H.261 with some extra extensions for digital storage applications. H.261 is targeted at video conferencing applications which normally contain a small amount of motion and in which reasonably low visual quality is acceptable. MPEG-1 on the other hand, requires that every frame of a sequence (which may have fast motion) be encoded to a high degree of quality (comparable to that of a video cassette recorder). This is not possible with H.261, even when operating in the mid-range of its p x 64 Kb/sec bandwidth [2]. Furthermore, MPEG-1 supports extra functionalities necessary for its target applications, such as random access (i.e. fast access to an arbitrary frame), high resolution still frame and variable mode video playback (forward, reverse, normal speed, fast speed).

MPEG-1 specifies three types of coded images [2]. The first is referred to as an I-picture and this is analogous to the I-frame of H.261. The second type of coded image is known as a P-picture which is analogous to the P-frame in H.261. A B-picture is encoded using both a past frame and a future frame in the sequence [2].

As in H.261/H.263, the basic coding unit in MPEG-1 is the macroblock. In an I-picture each macroblock in encoded in INTRA mode. Each 8×8 block in the macroblock is transformed into the DCT domain and the coefficients undergo quantization, zig-zag scanning and entropy encoding. The quantization takes place with reference to a user-defined quantization matrix which specifies the quantizer to be used for each coefficient (except the DC coefficient) [2]. The user-defined matrix ensures that quantization can be tailored to a particular application. The quantized DC coefficients undergo lossless encoding using a DPCM technique [2]. The quantized AC coefficients are not predicted,

but undergo zig-zag scanning and entropy encoding using RLC, in a manner very similar to H.261.

The macroblocks in a P-picture are encoded using motion compensation and DCT-based coding of the prediction error. Four coding modes exist for each macroblock. An MC/no MC decision can be made to indicate that no motion vectors are to be encoded for the macroblock. Motion vectors are encoded differentially with VLCs in a manner similar to that used in H.261. A macroblock in a P-picture can be encoded in INTRA or INTER mode depending on which gives the most efficient compression. In fact, a macroblock need not be encoded at all if the quantized DCT coefficients are zero and this is indicated with a coded/not coded mode decision. Finally, a quant/no quant decision is made which indicates whether or not the quantizer scale is to be changed in the macroblock. Visual image quality can be improved by varying the quantizer to reflect scene activity [2].

Decoded B-pictures need not be stored as they are not used for subsequent prediction. In order to encode a macroblock in a B-picture, three predictions of the macroblock corresponding to forward, backward and bi-directional are available. There are two types of motion vector to be encoded (forward and backward) and each is encoded with reference to a predictor of similar type [2]. Quantization and encoding of the DCT coefficients in a B-picture macroblock are performed in the same manner as in P-pictures.

2.4.4 MPEG-2: Generic Video Compression

The second standardisation work item of the ISO Motion Picture Experts Group is normally referred to as MPEG-2. It is a compression standard developed to support video transmission at bit-rates in the range 2 to 15 Mb/sec [2]. MPEG-2 is designed for applications such as DSM, electronic cinema, satellite broadcast, high definition television (HDTV) supporting existing television, multimedia, computer games, etc. The standard is very flexible, allowing specification of video compression for many applications between the high performance (or complexity) and low performance (or complexity) extremes. Because of this, the standard is designed to support a very wide range of bit-rates, spatial and temporal resolutions, qualities and services.

It is not practical to support all application options in a single bitstream syntax, and so a number of bitstream syntax subsets are defined and are referred to as *profiles* and *levels*. As a result, MPEG-2 is often considered as a generic tool-box for video compression. The basic video compression algorithm for MPEG-2 is very similar to that of MPEG-1. It also consists of INTRA frame compression and INTER frame prediction (both forward and backward). Only the high level differences between MPEG-1 and MPEG-2 are described here.

Compatibility and scalability are the two fundamental concepts underpinning MPEG-2. Compatibility ensures that an MPEG-2 decoder can decode the output of existing encoders, such as H.261 and MPEG-1 [2]. Conversely, an existing decoder can decode the MPEG-2 bitstream (or at least a part thereof). A bitstream is considered scalable if it possesses the property whereby part of it can be decoded independently of the rest in order to produce a meaningful output [2]. In this way, cheaper decoders can decode low quality and/or low resolution (both spatial or temporal) video, whereas a more expensive decoder can produce better quality video from the same bitstream. This is important for targeted applications such as multipoint video conferencing, database browsing, and video communications over asynchronous transfer mode (ATM) networks. Scalability consists of having a number of layers in the bitstream. The most basic layer (referred to as the *base layer*) can be decoded by the decoder of lowest complexity and the other layers ignored. Decoders of higher complexity can decode these other layers (referred to as *enhancement layers*) along with the base layer for enhanced functionality. MPEG-2 supports a number of different types of scalabilities such as spatial scalability, temporal scalability, SNR scalability, data partitioning and hybrid scalability [2].

When considering non-scalable encoding in MPEG-2, the main difference with respect to MPEG-1 is the consideration of interlaced video. Interlaced video is very important for many of MPEG-2's targeted application areas, particularly broadcasting. The main tools adopted in order to improve coding efficiency for interlaced video are a field/frame adaptive DCT, field/frame adaptive MC, and a new scanning order for DCT coefficients to cope with field pictures [2]. In addition, a non-uniform quantizing step size

assignment method is defined which provides finer control of the quantization process for high-quality coding [2].

2.5 Conclusions

The video compression standards described in this chapter have proven to be very successful. Although these standards are quite recent, products offering the envisaged services in each case are already becoming available. It is clear that the underlying core technology is successful in meeting its required objectives, in terms of efficiency and quality. H.261/H.263 can provide video-conferencing facilities of acceptable quality at low bit rates. MPEG-1/2 can provide higher quality video for either off-line or real-time higher bit rate applications.

The schemes described are all lossy in that compression of the original data is achieved by throwing out some image data that is considered redundant. Naturally, this loss is reflected in some reduction of the decoded image quality. The loss of information typically manifests itself as artefacts in the decoded picture. These artefacts, particularly at higher bit rates, may be transparent to the human visual system (e.g. the loss of very fine detail may not even be missed). However, at very low bit-rates the effect of these artefacts is more prominent and they become visually annoying. As could be expected of block-based schemes, the coding artefacts are themselves block-oriented. They arise primarily from coefficient quantization errors, due to the coarse quantization necessary at lower bit rates, which ensure that an exact block reconstruction is not obtained. Since the inter-correlation between blocks is not exploited this results in visible discontinuities among adjacent blocks (known as *blocking artefacts*). The regular structure of such artefacts when considered in natural images is visually disturbing. The effect of motion compensation can further reduce image quality by reinforcing the effect of this coarse quantization in the displaced block which itself may exhibit visible discontinuities with respect to its neighbours.

As outlined in chapter four, compression standards of the future have requirements for content-based functionalities. In this case, a purely block-based approach to compression is not appropriate. The initial investigations of the video compression community into non-block-based approaches in order to provide efficient compression,

whilst avoiding blocking artefacts and at the same time addressing content-based functionalities, are described in the next chapter.

3. SEGMENTATION-BASED VIDEO COMPRESSION

3.1 Introduction

In the block-based approach to video compression, the individual images to be encoded are modelled as a set of independently moving square blocks. This is a rather artificial description of an image and it gives rise to visually disturbing distortions known as blocking artefacts, particularly at low bit rates. Consequently, non-block-based approaches to video compression were also considered within the research community. The coding units of these new approaches were arbitrarily-shaped image segments which reflected actual scene content, and which were derived from a segmentation of each image prior to encoding. By describing these image segments in a compact manner, it was hoped to achieve more efficient compression. Furthermore, since these segments reflect the natural content of the scene, it was proposed that any coding artefacts arising from such an approach would be visually less disturbing than the unnaturally regular artefacts produced by block-based approaches.

As standardisation work on block-based compression schemes was coming to an end, it became clear that the next generation of video compression standards would have very different requirements. In addition to coding efficiency, future standards would also have to provide for the possibility of much higher levels of end user interaction[3]. The envisaged new levels of interaction would require the possibility for an end user to interact with actual scene *content*. In this way, future compression schemes are required to support content-based functionalities. A regular splitting of the image into square blocks cannot provide these functionalities. Segmentation-based encoding approaches, however, were considered ideal candidates for providing this. In light of this, and with the advent of a new standardisation work item, the new segmentation-based approaches to video compression received a lot of attention.

[3] Both MPEG-1 and MPEG-2 provide a certain level of user interaction but of a very restricted nature (fast-forward, reverse, applied to the entire scene)

3.2 Objects and Regions: Two different approaches

When segmentation-based compression is considered there are two approaches: object-based and region-based. An object is considered to be a grouping of pixels which forms a partition of an image which has some semantic meaning associated with it. In other words, the grouping reflects a human's interpretation of the scene. A region, on the other hand, is a grouping of pixels which are homogeneous according to some chosen criterion but which need not necessarily have a semantic meaning associated with it. An object is normally made up of a number of image regions and is thus not generally homogeneous according to a single criterion. For example, an object can be composed of many regions of different intensity, colour and motion.

The relationship between objects and regions is illustrated in Figure 3.1 below. In Figure 3.1(a) a selected semantic object in the scene (i.e. the foreground person) is composed of a number of image regions, homogeneous according to a luminance-based criterion, which in this case have a semantic meaning associated with them (e.g. hair, face, jacket, etc.). Figure 3.1(b) shows another example of a selected semantic object (i.e. both foreground figures) which is composed of a number of image regions, homogeneous according to the same criterion as above. However, in this case very few image regions can be considered to have semantic meaning associated with them.

Object-based video compression performs the segmentation of an image prior to encoding on the basis of objects present in the scene. Compression is obtained by exploiting the spatial and temporal redundancy within the object's segmentation in each image in the sequence, and by compactly representing object parameters. Region-based video compression performs the segmentation on the basis of detecting homogeneous regions in the scene and exploiting the temporal and spatial redundancy associated with these regions throughout a sequence. Both the object-based and the region-based approach to video compression are described in this chapter. In each case, individual compression schemes implementing these coding philosophies are described and discussed.

(a) An object is made up of regions with semantic meaning

(b) An object is made up of regions with no semantic meaning

Figure 3.1 - The relationship between objects and regions

3.3 Object-Oriented Analysis Synthesis Coding

The approach of object-oriented analysis synthesis coding (OOASC) is to segment each image to be encoded into objects which conform to an underlying source model. The source model is described by a set of well defined parameters. Model parameters are obtained by analysis of the input image at the encoder. Using the encoded and transmitted parameters, the objects are synthesised at the decoder. Analysis synthesis coding techniques have been published which use three-dimensional models, known as *wire frames*, in order to code special known objects in the scene [5][6]. These models represent the actual three-dimensional structure of the object to be coded (e.g. a person's head or torso). This approach to analysis synthesis coding is considered to be very specialised and limited to a narrow set of applications and as such, it is not described here.

The OOASC approach of *Musmann et al* [7] was to develop a more general approach to analysis synthesis coding, which did not restrict itself to coding known objects. In the proposed approach, a number of different generic models can be used. For example, in [7] and [8] the analysis and synthesis of both two-dimensional and three-dimensional models is examined. It is not required that the algorithm understand the actual semantic meaning of objects conforming to such models, thus making this approach applicable to a wider class of scene types than the wire frame approach referred to above. No user interaction or human intervention is required in the analysis step, as objects are detected

and their parameters extracted automatically, thus making this approach a suitable candidate for real-time video compression.

Object parameters are obtained by automatically analysing and segmenting the input image at the encoder [9]. Using the encoded and transmitted parameters, the object can be synthesised at the decoder and the input image reconstructed. The parameters extracted encapsulate the shape, motion and texture of each object. Thus, compared to block-based approaches, there is an extra data source to be encoded, corresponding to the shape information. In order to ensure coding efficiency, a very compact encoded representation of shape is required. However, the visual distortions introduced by encoding shape information tend to be geometric in nature and thus (hopefully) less visually disturbing.

Only the parameters of moving objects in the scene are transmitted for each image. In order to reconstruct an entire image, knowledge of static regions is required. This is achieved by using a colour (i.e. luminance and chrominance information, also referred to in this chapter as texture) memory for the sequence. Ideally, only decoded shape and motion parameters are required in order to synthesise an object. However, very often an object will consist of sub-regions which cannot be synthesised to a sufficient degree of quality in this manner. These correspond to regions within the object whose behaviour does not conform to the underlying source model, or regions which by their very nature are unpredictable (e.g. an eye opening/closing). Such regions are termed Model Failure (MF) regions. It is necessary to encode shape and texture parameters for MF regions. These parameters (which complete the parameter set for each object) are extracted by analysing the synthesised object.

3.3.1 SIMOC: The COST 211ter analysis synthesis encoder

The COST (Coopération Européenne dans la recherche scientifique et technique) Telecommunications activities provide an open and flexible framework for research and development collaboration in Europe. The COST 211bis Action was a project dealing with redundancy reduction techniques applied to video signals and COST 211ter is a follow-on project to this [10]. This collaborative research forum facilitates the creation and maintenance in Europe of a high level of expertise in the field of video

compression. The project contributed significantly to both ITU and ISO standardisation bodies. The group was partly responsible for the definition and development of H.261 and H.263, as well as contributions to MPEG-1 and MPEG-2.

When the standardisation effort on H.261/H.263 was coming to an end, COST 211ter turned its attention to segmentation-based representations of video, with the objective of performing collaborative research in this area with a view to eventual standardisation. To this end, an OOASC scheme was chosen as the focus of the group. The target applications of the scheme were real-time video conferencing and video telephony. A reference model known as the Simulation Model for Object-based Coding (SIMOC) [11] was defined, the purpose of which was to define a common specification of an OOASC algorithm which could be collaboratively developed by the members of COST 211ter. SIMOC was based on the work of *Musmann et al.*

3.3.2 A Description of SIMOC

The underlying source model in SIMOC is a single planar flexible natural object which moves translationally in the image plane (i.e. a two-dimensional object with two-dimensional motion) across a static background. The structure of the SIMOC encoder is shown in Figure 3.2. Overall, a DPCM approach is employed, whereby the encoder incorporates a decoding mechanism for prediction. Illustrative examples of various processing steps and results obtained using SIMOC are presented in Figure 3.3. These examples were generated using a software simulation of SIMOC developed by the author in the ANSI C programming language.

Image Analysis:

The task of image analysis is to automatically extract the parameters of the object present in the current image I_{k+1}. Analysis is carried out on I_{k+1} with reference to the previous reconstructed image $\hat{I}_k$. Shape, motion and texture parameters (S_{k+1}, M_{k+1}, T_{k+1} respectively) for the object are extracted. The individual processing steps are briefly described below.

Image Analysis - *Change Detection*:

The static background constraint of the SIMOC source model leads to the implicit assumption that any change or difference between successive images in a sequence is due to the presence of a moving object. Thus, the objective of the first step in the image analysis procedure is to obtain a binary segmentation of the current image indicating changed and unchanged regions. To perform the segmentation, the current image and the previous reconstructed image are subtracted. The difference image is filtered and thresholded in order to segment a changed area [11].

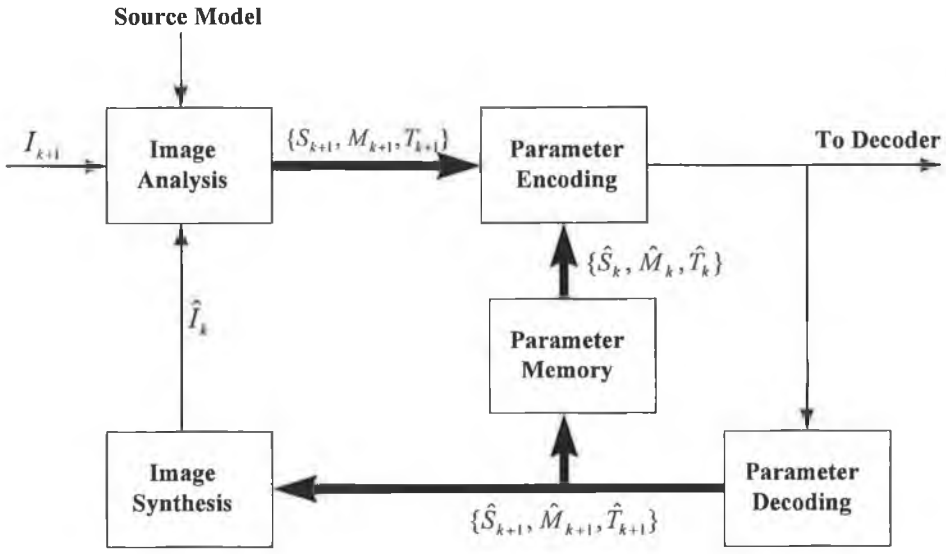
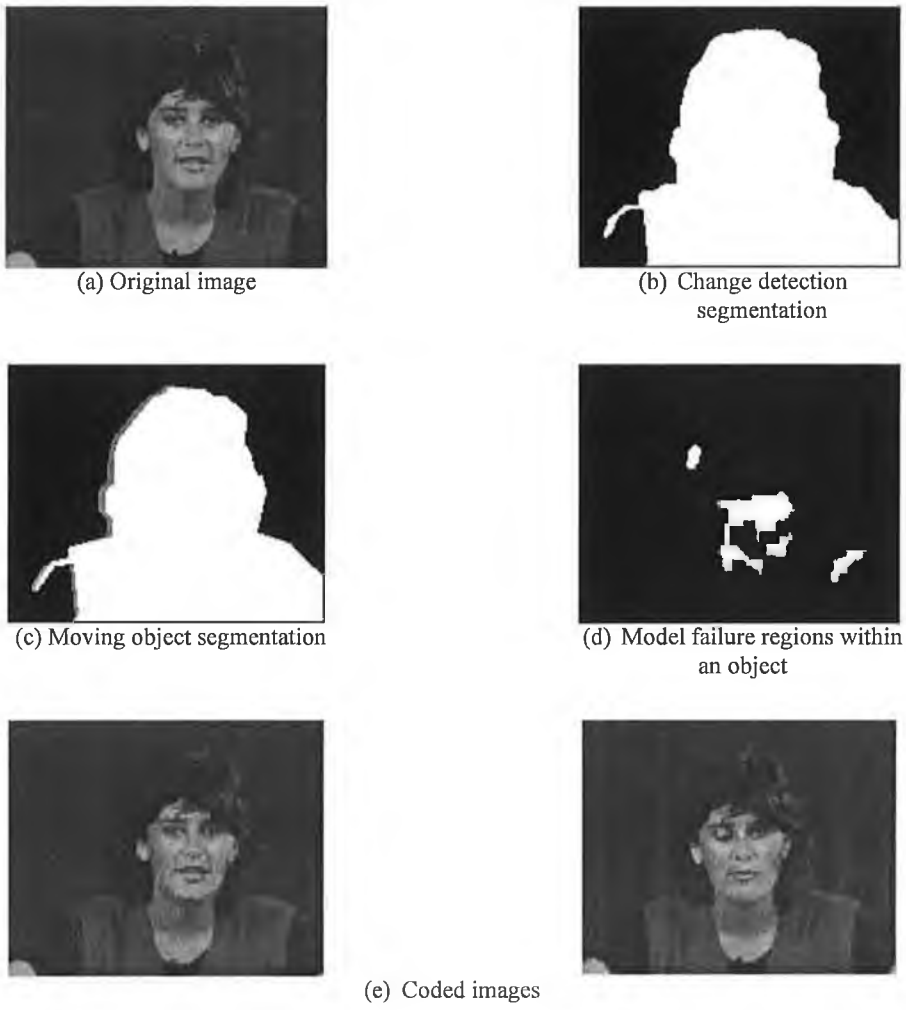


Figure 3.2 - The structure of SIMOC

Image Analysis - *Moving Object Segmentation*:

The change detection segmentation is further segmented into moving object, static background and uncovered background[4] image regions by reversing the effect of the estimated motion (see below) and comparing with the previous segmentation mask [11]. The shape of the moving object is approximated using a combined spline/polygon approach and the vertices of this approximation, together with the set of spline/polygon decisions, constitute the required shape parameters S_{k+1} [11].

[4] Texture for these regions must sometimes be transmitted. This is achieved using the same VQ approach as for MF regions.



(a) Original image

(b) Change detection segmentation

(c) Moving object segmentation

(d) Model failure regions within an object

(e) Coded images

Legend: (b): white = changed, black = unchanged
(c): white = object, grey = uncovered background, black = static background
(d): white = model failure

Figure 3.3 - Illustration of SIMOC encoding

Image Analysis - *Motion Estimation*:

The motion between successive images in the video sequence is estimated using a three-step hierarchical block matching algorithm, which is based on the work of *Bierling* in [12]. After estimation, the obtained motion vector field is interpolated to obtain a dense motion vector field. The change detection segmentation is used to constrain the estimation process [11]. This results in a set of motion parameters M_{k+1} for the object.

Image Analysis - *Model Failure Detection*:

By necessity, this analysis step takes place after synthesis (see below). The quality of the synthesis is evaluated in order to locate any MF regions. These are detected by evaluating the synthesis error over the object using a MAD criterion, and detecting

regions where this error is unacceptable [11]. The shape of these regions is approximated using a spline curve approximation with only four vertices. The texture of these regions is encoded using a vector quantization (VQ) approach [11]. The VQ indices constitute the texture parameters T_{k+1} for an object[5].

Parameter Memory:

The parameter memory is analogous to the frame store of traditional block-based encoders. It is used to store the object's decoded parameters from the previous frame $\{\hat{S}_k, \hat{M}_k, \hat{T}_k\}$, as well as the previous object segmentation mask. The parameter memory also includes the colour memory which is used to store texture already encountered in the encoding process. Texture parameters are stored in the parameter memory by updating the colour memory with the transmitted texture [11].

Parameter Encoding and Decoding:

The estimated parameters are predictively encoded via adaptive arithmetic encoding [13]. The coding events formed for the shape parameters are run-lengths based on the vertices and the spline/polygon decisions for the approximation [11]. The coding events for motion are the predicted motion vector differences, whilst the coding events for texture are VQ code-book indices [11]. Parameter decoding reverses the operation of the encoding. These decoded parameters $\{\hat{S}_{k+1}, \hat{M}_{k+1}, \hat{T}_{k+1}\}$ are then used to synthesise the object and reconstruct the current image.

Image Synthesis:

The shape of each object is first reconstructed using the decoded shape parameters. This shape is filled with texture from the colour memory, suitably motion compensated using the decoded motion parameters [11]. The shape of the MF regions is also reconstructed and the decoded texture is used to fill these regions, thereby replacing the equivalent synthesised regions within the object. Finally, in order to obtain a completely reconstructed scene, the texture for the static regions in the image is taken directly from the colour memory [11].

---

[5] The shape parameters for MF regions are added to S_{k+1}.

3.4 Region-based coding

The underlying objective in region-based coding is to take human sensitivity to image contours and textures into account in the coding process. The coding units are arbitrarily-shaped image regions which are homogeneous in texture according to some chosen criterion, and which reflect the content of the scene. The approach generally consists of three steps. The first is the region-based segmentation of the image to be encoded and is the crucial step which leads to a codec structure[6]. The second step deals with encoding the segmentation for transmission. Texture information (i.e. the luminance and chrominance information) within these regions is encoded in the third step. In a manner similar to OOASC, since the segmentation used in the encoding process reflects scene content, it is proposed that the type of distortions introduced by such an approach are less visually disturbing than blocking artefacts.

3.4.1 Morphological Tools for Region-based Segmentation

The most popular approaches to region-based coding are approaches which use morphological image processing tools. This is because morphological tools can produce the required segmentations very easily. Mathematical morphology is a geometric approach to image processing which can easily take advantage of such criteria as size, shape, contrast and connectivity. A number of tools have been developed which allow analysis and segmentation of a grey level image in these terms [14]. These tools can be easily extended to cope with sequences by considering the signal to be defined in three dimensions with time making up the third dimension [15]. This simply corresponds to changing the local neighbourhood, termed a *structuring element*, on which the tools are defined [15].

Many morphological tools rely on two basic transformations known as erosion and dilation [14]. Erosions and dilations allow the definition of size-oriented filters known as openings and closings. An opening removes the bright elements of the picture which do not fit within the structuring element, whereas a closing removes the dark elements [14]. Open-close or close-open filters can be defined when processing requires that both

---

6 The nature of the segmentation indicates how it can be most efficiently encoded.

dark and bright elements be considered. These filters can be used to simplify an image before segmentation, however they do not preserve the contour information present.

By introducing the notion of connectivity and a reference function, a set of geodesic morphological filters can be defined [14]. These filters are called reconstruction filters, as they attempt to reconstruct the reference function by successively dilating/eroding the input signal. When the reference function is the original image and the input function is a morphologically filtered version of that image, the output is the original image with the filtered elements removed but with contour preservation of the remaining image components [14].

The traditional morphological approach to segmentation relies heavily on the use of the morphological gradient which is the difference between a dilation and erosion of an image. The gradient transformation highlights contour points, but represents a loss of information [15]. This is not usually of much consequence for still images, but has more serious implications in the case of image sequences.

The watershed algorithm is a very powerful morphological segmentation tool which produces very clean spatial segmentations with excellent contour localisation [16]. The watershed segmentation is created by treating the morphological gradient as a topographic surface and by building a segmentation based on its local minima via immersion simulations [16]. If the watershed algorithm is applied directly to the gradient of the image, the result is extreme over segmentation [16]. This can be avoided by morphologically filtering the input image and extracting markers[7] for the required regions in the segmentation prior to the watershed process [16]. The results of the watershed algorithm and other morphological transforms are illustrated in Figure 3.4. These results were generated using software implementations of the morphological transforms, developed by the author in the ANSI C programming language.

[7] Markers can be extracted based on various criteria such as contrast or flatness.

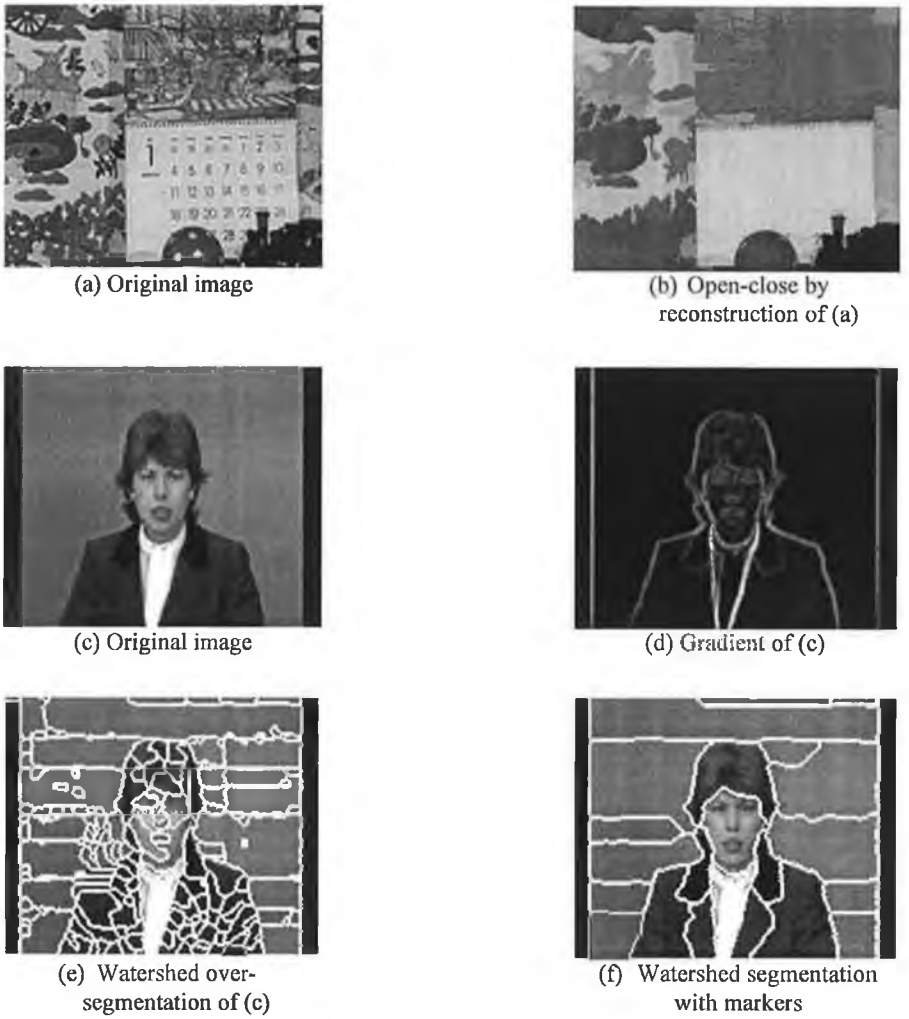


Figure 3.4 - Illustration of morphological processing

3.4.2 MORPHECO: A morphological region-based video codec

The RACE MORPHECO (Morphological Codec for Storage and Transmission) project used the above basic morphological tools (amongst others), suitably extended to three-dimensions, to produce an integrated morphological region-based video codec. The coding scheme is usually referred to simply as MORPHECO. It is a hierarchical segmentation-based coding scheme following a top down approach [15]. The first level of the hierarchy produces a very coarse segmentation which produces low visual quality. The segmentation is coarse in the sense that it contains very few image regions, however, the contours of the regions in the segmentation are very precise. Subsequent levels of the hierarchy introduce more regions into the segmentation, without altering the segmentations of preceding levels, leading to higher quality coded images. Each level passes the coding residue (see below), the current segmentation and the original

image to be encoded to the next level of the hierarchy. The operation of the coding scheme for one level of the hierarchy is shown in Figure 3.5.

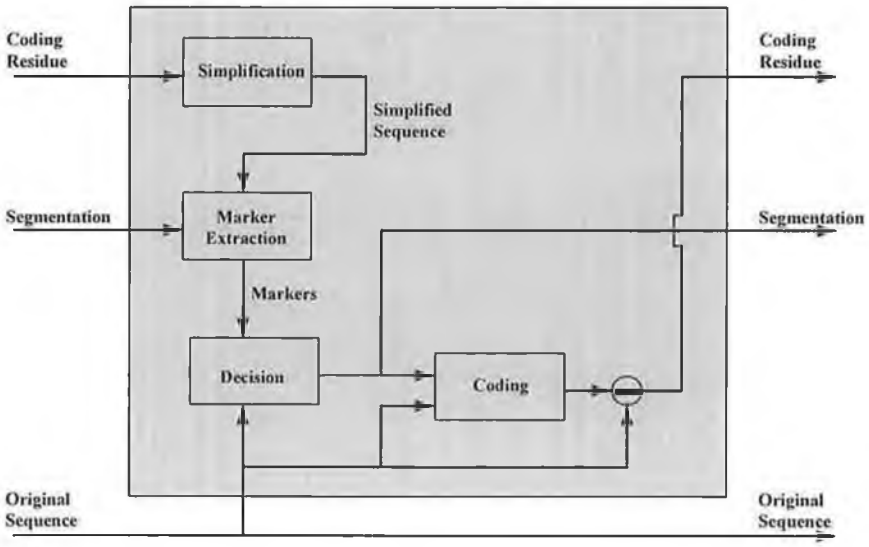


Figure 3.5 - One level of the MORPHECO coding hierarchy

The simplification controls the number of regions retained in a particular level of the hierarchy. Morphological filters based on reconstruction[8] of erosion or dilation are used [15]. The marker extraction step extracts markers for three-dimensional homogeneous regions. The segmentation from previous levels of the hierarchy is used as it already indicates regions which are homogeneous. Marker extraction is performed by extracting flat regions (corresponding to the interior of regions) and contrasted regions (corresponding to visually important regions) [15]. The decision block precisely locates the contours of the marked regions. In order to perform this segmentation, the watershed algorithm is used. Since contours of moving objects in an image sequence appear as thick image segments in a three-dimensional gradient, a modified watershed algorithm is applied to the original signal [15].

Quality estimation is carried out in the coding block by actually encoding the data in the segmentation and subtracting the resultant coded image from the original image. The difference between the original image and the encoded image is passed to lower levels of the hierarchy as the coding residue. For the first image of the sequence, both contours and texture are coded in INTRA mode using a modified chain code approach, low order

[8] Actually, consideration of video sequences requires partial reconstruction filters to be used [15].

polynomials and block truncation coding[9] [15]. In subsequent images, the contours and textures are predicted using motion compensation and the prediction error is encoded using a similar approach [15]. Motion information consists of one vector per region in the segmentation, which describes the translational motion of the region between images.

3.5 Discussion

Both SIMOC and MORPHECO were considered to be ideal candidates as starting points for a new video compression standard which would support content-based functionalities. The COST 211ter group intended to collaboratively develop SIMOC towards a description suitable for proposal for standardisation. In fact, MORPHECO was also eventually introduced into the COST 211ter framework with a similar view in mind. However, due to a number of limitations, neither MORPHECO nor SIMOC were eventually adopted as the basis for a new round of video compression standardisation work. The reasons for this are explained in this section.

The most fundamental limitation of SIMOC lies with the choice of source model. Using this model, successful OOASC is restricted to scenes containing only one object and a static background, in other words, to a class of sequences which exhibit the characteristics of a video conferencing application. Sequences exhibiting more than one moving object or a moving background (e.g. due to camera motion such as zoom and pan) cannot be efficiently encoded using this approach. Furthermore, objects moving with more complicated motion than simple translation (which is quite likely for many objects) will be poorly represented using this source model.

The choice of source model is reflected in the analysis procedure. Change detection assumes that any change between images is due to the presence of a moving object. In the case of sequences exhibiting background movement, this assumption does not hold. This is illustrated in Figure 3.6(a) and (b) where the change detection mask calculated by SIMOC for a sequence with a moving background (i.e. the Foreman test sequence) is

---

9 It should be noted that these are only examples of coding techniques which may be used within the MORPHECO framework.

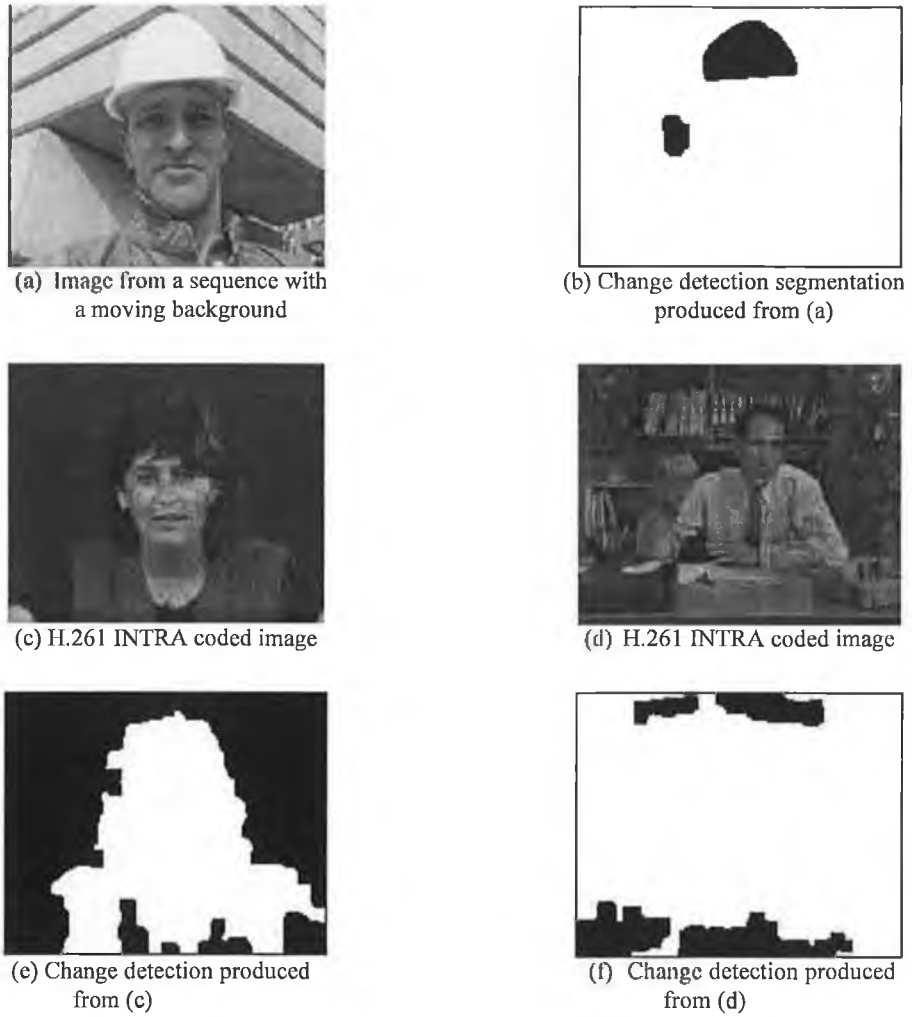
presented. The change detection mechanism breaks down completely and almost the entire image is detected as the "object". The encoded representation of this scene produced by SIMOC is neither efficient, nor does it reflect scene content [17].

Another limitation of the SIMOC algorithm is the approximation of MF regions. Even when dealing with sequences which conform to the underlying source model, these regions very often have complex shapes (e.g. see Figure 3.3) and an approximation with only four vertices is not suitable as it actually increases the size of such regions [17]. This leads to a decrease in coding efficiency since it requires encoding texture for regions which have already been sufficiently synthesised. Furthermore, MF regions will tend to grow in size over the course of a sequence, leading to a significant decrease in coded image quality [17].

SIMOC does not specify an INTRA encoding scheme. All practical compression schemes require a full frame update mode. This mode must at least encode the first frame of the source video sequence. Typically, a build-up phase is unavoidable and video quality is quite low during this time. This effect is not taken into account in SIMOC where the first reconstructed image is required for the initial change detection. In fact, a low quality reconstructed image will severely affect the change detection process [17] as illustrated in Figure 3.6. The first image of two test sequences encoded using the INTRA mode of H.261 is illustrated in Figure 3.6(c) and (d). The change detection segmentations obtained based on these are presented in Figure 3.6(e) and (f). The resultant segmentations exhibit a blocky unnatural contour, or else the entire change detection procedure fails.

One of the main advantages of block-based approaches is that since blocks are treated independently, a decoder can start to decode a block as soon as it is transmitted thus keeping overall delay to a minimum. However, in the case of a segmentation-based approach, the decoder must wait until the encoder analyses an entire image and transmits the shape information before it can start the decoding process. Thus, segmentation-based coding implies an implicit coding delay of one frame. This may not be an issue for off-line coding applications, but it is a severe restriction in real-time

applications such as video conferencing or video telephony at which SIMOC was targeted.



(a) Image from a sequence with a moving background

(b) Change detection segmentation produced from (a)

(c) H.261 INTRA coded image

(d) H.261 INTRA coded image

(e) Change detection produced from (c)

(f) Change detection produced from (d)

Legend: white = changed, black = unchanged

Figure 3.6 - Illustration of the limitations of SIMOC

One the main motivations of OOASC is to produce a compression scheme which produces visually acceptable coding artefacts. The assumption is that a human observer is less sensitive to distortions introduced by compressing object model parameters than the more visually disturbing blocking artefacts. However, examining the images of Figure 3.3(e) it can be seen that this is not always the case. Sometimes the distortions introduced are even more unnatural in nature and hence more disturbing (see image on right in Figure 3.3(e)).

MORPHECO suffered from very few of the problems associated with SIMOC. MORPHECO can be considered to have an underlying source model of moving image regions, which is a more general source model than that of SIMOC. As such, the MORPHECO approach can be applied to a much wider class of scenes. There is no limitation to a single object, as the segmentation is considered to be a tessellation of the image in terms of homogeneous regions which reflect the objects present. Furthermore, the morphological tools used to obtain this segmentation are very robust for a wide class of scene types. The hierarchical approach ensures that all region types are catered for, leading to an accurate segmentation reflecting image content. MORPHECO also has its own INTRA encoding scheme which is closely related to INTER frame coding. Thus, the effect of INTRA coding on the segmentation performance is already considered in the codec structure.

However, MORPHECO does suffer from the inherent delay associated with any segmentation-based approach. As in SIMOC, an entire image must be analysed and the complete segmentation performed before decoding can commence. This delay is exacerbated in the case of MORPHECO which uses three-dimensional morphological tools which reference both the preceding image and subsequent image to the image being coded. However, the MORPHECO codec was not targeted at real-time applications. Rather, as the title of the project suggests, a key application of MORPHECO was video compression for storage, which can be performed off-line.

The limitations associated with SIMOC ensured that the algorithm was eventually abandoned and never developed to a state suitable for proposal for standardisation. MORPHECO was not a suitable candidate either, but for different reasons. The key to understanding this lies in the consideration of the type of functionalities required of a new video compression standard. As stated in the introduction to this chapter, the main functionality required is the ability for the user to interact with scene content. As explained in the next chapter, scene content in this context refers to semantic objects. Thus, semantic object segmentations are required for the encoding process. Neither MORPHECO nor SIMOC produce semantic object segmentations. SIMOC, whilst being object-based, can only segment a restricted class of objects and even then the segmentation is not very accurate. MORPHECO, on the other hand, encodes image

regions and no semantic object segmentation is available. In both SIMOC and MORPHECO, the encoding scheme is very tightly linked to the image analysis process. Because of this, the overall quality and usefulness of the compressed representation is highly dependent on the segmentation process. In the next chapter, it is described how analysis and encoding were de-coupled to produce a compression scheme which supports content-based functionalities. The resulting approach incorporates the advantages of both segmentation-based and block-based compression. That is, objects are encoded independently (thereby allowing access to scene content), but in an efficient, flexible, and robust manner using modified block-based coding tools.

4. RECENT DEVELOPMENTS IN VIDEO COMPRESSION

4.1 Introduction

Recently, a large number of new application areas for encoded AV data have been proposed. These application areas include concepts such as content-based AV database access, AV home editing, tele-shopping, computer games, and remote monitoring and control. One common feature of all these application areas is that interaction with AV content is fundamental for the envisaged application or service. For multimedia applications of the future, it is no longer enough for the user to be a passive observer of AV data. Rather, it is desired that the user be able to react to what is seen or heard, in order to tailor the AV application or service to his/her own needs. Another fundamental concept encapsulated by these new application areas is the notion of re-using AV encoded data in an intuitive and flexible manner. Consider an application allowing the editing and manipulation of AV data where it is desired to create new AV content based on AV content in a library. In order to do this, so that the resultant AV content is seamless, the stored AV content should have an encoded representation which facilitates re-use. In this chapter, the third standardisation work item of ISO's Motion Picture Experts Group, normally referred to simply as MPEG-4, is described. This new standard addresses the needs of future AV applications and services by supporting content-based functionalities. Also briefly described in this chapter is the envisaged MPEG-7 standard which is just starting to be developed. These two standards are described because, as outlined in this chapter, this serves to high-light the importance of video segmentation for future multimedia applications.

4.2 MPEG-4: A new representation of audiovisual data

Considering the AV encoding standards described in chapter two, it is clear that interaction is limited to the temporal aspect of AV sequences, whereby the user can linearly traverse the sequence at variable speeds (in other words, fast-forward, reverse,

pause, etc. applied to an entire scene). Re-use of coded AV data is restricted as it corresponds to extracting a rectangular segment from a sequence and inserting it into another application, when perhaps only a particular object in the scene and not the entire scene is desirable in the new application. These deficiencies of the existing standards form the basis of the MPEG-4 standard. At the core of MPEG-4 is a new representation of AV data. The MPEG-4 standard has adopted a scene representation in terms of the individual AV objects which make up a scene, and each AV object is encoded independently [18]. As explained in the following discussion, this new representation of AV data enables MPEG-4 to achieve its main objectives of support for content-based functionalities and re-use of encoded AV data.

4.2.1 Objectives of MPEG-4

The major goal of MPEG-4 is to provide a new form of interactivity with AV coded data [19]. When viewing a scene a user may wish to interact with the content of the scene for a variety of reasons. The user may require increased quality or temporal resolution for a particular object because it is more important to him/her than the rest of the scene. They may wish to remove an object from the scene because it is obscuring what they really want to see/hear. Perhaps they would like to insert an object in the scene in order to create a new scene. The exact nature of the interaction will depend on the associated application. An object-based scene representation facilitates this type of interaction. Since each object is encoded independently it becomes possible to interact with that object alone, without affecting the rest of the scene. Furthermore, it allows the re-use of encoded AV data, as a stored AV object can be easily added to the scene, or the objects in the scene can be used at a later date in a different scene. Also, since the objects are treated independently, it is possible to encode each object using a coding scheme best suited to the nature of the object[10]. By adopting an object-based scene representation, the standard effectively removes the responsibility of *scene composition* from the decoder and allows the possibility of providing it as a functionality of the user's application.

---

10 In this way, objects of different media types can be represented and used to build a scene consisting of very different types of content.

The standard has other objectives such as scalable coding, transmission across heterogeneous and error-prone networks, hybrid natural/synthetic content coding and coding of multiple concurrent data streams. Whilst these objectives are greatly facilitated by an object-based scene representation, they are not described in detail here. The reader is referred to [19] for a more complete description.

4.2.2 An MPEG-4 System

Figure 4.1 shows a complete MPEG-4 encoding and decoding system for visual objects. Input objects are encoded and may be multiplexed with other stored objects for transmission. At the receiving side, the objects are de-multiplexed and decoded. A display is created by combining these decoded objects. They may be combined with other objects stored at the receiver. User interaction is allowed at any stage of the encoding/decoding process. The user may choose which objects to encode, how to encode them, and to what quality. The user can also decide which objects to decode and at what quality level to perform this decoding. Finally, the user is instrumental in creating the desired display as he/she can decide which objects are to be used to form the display, as well as how to display these objects (this is the process termed scene composition above).

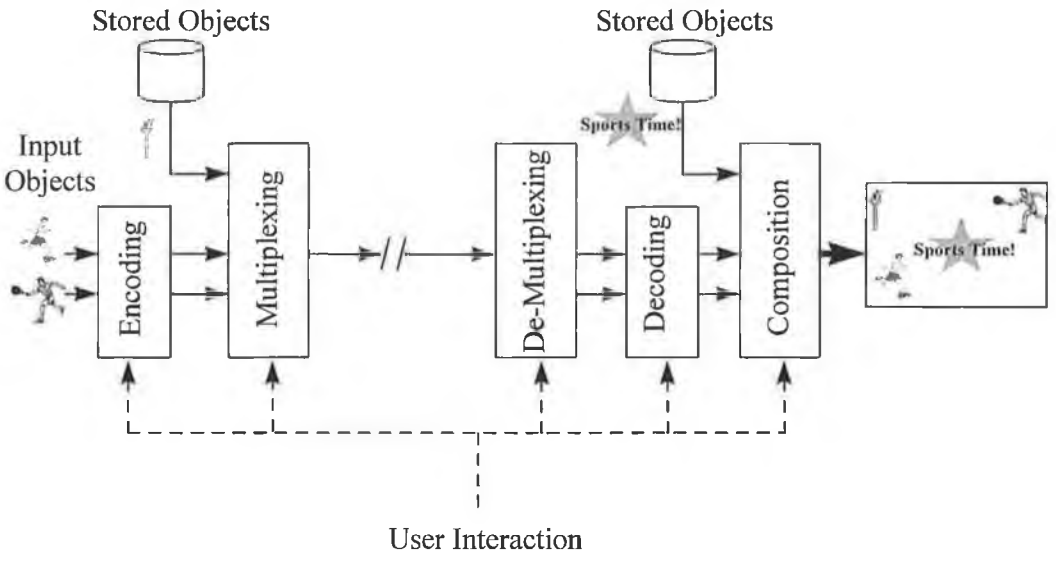


Figure 4.1 - The complete MPEG-4 encoding and decoding system for visual objects

4.2.3 Video Objects and Video Object Planes

In MPEG-4 video compression, objects are referred to as Video Objects (VOs). A VO is a semantic object in the video scene which evolves through time. A VO can be an arbitrarily-shaped object in a video scene, such as a foreground person, or can be a rectangular object corresponding to the entire scene. A VO evolves temporally through a video sequence and a snap-shot of the state of the VO at a particular time instant is termed a Video Object Plane (VOP). VOPs are analogous to frames in block-based coding techniques.

Each VOP has associated with it four pixel components which represent the object's shape and texture. Luminance and chrominance information (i.e. *Y,U* and *V* components) is used for texture exactly as in a rectangular video frame. The object's shape is represented using a pixel component termed the *alpha plane* (sometimes termed the *alpha channel*). This is a segmentation which defines the object's shape and transparency information. A binary alpha plane indicates a completely opaque object. Alternatively the alpha plane may contain different grey level values indicating parts of the object which are partially transparent. The concept of VOs, VOPs, YUV and alpha components is illustrated in Figure 4.2.

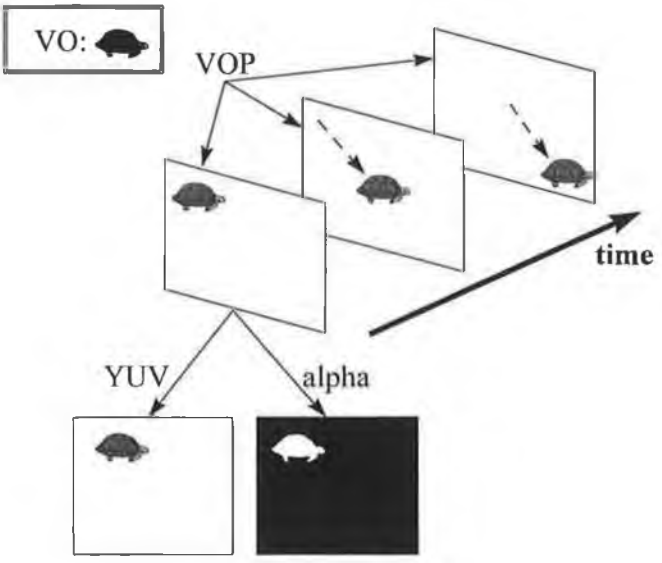


Figure 4.2 - Video Objects and Video Object Planes

4.2.4 Structure of an MPEG-4 Video Codec

A diagram of the high level structure of an MPEG-4 video encoder is shown in Figure 4.3. As explained by *O'Connor and Winder* in [20], a layered coding methodology is adopted to facilitate the object-based scene representation. Although it is not shown in the diagram, it is implicitly assumed that some composition information specifying the VO's spatial and temporal position is also encoded.

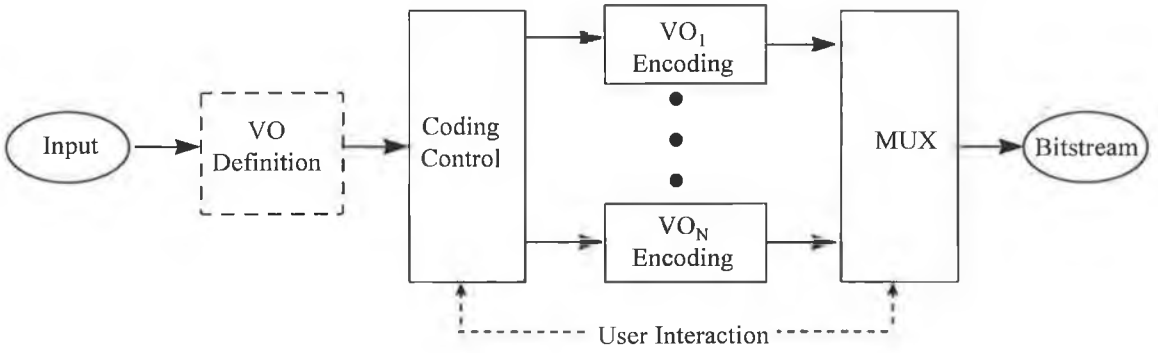


Figure 4.3 - High level structure of an MPEG-4 video encoder

The VO definition block has the task of defining meaningful objects in a scene with which the user may wish to interact. It is the process of creating a sequence of VOPs for a particular object, which amounts to an object segmentation process. The segmentation process may be automatic or semi-automatic or obtained via a chroma-keying process (i.e. blue screen technology, see section 6.2) in a studio. MPEG-4 will not standardise this segmentation process. In this way, MPEG-4 specifies a coded representation of objects, but not the way in which these objects were obtained (for more information on VOP creation and MPEG-4, see the final section of this chapter). Figure 4.3 shows the possibility of user interaction at the encoder[11]. User interaction may occur at the coding control block whereby the user may request that a particular VO be coded to higher quality than other VOs. User interaction can also occur at the multiplexer whereby the user may wish to change a VO's composition information prior to transmission. Alternatively, due to bandwidth limitations for example, the user may request that a particular VO is not transmitted at all.

[11] This implicitly assumes the availability of a return channel for the associated application.

The high level structure of an MPEG-4 video decoder is illustrated in Figure 4.4. Each VO is demultiplexed from the bitstream and independently decoded using its own decoder [20]. The composition block has the task of combining the decoded VOs to create a scene. This is actually an MPEG-4 systems issue and as such, the exact nature of this block is not outlined, except to note that user interaction may occur during composition. User interaction at the decoder may also occur at the demultiplexer. The user may request, for example, that only a subset of the available VOs be decoded, thereby lightening the load on the decoder.

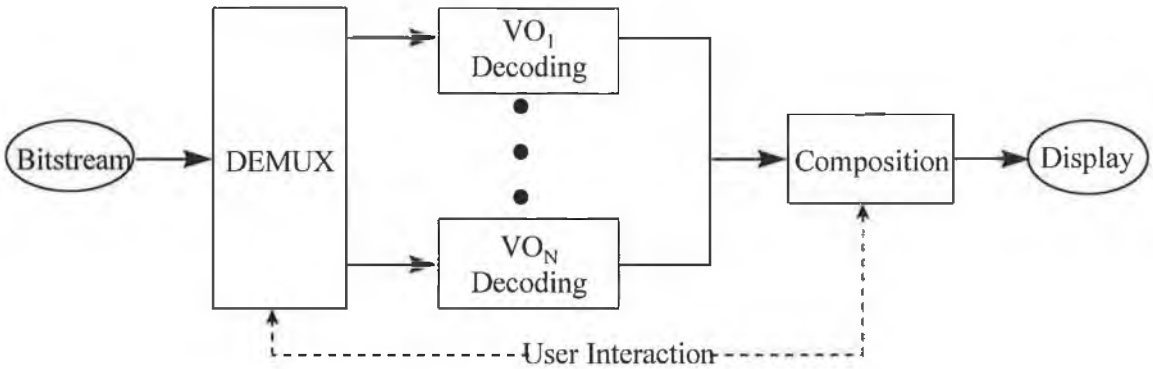


Figure 4.4 - High level structure of an MPEG-4 video decoder

4.2.5 MPEG-4 Compression Tools

In this section, the actual encoding tools adopted by MPEG-4 are described. The description is limited to tools used for encoding natural arbitrarily-shaped VOPs and even then only a subset of the available tools is explained. A more complete description of MPEG-4 coding tools can be found in [21]. There are three types of encoded VOPs referred to as I-, B- and P-VOPs, which are analogous to the I-, B- and P-pictures of the other MPEG standards.

A bounding box is placed around the VOP to be encoded, which limits the amount of data to be coded for each VOP. The bounding box is the tightest rectangle which includes all non-zero pixels in the VOP's alpha plane. This bounding box is divided into non-overlapping 16×16 blocks. Each block is referred to as an alpha block. The same block grid is applied to the texture component of the VOP and each texture block (referred to as a macroblock, as in H.261, H.263, MPEG-1 and MPEG-2) is encoded independently.

An alpha block (macroblock) may be completely outside the shape of the VOP and thus need not be coded at all. Alternatively, an alpha block (macroblock) may be completely inside the shape of the VOP. These are encoded in a manner very similar to existing block-based standards. Finally, an alpha block (macroblock) may be partially inside the shape of the VOP. These blocks need special consideration as they require the transmission of shape, and also the use of this shape information in motion compensation and texture encoding.

MPEG-4 Compression Tools - *Shape Coding*:

Each alpha block in the bounded VOP is encoded separately. In this section, only binary alpha plane encoding is described. In this case, an alpha block is referred to as a Binary Alpha Block (BAB). BABs are encoded using motion compensation and Context-based Arithmetic Encoding (CAE) [21]. A simple motion estimation procedure at full pixel resolution is carried out for each BAB. Each BAB's motion vector is predictively encoded using neighbouring shape and texture motion vectors. Shape update information for the BAB is encoded using the CAE scheme with reference to the motion compensated BAB.

MPEG-4 Compression Tools - *Motion Estimation and Compensation*

Motion estimation and compensation of macroblocks partially within the shape of the VOP uses two tools known as polygon matching and padding [21]. For macroblocks which are completely inside the shape of the VOP, the motion estimation procedure is very similar to that of existing standards. The estimation is carried out at half-pixel resolution. The advanced prediction mode of H.263 (corresponding to four motion vectors per macroblock) is available and OBMC is used to form the prediction. A padding scheme is also used to extend the reference VOP beyond its bounding box in order to provide unrestricted motion estimation, similar to H.263. Motion vector information is predictively encoded in a manner similar to other standards, although special rules are required to cope with the different macroblock cases and the arbitrary shape of the VOP.

MPEG-4 Compression Tools - *Texture Coding*

Texture coding consists of compressing raw pixel data in the case of I-VOPs and compressing the prediction error in P- and B-VOPs. Macroblocks completely within the shape of the VOP are coded using a conventional DCT scheme. Two alternatives exist for encoding macroblocks partially within the shape of the VOP [21]. In the first approach, a subblock is padded using the same technique as in motion estimation and a conventional DCT transform is then applied. In the second approach, a DCT transform modified to cope with arbitrarily-shaped image regions, known as the Shape Adaptive Discrete Cosine Transform (SADCT), is employed. DCT coefficients are quantized, zig-zag scanned and entropy encoded using VLCs as in H.263, MPEG-1, etc.

MPEG-4 Compression Tools - *Scalability and Error Robustness*

In a manner similar to MPEG-2, scalable coding is also supported by MPEG-4. Both temporal and spatial scalability are supported, although only temporal scalability may be used in the case of arbitrarily-shaped VOPs [21].

4.3 MPEG-7: A description of audiovisual Content

The Multimedia Content Description Interface, more commonly known as MPEG-7, is a new work item within the ISO MPEG standardisation process. This work item was initiated in recognition of the increasing availability of AV data of all kinds in many different locations. In order to use this AV content in some way, it must first be located. However, due to the proliferation of this type of information, it has become increasingly difficult for a user to locate and obtain AV content particularly suited to his/her needs. The overall objective of MPEG-7 is to specify a description of multimedia information which can then be used to index the data for storage and subsequent retrieval [22]. This description will be content-based reflecting the semantic content of the AV data. In this way, flexible, efficient queries tailored to a user's exact needs can be made on databases containing AV data described using MPEG-7.

The MPEG-7 standard will support a broad range of applications. Application areas such as digital libraries, broadcast media selection and multimedia editing have all been targeted as areas which will benefit from MPEG-7. Specific applications within these areas such as journalism (e.g. searching a database for a video/audio clip of a politician

or celebrity, searching for AV content on a specific historical event or a range of similar events) and tele-shopping (e.g. searching an on-line catalogue), have already been identified.

A very generalised illustration of an MPEG-7 system is shown in Figure 4.5. This indicates that descriptive features must be first extracted from the AV data. These features are then used to form the description of the data. MPEG-7 will not standardise the way in which the feature extraction is performed [22]. This is because interoperability does not depend on the actual feature extraction method, but on a common description of features. This is analogous to the way in which MPEG-4 does not standardise VOP creation. Similarly, MPEG-7 will not standardise the search engine used to locate AV content, nor indeed the way in which the results of queries are dealt with.

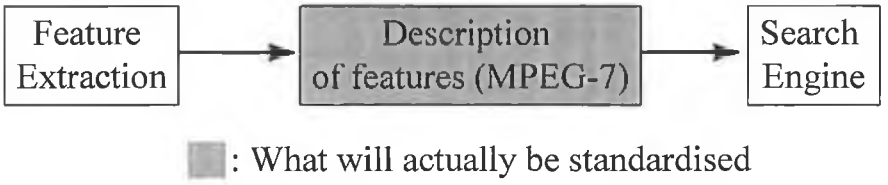


Figure 4.5 - A very generalised MPEG-7 system

4.3.1 Objectives of MPEG-7

Whilst a number of proprietary solutions exist for searching a database based on content, these solutions tend to be limited to a particular data type or a particular type of query. MPEG-7 intends to extend these solutions to incorporate more flexible queries based on a large range of data types such as still images, video clips, audio clips and 3-D models. MPEG-7 will also specify a way of linking the description of the AV data to the actual location of the data itself. In this way, the AV content need not be co-located with its description. The description of the AV content will be independent of the encoding process used to produce the content such as MPEG-1, MPEG-2, JPEG, etc. Although the description is independent of the encoding process, MPEG-7 will borrow heavily from MPEG-4. The object-based scene representation and subsequent encoding of MPEG-4 is a key mechanism for enabling content-based descriptions.

MPEG-7 will ensure that the features used to describe AV content can be tuned to a particular application. This will be achieved by the use of several levels of abstraction for the descriptive features. For example, low level features (e.g. size, shape, colour for visual data) may be extracted and described automatically. However, in the higher levels of abstractions, the features (e.g. plot synopsis, characters, actors, director) will require human intervention in the description process.

MPEG-7 will also support the description of non-content-based information which may be useful and/or necessary for the envisaged applications. These types of descriptive features will contain such information as (i) the form of the AV data (i.e. the coding scheme used), (ii) access conditions (e.g. copyright on the AV data), (iii) classification (e.g. parental rating or classification into pre-defined classes of content), (iv) links (i.e. to the actual location of the AV data, or related AV content), and (v) context (e.g. in the case of non-fiction AV content, it is very often necessary to know when, where and why the content was recorded).

4.4 Discussion

MPEG-4 restricts itself to standardising the way in which AV objects are represented. It will not define the way in which these objects are generated prior to the encoding process. MPEG-4 only specifies the bitstream syntax necessary to represent a video object so that it is MPEG-4 compliant. The process of obtaining video object parameters, i.e. the VO Definition block of Figure 4.3, is an image analysis task, corresponding to the segmentation of a semantic object at each time instant of a video sequence. In other words, it is the process of creating arbitrarily-shaped VOPs.

VOP creation is analogous to the analysis task in the OOASC approach and the image partition task in region-based coding. In these coding approaches, the encoding mechanism (and hence the coded representation), is very closely linked to the analysis procedure and is therefore dependent on the performance of this analysis. The coded representation specified by MPEG-4, however, is *independent* of the way in which the parameters describing the video object are obtained. For example, the shape coding technique divides each VOP's shape into blocks and treats each block independently. This is a very generic approach to shape representation which can be applied to any

shape: the shape of an image region, the shape of a person present in the scene, or even the shape of text in the scene. Irrespective of the form of the video object to be encoded, its shape and texture can be represented using the techniques specified by MPEG-4. In this way, MPEG-4 combines the advantages of both segmentation-based and block-based compression. The basic coding units are objects (in the form of a sequence of arbitrarily-shaped VOPs), thereby facilitating content-based functionalities, but they are encoded in a block-based manner to facilitate many other functionalities (compression efficiency, low-delay, error robustness, etc.).

The way in which the MPEG-4 standard restricts itself to simply specifying an encoded representation for a VOP is not a limitation on the part of the standard. In fact, it is a necessary restriction to ensure its commercial success. A specification of an encoded representation of VOPs ensures interoperability between MPEG-4 products. What will differentiate MPEG-4 products in the market place, however, is the manner in which the VOPs are created. In the absence of a segmentation process during video capture (e.g. if chroma-keying technology is unavailable, see section 6.2), an MPEG-4 application allowing the creation of VOPs in a flexible, robust and exact manner will be more attractive to users than one in which VOP creation is inaccurate, or fails for certain scene types. A further advantage of the approach taken by MPEG-4 is the fact that the standard will be able to avail of future technological advances in the field of video segmentation.

In a similar manner, MPEG-7 will not standardise the way in which features are extracted from an AV scene in order to construct a description of the scene. Again, this promotes a competitive aspect to the MPEG-7 standard: a good feature extraction method for a particular application will ensure a commercial edge in the market place. It also allows MPEG-7 to avail of any future advances in feature extraction technology. Low level feature extraction is an image analysis task which attempts to extract features such as the shape, colour or motion of objects in order to describe them. It can therefore be conjectured that segmentation will be important for extracting low level features for an MPEG-7 description.

The fact that neither MPEG-4 nor MPEG-7 will standardise segmentation serves to emphasise the importance of video segmentation as an enabling feature of future multimedia communications and applications. In the following chapters, a selection of existing (state of the art) approaches to video segmentation are described and their suitability for the task of VOP creation for MPEG-4 applications is examined. The potential use of segmentation in conjunction with MPEG-7 is further discussed in chapter ten.

5. UNSUPERVISED VIDEO SEGMENTATION

5.1 Introduction

In the previous chapter, it is explained how the development of a reliable robust segmentation technique is essential for the commercial success of MPEG-4 applications which require arbitrarily-shaped VOPs, and which do not have access to *a priori* segmentation information (i.e. if a segmentation process such as chroma-key cannot be used directly in the video capture process). It is further conjectured that segmentation will have an important role to play in the feature extraction process for MPEG-7 applications. In this chapter, a number of approaches to *unsupervised* video segmentation which may be used for the purposes of creating arbitrarily-shaped VOPs are reviewed. The term unsupervised refers to the fact that the segmentation process is completely automatic. In other words, no user interaction is allowed at any stage of the segmentation process. These approaches are reviewed because they are suitable for a particular subset of MPEG-4 applications. Based on the review in this chapter, however, it is explained in chapter six why these approaches are not suitable for all types of MPEG-4 applications, thereby explaining the motivation behind the investigation of a different category of segmentation approaches. The chapter concentrates on approach to segmentation based on motion. The various approaches are briefly described followed by a discussion of their suitability for providing the type of segmentations necessary for MPEG-4 applications.

5.2 Object-based v. Region-based Segmentation

As explained in chapter three, when video segmentation is considered in a compression framework, two approaches are available. The first is object-based, whereby an image is partitioned into regions which have a semantic interpretation (i.e. a moving object). The second is region-based, whereby an image is partitioned into regions which exhibit a degree of homogeneity. When considering automatic segmentation for the purposes of arbitrarily-shaped VOP creation, it is clear that an object-based approach is the best solution. The partition of a scene into semantic objects is dependent on a semantic

interpretation of the scene. This is an ill-posed problem for a computer unless some underlying object model is provided which an algorithm can use to approximate real-world considerations. A region-based segmentation, which does not consider such a model (see chapter 3, Figure 3.1), is ill-suited to the automatic extraction of objects. Simply stated, it is assumed that a semantic object generally consists of a number of homogeneous image regions, but a region-based segmentation algorithm has no means of knowing which regions to group in order to form the object segmentation. However, as is seen in this chapter (section 5.6.1), region-based segmentation tools can be used in order to enhance the result produced by an object-based segmentation process.

5.3 Motion-based Segmentation

In object-based unsupervised segmentation, the approach is to segment objects on the basis of observable motion, which can be estimated automatically. After estimating the motion between two images in a video sequence, areas of coherent motion within the image to be segmented are detected. The underlying assumption is that this coherent motion is indicative of a moving object present in the scene. A segmentation of the image on this basis corresponds to arbitrarily-shaped VOP creation. The estimated motion may be used to initialise the segmentation process in the next image of the sequence to be segmented, thus ensuring that the located objects can be tracked over time. This temporal coherence can be achieved by initialising the estimation of motion parameters in the next image with the calculated motion estimates for the current image, or by projecting the current segmentation into the next image.

5.4 Segmentation via Change Detection

Change detection was first introduced in chapter three where its use within SIMOC was described. As outlined in chapter three, the underlying philosophy of change detection segmentation is that any change between two successive images in a video sequence is due solely to the presence of a moving object. In SIMOC, the segmentation algorithm consists of a very simple global thresholding of the difference image between two successive images in a sequence to produce a change detection mask, and the subsequent processing of this mask based on estimated object motion to obtain an output object segmentation. Change detection-based segmentation can be considered to

be a very simple example of motion segmentation: initial image analysis is driven by the implicit assumption of a moving object present in the scene, and the result of this initial analysis is refined using the estimated motion. A number of drawbacks of the change detection approach were pointed out in chapter three. These drawbacks ensured that SIMOC could not perform efficiently as a segmentation-based video encoder. In this chapter, change detection is considered purely in a segmentation framework.

The main drawback of a restrictive source model (see section 3.5) corresponds to the central weakness of change detection as a segmentation approach. However, the approach also contains a number of other limitations. The first is the sensitivity of the segmentation result to the threshold used on the difference image. This threshold is very sensitive to the particular scene to be segmented. This is illustrated in Figure 5.1 below[12]. By slightly changing the difference image threshold, a much improved segmentation (i.e. one that more closely reflects the actual object) is obtained. Using an unsuitable threshold means that differences between two images, which simply correspond to noise and/or illumination variations, are classified as changed regions. Such threshold sensitivity is extremely undesirable in an automatic segmentation algorithm which is to be applied to a number of different scene types.

Another limitation of change detection is that very often the contours obtained do not accurately reflect the contours of the actual object. Consider the worst case scenario, where the sequence being segmented consists of a white square moving horizontally across a black background. In this case, the change detection mask of Figure 5.2(a) is obtained. This illustrates the fact that even though the required object is actually moving, this movement is not detectable in the difference image and hence it is impossible to obtain the required segmentation. Although this example may appear somewhat contrived, since the white square is not a natural object, the problem also exists for objects which conform to the underlying source model. Consider the image illustrated in Figure 5.2(b), which is taken from a sequence of a textured square moving horizontally across a black background. The change detection mask obtained is depicted in Figure 5.2(c). In this case, the correct shape of the square does not appear in the

12 All change detection segmentations were generated using the SIMOC software implementation of chapter three.

change detection segmentation, since even in the case of natural texture, the movement of parts of the object is not detected in the difference image.

(a) First frame in a sequence with illumination variations

(b) Second frame in a sequence with illumination variations

(c) Change detection mask obtained with a threshold of 2

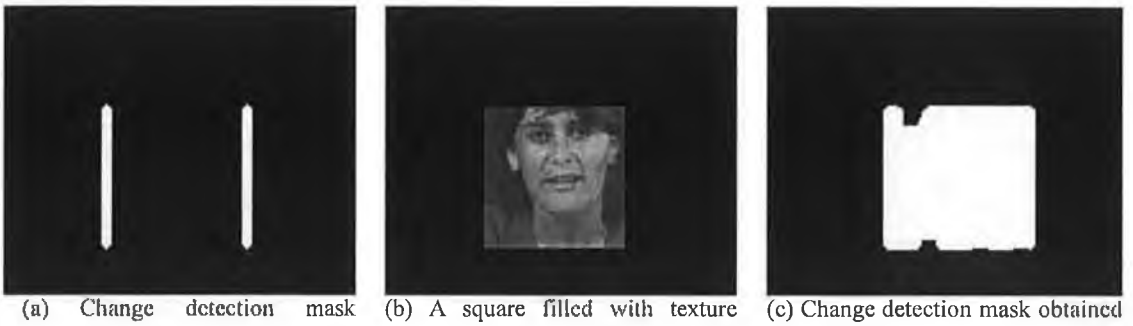
(d) Change detection mask obtained with a threshold of 3

Legend: (c) & (d): white = changed, black = unchanged

Figure 5.1 - The sensitivity of change detection to the difference image threshold

The change detection-based segmentation approach of *Mech and Wollborn* [23][24] attempts to address the drawbacks outlined above. The segmentation technique is illustrated in Figure 5.3 and the main processing steps are described below.

The first step is a global motion estimation step which attempts to compensate for the presence of a moving background or a moving camera in the sequence to be segmented. Given two successive frames of the video sequence I_k and I_{k-1}, an apparent camera motion, such as a zoom or pan, is estimated and the globally motion compensated frame I_{k-1}^{GMC} is produced [24]. This motion compensated frame is used in all subsequent processing.



(a) Change detection mask produced by a white square moving horizontally on a black background

(b) A square filled with texture from the Miss America sequence moving horizontally across a black background

(c) Change detection mask obtained using (b)

Legend: (a) & (c): white = changed, black = unchanged

Figure 5.2 - Change detection fails to extract the required contours

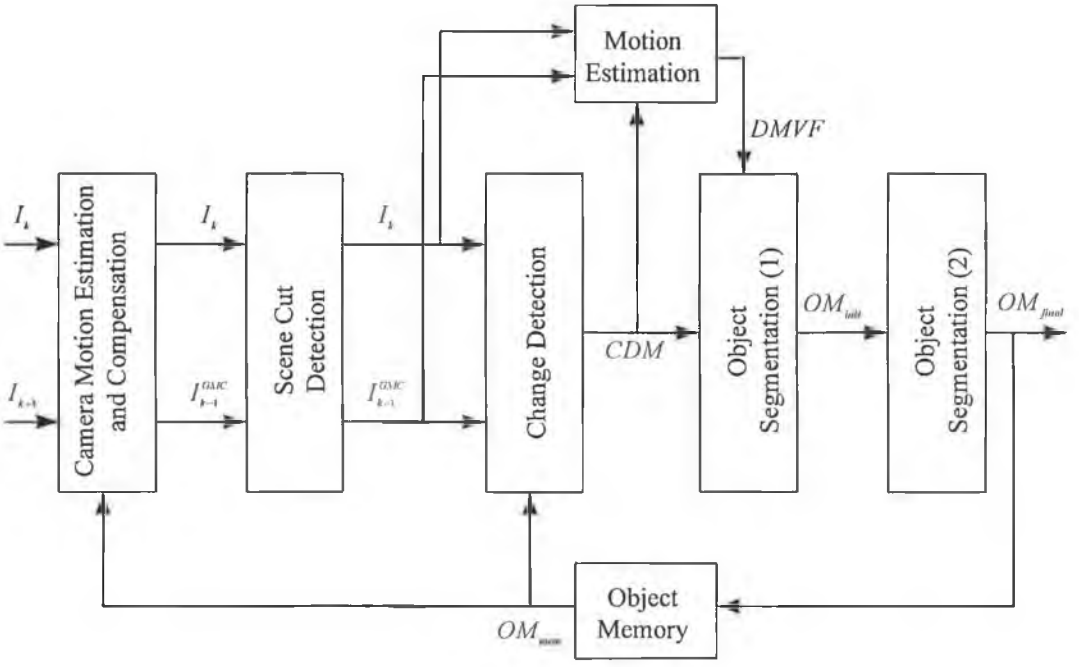


Figure 5.3 - An enhanced change detection segmentation algorithm

A scene cut in the video sequence being analysed is detected in the next step by analysing the difference between the motion compensated previous frame I_{k-1}^{GMC} and the current frame I_k [24]. If a scene cut is deemed to have occurred, the segmentation parameters are reset to their initial values and the segmentation process starts again as if from the beginning of the sequence.

The first task of change detection is to compute the difference image between the current frame I_k and the motion compensated previous frame I_{k-1}^{GMC}. A global threshold is applied to the difference image to obtain an initial change detection mask. The pixels of this mask then undergo an iterative relaxation process, whereby the threshold used to assign pixels to changed/unchanged regions is locally adapted based on (i) the assumed Gaussian camera noise, (ii) the estimated variance of the luminance signal within the changed regions and (iii) the local changed/unchanged configuration of the pixel under consideration [23]. All pixels classified to the object within a recent time interval are added to the change detection mask using the object memory, OM_{mem}, (corresponding to a mask of all pixels classified to the moving object in previous frames) [23]. Finally, the resultant change detection mask CDM is obtained by applying a morphological closing and eliminating small regions.

A dense motion vector field, $DMVF$, within the identified changed area is calculated using the same motion estimation algorithm as that of SIMOC [23]. The first object segmentation step is used to further segment the changed region into a moving object and an uncovered background region, in exactly the same as way as carried out in SIMOC [23]. This produces an initial object mask OM_{init}. The final segmentation step adapts the contours of the initial object mask OM_{init} to the actual contours present in the luminance image using a Sobel operator [23]. The result is the final segmentation mask OM_{final}.

5.5 Segmentation via Motion Model Clustering

The segmentation technique described in this section is embedded in an overall process known as layered decomposition of video, proposed by *Wang and Adelson* in [25][26][27]. This is a process whereby an entire video sequence is decomposed into a set of depth-ordered layers, with one layer for each object present in the scene. A layer is a complete representation of an object over the course of a sequence corresponding to the object's observable luminance, motion and shape information throughout the sequence [25]. To achieve a layered representation, it is necessary to segment each object in the scene at each time instant. These segmentations are then processed to extract occlusion and depth information in order to build the layers [25]. In this section,

the discussion is limited to the actual segmentation technique used to segment each object in each image, which corresponds to arbitrarily-shaped VOP creation.

The segmentation algorithm uses affine motion models which describe motions such as translation, rotation, zoom and shear. The physical interpretation of such models is the motion of a three-dimensional planar surface under orthographic projection [26]. The assumption is that such a model can represent the movement of a semantic object and the objective is to detect multiple such motions in the scene. Motion models are identified, estimated and allowed to iteratively compete for pixel support. The support of each object after the iteration terminates constitutes a segmentation of the scene. The structure of the algorithm is illustrated in Figure 5.4 below.

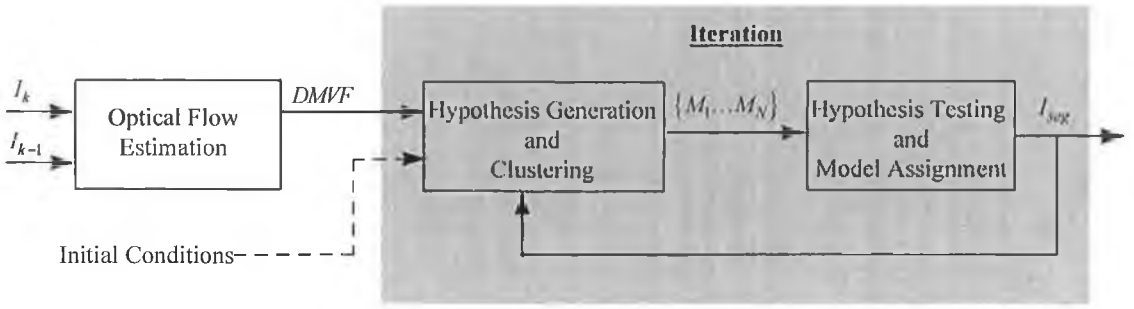


Figure 5.4 - Segmentation via motion model clustering

The first step in the segmentation process is the calculation of the optic flow between the previous image I_{k-1} and the current image I_k to be segmented using a multi-scale coarse-to-fine algorithm based on the gradient [26][27]. This results in a dense motion vector field *DMVF* which describes the motion at each pixel in the current image.

The segmentation iteration is initialised by deriving a set of motion model hypotheses (corresponding to the parameters of affine motion models). The hypotheses from the segmentation of the previous image can be used for the initial hypotheses. Alternatively, the initial hypotheses can be generated by projecting the previous segmentation into the current frame based on the estimated dense motion vector field[13] (these are the initial conditions referred to in Figure 5.4) [27]. When considering the first two images of the sequence, neither a previous segmentation nor a set of motion hypotheses are available.

---

[13] That is, model parameter estimation takes place over the set of pixels classified to the model in the new image.

In this case, the hypotheses are generated by dividing the image into non-overlapping square blocks and deriving motion hypotheses for each block [27]. A standard linear regression technique conditioned on the optic flow data is used for this [26][27]. Either way, in order to converge on the subset of correct models present in the scene, a *K*-means clustering algorithm[14] in the affine parameter space is employed [27]. The affine parameters of the cluster centres are taken as the required motion hypotheses $\{M_1 \ldots M_N\}$ where N is the number of motion hypotheses (corresponding to the number of detected objects). In the next step, each motion hypothesis is tested for each pixel in the image and the pixel is assigned to the motion hypothesis which best describes its motion [27]. The assignment decision is carried out by applying each motion model to the pixel under consideration and calculating the error obtained with this prediction compared to the error obtained using the optic flow motion vector for the pixel. The pixel is assigned to the motion model which produces the smallest error. This results in an output segmentation of the current frame I_{seg}. The entire segmentation process (i.e. hypothesis generation, clustering, hypothesis testing, and model assignment) is iterated until no pixels are reassigned within I_{seg} from one iteration to the next.

5.6 The COST 211ter Analysis Model

With the advent of MPEG-4 video compression, the COST 211ter group decided to change the main focus of its work away from video compression in order to address segmentation for arbitrarily-shaped VOP creation. Furthermore, in light of the new MPEG-7 initiative the group decided to also focus on image analysis for feature extraction [28]. A test model for collaborative development, known as the COST 211ter Analysis Model (AM), was specified. This consists of a set of tools and algorithms for image analysis which allows segmentation and feature extraction within a single framework. The envisaged application scenario for the AM is described in [28]. This application scenario depicts the integrated AM framework, referred to as the Kernel of Analysis for New multimedia Technologies (KANT), which has the task of producing content for an MPEG-4 (or MPEG-7) encoder. KANT consists of the complete AM, a user interface, and provision for user interaction.

---

[14] A detailed description of *K*-Means clustering is provided in chapter eight.

The AM has the task of providing a number of functionalities which are required to enable MPEG-4 and MPEG-7 applications. These functionalities are sometimes termed (i) *unsupervised object detection and tracking*, (ii) *supervised object detection and unsupervised tracking*, and (iii) *object feature extraction and tracking*. The first functionality implies that the AM must be capable of automatically detecting and tracking a semantic object present in a video sequence. The second functionality requires that the AM must support some form of user interaction in detecting and segmenting objects present in an image with the ability to then automatically track these objects. The third functionality requires that the AM must be capable of extracting features which characterise the objects present in the sequence, and of tracking these features over time. The AM is only considered here in terms of the first functionality, which corresponds unsupervised segmentation.

5.6.1 Segmentation via the COST 211ter Analysis Model

The approach taken by COST 211ter is to employ some useful region-based analysis tools in the overall segmentation process. The AM is partly based on the approach of *Alatan et al* described in [29]. This algorithm performs both a region-based segmentation and a motion-based segmentation independently, and uses these two segmentation results as inputs to an intelligent rule-based processor. The rule-based processor attempts to construct the correct segmentation by considering both results. The previously obtained segmentation is also an input to the rule-based processor so that detected objects can be tracked. The rule-based processor is a very flexible processing block, capable of accepting as many inputs as are available (once appropriate rules are defined).

The approach of *Alatan et al* within the AM is illustrated in Figure 5.5. Currently change detection can only be used in the AM via a single switch in the rule processor which essentially replaces the approach of *Alatan et al* with the approach of *Mech and Wollborn*[15] (the dotted lines in Figure 5.5). Since this approach is previously described in this chapter, the discussion here is limited to the structure of the AM based on [29].

---

15 The only difference is that the contours of the moving object are refined using the result of the region-based segmentation and not a Sobel operator as before.

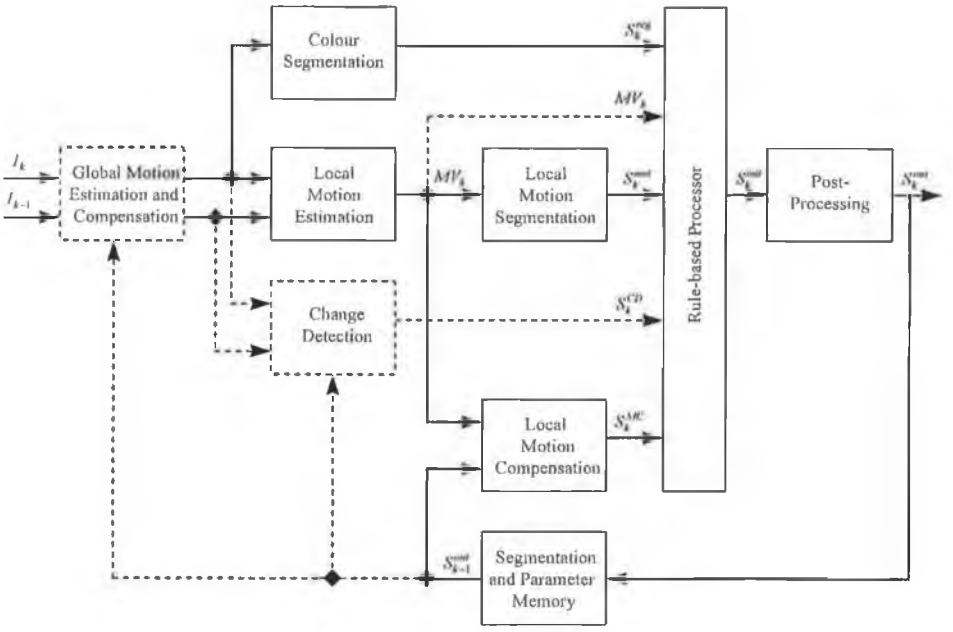


Figure 5.5 - The COST 211ter Analysis Model

Colour segmentation processing produces a region-based segmentation S_k^{reg} of the current frame I_k. It is assumed that I_k contains both luminance and chrominance information (i.e. Y,U and V image components). The region-based segmentation is produced using a Recursive Shortest Spanning Tree (RSST) method [29]. The algorithm starts by considering each pixel as a separate region. A distance metric is defined for calculating the distance between two neighbouring regions in terms of the colour difference, whilst considering the size of the regions. The regions with the smallest distance are merged. This merging process continues until only a required number of regions are present in the segmentation. S_k^{reg} is an input to the rule-based processing block.

Local motion estimation produces a dense motion vector field for the current image MV_k using the scheme of *Bierling* in [12]. Local motion segmentation has the task of producing a motion-based segmentation of the current frame. RSST is also used for this, but this time the algorithm is applied to the dense motion vector field [29]. The output of this processing block S_k^{mot} is also an input to the rule-based processor. The previously obtained segmentation mask S_{k-1}^{out} is motion compensated using the calculated local motion estimates. The result constitutes the final input S_k^{MC} to the rule-based processor.

The task of the rule-based processor is to construct the output segmentation based on the available inputs and a set of defined heuristics. In a first step, every region in S_k^{reg} is associated with one region in S_k^{mot} and S_k^{MC}. This is achieved by calculating the area of overlap of a region in S_k^{reg} with respect to each motion region, and assigning it to the motion region with the greatest area of overlap [29]. Each motion region in S_k^{mot} is then labelled as *moving* or *stationary* by calculating the average motion vector for the region and comparing it to a pre-defined threshold [29]. Each motion region in S_k^{MC} assumes the same *moving/stationary* label as the corresponding region in S_{k-1}^{out}. Each region in S_k^{reg} assumes the *moving/stationary* label of the motion region in S_k^{mot} it is assigned to. Finally, the heuristic rules are applied.

The application of the rules has the following effects. If all the regions assigned to an object in the motion compensated previous segmentation have the same *moving/stationary* label, then these regions are merged and the object is considered as successfully tracked. Otherwise, if the object in the previous segmentation was *moving*, and some regions assigned to this object have different *moving/stationary* labels, then obviously part of the object has stopped moving. Thus the *moving* regions of this object in the current frame are merged to produce the current tracked segmentation of this object. The *stationary* regions are also merged to form a new *stationary object*. If the object in the previous segmentation is *stationary*, and some regions of the current image assigned to this object have different *moving/stationary* labels, then obviously part of this object has started moving (e.g. a static background region may contain an object which starts to move at some arbitrary point in the sequence). Thus, all *stationary* regions of the object are merged to form the *stationary object* in the current frame, and the *moving* regions mapped to this region in the current motion segmentation are merged to form new *moving objects*. It should be noted that regions in the motion compensated segmentation which retain the *stationary* label for a pre-defined length of time are deemed to belong to the scene background. Thus, if part of an object stops moving, it will eventually be assigned to the background. The result of this processing is a segmentation of the current frame S_k^{init} in terms of moving and stationary objects (in which the latter, when considered collectively, forms the scene background).

Post-processing is carried out in order to obtain a refined segmentation. Small regions in the segmentation are detected and merged to larger neighbouring regions [29]. In a second step, the grey-level segmentation mask undergoes morphological filtering to smooth the contours of the regions present. This produces the overall final segmentation S_k^{out}.

5.7 Discussion

Experimental results have verified that the enhanced change detection approach of *Mech and Wollborn* performs much better than simple change detection segmentation [23][24]. The adaptive thresholding scheme reduces the sensitivity of the change detection process. The use of the Sobel operator ensures that output segmentations reflect the contours in the original images. Furthermore, by considering and compensating for the global motion in the scene, the algorithm can be successfully applied to a wider class of scene types. The experimental results obtained using the motion model clustering approach of *Wang and Adelson* indicate the promising performance of this technique [25][26][27]. It is clear from the presented results that different affine motions in the scene can be detected and accurate segmentations obtained for regions exhibiting these types of motion. Experimental results also indicate the promising performance of the approach of *Alatan et al* [29]. Significant motions in the scene are detected and successfully tracked due to the rule-based processor which reflects what actually occurs in the scene using appropriate heuristics. Furthermore, the contours obtained are very accurate due to the use of the region-based segmentation in determining the exact location of colour region boundaries.

Unsupervised segmentation has, however, fundamental unavoidable limitations. The implicit assumption of *Wang and Adelson* and *Alatan et al* is that an area of coherent motion corresponds to an object in the scene. This is not necessarily so, depending on the semantic interpretation of the scene. For example, a semantic object corresponding to a person can consist of multiple motions (e.g. a head moving one way, arms moving another). In the approach of *Alatan et al* and *Wang and Adelson* these motions will appear as separate regions in the final segmentation. If it is known *a priori* that only a

single moving object exists, then these regions can be labelled as a single object segmentation. If this is not the case, however, the algorithm cannot know how to group different motions to form a single object. Whilst in the approach of *Mech and Wollborn*, the motion need not be coherent in order to form an object segmentation (it must simply produce a change between one image and the next), the approach faces the same limitation when dealing with multiple objects: it cannot group different changed areas to form an object segmentation. In a similar manner, even when assuming a single object, a semantic object segmentation obtained using any of these schemes gradually builds up over time. Different parts of the entire object are not guaranteed to move at the same time. The parts which do not move are not detected as part of the object until they do so. Finally, of course, there is no segmentation for the first image in a sequence since the schemes rely on motion which must be estimated between successive images.

These limitations are unavoidable because unsupervised approaches rely on the relatively low-level feature of estimated motion. Another higher level semantic paradigm is required in order to recognise independent motions, or lack thereof, as belonging to the same "real world" semantic object required by the user. This is analogous to the problem of grouping image regions to form a semantic object associated with automatic region-based segmentation schemes.

The attraction of automatic approaches to segmentation is that, since no user interaction is necessary at any stage, they can potentially be used in real-time application environments. Consider, for example, a video surveillance application where it is desired to detect an intruder in a scene and transmit a coded representation of the scene to a central surveillance station for evaluation by a human observer. This scenario is an ideal candidate for a real-time MPEG-4 application. An unsupervised segmentation algorithm is used to detect the required video object (the intruder) and to create the associated arbitrarily-shaped VOPs from the source video. An MPEG-4 video encoder can then use these VOPs to produce a compressed version of the scene with the intruder object at a higher quality than the rest of the scene (using MPEG-4 scalability), thus aiding human evaluation. The first appearance of the object (i.e. the first non empty segmentation mask) could be used to trigger this evaluation. In this surveillance scenario, segmentations such as those produced by the approaches described in this

chapter are acceptable. It is enough to be able to detect that the intruder is present by detecting motions where none should be present, and obtaining the shape of these motions. A segmentation of the entire intruder may not even be necessary. It may be sufficient to segment only the moving part of the intruder and to transmit this section of the scene at higher quality to a human observer who can infer what is actually taking place in the scene.

Given the limitations associated with unsupervised approaches to segmentation, it is clear that such approaches are suitable for a particular class of MPEG-4 applications (i.e. real-time applications requiring content-based functionalities, which can tolerate inaccurate or partial object segmentations). However, as explained in the next chapter, there is another class of MPEG-4 applications which require accurate segmentations of entire "real-world" semantic objects. In this class of applications, the ill-posed problem of segmenting objects based on a human's semantic interpretation of the scene must be addressed. In this case, the segmentation results produced by automatic techniques such as those described in this chapter are not acceptable. However, a segmentation algorithm can be effectively relieved of the burden of semantic object definition through the introduction of user interaction. This has prompted the research community to turn its attention to a different category of segmentation approaches which allow user interaction, in order to address the requirements of these (normally off-line) applications.

6. SUPERVISED VIDEO SEGMENTATION

6.1 Introduction

For a certain class of multimedia applications it is necessary to obtain very accurate segmentations of the required semantic objects for an entire video sequence and thus, the techniques described in the preceding chapter are not suitable. In this case, it is necessary to include user interaction in the segmentation process in order to allow semantic meaning to be defined. This has prompted the investigation of a category of approaches to segmentation known as *supervised* segmentation. The term supervised here refers to the fact that user interaction is allowed. In this chapter, a number of supervised segmentation approaches are described and their overall performance, as well as the type of user interaction employed, is discussed. The most promising aspects of one approach in particular, leads to it being used as the basis of the author's further investigations into supervised segmentation in chapters eight and nine.

6.2 Why Supervised Segmentation?

Consider a content production/editing application in which it is required to extract an object from source video so that it can be combined with different objects in order to create a new scene. This scenario is an ideal candidate for an off-line (i.e. non-real-time) MPEG-4 application, consisting as it does of off-line video object compression and composition. If the content is being produced in a studio, a chroma-key system may be used to generate the required VOPs. A chroma-key system is one in which an object is filmed in front of a colour not contained in the object (normally a blue screen). By simply extracting the background colour from the resultant video sequence (a relatively simple segmentation process), it is possible to recover the shape of the required object. However, such a process may not be available due to the expense involved, or due to the fact that it is required to segment objects from an existing archive of frame-based video. In the absence of a chroma-key system, a segmentation process is necessary in order to create the required VOPs. This segmentation process should be as accurate as possible to avoid missing part of an object, or mixing part of the background of one sequence

with that of another during the composition process, in order that the resultant scene would appear seamless.

In the content production/editing scenario the drawbacks of automatic segmentation techniques (e.g. missing part of an object if it does not move like the rest of the object, or no segmentation available for the first image in a sequence, or slow build-up of the entire object segmentation) are unacceptable. However, with the real-time constraint relaxed, it is possible to allow user interaction in the segmentation process. Including user interaction in the process relieves the segmentation algorithm of the ill-posed problem of trying to extract semantic meaning from the scene, and places this task in the hands of the user. This in turn allows the user the possibility of obtaining exactly the segmentation he/she desires in terms of content (i.e. what is segmented) and quality (i.e. accuracy of the resultant segmented object).

Another advantage of a supervised approach to segmentation is that the segmentation process can be tailored to different end-user applications with little effort and is thus generally much more flexible than an automatic process. Figure 6.1, for example, illustrates two different semantic objects for different applications in the same scene. This first is the girl in the MPEG-4 Weather test sequence and the second is simply the girl's jacket. The first may be required in order to place the girl on a different background in an editing application (e.g. placing a different weather map behind the girl), whilst the second may be useful in creating content for a tele-shopping application (e.g. where the user may request: *"show me the jacket in a different colour"*). The same supervised segmentation approach with different forms of interaction can be used to segment both types of objects making this approach to segmentation applicable across a range of applications[16].

6.3 Object-based v. Region-based Segmentation Revisited

The previous chapter concentrates on object segmentation approaches based on estimated motion in a scene. This is because in a completely automatic framework, objects can only be detected based on their motion. Many tools exist for obtaining

16 The segmentations of Figure 6.1 were both generated using the algorithm presented in chapter nine which was developed by the author.

accurate region-based segmentations (e.g. the watershed algorithm, RSST) but as explained in the previous chapter, a purely region-based segmentation process cannot automatically group image regions to form a required object. However, if user interaction is allowed, it becomes possible to include region-based segmentation in an object segmentation process. Of the three supervised approaches described in this chapter, two are region-based and the other is object-based.

(a) An image from the Weather test sequence

(b) The semantic object is the weather girl

(c) The semantic object is the weather girl's jacket

Figure 6.1 - Illustration of the different natures of semantic objects

6.4 Requirements on User Interaction

There are certain considerations regarding the degree of user interaction which is desirable in a supervised segmentation process. The approach most often used in existing commercial applications is to allow a user to indicate every pixel in an image associated with the required object (or at least pixels belonging to its border). This is a very laborious and error-prone task, which quickly becomes impractical if it is to be repeated for every image in a lengthy video sequence. Ideally, the user should only have to indicate the shape of the object at a particular time instant in the sequence, with further instances of this object throughout the sequence segmented automatically. In other words, user interaction is used to segment the object in one image and this object is then automatically tracked throughout the rest of the sequence.

It is further proposed here that the amount of interaction necessary to segment the object in the initial image should itself not be excessive. The objective is to make the segmentation task as easy as possible for the user. The necessity of indicating every pixel of the object (or its border) should be avoided. The purpose of user interaction should be to provide an automatic segmentation algorithm with some clues as to the

nature of the user's requirements. The segmentation algorithm can then take these clues as input and use them to segment the required object as accurately as possible. The resultant segmentation can be presented to the user for approval. If it is not sufficiently accurate for the user's needs, he/she may decide to completely re-do the segmentation process. Alternatively, he/she may decide to manually refine the automatically generated segmentation. Either way, the entire segmentation process is less time consuming (and hence less expensive!) and the results obtained are tailored to the user's individual requirements. User interaction for refinement is not considered in this chapter (or indeed chapters eight and nine), but is discussed in chapter ten

6.5 Segmentation via Fuzzy Logic

An approach to supervised segmentation using fuzzy logic was proposed by *Steudel and Glesner* in [30] and [31]. The algorithm consists of iterative user interaction coupled to an automatic segmentation process to segment a required object in the initial image of a video sequence. The automatic segmentation algorithm employs fuzzy logic in order to consider multiple information sources in the segmentation process. The segmentation algorithm itself is presented in [31] where it is used to segment still images in order to perform region-based encoding. A fuzzy logic framework used to track the required object after it has been segmented in the initial image is proposed in [30].

6.5.1 User Interaction

The segmentation algorithm of *Steudel and Glesner* segments an image into regions which are homogeneous according to a given criterion. An object present in the image is considered to be made up of an arbitrary number of such image regions. User interaction consists of the user indicating to the algorithm the regions which make up the required object. This is achieved via a single mouse click by the user within each of the object's regions. This indicates to the algorithm a single pixel, sometimes called a *seed*, which is a member of the required region. The automatic segmentation technique is a region-growing process, whereby, starting from the seed pixel, adjacent pixels are added until no more pixels conform to the homogeneity criterion.

The entire process is illustrated in Figure 6.2. The user examines the initial image and decides which object he/she wishes to segment (e.g. the car in Figure 6.2) and also identifies the regions making up this object. The user then starts to segment the object. The algorithm starts with a blank segmentation mask (i.e. an empty VOP). The user clicks somewhere in the first region to be segmented. The region-growing process then segments all pixels which belong to this region. The resultant region is added to the segmentation mask as part of the object. This process is repeated for each image region and the result of the region-growing process is added to the object segmentation mask each time. The final result is a segmentation for the entire object (e.g. the final segmentation in Figure 6.2).

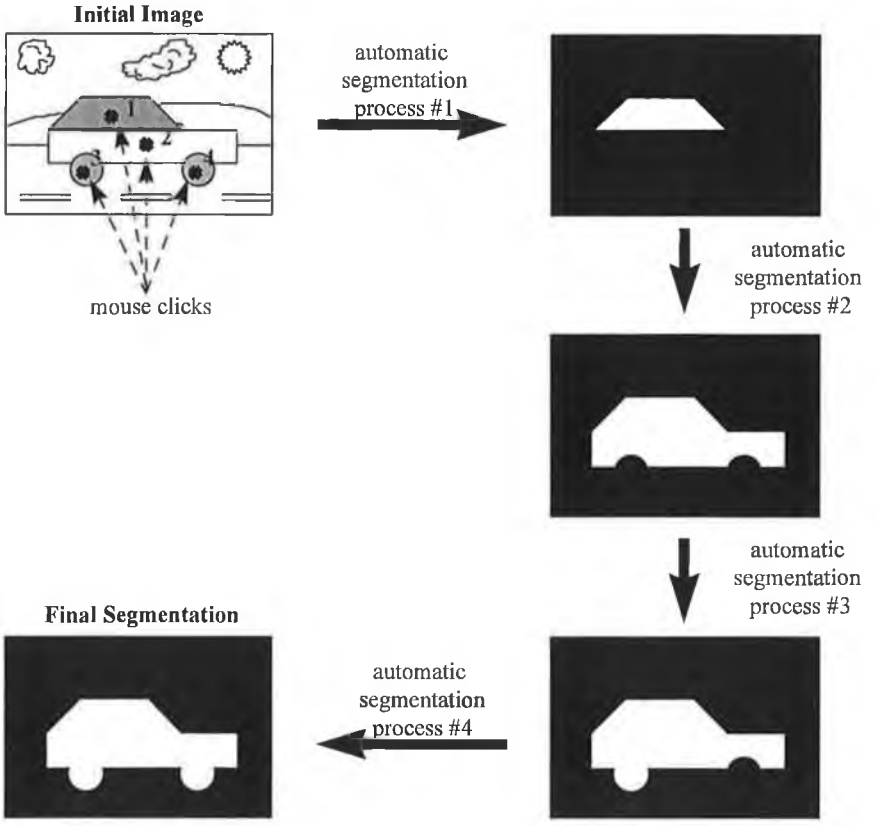


Figure 6.2 - Illustration of *Steudel and Glesner's* approach to user interaction

6.5.2 Object Segmentation

Starting from the seed pixel, region-growing consists of finding adjacent pixels which fulfil the homogeneity criterion. *Steudel and Glesner* use a fuzzy set approach which allows a number of different features to contribute to the homogeneity measure in a

intuitive manner, which mimics the semantic reasoning of a human [31]. The different features considered are (i) the difference in intensity between a pixel under consideration and the mean intensity of the region, (ii) the local intensity gradient at the pixel under consideration, (iii) the size of the region, and (iv) the smoothness of the contours of the region. These features are introduced as linguistic variables in the rule premises of the fuzzy rule-based system and result in single rule conclusions [31].

There are four linguistic expressions (and associated fuzzy sets) for the single rule conclusions, which correspond to four degrees of certainty with which a pixel belongs to a region (*not merge, probably not merge, probably merge,* and *merge*). The following heuristic rules are applied. If the intensity difference is small, then the pixel should be merged, otherwise it should not be merged (the resultant region should be homogeneous in terms of grey-level). If the gradient is small, then the pixel should probably be merged, otherwise not (a strong gradient indicates that the pixel is outside the region's boundary). If the region size is small, then the pixel should be merged (large regions are favoured in order to more quickly build up a complete object segmentation). If the contours of the region are smooth, then the pixel should probably be merged, otherwise probably not (most natural object boundaries are smooth).

Membership functions of the fuzzy sets used in the rule's premises are also defined [31]. The final output membership function is the union of the fuzzy sets of each conclusion after clipping its degree of membership at the degree of membership for the corresponding premise [31]. The final output value (i.e. merge or not) is generated from the output membership function using the centre of gravity method [31]. In summary, for each pixel adjacent to the region (starting from the seed pixel) the features are calculated, the rules evaluated, and a final region membership decision made. This process is repeated until no more pixels are added to the region. This entire process is iterated for each region making up the object.

6.5.3 Object Tracking

In order to track the object segmented in the initial image into the next image in the sequence, a two-step process is employed [30]. In the first step, the motion between the two images is estimated. A half-pixel accurate segmentation-constrained hierarchical

block matching algorithm is employed [30]. Bilinear interpolation is used to produce a dense motion vector field from the motion vector estimates. This motion vector field is used to project the contour of the object in the initial image into the next image to be segmented. The object's contour is represented using a vertex-based approach similar to that of SIMOC [31]. The result of this step is a projection of the object's shape into the current image to be segmented.

Simply projecting the object's shape from image one to the next is not sufficient to generate a segmentation with the required high degree of accuracy. This is due to the nature of the motion estimation/compensation technique, which (being block-based) is only an approximation to the object's true motion [31]. For this reason, the object's shape in the new image is refined in a second object tracking step. This refinement is carried out in a fuzzy rule-based framework, very similar to that used when segmenting the initial image [31]. Each compensated vertex is displaced in either direction along a line orthogonal to the two adjacent connecting line segments in the polygon approximation. The objective is to calculate the displacement which best refines the object's contour. A set of local image features are calculated and used in the fuzzy rule-based system to generate probabilities for each displacement position.

The features used are (i) the intensity gradient of the pixel at the displaced location, (ii) the displacement itself in pixel units, (iii) the smoothness of the contours, (iv) the colour gradient of the pixel under consideration and (v) special consideration for image boundaries [30]. In a manner similar to that outlined above, the calculated features are fuzzified with the membership functions used in the premises of the rules. The rules reflect the following three heuristics: (i) a small displacement is favoured because this is more likely than a large displacement (the vertices have already been motion compensated), (ii) a large gradient in the colour and luminance components indicates that the pixel is near the edge of the object, and (iii) a vertex coincident with the image border is not allowed move from this position unless the suggested displacement is large. Using the same method as above, the fuzzy conclusions for each rule are aggregated to form a single fuzzy main result which is then defuzzified. This produces a probability for each vertex displacement position. The displacement with the highest probability is the refined position for the vertex under consideration. The result of this

step is a refinement of the projected object contours based on local features. This two-step tracking process is repeated for every image in the sequence.

6.5.4 Discussion

The reported results of the segmentation approach of *Steudel and Glesner* are very promising (see [30] and [31]). The scheme allows the accurate segmentation and subsequent tracking of video objects in a variety of scene types. However, there are potential limitations with this approach. The amount of user interaction required may be excessive in certain scene types. It is possible that the object to be segmented may consist of very many small regions, and hence, a mouse click is required for every region (e.g. consider the worst case scenario where the object to be segmented is a person wearing a checked shirt!). In this case, user interaction does not meet the requirements of section 6.4 as it is excessive in nature and difficult to perform. The tracking step may also be problematic. If the object has a complicated shape, then a large number of vertices are required to describe its contour. Each must be successfully tracked in order to obtain accurate object segmentations. This may be difficult if the object changes shape dramatically or moves very fast.

6.6 Segmentation via Multidimensional Statistical Processing

A supervised segmentation approach based on statistical processing was proposed by *Chalom and Bove* in [32]. The underlying philosophy is to consider the scene as a collection of semantic objects, including a background object. The approach relies on user interaction performed on an initial image in order to obtain important clues as to the nature of the objects which the user wishes to segment. The objects are modelled on the basis of this user interaction and the scene itself is modelled as a collection of object models. Each pixel in the image is allowed to compete for membership of one of the available models in an automatic segmentation algorithm, and is eventually assigned to the most likely object model, thereby obtaining a segmentation of the scene. The segmented objects are then tracked in subsequent images in the sequence using one of two techniques.

6.6.1 User Interaction

The purpose of user interaction in the approach of *Chalom and Bove*, is to allow the user to indicate objects in the scene which he/she would like segmented. This is achieved by allowing the user to draw on the scene by simply dragging a mouse over the image. In this way, the user creates a set of *scribbles* for the objects to be segmented. The user gives a label to each scribble whereby each label corresponds to an object to be segmented. The minimum number of labels is two, corresponding to the required object and "everything else" (i.e. the background object). Two different scribbles can have the same label, indicating different parts of the same (possibly disjoint) object in the image. This user interaction process is illustrated in Figure 6.3. The user draws on the image with a mouse and labels the required object (in this case the car i.e. the unbroken scribble in Figure 6.3). The user then labels the background as the second object. In this case, it is more convenient for the user to use two scribbles of the same label in order to indicate the background object (i.e. the two dashed scribbles in Figure 6.3).

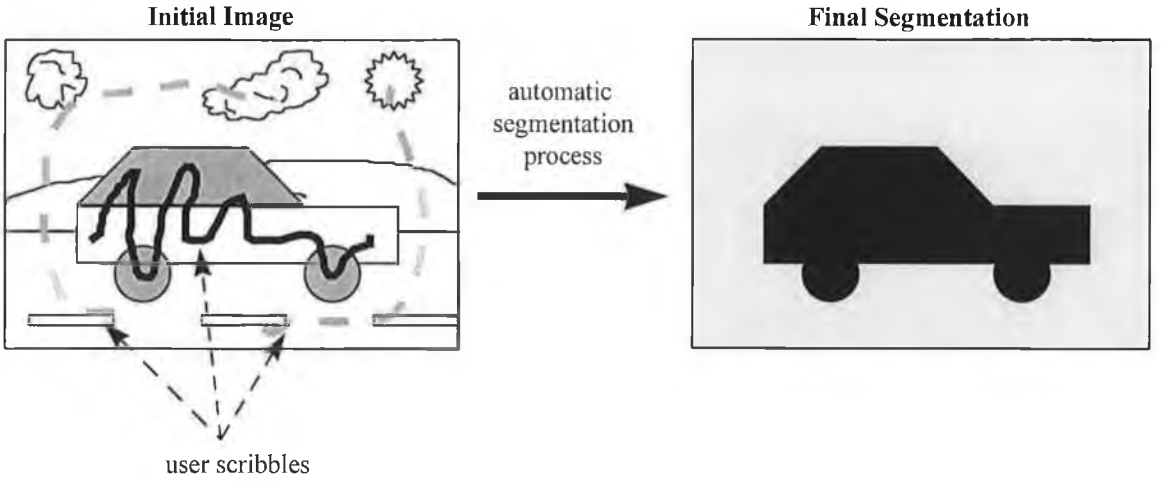


Figure 6.3 - Illustration of *Chalom and Bove's* approach to user interaction

The result of user interaction is a number of important inputs to the automatic segmentation process. The most important input is the number of required objects in the image, which is simply the number of different labels in the set of scribbles. The set of scribbles associated with a particular object also give a very good indication of the location of this object in the image. Finally, the scribbles for an object also give a very good indication of the nature of the object to be segmented (see section 6.6.2).

6.6.2 Object Segmentation

The automatic segmentation process of *Chalom and Bove*, like that of *Steudel and Glesner*, uses multiple information sources. A number of measurements (termed *features*) are made for each pixel in the image. These correspond to luminance, texture, colour, motion, and position information at each pixel [32]. The luminance and colour features for each pixel are simply the pixel's *Y, U* and *V* (or *RGB*) values. The position features are simply the pixel's horizontal and vertical co-ordinates within the image. The motion features for each pixel consist of the vertical and horizontal motion vector components of a dense motion vector field estimated for the image to be segmented using optic flow. Texture features are calculated as statistics on local luminance information. These features are arranged in a multidimensional *feature vector* (see section 8.2) for each pixel in the image [32].

Given this dense feature space for the image to be segmented, the probability distribution function (PDF) of each feature is modelled parametrically. It is assumed that the distribution of each feature across an object can be modelled as the sum of one or more Gaussian PDFs (this concept is further explained in section 9.2). Since this distribution is not known *a priori*, the results of user interaction are used to approximate each object's PDF. The feature vector of every pixel contained in the set of scribbles for a particular object (i.e. every pixel which lies on one of the scribbles) is taken as a member of a training data set for the object. This training data is used to estimate the parameters of the PDF of each feature across the object [32].

The main difficulty is deciding how many modes the PDF should have (i.e. how many Gaussians in the sum, see chapter nine). To determine this, the number of modes is allowed to vary between one and a maximum number and the required parameters are estimated each time using an Expectation-Maximisation (EM) algorithm [32] (the EM algorithm is explained in more detail in chapter seven). For each PDF, the distance between the model and the training data is calculated, and the most appropriate number of modes is chosen [32]. In this way, each object in the image is modelled as a multivariate multimodal Gaussian PDF [32].

The pixels used to construct the training data are considered to be labelled prior to the segmentation process. The task of the segmentation process is to assign one of the available object labels to every other pixel in the image. This is achieved via the estimated PDF model for each object using maximum *a posteriori* (MAP) hypothesis testing [32]. Assuming that the individual features are independent, this reduces to evaluating the PDF for each object for a particular pixel (i.e. feature vector), and assigning the pixel to the object with the highest associated probability [32]. This results in a segmentation of the scene in terms of the labelled objects.

6.6.3 Object Tracking

Two approaches can be used for tracking the segmented objects. The first and simplest approach consists of simply re-using the estimated PDFs in order to segment subsequent images. In this approach, the MAP hypothesis testing and pixel assignment is carried out for every pixel in the new image to be segmented [32]. In the second approach, the PDF estimates are updated for every new image to be segmented. This is achieved by tracking the training data from one image to the next [32]. A motion estimation algorithm is employed and the pixels making up the training data are motion compensated into each new image to be segmented. The feature vectors associated with the compensated training data in the new image constitute a new training data set. Given this training data, the segmentation process continues in the same manner as above.

6.6.4 Discussion

As in the approach of *Steudel and Glesner*, the reported segmentation performance of the approach of *Chalom and Bove* is very promising (see [32]). A number of objects in a scene, moving with complicated motion, can be successfully segmented and tracked. The scheme has, however, certain key advantages over that of *Steudel and Glesner*. The interaction required of the scheme is never excessive and is easy to perform: it simply requires the user to draw on an image, and very often a single scribble per object will suffice, thus meeting the requirements of section 6.4. It should be noted, however, that it is important that the training data derived from a scribble be representative of the entire object (see chapter nine for details of this for a similar scheme). The PDF models used to represent objects prove to be very successful, irrespective of the nature of the objects

to be segmented. Another advantage of this approach is the fact that multiple objects in the same scene can be segmented simultaneously.

There are, however, limitations associated with the approach of *Chalom and Bove*. The training data used (corresponding to user scribbles) in order to formulate object models is very sparse. That is, the number of pixels used to derive model parameters is small compared to the total number of pixels to be segmented. This may result in poor initial estimates of model parameters if, for example, the scribbles are small compared to the objects to be segmented. This in turn will result in poor segmentation results. The first tracking approach suggested (i.e. simply using the parameter estimates calculated in the initial image for MAP hypothesis testing in every subsequent image), takes no account of the temporal evolution of objects in the scene (e.g. object regions appearing or disappearing). Thus, the PDF parameters may become less appropriate for modelling objects over the course of a sequence, leading to a deterioration in segmentation accuracy. Furthermore, if the second approach to object tracking in section 6.6.3 is employed, unless the tracking of the sparse data set is very accurate, object modelling will also deteriorate from one image to the next, producing a gradual reduction in segmentation accuracy. Finally, in order to derive the required PDF estimates, a number of EM algorithms must be employed, which is a potentially computationally expensive and time consuming process.

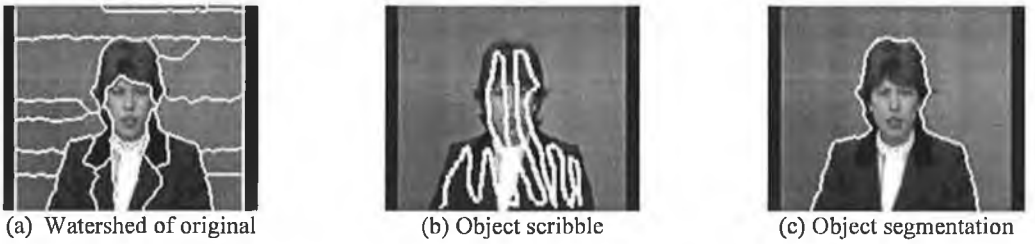
6.7 Segmentation via Mathematical Morphology

As explained in chapter three, morphological tools constitute a very powerful approach to region-based segmentation. The watershed algorithm, for example, produces a segmentation of an image with excellent localisation of image region contours. Grouping watershed image regions to produce a semantic object segmentation can produce very accurate object borders. Thus, applying user interaction to the result of morphological processing can allow an object to be segmented in an initial image in a sequence. Given this segmentation of the object, there exists morphological techniques for automatically tracking this object throughout the rest of the sequence.

6.7.1 User Interaction and Object Segmentation

The approach to user interaction in a morphological framework is different to the types of user interaction outlined thus far in this chapter. In both the approach of *Steudel and Glesner* and that of *Chalom and Bove*, the user indicates to the underlying automatic segmentation process the type of segmentation required (i.e. a scribble says *"this is the nature of the required object"*, a mouse click says *"segment a region starting here, similar to this point"*). In other words, user interaction is used to drive an automatic segmentation process. In the case of the morphological approach described here, however, user interaction is applied *after* the automatic segmentation process has been performed.

User interaction can be applied to the result of a watershed segmentation of the initial image in the sequence. This can be carried out in a number of ways. One example is to allow a user to perform a mouse click in each watershed region making up an object to progressively build up an object segmentation. Alternatively, a user could perform a mouse drag within the object to be segmented, thereby creating an object scribble. Any watershed region which contains at least one pixel of the object scribble is then included in the object segmentation. This latter technique is favoured here as it avoids the potentially excessive amount of interaction necessary with clicking on each region (see section 6.5.4). This technique is illustrated in Figure 6.4 below[17].



(a) Watershed of original luminance image

(b) Object scribble

(c) Object segmentation

Figure 6.4 - Illustration of user interaction on the result of morphological processing

6.7.2 Object Tracking

Given the segmentation of an object in the initial image in a sequence, morphological processing may be used to automatically track the object throughout the rest of the

[17] These results were generated using the same software simulation of the watershed algorithm as in chapter three.

sequence. An approach which has demonstrated reasonably accurate results is the double partition approach of *Marqués and Molina*, presented in [33]. In this approach, the object segmentation is not projected directly into the next image to be segmented. Rather, a fine partition of the previous image is projected into the next image. The regions in this fine partition are tracked and the object segmentation is reconstructed in the new image to be segmented.

Due to drawbacks associated with the watershed algorithm when applied simply to the luminance component (see next section), in the author's opinion, the fine partition of [33] should actually be calculated using a modified watershed algorithm which uses both luminance and chrominance information. The regions of this partition are then homogeneous in terms of colour. This segmentation technique can then be applied to both the initial image and the current image to be segmented. User interaction is applied to the initial image in order to produce the object segmentation in a manner similar to that outlined in Figure 6.4 above. The object is then tracked using the technique referred to above (termed *partition projection*).

The motion between the two images is estimated using a block-based motion estimation algorithm and the markers of the fine partition of the initial image are motion compensated [33]. These compensated markers are then adjusted to the fine partition boundaries in the image to be segmented. A two-step fitting procedure is used for this [33]. The first step, based on geometrical considerations, locates the centre of every region in the new image. In the second step, a cleaning process is applied to the marker regions based on analysis of the image gradient covered by the markers. The exact region contours associated with the markers in the new image are calculated using a watershed algorithm using both contour and texture information [33]. The result of this step is a final fine partition of the image to be segmented. Since the algorithm tracks the labels of the regions which make up the required object, these regions can be merged to produce a segmentation of the object in the current image. This tracking technique can then be repeated for all remaining images in the video sequence.

6.7.3 Discussion

As is clear from Figure 6.4, very accurate segmentations of the required object in the initial image are possible with the morphological approach. However, if a watershed segmentation of the luminance image component is used (as in Figure 6.4 and [33]) rather than the proposed watershed, it may be impossible to accurately segment the object. This is because the watershed will not extract colour region boundaries which do not appear in the luminance image (e.g. two adjacent eqiluminant regions with different colour). This may be avoided (except when the luminance contour is just not present) by using a very fine watershed segmentation, corresponding to the watershed applied to the gradient without filtering. The extreme over segmentation in this case, however, increases the amount of user interaction required (more interaction is needed and must be performed very carefully).

In a manner similar to that of *Chalom and Bove*, multiple objects can be segmented simultaneously with the morphological approach. The tracking results reported (see [32]), however, are not as impressive as those obtained by the author using a scheme similar to that of *Chalom and Bove* (see section 9.6.1 and 9.6.2). This is probably due to the partition projection approach, which may have difficulty in certain scene types to accurately track all the small regions composing an object (e.g. if there are a large number of small object regions and the object moves with complicated motion). The main advantage of the approach is that it requires the least amount of user interaction (i.e. there is no necessity for a background scribble, simply one for the required object).

6.8 Conclusions

All three approaches described in this chapter make use of multiple information sources in the segmentation process. The approach of *Studel and Glesner* uses multiple rules in the fuzzy logic rule-based system, whereby each rule is designed to process a local image feature. Similarly, in the approach of *Chalom and Bove*, local image features are arranged in a feature vector for each pixel. The watershed algorithm used in the approach of *Marqués and Molina* uses both contour and luminance information in order to build a fine partition. In this way, the resultant segmentation in each case is based on simultaneously processing information from a diverse range of sources such as motion,

colour, texture, luminance and shape. The segmentation techniques of *Wollborn and Mech* and *Wang and Adelson* described in the previous chapter deal only with a single information source (the approach of *Alatan et al* uses multiple information sources, corresponding to the use of colour in the RSST). Change detection works solely on the luminance difference image. Polynomial motion modelling considers only estimated motion. Such approaches attempt to segment a semantic object on the basis of a single parametric model, which is very often impossible. The approaches described in this chapter, however, make use of any available information which may be of use in distinguishing an object from the background. This factor, coupled with user interaction to indicate to the algorithm the nature of the required object, results in very useful and powerful segmentation techniques. For this reason, multiple information sources are employed in the author's further investigations into supervised segmentation described in chapters eight and nine.

All the approaches described in this chapter exhibit the type of flexibility illustrated in Figure 6.1. Using the approach of *Steudel and Glesner*, the interaction can be stopped after segmenting the weather girl's jacket or after segmenting the entire weather girl. Using the approach of *Chalom and Bove,* the weather girl can be labelled as one object, or alternatively, simply her jacket can be labelled as the object (this is how the results of Figure 6.1 were generated by the author). Similarly, in the morphological approach it is completely up to the user which regions to group in order to form an object.

The reported performance of these techniques (presented in [30][31][32] and [33]) indicate comparable segmentation results, although the results of *Chalom and Bove* are clearly the best. As outlined in this chapter, what distinguishes the techniques is the type of user interaction employed. The scribble-based approach is more attractive than a mouse-click approach in terms of the amount of user interaction necessary. Furthermore, the scribble-based approach is a more intuitive form of interaction for the end-user, particularly if the user is unfamiliar with the segmentation mechanism (as is very often the case). Drawing a scribble is a more "user-friendly" version of the existing approach of outlining every pixel on an object's border (less care and attention is required for a similar result). For this reason, this form of user interaction is adopted by the author in chapters eight and nine. Another advantage of both the approach of

Chalom and Bove and the morphological approach, is that multiple objects in the scene can be defined and segmented. In other words, the arbitrarily-shaped VOPs for two or more objects can be created simultaneously.

Although the least amount of interaction is required with the scribble-based morphological approach, the most promising segmentation results are those of *Chalom and Bove*. This is due to the method of object modelling which allows a good description of objects as a basis for segmentation. This approach to object modelling is adopted by the author as the basis of extending the region-based segmentation schemes presented in chapter eight to cope with objects.

Whilst the author's investigations in chapter nine are based on the approach of *Chalom and Bove*, the scheme developed attempts to address the limitations of this approach. The sparseness of training data is avoided by automatically augmenting this data (see section 9.3.2). The computationally burdensome iterative application of a number of EM algorithms is avoided by extending user interaction (see section 9.3.1). In the approach of *Chalom and Bove,* MAP testing is performed based on PDF parameters derived solely from the training data. In the author's approach, whilst the initial PDF parameter estimates are derived based solely on training data, *all* available information (i.e. all pixels in the image) are used to refine the parameter estimates. This results in object models which more appropriately reflect the nature of the object. Furthermore, the tracking scheme employed by the author uses *all* available segmentation information for a particular image in order to update the model parameters used to segment the next image in the sequence (as opposed to simply tracking training data to update parameters or using the parameter estimates of a single image). This ensure temporal coherency and reduces the chances of the gradual deterioration of segmentation accuracy possible with either tracking approach of *Chalom and Bove*.

The supervised segmentation techniques described in this chapter are quite different in nature. The approach of *Steudel and Glesner* and the morphological approach are region-based whilst that of *Chalom and Bove* is object-based. In chapter eight, two supervised region-based schemes investigated by the author, which incorporate the user interaction approach of *Chalom and Bove,* are presented. The development of these

schemes is necessary as they form the basis of the author's modified and enhanced version of the scheme of *Chalom and Bove*, presented in chapter nine.

7. MAXIMUM LIKELIHOOD ESTIMATION AND MIXTURE DENSITIES

7.1 Introduction

A commonly encountered task in the field of signal processing is the estimation of the parameters of a probability distribution function (PDF). This amounts to inferring the values of unknown or random quantities from a set of observations which are random variables. For example, consider using a set of observations drawn from a distribution of unknown mean in order to estimate the value of the mean. Alternatively, consider the task of estimating the parameters of a signal, the observations on which are corrupted by noise. These two examples are representative of two different classes of estimation problems known as *parameter estimation* and *random variable estimation,* respectively [34]. In this chapter, the parameter estimation problem is examined. A parameter estimation technique known as Maximum Likelihood (ML) estimation is introduced and an illustrative example of its use is presented. ML estimation of mixtures of PDFs is then presented, again with an example. A particular technique for obtaining ML estimates of mixture parameters in the presence of incomplete data, known as the Expectation-Maximisation (EM) algorithm, is then explained. The use of the EM algorithm in video segmentation (both supervised and unsupervised) is described, and its promising performance when used in this context is discussed. The mathematical techniques described in this chapter are basis of the supervised segmentation approach of *Chalom and Bove*, described in the previous chapter. Consequently, these techniques also form the basis of the author's further investigations detailed in chapters eight and nine: they are used in one of the region-based schemes of chapter eight, and the object-based scheme of chapter nine.

The examples of parameter estimation presented in this chapter (see section 7.2.1 and 7.3.1) were derived by the author and consider only multivariate Gaussian PDFs. Appendix A contains a derivation required by these examples, as well as some simpler examples considering only univariate Gaussian PDFs. It should be noted that whilst

these examples are intended to be illustrative in nature in the context of this chapter, these derivations are used directly in the segmentation techniques described in the following chapters.

7.2 Maximum Likelihood Estimation

Let x be data observed from a distribution X with PDF $p(x|\theta)$ which is completely defined by the parameter set $\theta = [\theta_1 \ldots \theta_m]^T$. Suppose that N observations, $x_1, \ldots, x_N$, are made on the outcomes of the random variables $X_1, \ldots, X_N$. Further, assume that X_i is independent of X_j for all $i \neq j$. The objective of ML estimation is to obtain an estimate of the parameters defining the PDF based on these observations. The approach taken is to view the PDF as a function of θ, for fixed values of the observations [34]. The ML estimate, denoted θ^{ML}, is then that value of θ which makes the given values of the observations the most likely.

The PDF when viewed as a function of θ for fixed values of $x_1, \ldots, x_N$ is known as the *likelihood function*, $l_x(\theta) = l_x(\theta|x_1, \ldots, x_N) = p(x_1, \ldots, x_N|\theta)$. The ML estimate is the value of θ which maximises this function:

$$\theta^{ML} = \arg\max_{\theta} l_x(\theta) \qquad \textbf{Equation 7-1}$$

Since it is the maximising value (i.e. the argument) which is important and not the maximum value itself, constants or terms which do not depend on θ in the likelihood function are often suppressed or ignored. Furthermore, very often it is more convenient to maximise the *log likelihood function*, $L_x(\theta) = \log l_x(\theta)$:

$$\theta^{ML} = \arg\max_{\theta} L_x(\theta) \qquad \textbf{Equation 7-2}$$

This is equivalent to maximising the likelihood function, since the logarithm is a monotonic function.

A necessary (but not sufficient) condition to maximise the (log) likelihood function is for the gradient to vanish at the ML value of θ [35]:

$$\nabla_\theta l_x(\theta)\Big|_{\theta=\theta^{ML}} = \nabla_\theta L_x(\theta)\Big|_{\theta=\theta^{ML}} = 0$$

$$\textbf{where } \nabla_\theta = \left[\frac{\partial}{\partial\theta_1} \quad \cdots \quad \frac{\partial}{\partial\theta_m}\right]^T \qquad \textbf{Equation 7-3}$$

It should be noted that when the observations are independent, the joint PDF of the observations can be written as the product of the individual PDF of each observation i.e. $p(x_1,\cdots,x_N|\theta) = p(x_1|\theta)\cdots p(x_N|\theta)$. In this way, the ML estimate can be written as:

$$\theta^{ML} = \arg\max_\theta \log\left[\prod_{j=1}^{N} p(x_j|\theta)\right] = \arg\max_\theta\left[\sum_{j=1}^{N}\log p(x_j|\theta)\right] \qquad \textbf{Equation 7-4}$$

7.2.1 ML Estimation of a Multivariate Gaussian PDF

In this section, as an example of ML estimation, the parameters of a multivariate Gaussian distribution are estimated. In this case, the PDF of the *random vector* takes the form:

$$p(x|\theta) = \frac{1}{(2\pi)^{\frac{k}{2}}|\theta_2|^{\frac{1}{2}}}\exp\left[-\frac{1}{2}(x-\theta_1)^T\theta_2^{-1}(x-\theta_1)\right]$$

which is completely defined by the parameters $\theta = \left[\theta_1 \quad \theta_2\right]^T$, where $\theta_1 = m_x$ is the mean vector, $\theta_2 = \Lambda$ is the covariance matrix of the distribution, and k is the dimension of the random vector. For the purposes of the following derivations, it is assumed that the individual components of the random vector are independent. The log likelihood function can be written (assuming independent observations of the vector):

$$L_x(\theta) = \sum_{j=1}^{N}\left[-\frac{k}{2}\log(2\pi) - \frac{1}{2}\log|\theta_2| - \frac{1}{2}(x_j-\theta_1)^T\theta_2^{-1}(x_j-\theta_1)\right]$$

To obtain θ_1^{ML}:

$$\frac{\partial}{\partial\theta_1}\sum_{j=1}^{N}\left[-\frac{k}{2}\log(2\pi) - \frac{1}{2}\log|\theta_2| - \frac{1}{2}(x_j-\theta_1)^T\theta_2^{-1}(x_j-\theta_1)\right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} -\frac{1}{2}\left[-\theta_2^{-1}(x_j-\theta_1) - (\theta_2^{-1})^T(x_j-\theta_1)\right] = 0$$

(see Appendix A for the exact details of this last step)

$$\Rightarrow \sum_{j=1}^{N}\left[\theta_2^{-1}(x_j-\theta_1)\right]=0$$

(since $\theta_2^{-1}=(\theta_2^{-1})^T$ when the components of the random vector are independent)

$$\Rightarrow \sum_{j=1}^{N}\left[(x_j-\theta_1)\right]=0$$

$$\theta_1^{ML}=m_x^{ML}=\frac{1}{N}\sum_{j=1}^{N}x_j \qquad \textbf{Equation 7-5}$$

In order to calculate θ_2^{ML}, it is first noted that because the individual components of the random vector are independent, θ_2 can be written as:

$$\theta_2=\begin{bmatrix}\sigma_1^2 & & 0\\ & \ddots & \\ 0 & & \sigma_k^2\end{bmatrix}$$

which in turn means:

$$|\theta_2|=\prod_{l=1}^{k}\sigma_l^2 \text{ and } (\mathrm{x}_j-\theta_1)^T\theta_2^{-1}(x_j-\theta_1)=\sum_{l=1}^{k}(x_{j_l}-\theta_{1_l})^2\frac{1}{\theta_{2_l}}$$

where x_{j_l} is the l^{th} component of x_j, θ_{1_l} is the l^{th} component of θ_1, and $\theta_{2_l}=\sigma_l^2$ is the l^{th} variance of θ_2. Thus, the log likelihood function can be written as:

$$L_x(\theta)=\sum_{j=1}^{N}\left[-\frac{k}{2}\log(2\pi)-\frac{1}{2}\sum_{l=1}^{k}\log\theta_{2_l}-\frac{1}{2}\sum_{l=1}^{k}(x_{j_l}-\theta_{1_l})^2\frac{1}{\theta_{2_l}}\right]$$

It is sufficient to maximise this function for any θ_{2_l}, and the result holds for all $l=1\ldots k$.

To obtain $\theta_{2_l}^{ML}$:

$$\frac{\partial}{\partial\theta_{2_l}}\sum_{j=1}^{N}\left[-\frac{k}{2}\log(2\pi)-\frac{1}{2}\sum_{l=1}^{k}\log\theta_{2_l}-\frac{1}{2}\sum_{l=1}^{k}(x_{j_l}-\theta_{1_l})^2\frac{1}{\theta_{2_l}}\right]=0$$

$$\Rightarrow \sum_{j=1}^{N}\left[-\frac{1}{2\theta_{2_l}}+\frac{1}{2}(x_{j_l}-\theta_{1_l})^2\frac{1}{\theta_{2_l}^2}\right]=0$$

$$\Rightarrow N\theta_{2_l}-\sum_{j=1}^{N}(x_{j_l}-\theta_{1_l})^2=0$$

$$\theta_{2_l}^{ML} = (\sigma_l^2)^{ML} = \frac{1}{N}\sum_{j=1}^{N}(x_{j_l} - \theta_{1_l})^2 \qquad \textbf{Equation 7-6}$$

From Equation 7-5 and Equation 7-6 above, it is clearly seen that the ML estimates of the parameters of a multivariate Gaussian PDF, based on a number of samples and assuming independence of random vector components, are simply the sample multivariate means and variances. This result is used in chapter eight to calculate initial estimates of the parameters of the PDFs used to model region types in the region-based EM segmentation approach (see section 8.4.2). It is also used to calculate initial estimates of the parameters of each mode of the multimodal PDFs used to model objects in chapter nine (see section 9.3.3).

7.3 Maximum Likelihood Estimation of Mixture Densities

In this section, PDFs are considered which can be represented as the sum of a number of PDFs. These are referred to as mixture densities. Mixture densities are considered here because, as illustrated in chapters eight and nine, a mixture density can be used to model PDFs in a segmentation framework: in chapter eight the PDF of the image to be segmented is modelled as a mixture of the PDFs of the types of region present (see section 8.4.1), whereas in chapter nine, the PDF of the image to be segmented is modelled as a mixture of the PDFs of the objects present (which themselves are modelled as mixtures of PDFs of image regions - see section 9.2).

Consider a random variable X drawn from a mixture of g groups, $G_1,\ldots,G_g$, which are mixed in the proportions $\pi_1,\ldots,\pi_g$ where $\sum_{i=1}^{g}\pi_i = 1$. The PDF of X can be represented in the finite mixture form as [36]:

$$p_X(x) = \sum_{i=1}^{g}\pi_i p_i(x) \qquad \textbf{Equation 7-7}$$

where $p_i(x)$ is the PDF of the i^{th} group, G_i. Very often, each PDF in the mixture is assumed to belong to the same parametric family. In this case, the parameters of the mixture are completely defined by a parameter set θ and the mixture density can be written as:

$$p_X(x|\theta) = \sum_{i=1}^{g} \pi_i p_i(x|\theta_i)$$ **Equation 7-8**

where $p_i(x|\theta_i)$ is the PDF of group G_i, which is completely defined by the parameter vector θ_i.

The mixing proportion π_i, can be viewed as the *prior* probability that X belongs to group G_i. The *posterior* probability of X belonging to G_i, having observed X as x, denoted τ_i, is [36]:

$$\tau_i = \Pr[X \in G_i | x] = \frac{\Pr[X \in G_i]\Pr[x \in G_i]}{\Pr[X]} = \frac{\pi_i p_i(x)}{p_X(x)} = \frac{\pi_i p_i(x)}{\sum_{i=1}^{g} \pi_i p_i(x)}$$

Equation 7-9

From Equation 7-9, it can be seen that the posterior probability that an observation x_j belongs to group G_i (where π_{ij} is the prior probability that x_j belongs to G_i) is given by:

$$\tau_{ij} = \frac{\pi_{ij} p_i(x_j|\theta_i)}{\sum_{i=1}^{g} \pi_{ij} p_i(x_j|\theta_i)}$$ **Equation 7-10**

The posterior probability is simply the result of the evaluation of the PDF in question at the observation value, weighted by the prior probability, and normalised by the sum of similar calculations for all PDFs in the mixture. This result is used to calculate posterior probabilities of region-type membership in the E-Step of the EM region-based segmentation scheme described in chapter eight (see section 8.4.3), and also to calculate both posterior mode and object membership probabilities in the E-Steps of the object-based scheme described in chapter nine (see sections 9.3.4 and 9.3.5).

In order to obtain ML estimates of the parameters of the mixture, N observations, $x_1, \cdots, x_N$, on the mixture are considered. As before, it is assumed that each observation is independent. The likelihood function is formed by evaluating the joint densities of the random variables at their observed values, conditional on posterior group membership probabilities [36]. It is thus possible to write the log likelihood function for the mixture as [36]:

$$L_x(\theta) = \sum_{i=1}^{g}\sum_{j=1}^{N} \tau_{ij} \log p_i\left(x_j|\theta_i\right) \qquad \textbf{Equation 7-11}$$

To obtain the required ML estimates for the parameters of the mixture, the log likelihood function is maximised as before:

$$\nabla_\theta L_x(\theta)\Big|_{\theta=\theta^{ML}} = \nabla_\theta \sum_{i=1}^{g}\sum_{j=1}^{N} \tau_{ij} \log p_i\left(x_j|\theta_i\right)\Big|_{\theta=\theta^{ML}} = 0 \qquad \textbf{Equation 7-12}$$

Since the individual PDFs of the mixture are from the same parametric family, it is only necessary to solve this equation once, for θ_i, which then holds for all $i = 1 \ldots g$:

$$\sum_{i=1}^{g}\sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial \theta_i}\left[\log p_i\left(x_j|\theta_i\right)\right] = 0 \qquad \textbf{Equation 7-13}$$

7.3.1 ML Estimation of Mixtures of Multivariate Gaussian PDFs

In this section, as an example of ML estimation of mixture density parameters, the situation whereby each PDF in the mixture is a multivariate Gaussian distribution is considered. In this case, each PDF can be written as:

$$p_i(x_j|\theta_i) = \frac{1}{(2\pi)^{\frac{k}{2}}\left|\theta_{i_2}\right|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}\left(x_j - \theta_{i_1}\right)^T \theta_{i_2}^{-1}\left(x_j - \theta_{i_1}\right)\right]$$

which is completely defined by the parameters $\theta_i = \begin{bmatrix}\theta_{i_1} & \theta_{i_2}\end{bmatrix}^T$, where $\theta_{i_1} = m_i$ is the mean vector of group G_i, $\theta_{i_2} = \Lambda_i$ is the covariance matrix of G_i, and k is the dimension of *random vector* X. As before it is assumed that the components of the random vector X are independent. Equation 7-13 thus becomes:

$$\sum_{i=1}^{g}\sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial \theta_i}\left[-\frac{k}{2}\log(2\pi) - \frac{1}{2}\log\left|\theta_{i_2}\right| - \frac{1}{2}\left(x_j - \theta_{i_1}\right)^T \theta_{i_2}^{-1}\left(x_j - \theta_{i_1}\right)\right] = 0$$

To obtain $\theta_{i_1}^{ML}$:

$$\sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial \theta_{i_1}}\left[-\frac{k}{2}\log(2\pi) - \frac{1}{2}\log\left|\theta_{i_2}\right| - \frac{1}{2}\left(x_j - \theta_{i_1}\right)^T \theta_{i_2}^{-1}\left(x_j - \theta_{i_1}\right)\right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij}\left[-\frac{1}{2}\left[-\theta_{i_2}^{-1}\left(x_j - \theta_{i_1}\right) - \left(\theta_{i_2}^{-1}\right)^T\left(x_j - \theta_{i_1}\right)\right]\right] = 0$$

(see Appendix A for the exact details of this last step)

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij} \theta_{i_2}^{-1} (x_j - \theta_{i_1}) = 0$$

(since $\theta_{i_2}^{-1} = \left(\theta_{i_2}^{-1}\right)^T$ when the components of the random vector are independent)

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij} x_j - \theta_{i_1} \sum_{j=1}^{N} \tau_{ij} = 0$$

$$\theta_{i_1}^{ML} = m_i^{ML} = \frac{\sum_{j=1}^{N} \tau_{ij} x_j}{\sum_{j=1}^{N} \tau_{ij}} \qquad \textbf{Equation 7-14}$$

Since the components of the random vector are assumed to be independent, θ_{i_2} can be written as:

$$\theta_{i_2} = \begin{bmatrix} \sigma_{i_1}^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_{i_k}^2 \end{bmatrix}$$

which in turn means that:

$$\left|\theta_{i_2}\right| = \prod_{l=1}^{k} \sigma_{i_l}^2 \text{ and } (x_j - \theta_{i_1})^T \theta_{i_2}^{-1} (x_j - \theta_{i_1}) = \sum_{l=1}^{k} (x_{j_l} - \theta_{i_{1l}})^2 \frac{1}{\theta_{i_{2l}}}$$

where $\theta_{i_{2l}} = \sigma_{i_l}^2$ is the l^{th} variance of group G_i, $\theta_{i_{1l}}$ is the l^{th} component of θ_{i_1}, and x_{j_l} is the l^{th} component of x_j. Equation 7-13 can then be written:

$$\sum_{i=1}^{g} \sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial \theta_i} \left[-\frac{k}{2} \log(2\pi) - \frac{1}{2} \sum_{l=1}^{k} \log \theta_{i_{2l}} - \frac{1}{2} \sum_{l=1}^{k} (x_{j_l} - \theta_{i_{1l}})^2 \frac{1}{\theta_{i_{2l}}} \right] = 0$$

It is sufficient to solve this equation for any $\theta_{i_{2l}}$, and the result holds for all $l = 1 \ldots k$, and all $i = 1 \ldots g$.

To obtain $\theta_{i_{2l}}^{ML}$:

$$\sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial \theta_{i_{2l}}} \left[-\frac{k}{2} \log(2\pi) - \frac{1}{2} \sum_{l=1}^{k} \log \theta_{i_{2l}} - \frac{1}{2} \sum_{l=1}^{k} (x_{j_l} - \theta_{i_{1l}})^2 \frac{1}{\theta_{i_{2l}}} \right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij} \left[-\frac{1}{2\theta_{i_{2l}}} + \frac{1}{2} (x_{j_l} - \theta_{i_{1l}})^2 \frac{1}{\theta_{i_{2l}}^2} \right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij} \left[\theta_{i_{2_l}} - (x_{j_l} - \theta_{i_{1_l}})^2 \right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij} \theta_{i_{2_l}} - \sum_{j=1}^{N} \tau_{ij} (x_{j_l} - \theta_{i_{1_l}})^2 = 0$$

$$\theta_{i_{2_l}}^{ML} = (\sigma_{i_l}^2)^{ML} = \frac{\sum_{j=1}^{N} \tau_{ij} (x_{j_l} - \theta_{i_{1_l}})^2}{\sum_{j=1}^{N} \tau_{ij}} \qquad \textbf{Equation 7-15}$$

From Equation 7-14 and Equation 7-15 above, it is clearly seen that the ML estimates of the parameters of the i^{th} PDF in a mixture of multivariate Gaussians, based on a number of samples and assuming independence of random vector components, are the weighted sample multivariate means and variances, where the weighting is based on the posterior probability with which each sample belongs to the i^{th} PDF (including a normalisation factor, i.e. the denominator of Equation 7-14 and Equation 7-15). This result is used in the M-Step of the region-based EM segmentation scheme described in chapter eight, where it is used to update the parameters of the region-type PDF models (see section 8.4.5). It is also used to update the parameters of each mode of the multimodal PDFs used to model objects in the object-based segmentation scheme described in chapter nine (see section 9.3.5).

7.4 The Expectation-Maximisation Algorithm

The Expectation-Maximisation (EM) algorithm is a general iterative approach to computing ML estimates. A complete review of the development of the EM algorithm, a review of the relevant literature, and a very general statement of the algorithm can be found in [37]. The algorithm can be used when the available observations are incomplete or can be viewed as incomplete [35].

To illustrate this concept of incompleteness, consider the example of estimating the parameters of multivariate Gaussians in a mixture (section 7.3.1). If the group membership of each observation drawn from the mixture is known, then the ML estimation of parameters is straightforward. The posterior probabilities for each

observation for each group are replaced by a group indicator vector. This is simply a vector containing an entry for each group, which is one if the observation belongs to a group, and zero otherwise. In this case, Equation 7-14 and Equation 7-15 revert to Equation 7-5 and Equation 7-6 respectively for each group. If the group membership information is not available, Equation 7-10 indicates a method for calculating the posterior group membership probabilities (assuming prior probabilities) from which it is then possible to obtain the required group indicator vectors (i.e. $z_{ij} = 1$, if $\tau_{ij} > \tau_{tj}$ for all $t = 1 \ldots g, t \neq i$, otherwise $z_{ij} = 0$, where z_{ij} is the i^{th} component of the group indicator vector z_j for observation x_j). Unfortunately however, Equation 7-10 depends upon the unknown parameter θ_i, which is required to be estimated in the first place. Clearly, it is necessary to classify each observation drawn from the mixture to a particular group in order to obtain estimates of group parameters.

It may not be possible to classify every observation prior to the estimation procedure. For example, it may only be practical to classify a small subset of the observations $x_1, \ldots, x_n$ ($n << N$), with the observations $x_{n+1}, \ldots, x_N$ remaining unclassified. The classified observations are normally referred to as *training data* [36]. It is required, however, to use both the training data (corresponding to the n observations and their group assignments) *and* the remaining observations in the estimation process. This is desirable if it is required to use a very rich data set in order to obtain parameter estimates (see the difference between the author's use of the EM algorithm, and that of *Chalom and Bove* outlined in section 7.5), or in order to update the estimated parameters as new observations (whose classification is unknown) become available in the future.

Formally, the *training data* is denoted $t = (y_1, \ldots, y_n)$, where $y_j = (x_j, z_j)$ which contains both the observations themselves, x_j, and the group indicator vectors, z_j, $j = 1 \ldots n$ [36]. The *unknown data* is denoted as $t_u = (x_{n+1}, \ldots, x_N)$. Such a formulation of the estimation problem is considered to consist of incomplete data, as the group indicator vectors $z_{n+1}, \ldots, z_N$ are missing.

Parameter estimation on the basis of both classified and unclassified observations can be carried out via ML estimation of the parameters of the mixture density:

$$p_X(x|\Psi) = \sum_{i=1}^{g} \pi_i p_i(x|\theta_i)$$

where $\Psi = [\pi \quad \theta]$ denotes the vector of all unknown parameters. The log likelihood function for Ψ formed by t, t_u can be written as [36]:

$$L(\Psi) = \sum_{i=1}^{g}\sum_{j=1}^{n} z_{ij} \log[\pi_i p_i(x_j|\theta_i)] + \sum_{j=n+1}^{N} \log p_X(x_j|\Psi)$$

As explained in the previous section, the ML estimate is obtained by calculating the value of Ψ at which the gradient vanishes, i.e. $\nabla_\Psi L(\Psi)\big|_{\Psi=\Psi^{ML}} = 0$. The complete data for this formulation of the estimation problem is taken as t, t_u and the unknown $z_j, j = n+1 \ldots N$. In this case the log likelihood function can be written as [36]:

$$L_C(\Psi) = \sum_{i=1}^{g}\sum_{j=1}^{N} z_{ij} \log p_i(x_j|\theta_i) + \sum_{i=1}^{g}\sum_{j=1}^{N} z_{ij} \log \pi_i$$

The EM algorithm operates by considering the unknown group indicator vectors as missing data. The first step, known as the Expectation Step (E-Step), is the calculation of the expectation of the entire log likelihood function, conditioned on the observed data and using an initial estimate Ψ^0 for the required parameters:

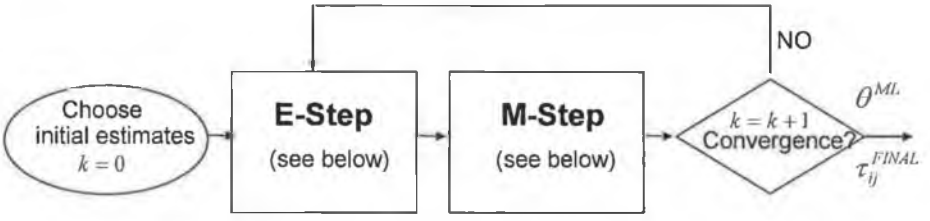
$$U(\Psi, \Psi^0) = E\{L_C(\Psi)|t, t_u, \Psi^0\}$$

This corresponds to simply replacing each unobserved group indicator with its expected value conditional on x_j, i.e. $E\{z_{ij}|x_j, \Psi^0\}$ [36]. In other words, z_{ij} is replaced with the initial estimate of the posterior probability that the observation x_j belongs to group i. This is another way of stating Equation 7-10 given that an initial value of the required parameter θ_i and initial prior probabilities are assumed.

The second step of the EM algorithm, known as the Maximisation Step (M-Step), consists of obtaining that value of θ which maximises $U(\Psi, \Psi^0)$. This is equivalent to solving Equation 7-13, using the posterior probabilities calculated using Equation 7-10. Very often, a closed form solution exists for this, making the EM algorithm a practical

approach. Consider Equation 7-14 and Equation 7-15, for example, which are the required closed form solutions in the case of mixtures of multivariate Gaussian distributions.

The E-Step and M-Step are iterated and the obtained estimates of each iteration (both parameter estimates and group membership probabilities) become the initial estimates for the next iteration. The EM algorithm is stated and the entire process illustrated graphically in Figure 7.1.



- Choose an initial estimate Ψ^0 , consisting of initial parameter estimates θ^0, and an initial set of membership probabilities $\pi_{ij}, i=1\ldots g, j=1\ldots N$
- For $k=0,1,2,\ldots$

E-Step

Compute $U(\Psi,\Psi^k)=E\{L_C(\Psi)|t,t_u,\Psi^k\}$

i.e. $\tau_{ij}^k=\dfrac{\pi_{ij}^k p_i(x_j|\theta_i^k)}{\sum_{i=1}^{g}\pi_{ij}^k p_i(x_j|\theta_i^k)}$ for all $x_j, j=1\ldots N$ and for all $i=1\ldots g$

M-Step

Calculate Ψ^{k+1} such that $\Psi^{k+1}=\arg\max_{\Psi} U(\Psi,\Psi^k)$

i.e. solve $\sum_{i=1}^{g}\sum_{j=1}^{N}\tau_{ij}^k\dfrac{\partial}{\partial\theta_i^k}\left[\log p_i\left(x_j|\theta_i^k\right)\right]=0$

set $\pi_{ij}^{k+1}=\tau_{ij}^k$ for all $j=1\ldots N$ and for all $i=1\ldots g$

set $\theta_i^{k+1}=\theta_i^k$ for all $i=1\ldots g$

Figure 7.1 - The EM Algorithm

The EM algorithm ensures that the likelihood function for the incomplete data cannot be decreased from one iteration to the next, thereby guaranteeing convergence [36]. Convergence is obtained when the calculated estimates remain the same from one

iteration to the next (or at least when the difference between these, calculated using a suitable distance metric, is sufficiently small). More information on the convergence of the EM algorithm, as well as proof of this convergence can be found in [35]. After the EM algorithm has converged, the result is a final ML estimate of the parameters of the mixture, θ^{ML}, and a final set of posterior group membership probabilities for all observations used in the estimation process, $\tau_{ij}^{FINAL}, i = 1 \ldots g, j = 1 \ldots N$. The latter is a very useful by-product of the EM algorithm as these probabilities may be used in a subsequent classification process (e.g. deriving group membership information for the unclassified observations).

7.5 Applications and Discussion

The EM algorithm has been successfully applied in a wide variety of signal processing applications. For example, *Moon* in [35] outlines the use of the EM algorithm for an Emission Tomography (ET) application as well as an Active Noise Cancellation (ANC) application. In [38], *Feder and Weinstein* describe in detail the estimation of the parameters of a number of superimposed signals (in the presence of additive noise) using the EM algorithm. In this section, the discussion is limited to video segmentation applications of the EM algorithm.

As stated in the previous section, the EM algorithm is suited to estimation problems in the presence of incomplete data. Video segmentation can be considered an incomplete data problem. Video segmentation consists of classifying each pixel in an image to one of a number of available groups based on underlying models for the groups. The more accurately the models reflect the groups, the more accurate the segmentation. Unfortunately, it is impossible to accurately estimate the parameters of these groups without knowing the classification[18].

Consider the case of unsupervised segmentation. The motion in a scene often consists of very different types of motion corresponding to camera/background motion, object motion, and even multiple motions within an object. The motion cannot be described by a single model, but rather as a mixture of a multiple different motion models. In order to

---

[18] This is the "chicken and egg" problem referred to by *Alatan et al* in [29].

obtain good estimates of the parameters of these models the pixel support for each model is required. The complete data in this case consists of the pixel luminance data (on which the motion is measured), which is observable, and the group/model assignment of each pixel, which is not. An example of performing unsupervised motion segmentation using an EM-based framework is presented by *Brady and O'Connor* in [39]. In this approach the motions in a scene are modelled as a mixture of polynomial (either affine or quadratic) motion models. The overall approach has three steps consisting of a detection step, a tracking step and a validation step.

The tracking step of *Brady and O'Connor* takes as input the previously obtained motion parameters and the segmentation of the previous image based on these parameters. A spatio-temporal EM formulation is used to segment the current image based on these inputs [39]. Initial pixel model membership probabilities are calculated based on the previous segmentation (i.e. temporal constraints are employed) [39]. The E-Step computes posterior model membership probabilities based on the current estimates of motion parameters. The probability of each pixel is adjusted based on the pixels in the same region in a fine watershed partition of the current image (i.e. spatial constraints are employed) [39]. Based on these probabilities, each pixel is assigned to a particular model i.e. the group indicator vectors are determined as in section 7.4. The M-Step uses these indicator vectors to update the model parameters [39]. These steps are then iterated until convergence. The classification obtained after convergence is a tracked segmentation.

The detection step of *Brady and O'Connor* attempts to formulate hypotheses about the presence of new moving objects in the scene. Pixels within a tracked object's segmentation whose motion cannot be well described using any of the available models are identified. These are referred to as *outliers* [39] (see chapters eight and nine for more details on outliers). A large homogeneous region of outliers indicates the possibility of a new moving object. A motion model hypothesis is formed for this possible new object and this model is allowed to compete for support with existing models using another EM algorithm, which is spatially constrained only [39]. The classification obtained after convergence of this EM algorithm is a proposed segmentation of the current image.

The validation step of *Brady and O'Connor* is used to determine which of the motion hypotheses is correct. A tool known as the Minimum Description Length (MDL) estimate is used for this [39]. This estimates the cost of encoding a particular segmentation. The underlying philosophy is that the best segmentation and motion estimates will result in the most compact representation. To identify motion hypotheses which may be extraneous, each new motion model hypothesis is removed and the MDL estimate is recalculated[19]. If the MDL estimate decreases with the removal of a motion hypothesis, then the model is deemed unnecessary and is removed from the scene [39]. The output of this step is the final segmentation of the image.

A similar application of the EM algorithm in a supervised segmentation scheme is presented in the approach of *Chalom and Bove* [32], described in chapter six. In this approach, training data in the form of sets of pixels assigned to particular objects, is supplied by the user. The approach uses an EM algorithm in order to estimate the parameters of the multimodal PDFs used to model each object (see section 6.6.2). There is no PDF mode information associated with the training data for a particular object. Thus, in this case, the complete data consists of the training data for the object, which is observable, and the mode classification information, which is not.

Assuming initial mode membership probabilities for the pixels in the training data and an initial set of mode parameters, the EM algorithm can be applied to obtain refined mode parameters [32]. As explained in chapter six, a series of such EM algorithms, each with a different number of modes, is applied to determine the best number of modes that should be used to model the object [32]. In a final step, MAP testing is used to derive the segmentation of the scene based on the estimated PDF parameters. The PDF of each object in the scene is evaluated for each pixel in the image, and the pixel is assigned to the object with the highest associated probability (this is analogous to a single E-Step followed by calculation of the group indicator vectors).

As is clear from the approach of *Chalom and Bove*, supervised video segmentation can also be formulated as an incomplete data problem in a manner similar to unsupervised segmentation. Whilst the author's supervised scheme, presented in chapter nine, is based

---

[19] MDL validation is also applied in the tracking step in order to eliminate objects which have disappeared.

on the approach of *Chalom and Bove*, a somewhat different formulation of the EM algorithm is used. In this case, the complete data set consists of all pixels in the image and their associated classifications. In this data set, only the classifications of the supplied training data (i.e. the result of user interaction) are observable. The training data is used to calculate initial parameter estimates and initial object membership probabilities. The EM iteration is then applied using the *complete* data set (i.e. all initially unlabelled pixels are used to update the model parameters in the M-Step)[20]. As explained in section 7.4 and chapter nine, this results in more appropriate object models as a richer data set is used to derive the final model parameters.

The results presented for these EM-based approaches to video segmentation indicate the promising performance of each. In the case of unsupervised motion segmentation, the successful segmentation of a scene into a number of coherent motion regions using the EM algorithm is presented by *Brady and O'Connor* in [39]. In the case of EM-based supervised approaches, the successful segmentation of a variety of semantic objects is presented by *Chalom and Bove* in [32] and by the author in chapter nine. It is clear from these results, and the discussion above, that formulating the segmentation problem in terms of parameter estimation in the presence of incomplete data, and using the EM algorithm as a means of solving this problem, is an elegant way of addressing the "chicken and egg" problem of video segmentation. As such, the EM algorithm constitutes a very powerful tool for the purposes of video segmentation.

---

[20] The explanation of the EM algorithm in this chapter is outlined considering such an approach.

8. TWO SUPERVISED REGION-BASED SEGMENTATION SCHEMES

8.1 Introduction

In this chapter, two approaches to supervised segmentation developed by the author for the purposes of creating arbitrarily-shaped VOPs for off-line MPEG-4 applications are presented. In each case, the form of user interaction found to be most promising in chapter six, i.e. the scribble-based approach of *Chalom and Bove* [32], is employed. Unlike the approach of *Chalom and Bove*, however, the approaches presented here are region-based and in this respect are similar to both the morphological approach and the approach of *Steudel and Glesner* [30] described in chapter six. A semantic object in a scene is considered to be a collection of image regions. A user indicates the approximate location of these regions and an automatic segmentation algorithm then attempts to segment exactly these regions. Finally, the resultant region-based segmentation is used to create the required semantic object segmentation.

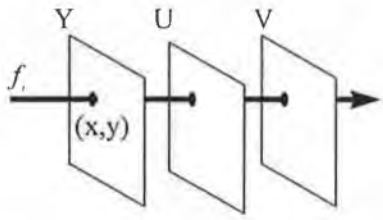
The first approach described in this chapter is based on clustering multidimensional image attributes. The second approach is based on the estimation techniques described in chapter seven, again using multiple image attributes. These two techniques are very closely related. In fact, the second approach is an enhanced version of the first, which allows misclassifications in the segmentation process to be detected and avoided. The description of these approaches concentrates on their usefulness for segmenting objects in still images. As explained in this chapter, both approaches include limitations which make them unsuitable for this task. Object tracking with these approaches is feasible but problematic and given the poor segmentation results obtained, it is not explored here. Whilst the two approaches are ultimately found to be unsuitable for the task of creating arbitrarily-shaped VOPs, they are described here because they form the basis of the author's enhanced version of the scheme of *Chalom and Bove*, presented in chapter nine: the second approach (which itself is an enhanced version of the first) can be extended using the object PDF model of *Chalom and Bove* to a complete object-based

segmentation and tracking approach which outperforms that of *Chalom and Bove* (see section 9.6.4).

Segmentation results obtained using both approaches are presented in the course of this chapter. In each case, these results were obtained by simulating the segmentation algorithm in the ANSI C programming language. The software simulations were carried out under the Unix operating system on a SPARC Ultra workstation. The programs were developed within the software platform used to develop the implementation of the MPEG-4 video encoder which was used by *O'Connor and Winder* in [20]. This software platform has proven to be flexible and portable.

8.2 Segmentation using Multiple Image Features

As explained in section 6.8, incorporating multiple information sources in a segmentation algorithm can produce very good results. For this reason, this approach is employed in both segmentation approaches described in this chapter. Multiple information sources are introduced to the segmentation process in exactly the same manner as proposed by *Chalom and Bove* in [32]. A number of *features* are measured at each pixel in the image and arranged in a multidimensional *feature vector* for each pixel. The actual features used here are a subset of those used by *Chalom and Bove*. Although the results presented in this chapter are not as promising as those of *Chalom and Bove*, ultimately a subset of their features was found to produce comparable segmentation results (see section 9.6.4) and as such, the computational cost associated with the unused features (i.e. calculating optical flow and local texture features for each pixel) was avoided. The features used are each pixel's luminance (Y) and chrominance information (U and V), and each pixel's spatial location within the image. The latter is simply the pixel's horizontal co-ordinate in the image (x) and the pixel's vertical co-ordinate in the image (y). The concept of a feature vector for a pixel is illustrated graphically in Figure 8.1.



$$f_i = \begin{bmatrix} Y & U & V & x & y \end{bmatrix}^T$$

• - Pixel position in image

Figure 8.1 - A multi-dimensional feature vector

8.3 Segmentation via Clustering

Clustering is a technique which can be used to segment an image into a number of regions, each of which exhibit a measure of homogeneity according to a chosen criterion. In the case of the segmentation approach described here, the homogeneity criterion is defined by the choice of features in the feature vector. The features chosen attempt to ensure image regions homogeneous in terms of colour and spatial location. The entire set of feature vectors for the image to be segmented constitutes a multidimensional feature space. The objective is to look for clusters in this feature space. A cluster is a grouping of vectors which are considered "similar" according to an appropriate measure of similarity. Usually this similarity measure is defined as the proximity of vectors in the feature space according to a suitable distance metric. Each cluster has a representative vector associated with it, termed a cluster *centre* (or *centroid*). Iterative approaches are employed to detect and calculate these cluster centres. Implicit in this iterative procedure is the classification of each feature vector (and hence pixel) to one of the available cluster centres, thereby obtaining a segmentation of the image.

The main difficulty in clustering is the initial choice of cluster centres in order to start the iterative process[21] [40]. The segmentation approach described in this chapter avoids this difficulty by introducing user interaction. The user interaction informs the automatic clustering process of the number of required cluster centres, and allows the calculation of initial estimates of these centres. After the clustering iteration has terminated, the result is a region-based segmentation of the image. In order to obtain a semantic object segmentation, subsequent user interaction is required. A graphical

[21] This is equivalent to the problem of extracting semantic meaning faced by any automatic segmentation technique.

overview of the entire segmentation process is presented in Figure 8.2. In the following discussion, the algorithmic details of the various processing steps are explained and the results obtained using this technique are presented.

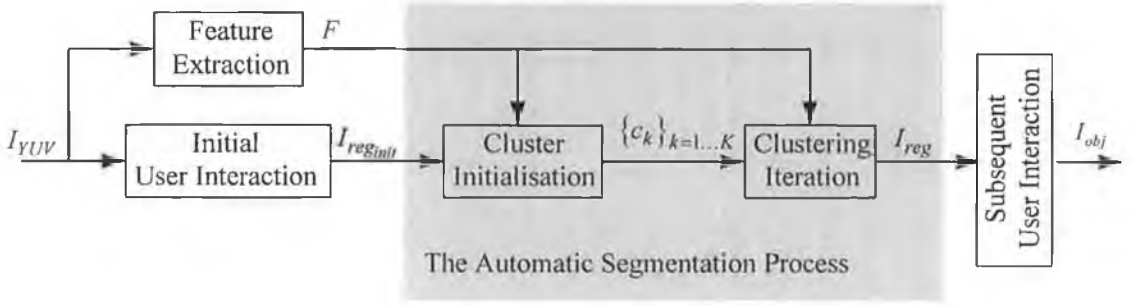


Figure 8.2 - A supervised clustering scheme for object segmentation

8.3.1 Image Segmentation via Clustering: An overview

Clustering in its simplest form can be explained by considering a set of samples $F = \{f_1 \ldots f_N\}$, each of which is to be classified to one of K available classes $\{c_1 \ldots c_K\}$. This classification can be represented by a set of class labels $C^{(n)}$ (since an iterative procedure is used, $C^{(n)}$ is used to denote the classification at the n^{th} iteration). This set has an entry for each sample, and each entry is one of the available class labels. It is assumed that a suitable error function $E(F, C^{(n)})$ exists which measures the optimality of the classification.

The clustering procedure starts by choosing an initial classification, $C^{(0)}$. This classification is then iteratively modified so that the value of $E(F, C^{(n)})$ decreases from one iteration to the next. In this way, the process converges on the optimum value of $E(F, C^{(n)})$. The iteration stops when the classification does not change from one iteration to the next, or in other words, when it is impossible to further reduce $E(F, C^{(n)})$.

In the segmentation approach described here, the set of samples $F = \{f_1 \ldots f_N\}$ corresponds to the feature space (where N is the number of pixels in the image), the set

of classes $\{c_1 \ldots c_K\}$ corresponds to a set of region labels (where K is the number of labels), and the classification $C^{(n)}$ to a segmentation of the image. The initial classification required to start the iteration is obtained via user interaction.

Using the Euclidean distance as the measure of similarity between feature vectors, the function $E(F, C^{(n)})$ can be defined as:

$$E(F, C^{(n)}) = \sum_{k=1}^{K} \sum_{f_i \in c_k} |f_i - \mu_k|^2 \qquad \textbf{Equation 8-1}$$

where the second summation is performed over all samples classified to the k^{th} cluster, and μ_k is the k^{th} cluster centre. It can be shown, that for a fixed set of samples and a fixed set of class assignments, $E(F, C^{(n)})$ is minimised by choosing μ_k as the mean of the k^{th} cluster [40]. In this case, the clustering process is known as K-Means clustering [40] and the algorithm can be compactly described as follows: (i) an initial classification is chosen, (ii) the mean of each cluster is computed, (iii) each sample is reassigned to the cluster with the closest mean, (iv) if the classification of samples has not changed, the iteration is terminated, otherwise it starts again from step (ii).

8.3.2 Feature Extraction

The first step of the segmentation process is to build the feature space F for the image to be segmented I_{YUV}. Calculating luminance and spatial location features for each pixel is a trivial task, as it simply corresponds to extracting each pixel's grey-level value and co-ordinates within the image. Since the 4:2:0 format for progressive video is used, the resolution of the chrominance components is half that of the luminance components. The chrominance components are therefore bilinearly interpolated to the same resolution as the luminance component in order to obtain chrominance features for every pixel.

The values of the different features can have different dynamic ranges. A feature which exhibits larger characteristic values in the distance calculation will tend to dominate the clustering process. In this scheme, it is desired to allow each feature to contribute equally to the clustering process so that the resultant regions in the segmentation are

indeed homogeneous in terms of the chosen features. The features are therefore normalised so that their values are bounded by the range [0,1]. Special consideration must be given to the spatial location features. Even when normalised, these features tend to dominate the distance calculation, and their values are therefore down-weighted by a factor of ten before normalisation. This down-weighting value was chosen based on experimental results. The clustering process was successively applied to a number of test images with down-weighting factors ranging from one to twenty. A factor of ten produced the most accurate segmentations (both in terms of region-based segmentations and the resultant object segmentations), for the largest number of test images. Using other factors, particularly at the extremes of the investigated scale, produced region-based segmentations which were not at all homogeneous, or alternatively, segmentations which were simply a tessellation of the x-/y-plane (thus making it impossible to extract the required semantic object). The factor ten seemed a good compromise between allowing some spatial homogeneity whilst avoiding a spatial tessellation.

8.3.3 Initial User Interaction

User interaction for the clustering segmentation technique follows the mouse drag approach of *Chalom and Bove* [32], albeit in a slightly different manner due to the fact that this approach is region-based, and indeed a specific type of region-based approach (whilst that of *Chalom and Bove* is object-based). When clustering is applied to image segmentation, a one-to-one mapping between cluster centres and image regions is not guaranteed (see section 8.3.7). A particular cluster centre can map to multiple image regions which are spatially disjoint but which exhibit the same type of homogeneity. For this reason, user interaction consists of labelling certain *types* of regions and disconnected scribbles of the same label are allowed. Interaction proceeds by allowing the user to scribble on the original colour image and thereby label types of image regions which are required in the final segmentation. The output of this user interaction is an initial segmentation. This initial segmentation $I_{reg_{init}}$ is very sparse, as it simply consists of the labelled scribbles with all other pixels in the image considered as unlabelled. Clearly, the number of differently labelled scribbles in $I_{reg_{init}}$ gives the number of required cluster centres K.

8.3.4 Clustering Initialisation

Given the initial segmentation, which corresponds to an initial classification for the clustering process, it is possible to calculate initial estimates of the cluster centres. The initial segmentation defines an initial labelling of the feature space. However, only feature vectors corresponding to pixels coincident with a scribble have a label at this time. The set of feature vectors for each set of similarly labelled scribbles is used to calculate an estimate for the associated cluster centre. The k^{th} cluster centre, corresponding to the k^{th} label in the initial sparse segmentation, is calculated as the mean feature vector of this set:

$$\mu_k = \frac{1}{n_k} \sum_{f_i \in c_k} f_i \qquad \textbf{Equation 8-2}$$

where n_k is the number of pixels in the set of similarly labelled feature vectors c_k.

8.3.5 Clustering Iteration

The clustering iteration starts by assigning each unlabelled feature vector to one of the cluster centres. Assignment is based on the Euclidean distance metric, which can be calculated between two feature vectors f_i and f_j as:

$$d(f_i, f_j) = \left[(f_i - f_j)^T (f_i - f_j) \right]^{\frac{1}{2}} \qquad \textbf{Equation 8-3}$$

Each unlabelled feature vector is assigned to the closest cluster centre under Equation 8-3. Given this new classification, the cluster centres are updated using Equation 8-2 and the process is repeated. Each new classification of the feature space constitutes a segmentation of the scene. The iteration is terminated either when the optimality of the classification (calculated using Equation 8-1) or the actual segmentation itself does not change from one iteration to the next (these two conditions are equivalent).

8.3.6 Subsequent User Interaction

The result of the clustering process is a region-based segmentation of the image I_{reg}. However, it is a special case of a region-based segmentation. There are only as many

labels in this segmentation as were specified by the user, and very often spatially disjoint regions have the same label. For the purposes of generating a semantic object segmentation, it is necessary to treat each of these regions as independent entities. In other words, object segmentation takes place at the region level and not the cluster level.

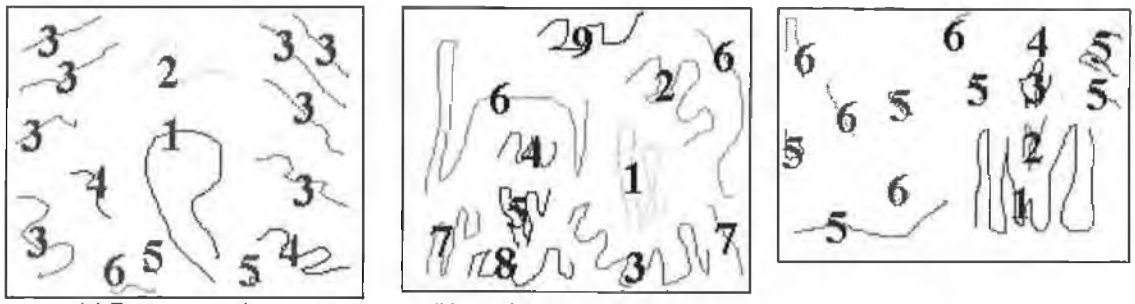
As explained in chapter six, in order to build an object from a region-based segmentation, subsequent user interaction is required on the result of the automatic segmentation process in order to indicate which regions compose the object. This is achieved in this segmentation approach by a mouse drag over the region-based segmentation indicating regions in the required semantic object segmentation. All the pixels of any region coincident with a mouse drag are included in the object segmentation.

8.3.7 Results

Segmentation results for this scheme were generated using three test images, each of which was selected from an MPEG-4 test sequence. The sequences used were QCIF versions of Foreman, Mother and Daughter, and Weather and in each case the initial image in the sequence was chosen in order to generate results. The supplied user interaction overlaid on the original image in each case is illustrated in Figure 8.3. The initial sparse segmentations defined by this user interaction are illustrated in Figure 8.4 as images in which the colour white indicates the unlabelled pixels in the image. The number beside a scribble in these images indicates the label of the associated region type. The segmentations obtained by the clustering process in each case are illustrated in Figure 8.5. In these images, each grey level indicates a different region type label.

(a) Foreman test image (b) Mother and Daughter test image (c) Weather test image

Figure 8.3 - User interaction for supervised clustering process



(a) Foreman test image (b) Mother and Daughter test image (c) Weather test image

Figure 8.4 - Sparse segmentations defined by Figure 8.3

(a) Foreman test image (b) Mother and Daughter test image (c) Weather test image

Figure 8.5 - Segmentation results obtained using the supervised clustering process

The approach produces a region-based segmentation of a scene in terms of homogeneous colour regions. This is due to the consideration of both luminance and chrominance information in the segmentation process. The advantage of considering both these information sources in the segmentation process is illustrated in Figure 8.6(a) which shows the segmentation of the Foreman test image using only one information source, corresponding to a single luminance feature for each pixel. The result is that regions which can be segmented on the basis of colour information (e.g. the man's hat and the background building) cannot be segmented on the basis of luminance information alone. This effect can be avoided by considering chrominance information, but only if the features used are normalised so that they contribute equally to the clustering process. This is illustrated in Figure 8.6(b) which depicts the segmentation obtained for the Foreman test image using both luminance and chrominance features but without normalising the feature space. The feature with the largest dynamic range (the luminance information in this case) contributes most to the clustering process and the final result is not much better than that of Figure 8.6(a).

(a) Single information source

(b) Un-normalised multiple information sources

(c) the effect of spatial features without down-weighting

Figure 8.6 - The effect of multiple information sources in the segmentation process

The spatial location features are included in the feature vectors in an attempt to produce spatially homogeneous regions in the output segmentation. As stated previously, it is not desirable to allow these features to contribute equally to the segmentation process. Figure 8.6(c) shows the segmentation result obtained using all features, but when the spatial co-ordinates are not down-weighted as previously outlined. The spatial features tend to dominate the segmentation process and the result is simply a spatial tessellation of the image. Choosing the down-weighting factor in order to avoid this effect and yet allowing some contribution of spatial constraints is non-trivial. For example, choosing a factor of five will produce more homogeneous regions in the segmentation of the Mother and Daughter test image, but will have an adverse effect on the results for the Foreman test case. This is due to the fact that the amount of down-weighting required is sensitive to the nature of the scene to be segmented and the applied user interaction. The Mother and Daughter test image, for example, is labelled for the most part with one scribble per region type (see Figure 8.4(b)) because the scene is more or less a collection of spatially disjoint homogeneous region types. This lends itself very well to considering spatial constraints in the segmentation process. The Foreman test image, on the other hand, can be considered to consist of large numbers of similar region types (see Figure 8.4(a)). A strong influence of spatial constraints is not appropriate in this case, because the individual regions of a particular type are distributed around the image. The factor ten chosen for the results depicted in Figure 8.5 has proven empirically to be a good compromise for the three test cases.

The results presented in this section indicate the possibility of obtaining a region-based segmentation of colour images using a very simple segmentation process. However, it is

clear from the results of Figure 8.5 that this approach also has a number of drawbacks associated with it. The most obvious drawback is the fact that if all types of region in a scene are not labelled (as is the case in all the test images presented here for example), then there exists image regions which do not have a suitable cluster centre to which to be assigned. The pixels of such image regions will be assigned to the "best" cluster centre of those available (corresponding to the closest in Euclidean distance terms) even though it may be a very poor choice.

The misclassification obtained is illustrated in Figure 8.5(b) and Figure 8.5(c), in which the original user scribbles are visible in the final segmentation. The pixels contained in a scribble are considered labelled prior to the segmentation process and their label cannot be subsequently changed. The pixels of a region which does not have an appropriate cluster centre available are assigned to an unsuitable centre (e.g. the contours on the weather map in Figure 8.5(c) are assigned to the same cluster centre as the weather girl's shirt, the black vertical stripe at the left edge of the Mother and Daughter image is assigned to the cluster centre corresponding to the scribble on the mother's hair). The cluster centre is then updated using these poorly classified pixels and the centre diverges away from the original estimates, to finally represent the unlabelled region type in which the user had (obviously) no interest! Similarly, if two region types which are in fact similar are labelled with different labels (e.g. the two human faces in Mother and Daughter), then there exists an extra unnecessary cluster centre which may not have many pixels assigned to it (as they have already been assigned to the other almost identical centre). The worst case scenario is that no pixels are assigned to this centre at all and it appears as a scribble in the final segmentation with the adjacent pixels assigned to the competing cluster centre (see Figure 8.5(b)).

A threshold on the result of the distance metric used to calculate assignment to avoid this misclassification is feasible. However, it is likely that this threshold would have to be chosen on the basis of consideration of the image to be segmented, and would very likely be different for each new image considered. Since one of the requirements on user interaction in a supervised scheme is that it be kept to a minimum, whilst being easy to perform (see section 6.4), it was desired to avoid such sensitivity for the techniques investigated here and this alternative was therefore not investigated further. Actually,

this effect is avoided by the second region-based approach described in this chapter, which can at least identify unsuitable classifications.

As stated previously, in order to build a semantic object segmentation, it is necessary to treat each spatially disjoint region in the segmentation as a separate entity. It is not enough to work on the basis that certain labels make up an object, but rather on the basis that certain regions of certain labels make up an object. This is illustrated in Figure 8.7(a)-(c), where the subsequent user interaction employed in order to build a semantic object segmentation for each of the test images is overlaid on the segmentation result. The resultant object segmentation (with the image data "painted" inside the object for visualisation purposes) is illustrated in Figure 8.7(d)-(f). Also presented is the inverse of the "painted object" in each case (see Figure 8.7(g)-(i))[22].

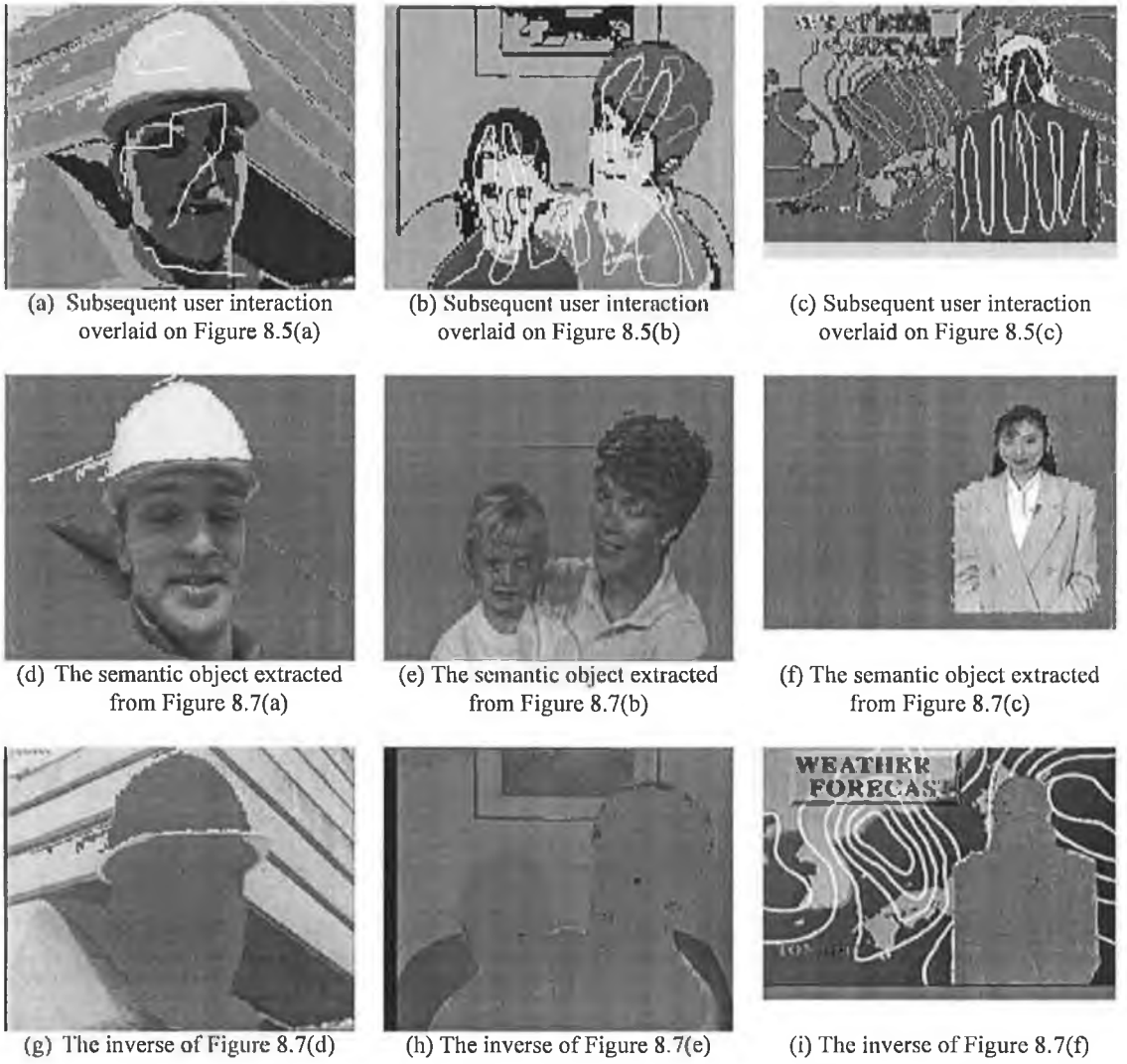
From the results presented in Figure 8.7, it is clear that it is possible to obtain semantic object segmentations using this approach. However, further drawbacks of the approach are evident in Figure 8.7. In each test case, significant user interaction is applied and yet the semantic object segmentations contain "holes" corresponding to small regions which are part of the object but which are not encompassed by the user interaction. This is due to the fact that the regions in the segmentation are not very homogenous and the overall segmentation appears "noisy" (i.e. large numbers of very small regions). The inclusion of spatial features is intended to prevent this. However, the down-weighting factor used is not optimal and the result is that these features have less than the desired effect in the segmentation process.

It is also clear that the subsequent user interaction must be performed very carefully. For example, two disconnected scribbles are required in the case of the Foreman test image in order to avoid large portions of the background being included in the object segmentation. Also, including required regions in the object segmentation (such as the man's shoulder in the Foreman test image) can result in undesirable regions being included in the resultant object segmentation.

---

22 This method of presenting results is widely used in the field of object segmentation as it allows a more complete evaluation. It indicates not only image pixels which are classified to an object but also pixels which are not. In fact, this method has been adopted by the COST 211ter Simulation Subgroup in order to evaluate segmentation results.

In summary, the segmentation results obtained with this scheme are not as accurate as the results obtainable with other supervised approaches, such as those described in chapter six. Furthermore, user interaction in this scheme meets none of the requirements of section 6.4, since the amount of interaction is excessive and is difficult to perform: it is required both before and after automatic segmentation processing, and must be carried out very carefully in the latter case. It is concluded that this scheme is unsuitable as a candidate for easily creating object segmentations with the high degree of accuracy required by future MPEG-4 applications.



(a) Subsequent user interaction overlaid on Figure 8.5(a)

(b) Subsequent user interaction overlaid on Figure 8.5(b)

(c) Subsequent user interaction overlaid on Figure 8.5(c)

(d) The semantic object extracted from Figure 8.7(a)

(e) The semantic object extracted from Figure 8.7(b)

(f) The semantic object extracted from Figure 8.7(c)

(g) The inverse of Figure 8.7(d)

(h) The inverse of Figure 8.7(e)

(i) The inverse of Figure 8.7(f)

Figure 8.7 - Semantic object segmentations using the supervised clustering process

8.4 Segmentation via a Region-based EM Algorithm

In this section, another supervised semantic object segmentation technique developed by the author is presented. The overall approach is exactly the same as that described in the previous section. User interaction (both initial and subsequent) is employed in exactly the same manner. The difference between the two segmentation schemes is the nature of the automatic segmentation process. The clustering approach of Figure 8.2 is replaced with a different automatic segmentation process. In actual fact, the new process is a more sophisticated version of the clustering scheme which addresses some of the limitations of clustering as outlined in the previous section. It produces more homogeneous regions in the automatically generated region-based segmentation.

As in the clustering approach, multiple information sources are considered in the segmentation process to produce a region-based segmentation of the input colour image. The exact same feature vector formulation is used for this, although this segmentation scheme requires that features are treated in a slightly different manner. The automatic segmentation process employed makes use of the ML estimation techniques described in chapter seven. The image to be segmented is assumed to be composed of a number of different types of regions. Thus, the PDF of the feature vectors of the image to be segmented is represented as a mixture of PDFs, where the component PDFs of this mixture are the PDFs of the different region types. Multivariate PDFs from the same parametric family are used to model the distribution of feature vectors across image region types. The underlying approach is to estimate the parameters of the mixture. The mixture model parameters are estimated using the EM algorithm, which as explained in chapter seven, is a robust iterative approach to obtaining ML estimates of the mixture parameters. A by product of this estimation procedure is a classification of each pixel to one of the PDFs of the mixture, thereby producing a segmentation of the original image. In a manner similar to the clustering scheme, the initial estimates used to start this iterative procedure are obtained using the sparse segmentations which are the output of the initial user interaction process.

The use of the EM algorithm in this context is somewhat different to that of *Chalom and Bove* in [32]. *Chalom and Bove* use the EM algorithm (actually, a number of EM algorithms) to derive initial mixture parameters, whereas in the approach described here,

the EM algorithm is used to refine initial parameter estimates (and thereby obtain a segmentation). In the author's approach, initial parameters are calculated using straightforward ML estimation. This is possible because, since the approach is region-based, the training data consists of feature vectors labelled to a single PDF in the mixture. Another difference in the use of the EM algorithm is that, in the author's approach, *all* available observations (corresponding to all pixels/feature vectors in the image) are used, whilst *Chalom and Bove* limit the EM algorithm to the training data. Considering the different nature of the scheme presented here to that of *Chalom and Bove* (region-based as opposed to object-based), the effect of this difference is not immediately apparent. However, this is a fundamental advantage of the author's extended EM-based approach (see chapter nine) over that of *Chalom and Bove* (see section 9.6.4).

The automatic segmentation process is illustrated graphically in Figure 8.8 and in the following discussion the algorithmic details of the main processing steps are explained. This followed by a presentation of the segmentation results obtained using this scheme.

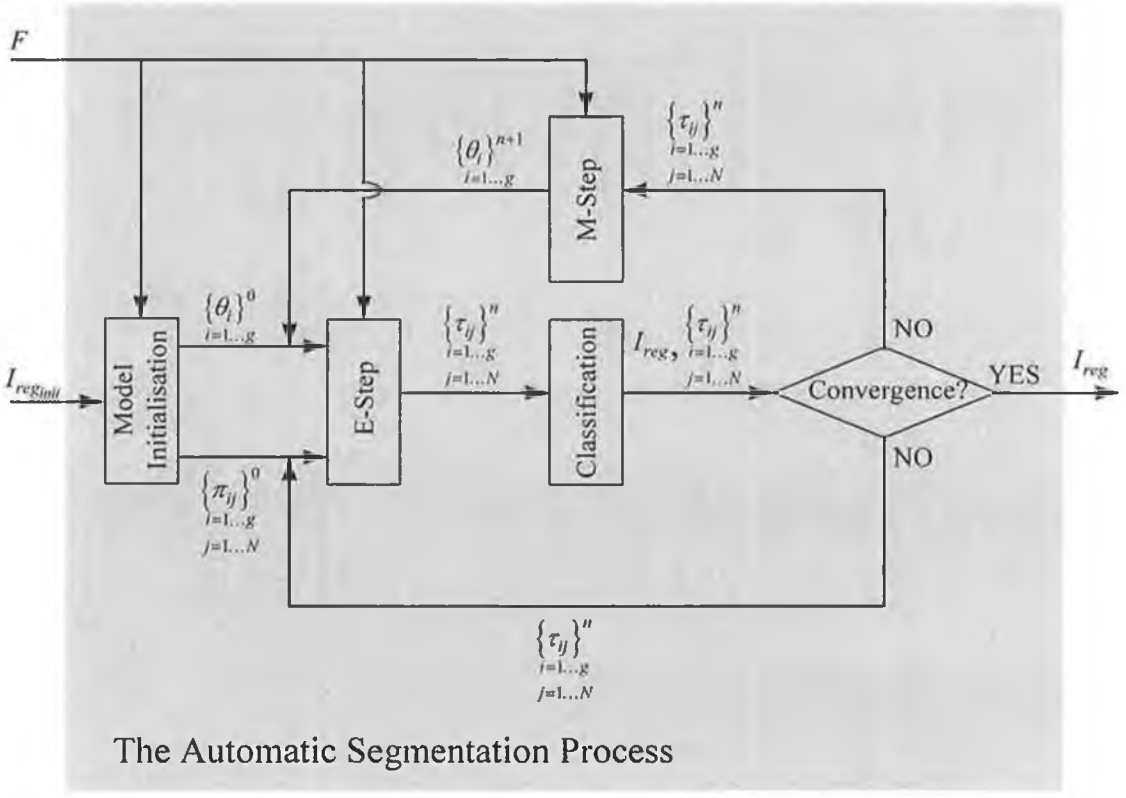


Figure 8.8 - The region-based EM segmentation algorithm

8.4.1 Model Formulation

The distribution of feature vectors across an image region type is modelled using a multivariate Gaussian PDF. The individual features in a feature vector are assumed to be independent. Whilst this may not be strictly true, the results presented indicate that this is a reasonable working assumption. Gaussian PDF models were chosen because as reported in [32] they perform well for natural images in a similar EM-based segmentation formulation. The number of PDFs in the mixture is indicated by the number of different region types in the user supplied sparse segmentation (i.e. the number of different region type labels).

The PDF of the i^{th} region type can be written as:

$$p_i(f|\theta_i) = \frac{1}{(2\pi)^{\frac{k}{2}}|\theta_{i_2}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}\left(f-\theta_{i_1}\right)^T \theta_{i_2}^{-1}(f-\theta_{i_1})\right]$$

which is completely defined by the parameters $\theta_i = \begin{bmatrix}\theta_{i_1} & \theta_{i_2}\end{bmatrix}^T$, where $\theta_{i_1} = f_{m_i}$ is the mean feature vector of region type i, $\theta_{i_2} = \begin{bmatrix}\sigma_{i_1}^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_{i_k}^2\end{bmatrix}$ where $\sigma_{i_k}^2$ is the variance of the k^{th} feature in region type i, and k is the dimension of the feature vector. The mixture of PDFs can be written as:

$$p(f|\theta) = \sum_{i=1}^{g} \pi_i p_i(f|\theta_i)$$

where g is the number of PDFs in the mixture.

The complete feature space, $F = \{f_j\}_{j=1\ldots N}$, (where N is the number of pixels in the image) constitutes a set of independently observed samples on the mixture which can be used to derive the required ML estimates of model parameters. Since features are considered independent during ML estimation, the feature extraction process is in fact simpler than that used in the clustering scheme. The extracted features need not be normalised nor down-weighted because, since each feature is treated independently in the segmentation process, all features are guaranteed to contribute equally to the segmentation process.

8.4.2 Model Initialisation

Model initialisation is performed in two steps, both of which make use of the user-supplied sparse segmentation $I_{reg_{init}}$. The first step consists of deriving initial estimates for the mixture model parameters based on the training data. The feature vectors of pixels contained in a set of scribbles for a particular region type are taken as members of the training data set for that region type. ML estimates of the required parameters are calculated based on these samples. From Equation 7-6 it can be seen that an ML estimate for the mean feature vector of region type i can be easily calculated as:

$$f_{m_i} = \frac{1}{n_i} \sum_{f_j \in S_i} f_j \qquad \textbf{Equation 8-4}$$

where S_i is the set of feature vectors for region type i and n_i is the number of feature vectors in this set. From Equation 7-5 it can be seen that an ML estimate of the variance of the k^{th} feature of region type i can be calculated as:

$$\sigma_{i_k}^2 = \frac{1}{n_i} \sum_{f_j \in S_i} (f_{j_k} - f_{m_{i_k}})^2 \qquad \textbf{Equation 8-5}$$

where $f_{m_{i_k}}$ is the k^{th} component of f_{m_i}. Parameter estimates are calculated for each PDF in the mixture, resulting in an initial set of mixture parameters $\{\theta_i\}^0_{i=1\ldots g}$.

In the second step, the prior probability of a feature vector belonging to a particular PDF in the mixture must be initialised. These prior probabilities are used in the EM iteration when refining the initial parameter estimates. They are represented with a probability vector π_j for each feature vector. This vector has g components and each component is the prior probability with which the feature vector can be assigned to a PDF in the mixture (note: $\sum_{i=1}^{g} \pi_{ij} = 1$, where π_{ij} is the prior probability that f_j belongs to i^{th} PDF). The model membership of feature vectors in the training data set is assumed known. Thus, the prior probability vectors for these feature vectors contain one in the associated region type's entry with zero for all other components (i.e. π_{ij} becomes the group indicator vector z_{ij} for these pixels, see section 7.4). Equal prior probabilities are

assigned for all models for feature vectors in the unknown (i.e. initially unlabelled) data set. This results in an initial set of prior probabilities for the entire image $\left\{\pi_{ij}\right\}^{0}_{\substack{i=1\ldots g \\ j=1\ldots N}}$.

8.4.3 The E-Step

The expectation step of the EM algorithm consists of estimating the posterior PDF membership probabilities of all feature vectors conditioned on *all* available observations, and using the current estimates of the PDF parameters. The posterior group membership probabilities are represented in exactly the same manner as prior probabilities. Each feature vector has associated with it a posterior probability vector τ_j with an entry for each PDF in the mixture (note: $\sum_{i=1}^{g}\tau_{ij}=1$, where τ_{ij} is the posterior probability that f_j belongs to group i). From Equation 7-10, it is clear that these posterior probabilities may be calculated as:

$$\tau_{ij}=\frac{\pi_{ij}p_i(f_j|\theta_i)}{\sum_{i=1}^{g}\pi_{ij}p_i(f_j|\theta_i)} \qquad \textbf{Equation 8-6}$$

In the first iteration of the EM algorithm, the initial parameter estimates and the initial prior probabilities as described above, are used to calculate the first set of posterior probabilities. In subsequent iterations, the posterior probabilities of the previous iteration are used as the prior probabilities for the current iteration. Similarly, the most recently updated set of mixture parameters (calculated in the M-Step of the previous iteration, see below) are used as model parameters for the current iteration.

In addition to posterior probability estimation, the E-step contains a process known as *outlier* detection, which is vital to the performance of the overall segmentation algorithm. An outlier is a feature vector which cannot, with a significant degree of probability, be assigned to any PDF in the mixture. These outliers can be detected based on the result of the evaluation of each PDF at this feature vector. If the result is zero for all PDFs in the mixture then this feature vector is an outlier. In actual fact, outliers are defined as feature vectors whose probabilities are *close* to zero for all PDFs. Outliers are identified by detecting such feature vectors. An outlier decision is carried out

independently for each feature of a feature vector for a particular PDF. It is based on calculating the distance of the feature being tested from the PDF mean value for this feature, and thresholding this distance as follows: if the distance is greater than 2.5 times the standard deviation, it is assumed that the evaluation of the PDF for this feature is a value very close to zero. If a feature vector contains one such feature, its membership probability for the entire PDF is set to zero. If a similar result is obtained for all PDFs in the mixture with this feature vector, then it is classed as an outlier. This method of outlier detection is used in the motion segmentation technique of *Brady and O'Connor* (presented in [39]) where it proves to be quite effective. Detected outliers are removed from subsequent processing of the image and result in unlabelled regions in the final segmentation. Outlier detection and removal is necessary because otherwise these inappropriate feature vectors will be used when updating model parameters in the M-Step. This will contaminate the estimates of model parameters, cause them to diverge from the correct estimates, and eventually result in misclassifications.

8.4.4 Classification and Convergence Testing

Given the posterior probabilities, it is possible to derive a segmentation, I_{reg}, of the scene. This is achieved by classifying each feature vector (and hence it's associated pixel) to a particular PDF in the mixture. The feature vector is classified to the PDF with the highest associated probability in the posterior probability vector (i.e. by calculating the group indicator vectors of all pixels in the image as outlined in section 7.4). This classification procedure is used to control the EM iteration. The segmentation derived from the current iteration of the algorithm is compared with that derived from the previous iteration. If no pixels have been reassigned between one iteration and the next, then the iteration is terminated, and the current classification is output as the region-based segmentation of the scene. Otherwise, the posterior probabilities are passed to the M-Step and the iteration continues.

8.4.5 The M-Step

The objective of the maximisation step is to calculate updated parameter estimates, based on the observed data and the posterior probabilities of group membership

calculated in the E-step. From Equation 7-14, it can be seen that given the posterior probabilities, the updated mean feature vector of the i^{th} PDF can be calculated as:

$$f_{m_i} = \frac{\sum_{j=1}^{N} \tau_{ij} f_j}{\sum_{j=1}^{N} \tau_{ij}} \qquad \textbf{Equation 8-7}$$

Similarly, from Equation 7-15, the variance of the k^{th} feature in the i^{th} PDF can be updated as:

$$\sigma_{i_k}^2 = \frac{\sum_{j=1}^{N} \tau_{ij} (f_{j_k} - f_{m_{i_k}})^2}{\sum_{j=1}^{N} \tau_{ij}} \qquad \textbf{Equation 8-8}$$

Assuming that the input to the M-Step is a set of posterior probabilities calculated in the n^{th} iteration of the algorithm $\left\{\tau_{ij}\right\}_{\substack{i=1\ldots g \\ j=1\ldots N}}^{n}$, the result of the above calculations is a set of mixture parameters to be used in the next iteration $\left\{\theta_i\right\}_{i=1\ldots g}^{n+1}$.

8.4.6 Results

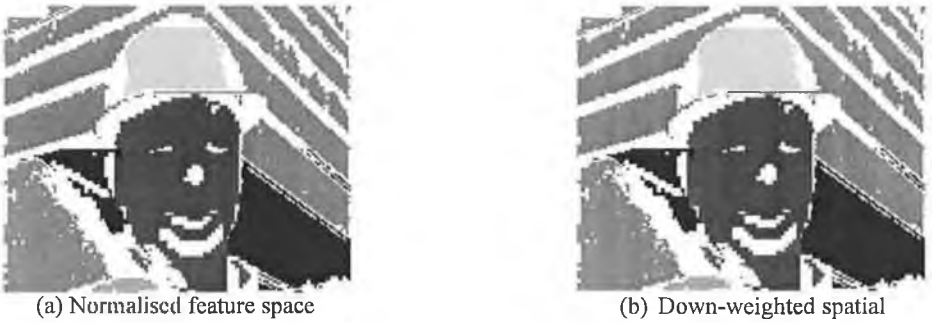
In this section, the segmentation results obtained using this scheme are presented. In order to compare the performance of this scheme with the clustering approach, the exact same test conditions are employed. The same user supplied scribbles as illustrated in Figure 8.3 and Figure 8.4 are used for the initial sparse segmentations. The region-based segmentations obtained based on this user interaction are illustrated in Figure 8.9. As before, the results are presented as images in which each different grey level represents a region type label. The white regions in these images indicate outliers in the segmentation process.

Figure 8.10(a) shows the segmentation result obtained for the Foreman test image with normalisation of the feature space. The image of Figure 8.10(b) depicts the segmentation result obtained using down-weighted spatial location features. The results obtained are identical to that of Figure 8.9(a). This to be expected considering the

feature independence assumption incorporated into the ML estimation process. This result holds for all test images considered.

(a) Foreman test image (b) Mother and Daughter test image (c) Weather test image

Figure 8.9 - Segmentation results obtained using the supervised EM-based process



(a) Normalised feature space (b) Down-weighted spatial location features

Figure 8.10 - The independence of features in the EM-based process

It is clear from these results that the supervised EM-based segmentation approach produces much cleaner segmentations than that of the clustering approach. Regions in this segmentation are more homogeneous, because since outliers are considered, very little misclassification occurs. The resultant segmentation also reflects the user's labelling of the scene. However, there are a large number of outliers present in the final segmentation. The nature of these outliers for each test image is illustrated in Figure 8.11 by "painting" the actual image data onto the outlier locations.

Outliers represent image region types which were not labelled by the user during user interaction (e.g. in the case of Foreman, the logo in the top left of the image or the diagonal stripes on the building). They also very often represent region contours. This is due firstly to the fact that a region's contour may not appear in the same location in both luminance and chrominance components, and secondly due to the fact that regions are

modelled as Gaussian distributions which do not expect a sharp cut off at region boundaries. It can also be seen that some outliers represent sub-sections of region types which were actually marked by the user (e.g. parts of the man's hat in Foreman, parts of the background wall in Mother and Daughter, and parts of the woman's torso in Weather). This is because these sub-regions are in fact separate homogenous colour regions, even though they may not appear so to a user. For example, a shadow or strong illumination variation within what may be user-defined as one region type, will appear as an outlier region.

(a) Foreman test image (b) Mother and Daughter test image (c) Weather test image

Figure 8.11 - The nature of outliers in the EM-based process

Outliers representing image region types not indicated by the user are not undesirable in the final segmentation. The segmentation algorithm cannot be relied upon to correctly classify the pixels of such regions. In fact, an automatic classification technique for outliers will produce inappropriate classifications such as those which occur in the clustering process. This will affect the overall segmentation result in an adverse manner as it means that the estimates of model parameters become contaminated with inappropriate samples. In the case of a supervised approach, it makes sense to present these outliers to the user along with the segmentation result in order to obtain further instruction on how to treat these outliers (segmentation refinement is further discussed in chapter ten). Outliers corresponding to sub-regions within labelled region types, however, are undesirable in the final segmentation as they lead to less homogeneous region types in the final segmentation.

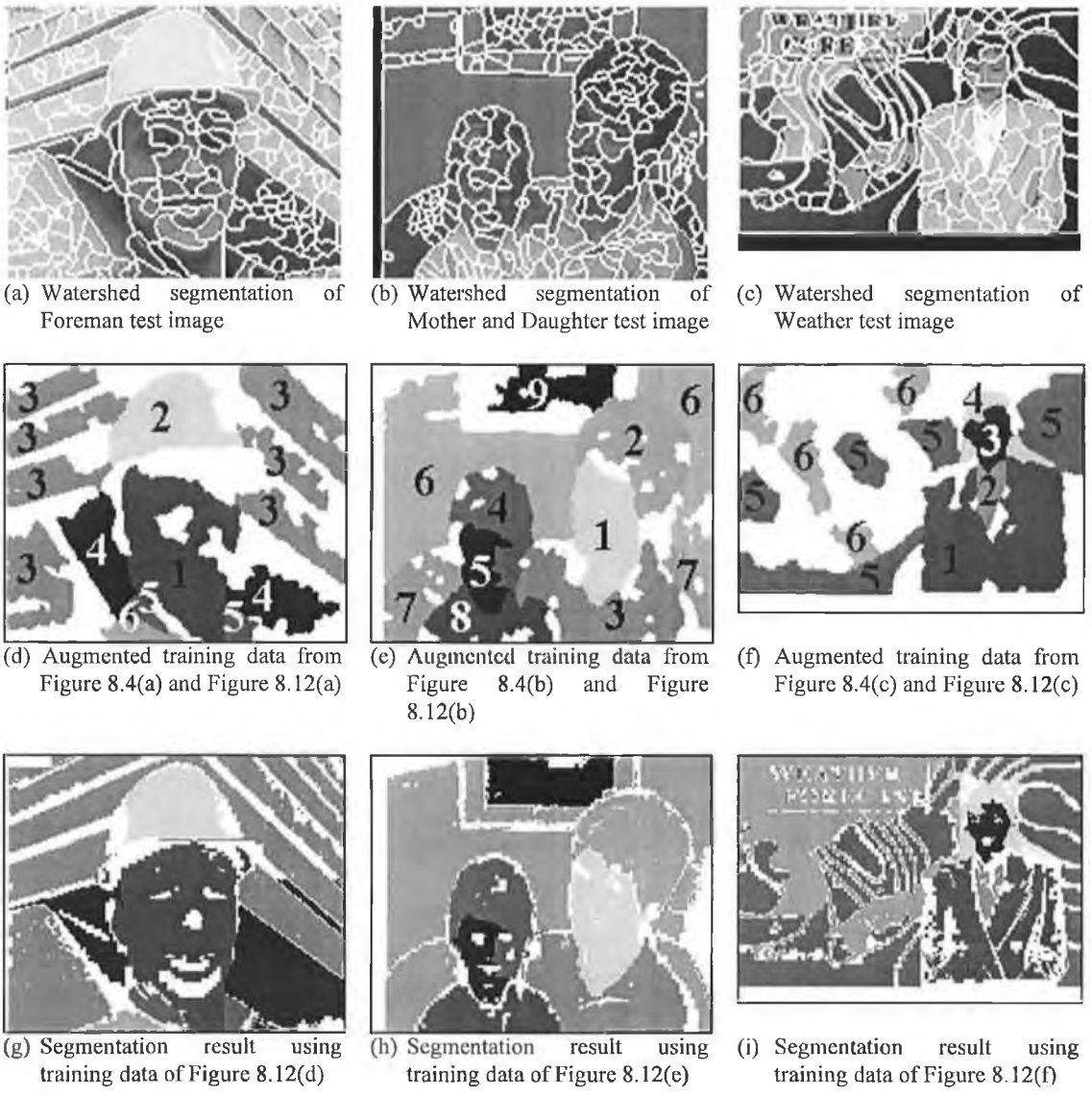
Outliers corresponding to sub-regions occur due to the sparse nature of the training data for each region type. A scribble consists of a small number of pixels which may not adequately reflect the variance of features within a region type. A method of reducing

the number of such outliers is to automatically augment the available training data prior to the automatic segmentation process. This is achieved using a watershed segmentation of the luminance component of the original image. The watershed is calculated on a morphologically filtered version of the original image. An open-close filter of size 3×3 is used to ensure a fine watershed partition. The augmentation proceeds by considering each scribble in conjunction with the watershed segmentation. All pixels of any watershed region, which contain at least one scribble pixel, are included in the associated training data set. The watershed segmentation of each test image is shown in Figure 8.12(a)-(c). The augmented training data sets in each case are depicted in Figure 8.12(d)-(f). Each augmented scribble is shown as a different grey level region with the label of the associated scribble drawn within each region. White regions in these images correspond to pixels not included in the training data (refer to Figure 8.4). The segmentation results obtained using the augmented training data are depicted in Figure 8.12(g)-(i) (the white regions in these images correspond to outliers).

It is important to note that the classification of training data is assumed known prior to the segmentation process, and thus, need not be re-estimated. However, pixels within the training data can be detected as outliers. This is necessary to allow for small misclassifications of training data (e.g. if a user's scribble is slightly outside an image region, or the watershed augmentation in the luminance component results in chrominance information from neighbouring regions being included). It results in a gradual build up of outliers as the estimation process gradually converges on the most dominant sub-region within a labelled region-type (after all, a region type is modelled using a single Gaussian PDF). This gradual build up of outliers for the Mother and Daughter test image is illustrated in Figure 8.13, where again, the original image data has been "painted" onto outlier locations.

In order to make the overall segmentation process less sensitive to sub-regions within region types, the iteration is terminated early in order to avoid this gradual build-up of outliers. The iteration is stopped when only a small percentage of pixels is reassigned between iterations. The results depicted in Figure 8.14 show the segmentations obtained by stopping the iteration when only 10% of all pixels in the image are re-assigned between one iteration and the next (as before the outliers are presented as white

regions). Using the augmented training data and early termination, the overall number of outliers is dramatically reduced, and the EM-based process results in very clean segmentations.



(a) Watershed segmentation of Foreman test image

(b) Watershed segmentation of Mother and Daughter test image

(c) Watershed segmentation of Weather test image

(d) Augmented training data from Figure 8.4(a) and Figure 8.12(a)

(e) Augmented training data from Figure 8.4(b) and Figure 8.12(b)

(f) Augmented training data from Figure 8.4(c) and Figure 8.12(c)

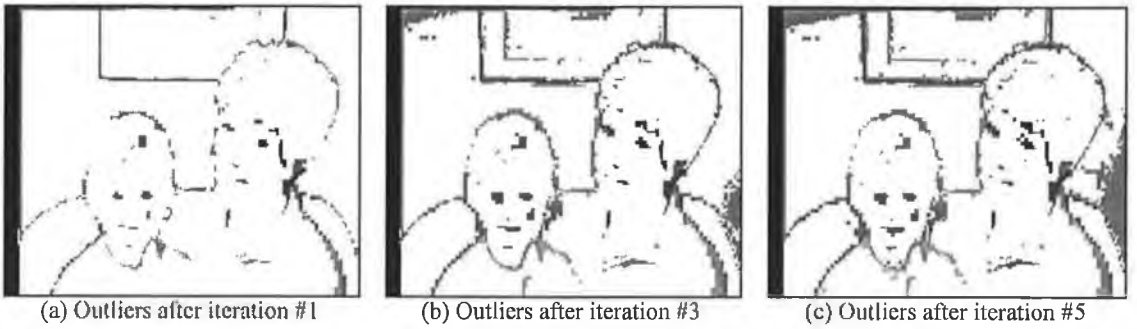
(g) Segmentation result using training data of Figure 8.12(d)

(h) Segmentation result using training data of Figure 8.12(e)

(i) Segmentation result using training data of Figure 8.12(f)

Figure 8.12 - Segmentation results after automatic augmentation of training data

The EM-based approach can be seen to perform much better than the clustering approach. The individual regions in the EM region-based segmentations are more homogeneous. There is a direct one-to-one mapping between a user scribble and a region-type in the final segmentation. Pixels whose misclassification would degrade the overall performance of the algorithm are identified and removed from further processing (and can optionally be presented to the user for further classification).



(a) Outliers after iteration #1 (b) Outliers after iteration #3 (c) Outliers after iteration #5

Figure 8.13 - The gradual build-up of outliers in the EM-based process

(a) Foreman test image (b) Mother and Daughter test image (c) Weather test image

Figure 8.14 - Segmentation results via early termination of the EM iteration

The improved performance of the EM-based algorithm is not surprising since this approach can be considered to be an improved version of the simple *K*-Means clustering approach. Both approaches attempt to classify samples between a number of competing groupings. In the clustering approach, this is achieved based on the distance between the sample and the various groupings within the feature space. This takes no account of the variance that normally exists within the groupings (particularly when these groupings correspond to image region types in natural images). The EM-based approach, on the other hand, takes this variance into account by using a Gaussian PDF to model region type PDFs. In order to update grouping parameter estimates in the clustering approach, it is necessary to make a "hard" classification of the available samples. In this way, the approach is sensitive to misclassification which will propagate over time. The EM approach avoids this by allowing a sample to contribute to a suitable degree to the parameter estimates of a number of competing groupings. Furthermore, probable misclassifications can be detected (via outlier detection) and removed from subsequent processing, in order to avoid contamination of parameter estimates. A similar

functionality in the clustering approach would necessitate an image dependent threshold on the classification metric (and even this may not be sufficient).

The updated graphical representation of the complete semantic object segmentation algorithm, including the augmentation of the training data, is illustrated in Figure 8.15. In this diagram $I_{reg_{aug}}$ is the augmented sparse segmentation.

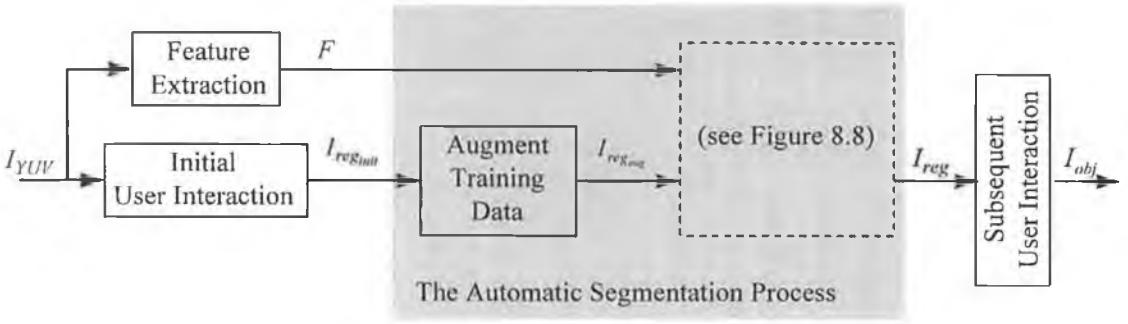
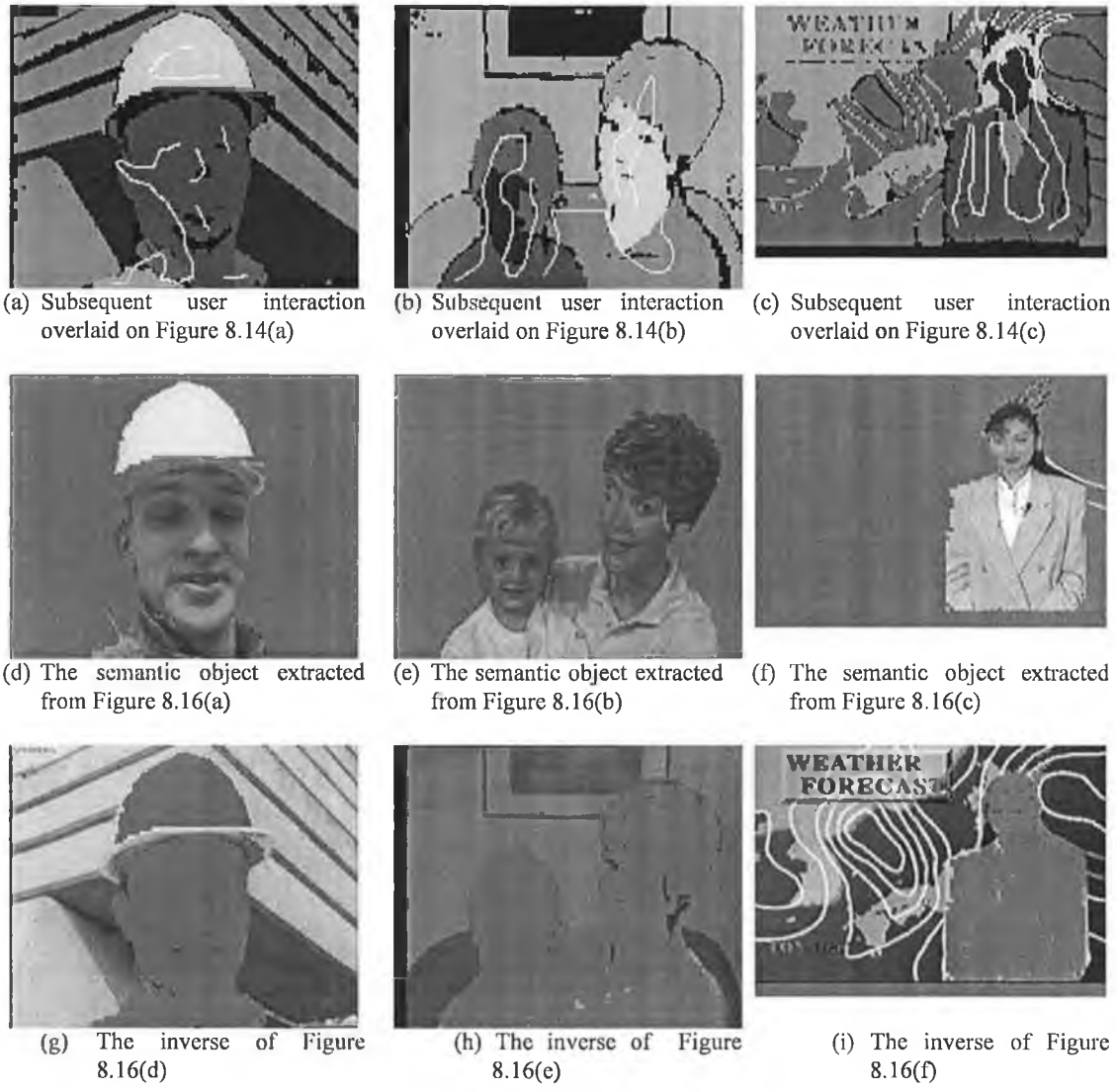


Figure 8.15 - A supervised EM-based scheme for object segmentation

As in the case of clustering-based segmentation, semantic object segmentations may be extracted from the region-based segmentation results. As before, in order to achieve this, it is necessary to treat each spatially disjoint region as a separate entity. Special consideration must also be given to outlier image regions. The subsequent user interaction employed to segment objects, overlaid on the results of Figure 8.14, is illustrated in Figure 8.16(a)-(c). All the pixels of an outlier region which contains at least one scribble pixel are included in the associated object segmentation. In Figure 8.16, subsequent user scribbles are presented as white pixels, and outlier regions as black pixels. The objects generated based on this interaction are illustrated using the dual "painted object" method in Figure 8.16 (d)-(i).

It is clear that the supervised EM-based approach can be used to derive semantic object segmentations. When the improved segmentation performance of this approach is considered, these segmentations could be expected to be of higher quality than those produced using clustering. However, the objects generated using the EM-based approach can be seen to be less accurate. Object contours are inaccurate and the objects contain "holes". This is a result of the way in which outliers are dealt with in the

subsequent user interaction process. The "holes" are due to small regions of outliers not included in a user's scribble. Outliers tend to constitute region contour pixels, as well as unlabelled region types and sub-regions within labelled region types. The entire set of outliers for an image can thus consist of a connected set of outlier groupings. Allowing user interaction to simply include an outlier region, corresponding to an image region contour, results in large regions of the image assigned to the wrong object.



(a) Subsequent user interaction overlaid on Figure 8.14(a)
(b) Subsequent user interaction overlaid on Figure 8.14(b)
(c) Subsequent user interaction overlaid on Figure 8.14(c)
(d) The semantic object extracted from Figure 8.16(a)
(e) The semantic object extracted from Figure 8.16(b)
(f) The semantic object extracted from Figure 8.16(c)
(g) The inverse of Figure 8.16(d)
(h) The inverse of Figure 8.16(e)
(i) The inverse of Figure 8.16(f)

Figure 8.16 - Semantic object segmentations using the EM-based process

This "knock-on" effect of outlier inclusion is avoided in the user interaction employed in Figure 8.16, by simply excluding outlier regions which cause this effect. Predictably, this results in less accurate object contours and missing regions in the final segmentation. Clearly, a more sophisticated approach to user interaction in order to deal with outliers is required. One possibility is to allow a user to "edit" outlier regions (e.g.

allow a user to disconnect outlier groupings into contour outliers and region outliers), and to treat each outlier region separately. A possible way of automating this "edit" functionality would be to reapply the segmentation algorithm considering only the detected outliers (e.g. initial user interaction on outlier regions to mark outlier region types, and then subsequent EM-based segmentation between these regions). However, considering the complex nature of this enhancement, which in addition places a further burden on the user, this approach was not investigated. Even in the absence of such an enhancement, the user interaction required of this scheme does not meet the requirements of section 6.4, for the same reasons as the clustering-based scheme, outlined in section 8.3.7. In summary, considering the excessive amount of user interaction required by this scheme, and the inaccurate segmentation results obtained, it is clear that this approach is not suitable for creating arbitrarily-shaped VOPs with the high degree of accuracy required by future off-line MPEG-4 applications.

8.5 Conclusions

Neither of the approaches described in this chapter fulfil the requirements on user interaction outlined in section 6.4. Both approaches require initial interaction prior to an automatic segmentation process, as well as subsequent user interaction to extract an object from the resultant segmentation. The nature of the initial interaction affects the quality of the automatically generated region-based segmentation. It is important in the case of the clustering approach that the user label all types of region present in the scene in order to obtain a region-based segmentation reflecting his/her requirements. This is not as critical in the case of the EM-based scheme in which unlabelled region-types will appear as outliers. Subsequent user interaction in both cases is not as demanding and yet must be performed very carefully in order to produce an accurate object segmentation. In the case of clustering, the user must be careful to encompass every small disconnected region in the segmentation which forms part of an object. Similar care must also be taken in the case of the EM-based segmentation. However, also included in this process is the burden of dealing with outliers which, as outlined in section 8.4.6, is not a trivial task. It is clear that even in the case of substantial initial and subsequent

interaction[23] the resultant semantic object segmentations are not as accurate as the results presented in [30][32][33] for the techniques described in chapter six.

The description of the two semantic object segmentation approaches described in this chapter restricts itself to the application of these approaches to still images. As stated in chapter six, it is desirable to extend such approaches so that the object can be defined in an initial image via user interaction, with automatic object tracking throughout a sequence thereafter. Since the segmentation approaches described in this chapter build objects in a manner similar to the morphological approach described in chapter six, a similar approach to object tracking could be employed.

The region-based segmentation of the clustering approach could be tracked into subsequent images in the sequence using the estimated cluster centres after the clustering iteration has terminated. These centres could be used to perform an initial classification of every pixel in a new image. Given this initial classification, the clustering iteration could take place as usual, producing a region-based segmentation of the new image. However, in order to automatically track the required object, it would be necessary to track the individual regions making up the object in the new segmentation. This is problematic considering the nature of the segmentations produced by the clustering process. The segmentations appear noisy with many small regions present. Establishing a correspondence between two such successive region-based segmentations is a difficult task considering that some regions may disappear, or change shape, or even be obscured by newly appearing regions.

The region-based segmentation produced by the EM-based approach could also be tracked in a similar manner. The PDF parameter estimates converged upon for one image could be used to initialise the segmentation process for the next image. In such a scenario, every pixel in the new image is initially unlabelled with equal prior probabilities of belonging to one of the available PDFs. The posterior probabilities of every pixel could then be computed using the PDF parameter estimates from the previous image. The EM iteration could then be employed in order obtain a region-based segmentation of the new image. Due to the homogeneous regions produced by

[23] Much more than required for the techniques described in chapter six.

this segmentation process, the region correspondence problem, which arises when re-building object segmentations, should not be as severe as in the clustering case. However, as in the case of the clustering process, it would be necessary to deal with new regions appearing in the scene, or existing regions disappearing.

The extension of the schemes described in this chapter into the temporal domain is not investigated. This is because, considering the excessive amount of user interaction required, the resultant inaccurate segmentations obtained, and the problematic nature of object tracking, neither scheme is a suitable candidate for creating arbitrarily-shaped VOPs in an MPEG-4 application. However, as described in the next chapter, the EM-based scheme can be extended to a full object-based segmentation and tracking scheme. The extended scheme places less burden on the user and the object segmentations obtained for the initial image are very accurate. These segmentations also facilitate object tracking in a straightforward manner. Furthermore, newly appearing or disappearing object regions can be dealt with quite efficiently based on outlier processing. The method of extending the scheme is to include the object model used by *Chalom and Bove* [32]. By additionally improving the method of object tracking, the overall result is a modified and enhanced version of the approach of *Chalom and Bove*, which addresses the limitations of this scheme as discussed in chapter six, and results in improved performance.

9. A SUPERVISED OBJECT-BASED SEGMENTATION SCHEME

9.1 Introduction

This chapter presents a supervised segmentation approach developed by the author for segmenting and tracking semantic objects in natural video sequences. Like the region-based approaches presented in the previous chapter, the approach consists of user interaction coupled to an automatic segmentation process. However, the segmentation process is an enhanced version of the EM-based approach of the previous chapter which models actual semantic objects in the scene, thereby producing an object-based segmentation. This object-based approach was developed due to the limitations of the approaches described in the previous chapter, and because an object-based approach is more attractive as an arbitrarily-shaped VOP creation tool for a number of reasons. Firstly, an object-based segmentation approach eases the burden of initial user interaction as it simply requires a user to mark an object and not construct an object based on a previous result. Secondly, subsequent user interaction for refinement (see chapter ten for more details on this) is also easier to perform since this interaction takes place at the object level, and not the more abstract region or region-type level. Finally, unlike region-based approaches (see section 8.5), an object-based scheme very often suggests a straightforward method of object tracking. The main advantages of an object-based scheme in terms of usability, flexibility and accuracy are described and demonstrated in this chapter.

The author's approach is based on the approach of *Chalom and Bove* [32] which was found to be the most promising of the supervised segmentation approaches reviewed in chapter six. The same scribble-based form of user interaction is employed and the object PDF model of *Chalom and Bove* is used to extend the region-based EM segmentation approach of the previous chapter. The differences and new contributions are as follows. User interaction is extended by an easily performed extra step which allows a user to more completely define the nature of the models used to represent objects. This avoids

the computationally burdensome iterative application of a number of EM algorithms used by *Chalom and Bove* for this purpose (see section 6.6.4). The watershed-based adjacency criterion described in the previous chapter is introduced to improve the model initialisation step. This addresses the limitations of the sparse training data used by *Chalom and Bove* (see section 6.6.4). *Chalom and Bove* use EM algorithms to perform model initialisation based on training data, followed by MAP hypothesis testing on the complete data (i.e. all available observations) as a segmentation process [32]. In the approach presented here, a modified clustering process applied to the training data is used for model initialisation, followed by an EM algorithm applied to the complete data to iteratively refine these estimates and converge on a final segmentation. In this way, all pixels in the image contribute to a greater or lesser degree in the parameter estimation process. This rich data set for parameter estimation yields very good object models whilst using only a subset of the image features required by *Chalom and Bove* (corresponding to those which are the easiest to compute). Furthermore, a different tracking scheme to that of *Chalom and Bove* is employed. A simple motion model is employed, yet the potential errors of tracking training data are avoided since the author's tracking scheme ensures robust temporal coherence between object segmentations, even in complicated scene types. Unlike either tracking approach of *Chalom and Bove*, the author's approach considers phenomena such as object regions appearing or disappearing throughout the course of a sequence. As a result of these enhancements, the author's scheme can be applied to a wider class of scene type than that of *Chalom and Bove* (who only present results for one scene type corresponding to a scene with a simple and static background).

The complete segmentation approach consists of two high-level steps: segmentation of an initial image (which requires user interaction) and automatic tracking of the segmented objects throughout a video sequence. These two high-level steps are presented in the following sections. A system overview of each step is presented, followed by the algorithmic details of the individual low-level processing steps. In each case, the segmentation results obtained are presented. In order to generate these results, the entire segmentation approach was simulated in the ANSI C programming language under the Unix operating system on a Sparc Ultra workstation. The MPEG-4 compliant

software development environment used to simulate the segmentation schemes of the previous chapter was also used here.

9.2 Object and Scene Modelling

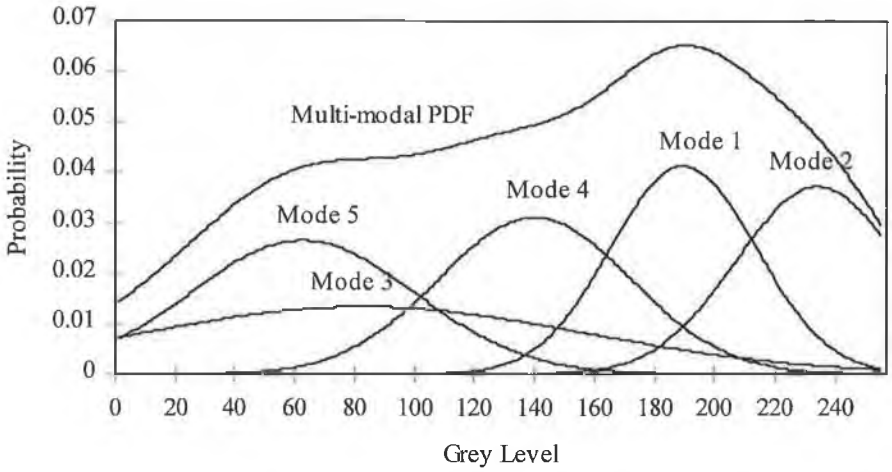
The region-based segmentation approaches of the previous chapter consider an image to be composed of a number of different region types. User interaction consists of marking these region types via a user scribble generated by a mouse drag over each region type present in the scene. In the object-based approach presented here, the scene is considered to be a collection of different semantic objects. Clearly, the objects making up a scene depend on the user's interpretation of the scene and the nature of the application in which the object segmentations will be used (see section 6.2). This semantic information is conveyed to the segmentation process via object-based user interaction. It should be noted that the minimum number of objects in a scene is two, corresponding to an object which the user wishes to segment and "everything else". However, as in the approach of *Chalom and Bove*, multiple objects (i.e. more than two) can be segmented simultaneously.

Each object in the scene is considered to be composed of a number of different image regions or region types. This is the basis of all the approaches described in chapter six and also the basis of subsequent user interaction in the clustering and EM-based approaches of the previous chapter. As in the approach of *Chalom and Bove*, this assumption is incorporated into the method of modelling objects. Multivariate Gaussian PDFs modelling the distribution of luminance, chrominance and spatial features across image region types are shown to perform well in the previous chapter (i.e. they produce homogenous region-based segmentations). As such, in the scheme presented here, the PDFs of region-types composing an object are modelled in exactly the same manner. The PDF of an object is modelled as a multimodal Gaussian PDF which is simply the normalised sum of the independent multivariate Gaussian PDFs of the region types making up the object. Object modelling using multimodal PDFs is illustrated in one dimension in Figure 9.1 below. Figure 9.1(a) illustrates the luminance feature of each pixel in an object. Figure 9.1(b) illustrates how the object is composed of a number (five in this case) of luminance image region types. Each image region type corresponds to a mode in the multimodal PDF modelling the distribution of the luminance features across

the object as depicted in Figure 9.1(c)[24] (for illustration purposes the multimodal PDF is not normalised).

(a) Luminance features of an object

(b) Luminance region-types making up the object



(c) Multimodal Gaussian PDF of luminance features of the object

Figure 9.1 - A multimodal PDF for an object in one dimension (luminance)

The PDF of the image to be segmented is modelled as a mixture density of the individual multimodal multivariate PDFs of the objects present in the scene. An EM formulation is employed to obtain the parameters of the mixture and the overall segmentation approach is very similar to the region-based EM segmentation process of the previous chapter. In fact, the approach is an extension of the previous approach, using the object PDF model of *Chalom and Bove* in order to cope with objects instead of region types (i.e. the Gaussian PDFs for each *region type* in chapter eight are replaced

[24] The segmentation of the object and its segmentation into different region types as depicted in Figure 9.1 was generated using the author's segmentation approach. The parameters used to plot the PDFs were those estimated by the segmentation process.

with multimodal Gaussian PDFs for each *object*). The PDF of the image to be segmented can be written in finite mixture form as:

$$p(f) = \sum_{i=1}^{g} \pi_i p_i(f|\theta_i) \qquad \textbf{Equation 9-1}$$

where g is the number of objects in the mixture, $p_i(f|\theta_i)$ is the PDF of the i^{th} object with M modes, completely defined by the parameter set $\theta_i = \left\{\left(\theta_{i_1}^1, \theta_{i_2}^1\right)\ldots\left(\theta_{i_1}^M, \theta_{i_2}^M\right)\right\}$. The parameter $\theta_{i_1}^j = f_{m_j}^j$ is the mean feature vector of the j^{th} mode, and $\theta_{i_2}^j = \begin{bmatrix} \left(\sigma_{i_1}^2\right)^j & & 0 \\ & \ddots & \\ 0 & & \left(\sigma_{i_k}^2\right)^j \end{bmatrix}$, where $\left(\sigma_{i_k}^2\right)^j$ is the variance of the k^{th} feature in mode j and k is the dimension of the feature vector.

9.3 Object Segmentation in the First Image

The EM algorithm is employed to converge on ML estimates of the parameters of the mixture and a by-product is a classification of each pixel in the image to one of the models (i.e. objects) in the mixture. As usual, in order to start the EM iteration, initial estimates of the mixture parameters and initial prior membership probabilities are required. As in the region-based EM approach, these are calculated on the basis of user interaction. The EM algorithm is applied in the same manner as in the previous chapter, which is different to the use of the EM algorithm by *Chalom and Bove*. The training data is used to derive initial estimates but *all* available data is used to refine these estimates (whereas *Chalom and Bove* limit parameter estimation to the training data). A graphical overview of the entire segmentation approach for the first image in a video sequence is presented in Figure 9.2. In the following discussion the individual processing steps of Figure 9.2 are explained in more detail.

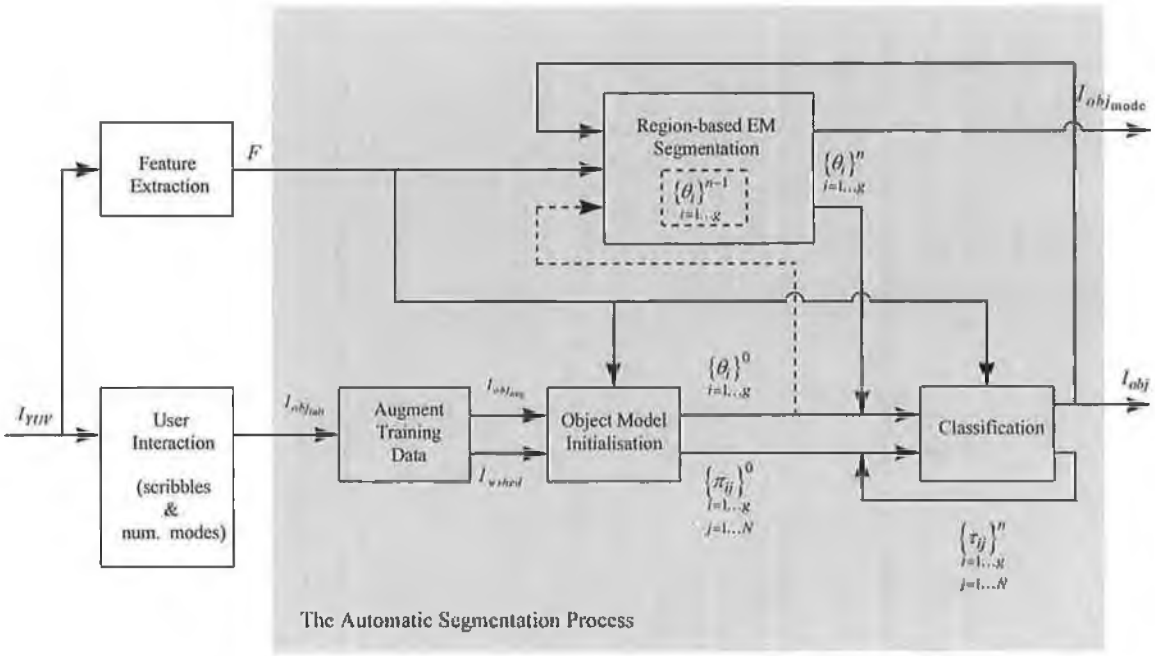


Figure 9.2 - The supervised object-based EM segmentation scheme for still images

9.3.1 User Interaction and Feature Extraction

Feature extraction takes place in exactly the same manner as for the region-based EM segmentation approach described in the previous chapter. Luminance, chrominance and spatial features are extracted from the original image I_{YUV} and arranged in a feature vector for every pixel in the image, thereby producing a feature space F.

User interaction for this approach also takes place in a manner very similar to that employed in the schemes described in the previous chapter (albeit with an extra step). Since the approach is object-based, the first step of user interaction is to allow the user to mark (and thereby label) objects present in the scene which he/she would like segmented. In exactly the same way as performed by *Chalom and Bove*, this is achieved via a mouse drag over the input image in order to create scribbles which label objects in the scene. Disconnected scribbles of the same label are permitted so that the user can mark different parts of the same (possibly disjoint) object (see Figure 6.3). The result is a sparse segmentation of the image $I_{obj_{init}}$ consisting simply of the labelled scribble pixels with all other pixels unlabelled. The number of different scribble labels in $I_{obj_{init}}$ is the number of objects to be segmented g. Theoretically there is no limit on the allowed number of objects. The feature vectors of pixels contained in a set of similarly labelled

scribbles constitute the training data for the associated object, which is used to initialise mixture parameters.

The second step of the user interaction process, which extends the user interaction process of *Chalom and Bove*, is to allow the user to specify the number of modes for each object. This corresponds to the user deciding how many region types are contained in an object and inputting this information into the segmentation algorithm. This may be difficult, particularly given that illumination variations and small colour changes may appear to the segmentation process as different region types within what a user may subjectively term one region type (see section 8.4.6). However, it is only necessary for the user to specify an approximation of the number of modes. The segmentation algorithm automatically corrects this parameter to a more suitable value as it iterates. Normally, the number chosen by the user based on a subjective evaluation of the region types making up an object is an appropriate initial estimate. This number of modes specification step is not necessary in the approach of *Chalom and Bove* where this is estimated automatically based on the results of a number of EM algorithms, each of which estimates the object model parameters for a specified number of modes [32]. Such an automatic process could be introduced into the approach described in this chapter, but it is difficult to justify the extra complexity this would entail given that an approximate initial estimate (which can be subsequently refined) of this non-critical information can be obtained simply by the user specifying a single number for each object. Thus, whilst user interaction is extended, this extension requires very little effort of behalf of the user, and it is easy to perform.

9.3.2 Augmentation of Training Data

As in the EM-based approach of the previous chapter, the training data indicated by $I_{obj_{init}}$ is very sparse. It normally consists of a small number of pixels (in the order of hundreds) for each object, whereas the number of unlabelled pixels in the scene is much greater. For this reason, the training data is automatically augmented in the same manner as performed in the previous chapter. A fine watershed partition of the luminance image to be segmented is calculated (I_{wshed}). All the pixels (and hence feature vectors) of a watershed region which contains at least one scribble pixel are

included in the training data set for the associated object. This produces an augmented training data set, indicated in Figure 9.2 by the less sparse segmentation $I_{obj_{aug}}$. This augmentation constitutes a much richer data set for model initialisation than that of *Chalom and Bove*. This, in association with the user supplied number of modes, allows very good initial object model estimates to be derived using a smaller number of features for each pixel, whilst avoiding the application of a number of EM algorithms.

It should be noted, however, that this augmentation of the training data places a restriction on user interaction. It is imperative that when a user scribbles on an object the scribble is completely contained within the object's borders. If this is not the case (e.g. if even one scribble pixel is outside the object), then watershed regions belonging to the adjacent object will be included in the training data. This may result in a mode being added to the object PDF for these regions, and will cause misclassifications between the adjacent objects in the segmentation process (see Figure 10.2). This effect is avoided in the EM-based approach of the previous chapter by simply detecting such training data as outliers, which is possible since each region is modelled by a single PDF. However, it is an unavoidable limitation of the approach described here. During the model initialisation process, the algorithm cannot identify regions which do not belong to the object, as this is analogous to the fundamental inability of automatic segmentation techniques to extract semantic meaning. However, given a relatively large object compared to the scene size, this restriction can easily be avoided by the user.

9.3.3 Object Model Initialisation

As in any EM-based estimation framework, model initialisation takes place in two steps: initial model parameter estimation and initial model membership probability estimation. Initial parameter estimates are calculated independently for each object. The input to the parameter estimation step consists of (i) the augmented sparse segmentation which defines an initial labelling of the feature space, (ii) the feature vectors for this labelling, (iii) the user-defined number of modes for each object and (iv) the watershed segmentation of the input image. The objective is to segment each object's training data into a number of region types corresponding to the specified number of modes, thereby obtaining the parameters of these modes. The training data is composed of a number of

small image regions corresponding to those watershed regions included by the augmentation process. Since the watershed partition is an over segmentation of the scene, there normally exists a larger number of watershed regions than the required number of modes. Initial parameter estimation proceeds by successively merging watershed regions within a clustering framework until the required number of region types (i.e. modes) are obtained. In the following two paragraphs, this initial parameter estimation process is outlined for a single object. It should be noted that the overall goal is to divide an object (more specifically its training data) into a number of different region types. It is possible that each region type may have more than one homogenous region associated with it in an object (e.g. consider a person's arms). Thus, it is undesirable to spatially constrain this merging process and because of this, the spatial co-ordinate features are not considered in the clustering process.

Each small watershed region assigned to the object's training data constitutes a clustering in the feature space. Centres for each cluster, corresponding to the mean feature vector, are calculated using Equation 8-2 applied to each watershed region. Given this large set of centres, the distance between each centre is calculated using Equation 8-3. The two cluster centres with the smallest distance are merged. The pixels of both regions are assigned the same label in the watershed segmentation and the new cluster centre is calculated, again using Equation 8-2. This merging process is then repeated until there are only as many region labels as the specified number of modes. These regions are taken to be the region types making up the object. Initial ML estimates of the PDF parameters of the j^{th} region type, corresponding to initial estimates of mode j, may be calculated according to Equation 7-5 and Equation 7-6 as:

$$f_{m_i}^j = \frac{1}{n_j} \sum_{f_l \in \text{mode } j} f_l \qquad \textbf{Equation 9-2}$$

$$\left(\sigma_{i_k}^2\right)^j = \frac{1}{n_j} \sum_{f_l \in \text{mode } j} (f_{l_k} - f_{m_k}^j)^2 \qquad \textbf{Equation 9-3}$$

where n_j is the number of pixels in mode j, and $f_{m_k}^j$ is the k^{th} component of $f_{m_i}^j$.

It may happen that the number of watershed regions contained in the training data is actually smaller than the required number of modes. Consider, for example, a small

object for which the user mistakenly specifies too large a parameter for the number of modes. In this case, the user-specified number of modes is ignored and the merging process is skipped. The number of modes is taken as the number of watershed regions making up the training data, and PDF parameters are estimated directly from these regions. In other words, the user-specified number of modes is a maximum allowable number of modes in the initial parameter estimation process. The process will choose a number of modes up to this value, depending on the watershed segmentation. As explained below, the number of modes is further automatically modified during the segmentation process.

The second step of model initialisation is to assign prior object membership probabilities to each pixel. As before (see section 8.4.2), these probabilities are represented by a probability vector π_j for each feature vector. The pixels contained in a training data set for an object contain a one in the appropriate component with zero elsewhere (i.e. they are the group indicator vectors of section 7.4). The probabilities for unlabelled pixels in the image are set to be equally likely for each object. The final output of the model initialisation step is a set of mode parameters for each object in the mixture $\left\{\theta_i\right\}^0_{i=1\ldots g}$ and a set of prior object membership probabilities for each pixel in the image, for each object $\left\{\pi_{ij}\right\}^0_{\substack{i=1\ldots g \\ j=1\ldots N}}$.

9.3.4 Classification

The classification step can be considered to be the E-Step of the object-based EM segmentation algorithm [41][42]. The goal is to estimate posterior object membership probabilities for each pixel in the image (i.e. the complete data set). The posterior probability for a single pixel and a single object is calculated by evaluating the object's multimodal multivariate PDF for the associated feature vector. This produces a probability which is adjusted based on the prior membership probability of the pixel of belonging to the object, and normalised based on the sum of similarly adjusted probabilities for all objects in the mixture. This calculation is essentially the E-Step of Equation 8-6 for object PDFs, written as:

$$\tau_{ij} = \frac{\pi_{ij} p_i\left(f_j|\theta_i\right)}{\sum_{i=1}^{g} \pi_{ij} p_i\left(f_j|\theta_i\right)}$$ **Equation 9-4**

where τ_{ij} is the posterior probability with which pixel j belongs to object i, π_{ij} is the associated prior probability, and $p_i\left(f_j|\theta_i\right)$ is the PDF of the i^{th} object evaluated at feature vector f_j.

As in the E-Step of the region-based segmentation algorithm of the previous chapter, outliers are also detected in this step. In this segmentation process, outliers are considered in two separate ways. The first considers outliers between objects (termed naturally enough, *between object* outliers). These are pixels whose posterior membership probability is zero for every object (i.e. the denominator of Equation 9-4 above evaluates to zero). These correspond to pixels in the image which, based on the training data, do not really belong to any object. These are normally pixels of region types contained in an image but not labelled to any object by a user's scribble. The segmentation algorithm has no means of knowing to which object these outliers belong and they are removed from further processing as they are detected. Outliers within objects (termed *within object* outliers) are also considered. These are pixels which can be assigned to an object but whose mode membership is not easily ascertained. As described in section 9.3.5, these are detected as in the EM-based approach described in chapter eight.

The labelling of pixels in the training data is assumed to be known prior to the segmentation process. However, these pixels are still included in the entire EM process. This is because the objective is not simply to segment the initial image but also to track the segmentation thereafter. For successful tracking, a good object characterisation is necessary. The training data pixels are considered in order that they can be detected as *within object* outliers and new modes assigned to the object, if necessary, in order to obtain a more complete characterisation of the object (see section 9.3.5 below). However, unless they are detected as outliers, their classification will not be changed from the initial *a priori* labelling.

The result of the E-Step calculation is a set of posterior object membership probabilities $\left\{\tau_{ij}\right\}^{n}_{\substack{i=1\ldots g \\ j=1\ldots N}}$ (since an iterative scheme is employed, these are the posterior probabilities of the n^{th} iteration). This essentially constitutes a *soft* segmentation of the image, whereby each pixel belongs to each object with varying degrees of probabilities. A *hard* object-based segmentation is easily obtained by simply examining the posterior probabilities of a pixel for the objects in the mixture, and assigning the pixel to the object with the highest associated probability (i.e. generating the group indicator vector for each pixel as outlined in section 7.4). This produces an object-based segmentation of the scene I_{obj} (see Figure 9.2). This segmentation is used to refine each object's PDF parameters (see below), after which, the classification step is re-iterated. For every iteration except the first, the posterior probabilities calculated in the previous classification step are used as the prior probabilities of the current step[25]. Convergence of the overall algorithm is obtained when the segmentation between two iterations remains the same. In actual fact, the iteration is stopped when only a small percentage of pixels in the image (i.e. 10%, as in the EM-based approach of the previous chapter) are reassigned.

9.3.5 Region-based EM Segmentation

The region-based EM segmentation step is the M-Step of the object-based EM segmentation algorithm [41][42]. The task of this step is to update the object mode parameter estimates so that the classification (E-Step) can be reiterated. Each object's parameters are updated independently. This is performed using the EM algorithm of chapter eight. In this case, rather than segmenting an image into different region types, the algorithm segments an *object* into different region types (corresponding to modes) and in the process, obtains updated estimates of the region type (mode) parameters. In other words, the algorithm of chapter eight is constrained to operate within each object's partition in I_{obj}, in order to implement the M-Step of the object-based EM algorithm. A brief overview of the algorithm used in this context is presented in the following sub-sections.

---

[25] In the first iteration, the prior probabilities used are those produced by the model initialisation step.

Region-based EM Segmentation - *Model Initialisation*:

The object is considered to be a mixture of Gaussian PDFs where each PDF corresponds to a mode. For the first iteration of the object-based algorithm (i.e. the first application of the region-based EM algorithm for each object), the initial estimates of the parameters of these PDFs are available as the output of the object model initialisation step. For every other iteration of the object-based EM algorithm, the initial parameters are the stored final parameters of the previous application of the region-based EM algorithm ($\{\theta_i\}_{i=1\ldots g}^{n-1}$ in Figure 9.2). For each application of the region-based EM algorithm, the prior mode membership probability of each pixel in the object's partition is set to be equally likely for each mode.

Region-based EM Segmentation - *E-Step:*

The posterior mode membership probabilities are estimated based on the current estimate of mode parameters, and the prior mode membership probabilities. These posterior probabilities can be used to obtain a mode segmentation of the object using the classification method of section 8.4.4, and are also used as prior mode membership probabilities in the next iteration of the region-based EM algorithm. The region-based iteration terminates with the same convergence criterion as described in section 8.4.4. The output is a mode-based segmentation of each object $I_{obj_{mode}}$. The equation for calculating posterior mode membership probabilities can be written as:

$$\tau_{ij} = \frac{\pi_{ij} p_i\left(f_j|\theta_i\right)}{\sum_{i=1}^{M} \pi_{ij} p_i\left(f_j|\theta_i\right)} \qquad \textbf{Equation 9-5}$$

where τ_{ij} is the posterior probability with which feature vector f_j ($f_j \in$ the feature vectors assigned to this object) can be classified to mode i, and M is the number of modes for this object.

Region-based EM Segmentation - *Within Object Outlier Processing*:

This is an important step of the EM algorithm which is not performed in the region-based segmentation approach of the previous chapter. In the previous chapter, outliers are simply removed from the segmentation process. This is because the algorithm

cannot know to which region these pixels should be assigned and so they are presented to the user for further instruction. However, in the case of the region-based EM segmentation process described here, the algorithm knows that such outliers belong to the associated object. A reasonable assumption is that they belong to extra modes in the object not specified by the user. *Within object* outliers are detected in exactly the same manner as in section 8.4.3, and collected over the iterations of the region-based EM algorithm for the current object. After the region-based EM iteration has terminated, the collected outliers are median-filtered and homogeneous groupings are detected. The largest such grouping (i.e. the largest number of pixels) is selected as an extra mode. The multivariate mean and variances of this grouping are calculated using Equation 9-2 and Equation 9-3, and a new mode is added to the object's PDF. Only one mode may be added for each complete application of the region-based EM algorithm. In this way, the number of modes specified by the user for each object is adjusted automatically. Outlier pixels which are not included in a new mode are temporarily given the same mode label as the closest labelled pixel. These reappear as outliers during the next application of the region-based EM algorithm and thus, have the possibility to form yet another mode for this object.

Region-based EM Segmentation - *M-Step:*

Each mode's parameters are updated using the M-Step of section 8.4.5 (i.e. using Equation 7-14 and Equation 7-15), except that the processing is restricted to within the current object's partition in I_{obj}.

9.3.6 Post-processing

After the object-based EM iteration has terminated, two outputs are available. The first is the object-based segmentation of the image (i.e. I_{obj} in Figure 9.2). The second is the final mode segmentation of each object (i.e. $I_{obj_{mode}}$ in Figure 9.2). Actually, $I_{obj_{mode}}$ is a superset of I_{obj} which contains not only the object partition, but also the partition of the modes making up the object. There is no guarantee that adjacent pixels will be assigned to the same object in the segmentation process. An object's partition in I_{obj} can consist of one or more large regions (corresponding to the actual object) and some small disconnected regions (corresponding to misclassifications). For this reason, the

segmentation undergoes post-processing based on the assumption that pixels contained in the object are adjacent (in a region growing sense) to one of its user scribbles. The scribbles are overlaid on the object segmentation, and only partitions containing an object scribble are retained for display. The effect of post processing should be reflected in the mode segmentation, which is retained along with the object segmentation for tracking purposes. However, this is not investigated here. Thus, object misclassifications are tracked in the present scheme (see section 9.6.1).

9.4 Results

In this section, segmentation results obtained using the object segmentation process for a single image are presented. In each case, the results are generated using an image (the first image unless otherwise stated) from a selection of the MPEG-4 test sequences. This section is intended to illustrate the nature of the segmentation process, and to indicate its performance. The section is divided into a number of subsections in order to illustrate the different aspects of the scheme.

9.4.1 Illustration of the Segmentation Process

The entire segmentation process is illustrated in Figure 9.3 where a single test image, corresponding to the first image in the MPEG-4 Foreman test sequence, is segmented. Figure 9.3(a) shows the user interaction applied to the test image. This illustrates a user marking two objects in the scene. One object is the foreground figure of the man, indicated by the single white scribble, and the other is the background (i.e. everything else in the scene) indicated by two user scribbles of the same label (i.e. the darker scribbles). The training data generated for this user interaction, after automatic augmentation via the watershed partition, is depicted in Figure 9.3(b). The same watershed segmentation of the scene as illustrated in Figure 8.12 was also used here. In this image, the white pixels represent the training data for the foreground figure, the grey pixels the training data for the background object, and the black pixels are unlabelled.

Four modes are specified for the foreground object, corresponding to the face, hat, shirt and shoulders region types. Five modes are specified for the background object,

corresponding to two modes for the background building and three modes for the part of the building in shadow. The segmentation of the augmented training data into these modes via watershed partition clustering is illustrated for each object in Figure 9.3(c) and (d). In these images, each different grey level within the training data indicates a mode of the object (black pixels indicate pixels not included in the training data). It should be noted that the modes reflect very closely the different types of regions contained within each object.

The output of the entire segmentation process before post processing (i.e. I_{obj} in Figure 9.2) is illustrated in Figure 9.3(e) and (f). Each object segmentation may contain small disconnected regions giving the overall segmentation a noisy appearance. For example, a small part of the logo in the top left corner of the image, and a region in the bottom right of the background are classified to the foreground object. These image regions were not included in user interaction. The misclassification occurs because (i) they are similar (in terms of colour) to parts of the foreground object and (ii) they are not different enough from either background or foreground object to be classified as an outlier region. The final mode segmentation of each object (i.e. $I_{obj_{mode}}$ in Figure 9.2) is illustrated in Figure 9.3(g) and (h). In these images, black pixels represent pixels outside the object segmentation and each mode is represented with a different grey level. Clearly, each object is accurately segmented in terms of the region types composing the object. This, in conjunction with the actual object segmentation, represent a good characterisation of each object, and are thus suitable for tracking purposes. The final object segmentations obtained after post processing are illustrated in Figure 9.3(i) and (j). Small disconnected regions assigned to each object are removed and the overall result is a "cleaner" segmentation.

Figure 9.4 illustrates the nature of outliers in the segmentation process. Figure 9.4(a) depicts the *between object* outliers collected over the segmentation process (these gradually build up as the entire segmentation algorithm iterates). In this case, there are only a very small number of outliers. They mostly correspond to a region type in the original image not included in the training data for either object (i.e. most of the logo in the top left corner of the image). This region type is not characterised by either object PDF model and these pixels cannot be classified. They are removed from the process as

they are detected. Figure 9.4(b) and (d) illustrate the nature of *within object* outliers detected during the region-based EM segmentation process. The outliers within the foreground object, after the EM region-based segmentation has terminated for the first time, are illustrated in Figure 9.4(b) (where white pixels indicate outliers). They correspond to region types within the object (e.g. eyes, the brim of the man's hat, etc.) which are not described by the object's PDF model, and thus constitute new modes. The outliers are filtered, and the largest homogeneous region is selected for further processing as described above. The outlier region selected is illustrated in Figure 9.4(c) where the black pixels indicate outliers in this case. The outliers detected after the region-based EM segmentation has terminated for the second time are illustrated in Figure 9.4(d). The number of outliers is reduced since the object PDF model is more complete.

Processing of outliers is the mechanism by which the number of modes in an object's PDF is automatically adjusted away from the user-specified parameter to a more suitable parameter, reflecting actual object content. In the case of the Foreman test image, for example, it may not be intuitively apparent to a user that the brim of the hat is a region type separate to the hat (hence it was not included in the initial list of modes). In fact, the extra region type present may be due to shadows or illumination variations within what the user perceives as a single region type (see section 8.4.6). However, in terms of the object PDF model, this section of the image is regarded as a separate region type and must be added to the PDF. In actual fact, the user-specified number of modes is not a critical parameter and an approximate value for this will suffice. Furthermore, a good object segmentation can be achieved without a very good characterisation of an object by its PDF model[26]. This is because the segmentation is derived based on models competing for pixel support. As long as the models are reasonably different, and reflect a reasonable characterisation of the object, an accurate segmentation is possible. This is illustrated in Figure 9.5 which depicts segmentations of the same scene with different number of mode parameters for each object[27]. The segmentation results obtained are comparable in terms of accuracy to those of Figure 9.3.

---

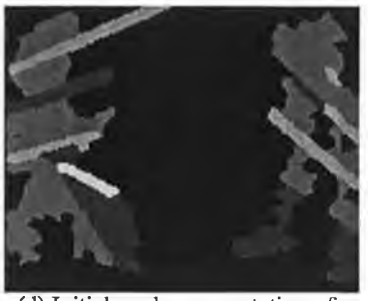
26 However, in general a good characterisation is necessary for tracking purposes.

27 For conciseness, the object-based segmentation is depicted as a single image with object and outlier contours overlaid on the original dimmed image.

(a) User scribbles

(b) Augmented training data

(c) Initial mode segmentation of foreground object



(d) Initial mode segmentation of background object

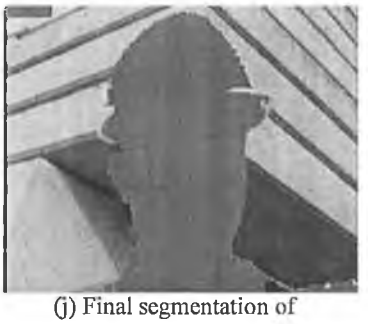
(e) Segmentation of foreground object without post processing

(f) Segmentation of background object without post processing

(g) Final mode segmentation of foreground object

(h) Final mode segmentation of background object

(i) Final segmentation of foreground object



(j) Final segmentation of background object

Figure 9.3 - Selected processing steps of the object-based segmentation process

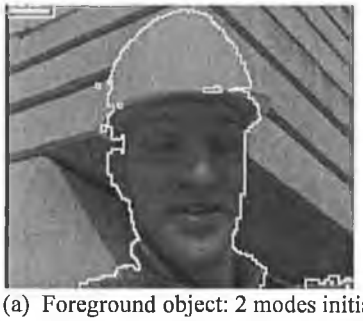
(a) *Between object* outliers

(b) *Within object* outliers (first iteration)

(c) New mode added to object

(d) *Within object* outliers (second iteration)

Figure 9.4 - Outliers in the object-based segmentation process



(a) Foreground object: 2 modes initially
Background object: 3 modes initially

(b) Foreground object: 7 modes initially
Background object: 3 modes initially

Figure 9.5 - The effect of the number of modes parameter on the segmentation process

9.4.2 Performance of the Segmentation Process

Segmentation results obtained by applying the segmentation approach to a number of test images from a selection of MPEG-4 test sequences are illustrated in Figure 9.6. In each test image, two objects (one foreground and one background) were selected and the user interaction for this is illustrated in Figure 9.6(a)(c)(e)(g) and (i). The segmentations of the foreground objects are depicted in Figure 9.6(b)(d)(f)(h) and (j). Very accurate object segmentations are obtained in each case. These segmentation results for an initial image are more complete than those of *Chalom and Bove* [32], who only present results

for two sequences: a sequence of people dancing and the Table Tennis test sequence. In both these sequences the background is composed of a small number of large regions of distinctive colours, thereby greatly facilitating segmentation. Clearly, the author's segmentation approach can be applied to a wider class of scene type (e.g. scenes with complicated backgrounds such as the Foreman, Weather, Mobile and Calendar, and Container test sequences). This is undoubtedly due to the superior object modelling carried out by the author: the augmentation of training data, and the use of the complete data in deriving estimates, yields PDFs very suitable for segmentation purposes.

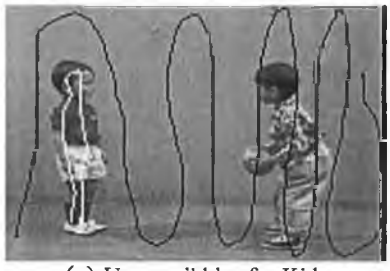
9.4.3 Flexibility of the Segmentation Process

There is no limitation on the types of objects to be segmented. As illustrated in Figure 9.6, the algorithm performs well for human objects and non-human objects. The nature of the object to be segmented is decided by the user. This demonstrates the flexibility of a supervised object-based approach, particularly when used as a means of content generation for off-line MPEG-4 applications as discussed in section 6.2. In fact, the illustrative images of Figure 6.1 were generated using the segmentation approach described here applied to the same image, but with different objects selected (i.e. the weather girl in one case and simply the weather girl's jacket in the other).

There is also no limitation on the number of objects which can be segmented at one time. For example, if a scene contains three objects which the user wishes to segment, then each can be segmented as an object-background pair separately (as in Figure 9.6). Alternatively, the user may decide to segment all three objects simultaneously which necessitates only one application of the segmentation process to the image. Either way, the segmentation results are comparable in terms of accuracy. Examples of segmenting more than two objects simultaneously are presented in Figure 9.7. This is an advantage of this approach (and that of *Chalom and Bove*) over the approach of *Steudel and Glesner* [30][31], which is limited to segmenting a single object in an iterative fashion.

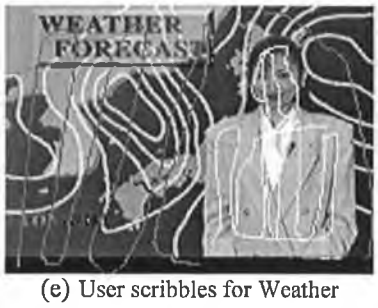
(a) User scribbles for Mother and Daughter
(mother and daughter - 6 modes, b/g - 4 modes)

(b) Mother and Daughter object



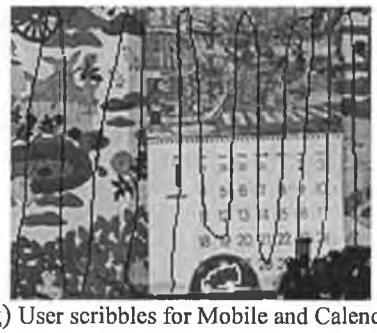
(c) User scribbles for Kids
(child - 4 modes, b/g - 7 modes)

(d) Child object



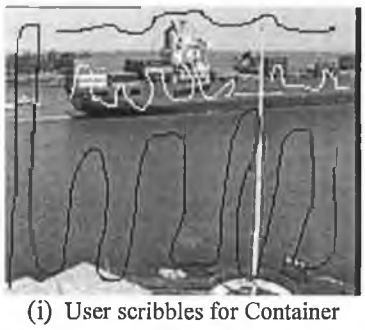
(e) User scribbles for Weather
(weather girl - 4 modes, b/g - 4 modes)

(f) Weather girl object

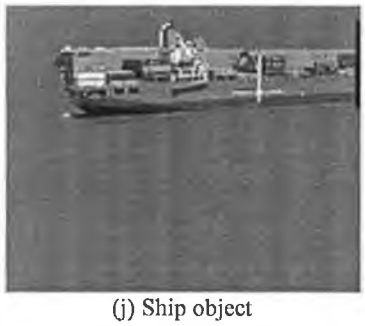


(g) User scribbles for Mobile and Calendar
(ball - 2 modes, b/g - 6 modes)

(h) Ball object



(i) User scribbles for Container
(ship - 3 modes, b/g - 6 modes)



(j) Ship object

Legend: b/g = background (i.e. everything in the scene bar the named object)

Figure 9.6 - A selection of objects segmented from various MPEG-4 test sequences

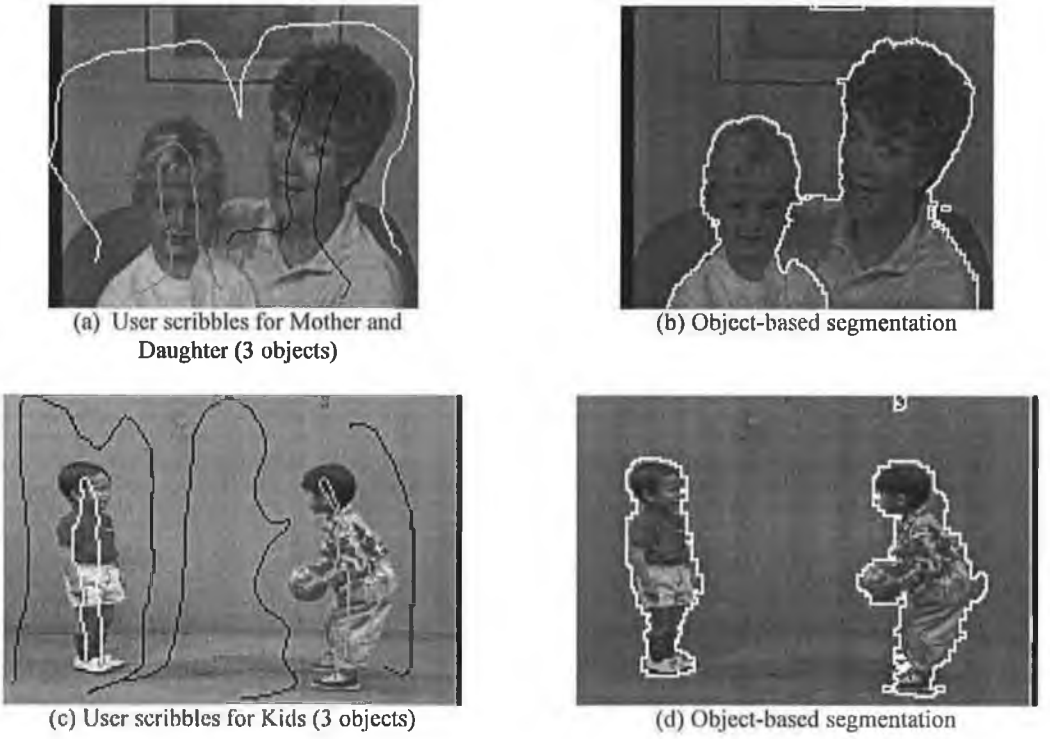
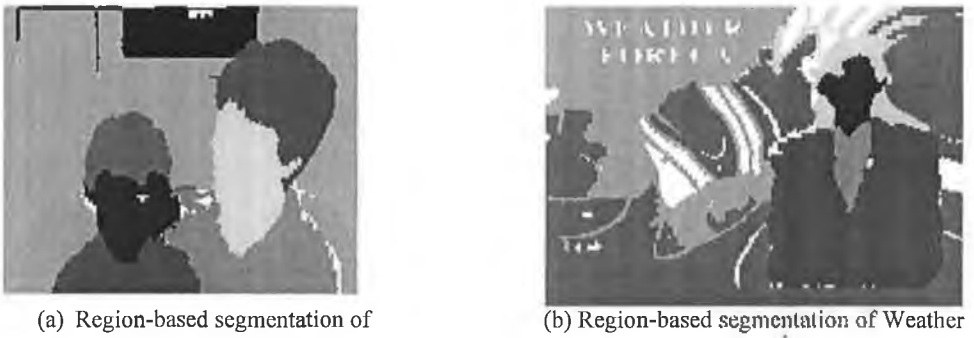


Figure 9.7 - Multiple objects segmented simultaneously

The segmentation results presented thus far have indicated a minimum amount of user interaction, consisting of very simple user scribbles and inputting a single number (i.e. the initial number of modes) for each object. This represents much less user interaction than that required by the region-growing approach of *Steudel and Glesner* [30] and both approaches described in chapter eight. Even though the segmentation results are not pixel-accurate (e.g. missing part of the child's face or including part of the background in the weather girl in Figure 9.6), they are of sufficient accuracy for many MPEG-4 applications. Furthermore, if pixel-exact segmentations are required, then this segmentation approach is a first step in the process which provides the user with very good results for further processing. The results require a minimum of subsequent user interaction/editing in order to obtain pixel-exact representations (for proposals on how to carry out this editing see chapter ten). These results can be slightly improved by performing more user interaction. However, a large increase in performance in terms of accuracy is not noticeable. This is due to the automatic augmentation process which can be considered as compensation in the case of minimal user interaction. The minimal approach to user interaction presented here is actually desirable, since one of the

requirements on a supervised segmentation process is to relieve a user of the tedious process of outlining manually every pixel on an object's contour (see section 6.4).

The flexibility of the segmentation approach is further illustrated in Figure 9.8, where the object-based segmentation approach is used to generate region-based segmentations of two test images. In order to generate a region-based segmentation, each region type in the image is selected as a separate object and the number of modes for each object is set to one. The user scribbles used to generate the results of Figure 9.8 were those used in chapter eight for the same test images (see Figure 8.3 and Figure 8.4). The segmentation results of Figure 9.8 are actually better than those generated using the region-based EM segmentation approach (see Figure 8.14). This is due to *within object* outlier processing in the object-based approach, which will actually add a mode to a single region-type if necessary (e.g. to cope with shadows and illumination variations). *Between object* outliers are depicted in Figure 9.8 as white pixels, and there are fewer of these than in the results of Figure 8.14.



(a) Region-based segmentation of Mother and Daughter

(b) Region-based segmentation of Weather

Figure 9.8 - Region-based segmentations via object-based approach

9.4.4 Failure of the Segmentation Process

The segmentation algorithm does not perform well for every test sequence. Figure 9.9(b) depicts the segmentation result obtained when attempting to segment the person in the 10th image of the Hall Monitor test sequence. This result is understandable when the nature of the two objects to be segmented is considered. Both are very similar in terms of colour and thus, compete more or less equally for pixel support. To make matters worse, because the person is small with respect to the size of the image, it is difficult to draw a scribble over the person which does not subsequently include a

background watershed region in the augmented training data. This background region is treated as a mode in the person object, and the result is the classification of neighbouring background (constrained by the spatial features) to the person. Figure 9.9(d) indicates an even worse case of the segmentation algorithm failing to extract the required objects. In the case of the News test image, there is a lot of similarity between the selected objects. The watershed partition itself (which is derived solely from the luminance component) does not reflect the objects' contours and thus (due to the augmentation process), the training data for each object consists partly of pixels from other objects in the scene. The object PDF models compete equally for support and the result is an inaccurate noisy segmentation.

(a) User scribbles for Hall Monitor
person - 2 modes, b/g - 6 modes

(b) Person object

(c) User scribbles for News
male - 3 modes female - 4 modes
dancer - 1 mode b/g - 5 modes

(d) Object-based segmentation

Legend: b/g = background (i.e. everything in the scene bar the named objects)

Figure 9.9 - When the object-based segmentation process fails

In summary, it is clear that given an image in which the objects to be segmented are very similar in terms of colour (and particularly in terms of the luminance component) it is difficult to obtain accurate segmentations. However, under such circumstances, many segmentation approaches (including those described in chapter six) will face similar difficulties. In many cases (in fact, in all but two of the test sequences considered) the supervised approach described in this chapter constitutes a very powerful segmentation

process which is ideal for creating arbitrarily-shaped VOPs in still images. It can be more widely applied than the approach of *Chalom and Bove* (on which it is based) and exhibits a high degree of segmentation accuracy. In order to further improve the results, subsequent user interaction is necessary. This subsequent user interaction should include the processing of *between object* outliers. These should be displayed to the user after the segmentation process has terminated, in order that they can be assigned to an available object. This is considered an important area for future research. Currently, *between object* outliers are simply removed from the segmentation process. In the absence of subsequent user interaction, this is a reasonable approach since there are normally very few *between object* outliers, as the detection process for these is (deliberately) less strict than that for *within object* outliers.

9.5 Tracking the Segmented Objects

Given a segmentation of the initial image in terms of the required objects, the segmentation approach attempts to automatically track the temporal evolution of each object throughout the rest of the sequence. Objects are tracked by applying the object-based segmentation approach to each successive image, initialised with the segmentation results for the previous image. An extra step is incorporated prior to initialisation in order to account for any motion present in the video sequence. The overall process is illustrated in Figure 9.10 and the individual processing steps are described in the following discussion.

Chalom and Bove propose two techniques for tracking semantic object segmentations [32]. The first consists simply of applying MAP testing to every pixel of each new image based on the object parameters estimated for the initial image. The second consists of tracking the training data of the first image, via motion estimation, and using this to initialise new object parameter estimates. As outlined in section 6.6.4, tracking training data points can be problematic in the case of complicated object motions. Furthermore, neither approach of *Chalom and Bove* addresses the issue of object regions appearing or disappearing (which should be reflected in the objects' PDFs) due to the object's or the scene's temporal evolution. As such, neither approach is adopted as the basis of the author's tracking described here. Rather, the segmentation of the initial image is projected into the new image to be segmented and then further refined to

account for object regions appearing or disappearing. In this way, the tracking procedure described here is more robust (i.e. it can be applied successfully to a wider class of scene types) than that of *Chalom and Bove*.

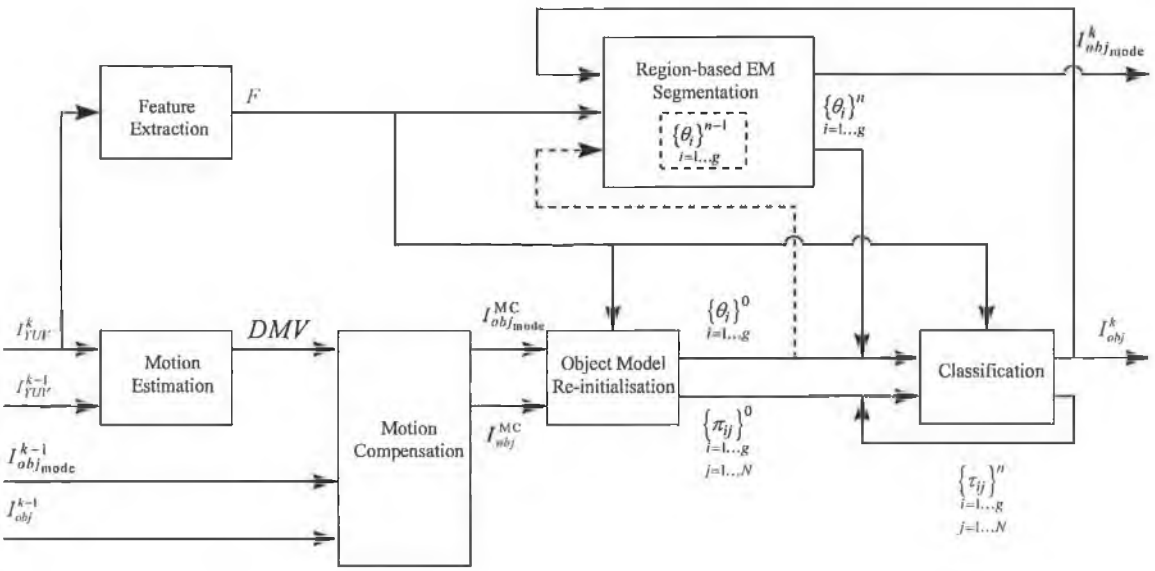


Figure 9.10 - The automatic EM-based segmentation scheme for object tracking

9.5.1 Feature Extraction

The current image to be segmented is denoted I^{k}_{YUV} in Figure 9.10, and features are extracted for this image in exactly the same manner as for the first image in the sequence.

9.5.2 Motion Estimation

The motion estimation step attempts to calculate the motion between the current image to be segmented, and the previous image in the sequence I^{k-1}_{YUV}. A very simple motion estimation scheme is employed, although any motion estimation scheme which produces a dense motion vector field[28] could be used. The actual scheme employed is based on the hierarchical block-matching scheme proposed by *Bierling* in [12], and employed by *Alatan et al* in [29]. A motion vector is estimated for every 4th pixel in the horizontal and vertical positions. Three hierarchical block-matching levels are used with

[28] Alternatively, any motion estimation algorithm from which a dense motion vector field can be derived (e.g. via interpolation) can be used.

parameters (in pixels) as specified in Table 9.1 below. In Table 9.1, "Block Size" is the size of the block used in the matching criteria (MAD is used for this), "Search Range" is the maximum horizontal and vertical displacement of this block in the search process, and "Step" is the step size used in the search. The result of this step is a motion vector for every 4×4 pixel block in the image. This is not really a dense motion vector field (DMVF), but it approximates the motion at every pixel.

Table 9.1 - Motion estimation parameters for object-based tracking process

	Level 1	Level 2	Level 3
Block Size	32	16	4
Search Range	8	4	2
Step	2	2	1

9.5.3 Motion Compensation

The motion vectors calculated using the motion estimation scheme described above are used to compensate for the motion present between two images in the sequence. The object-based segmentation of the previous image I_{obj}^{k-1} and each object's mode segmentation $I_{obj_{mode}}^{k-1}$ are motion compensated[29]. This produces an initial segmentation for each object, I_{obj}^{MC}, and its modes $I_{obj_{mode}}^{MC}$, in the current image. This information is used to re-initialise each object's PDF parameters.

9.5.4 Object Model Re-initialisation

ML estimates of the parameters of each mode[30] of each object are calculated based on the motion compensated mode segmentation using Equation 9-2 and Equation 9-3. Initial prior object membership probabilities are calculated on the basis of the motion compensated object segmentation I_{obj}^{MC}. Each object partition is represented as a binary image which undergoes mean value filtering with a large filter window (25×25 pixels). Membership probabilities are calculated for each pixel by extracting the normalised

---

29 In practice, only one segmentation image ($I_{obj_{mode}}^{k-1}$) needs to be motion compensated as this is essentially a superset of I_{obj}^{k-1} (i.e. it contains the object-based segmentation of the image but also each object's segmentation into modes).

30 It is possible that not all modes will appear in the motion compensated mode segmentation of an object (e.g. a mode with very fast motion, or motion not approximated by the motion model). In this case, the missing mode will not be tracked.

values from the filtered images. This is similar to the method of extracting temporal constraints used in the context-constrained EM algorithm for automatic motion segmentation described by *Brady and O'Connor* in [39].

9.5.5 Classification and Region-based EM Segmentation

Given initial estimates of the mixture parameters, the segmentation process continues in almost the same manner as for the first image in the sequence. There is, however, an extra step in the region-based EM segmentation process, which is necessary when tracking objects. The extra step is necessary in order to cope with disappearing object regions. As an object moves, parts of it may become occluded or actually move outside the image. Also, the illumination effects and shadows within a region type may vary from one image to the next. In section 9.3.5, it is explained that in the initial image, the latter lead to new modes added to an object as it is segmented. Both the effect of object motion and the transitory nature of illumination variations imply that an object mode can disappear from one image to the next. This must be detected, and the object PDF must be modified, in order to maintain a reasonable characterisation of the object for future tracking. Even if a mode has disappeared from one image to the next, it will appear in the motion compensated mode segmentation used in model re-initialisation (see section 9.5.4). However, the pixels used to generate new initial estimates for this mode's parameters no longer reflect a separate region-type in the object, but actually correspond to an existing mode which is itself also initialised (the latter is the original region type without shadows in the case of illumination variations). Thus, the same image region type may be initially represented twice in the object's PDF. These two modes will compete for assignment of pixels. The probabilities of pixels of this region type will, over a number of iterations, gradually tend to 1 for one mode, leaving the other mode with all zero probabilities for these pixels. This is indicated by τ_{ij} (as calculated in Equation 9-5) evaluating to zero for all feature vectors f_j ($f_j \in$ the set of pixels assigned to this object) for a particular mode i. This occurrence is detected and the redundant mode is removed.

When an object moves, new region types within the object may become visible in the scene (e.g. a hand being raised). In this way, it may become necessary to add new

modes to the objects. This does not require an extra step in the region-based EM segmentation process, as it is already handled by *within object* outlier processing. If a new region of an object appears in the current image to be segmented within the object's motion compensated segmentation in the previous image, then the pixels of this new region will be included in the re-initialisation of the object's mode parameters. These pixels will then be detected as *within object* outliers, and a new mode will be added to the object to account for their appearance. Unfortunately, as illustrated in the results section, new object regions do not always enter the scene within the motion compensated previous object partition and this can cause misclassification of these new regions.

The final output (i.e. after convergence) of the object-based segmentation iteration I^k_{obj}, and the final output of the region-based segmentation iteration for each object $I^k_{obj_{mode}}$, are retained in order to initialise the segmentation process in the next image to be segmented. As in the case of the initial image in the sequence, *between object* outliers detected during the segmentation process for each image are not considered in the tracking process. They are simply removed from processing. No memory of these outliers is maintained from one image to the next.

9.5.6 Post-processing

In order to derive the final segmentation of an image, similar post processing to that used for the first image in the sequence is applied to I^k_{obj}. For tracked segmentations, it is assumed that the segmentation of the required object is close to the segmentation of the object in the previous image. The previous segmentation is overlaid on the current segmentation and only partitions which overlap are retained. The effects of this post-processing should be incorporated into the object mode segmentation, and thus reflected in the object characterisation. However, as in the case of the first image in the sequence, this is not investigated here.

9.6 Results

In this section, object tracking results are presented for a selection of MPEG-4 test sequences. In each case, the segmentation of the initial image used is the segmentation presented in the previous results section for the corresponding sequence. Object tracking, when the segmentation of the initial image does not perform well, is not investigated. If the approach cannot segment the objects in the first image based on user interaction, it will not be able to do this automatically in subsequent images. This is because (as is clear from the results presented here), the segmentation process cannot correct itself. It is sensitive to unsuitable initial parameter estimates. Given such estimates in the form of an inaccurate segmentation of the initial image, the algorithm will not obtain reasonably different characterisations of the objects in the second image and tracking results will be poor. These results will then deteriorate with every new image segmented, since the segmentation of each new image is initialised with the previous result. Results are presented here which indicate tracking performance given a reasonably accurate segmentation of the first image.

One hundred images (with no frame skipping) were segmented for each test sequence. This number of test images was chosen in order to reflect the use of the approach in an off-line MPEG-4 VOP creation application. Ideally, the technique should automatically segment every available image. In practice however, this is unlikely. One can imagine a scenario whereby the algorithm is used to segment a similar set of images in a sequence. If the scene changes substantially (e.g. objects leaving or entering: phenomena which are not considered by this approach), or the tracking procedure starts to fail, then the user would most probably stop the tracking process and re-initialise the segmentation process. As is demonstrated in this section, tracking throughout one hundred images in the case of certain test sequences is optimistic, given some limitations associated with the tracking process. As in the previous results section, the presented results are divided into subsections in order to illustrate the nature and performance of the tracking process.

9.6.1 Illustration of the Tracking Process

The re-initialisation step of the tracking process for the second image of the Foreman test sequence is illustrated in Figure 9.11. The motion compensated mode segmentation

is presented for each object in Figure 9.11(a) and (b). In these images, each mode is represented by a different grey level. Black pixels indicate pixels not classified to the object. The method of deriving initial prior object membership probabilities is illustrated in Figure 9.11(c) and (d). In these images, each object's partition in the motion compensated object segmentation is represented as a binary image which undergoes undergo mean value filtering. These images can be considered to be probability images, whereby the brighter the grey level, the more probable that the associated pixel is a member of the object. The filtering process "blurs" the probabilities around the object's boundaries (which are likely to change from one image to the next) whilst maintaining object membership in the centre of the object, thus enforcing temporal coherence. The initial object membership probabilities for each pixel are extracted as the normalised values (i.e. the filtered grey levels mapped into the range [0,1]).

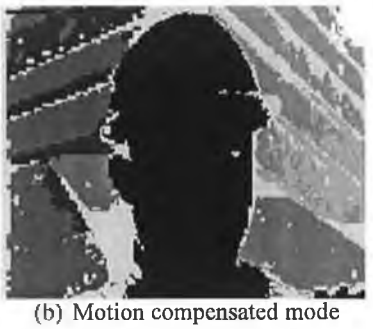
The tracked foreground object prior to post-processing, is presented in Figure 9.11(e). As can be seen, the bottom right background region remains classified to this object in the second image since the results of post-processing in the initial image are not reflected in the object's PDF. The final output foreground object segmentation for this image is presented in Figure 9.11(f). The object tracked into different images of the sequence is presented in Figure 9.11(g) and (h). Figure 9.11(h) illustrates the effect of new object regions entering the scene. The man's hand enters the bottom right of the image. As it does so, it occludes an existing region for this object (the misclassified background region) and becomes incorporated into the object segmentation. As illustrated in Figure 9.13, the effects of newly appearing object regions is not always as fortuitous.

9.6.2 Performance of the Tracking Process

This section illustrates the tracking of a selection of the objects segmented in section 9.4. The tracking results obtained for the Mother and Daughter test sequence considering two objects (corresponding to the foreground object and the background object) are presented in Figure 9.12. As in the Foreman case, the tracking process performs very well for this sequence. This is undoubtedly due to the nature of the objects being segmented. Objects are large compared to the image size and a good

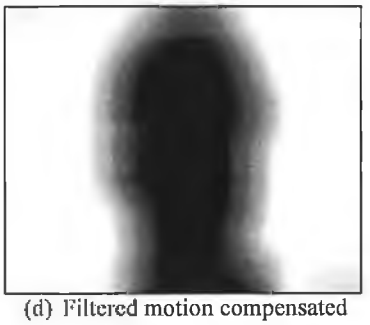
characterisation of each is possible. The effect of trying to segment similar (in terms of colour) adjacent objects is evident here. The frame of the picture in the background has almost the same colour characteristic as the mother's hair. It is included (albeit spatially constrained, thanks to the spatial features) in some segmentations.

(a) Motion compensated mode segmentation of foreground object



(b) Motion compensated mode segmentation of background object

(c) Filtered motion compensated object segmentation of foreground object



(d) Filtered motion compensated object segmentation of background object

(e) Tracked foreground object before post processing

(f) Tracked foreground object after post processing

(g) Tracked object in 40th image

(h) Tracked object in 90th image

Figure 9.11 - Illustration of the object-based tracking process

(a) Tracked foreground object in 2nd image

(b) Tracked foreground object in 40th image

(c) Tracked foreground object in 60th image

(d) Tracked foreground object in 84th image

Figure 9.12 - Tracked foreground object of Mother and Daughter test sequence

The tracked object-based segmentation of the Mother and Daughter test sequence considering three objects is presented in Figure 9.13. In these results, the disadvantages of the appearance of new object region-types are illustrated. The mother's arm region appears in the scene, but as it appears it occludes the daughter object (see Figure 9.13(c)). The pixels of the arm region type are included in the initialisation of the daughter object PDF parameters, and the arm region type is then classified to the daughter object. This limitation of the segmentation process is unavoidable. The algorithm cannot know how to classify appearing regions (it is the same problem faced by unsupervised approaches which cannot extract semantic meaning from a scene). This is a strong argument in favour of subsequent user interaction on segmentation results. Whilst the segmentation and tracking algorithm provide accurate segmentations, to achieve exact segmentations (based on a user's perception of the actual objects present), it is necessary to edit/modify these results. This modification should include the possibility of assigning new regions entering the scene to a particular object. The advantage of the approach presented in this chapter, is that this subsequent interaction can be kept to a minimum (e.g. the hand is a mode which could be swapped between objects).

(a) Object-based segmentation of 2nd image

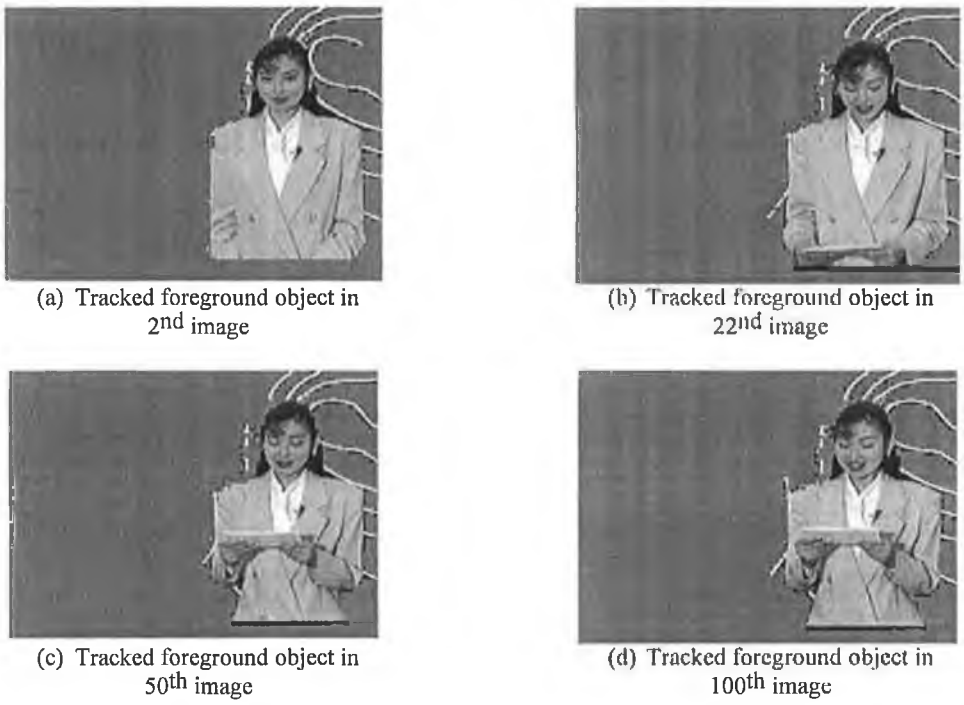
(b) Object-based segmentation of 40th image

(c) Object-based segmentation of 60th image

(d) Object-based segmentation of 84th image

Figure 9.13 - Tracked object-based segmentation of Mother and Daughter test sequence considering three objects

The tracked foreground object of the weather sequence is illustrated in Figure 9.14. A limitation of the tracking process is clearly seen in these results. The segmentation of the foreground object in the initial image contains parts of the background: part of the weather map is assigned to the girl's hair (see Figure 9.6). This occurs because this part of the background is assigned to the training data of the weather girl by the augmentation process (the contour between the background and the girl's hair is not present in the watershed partition of the image). This background region is included as a mode in the foreground object and contaminates the initial estimates of this object's parameters in subsequent images. The result is that background pixels start to get assigned to the object (constrained by the spatial features). These misclassifications then further enforce the effect of the contamination and as the sequence progresses, the segmentation diverges from the required object.



(a) Tracked foreground object in 2nd image

(b) Tracked foreground object in 22nd image

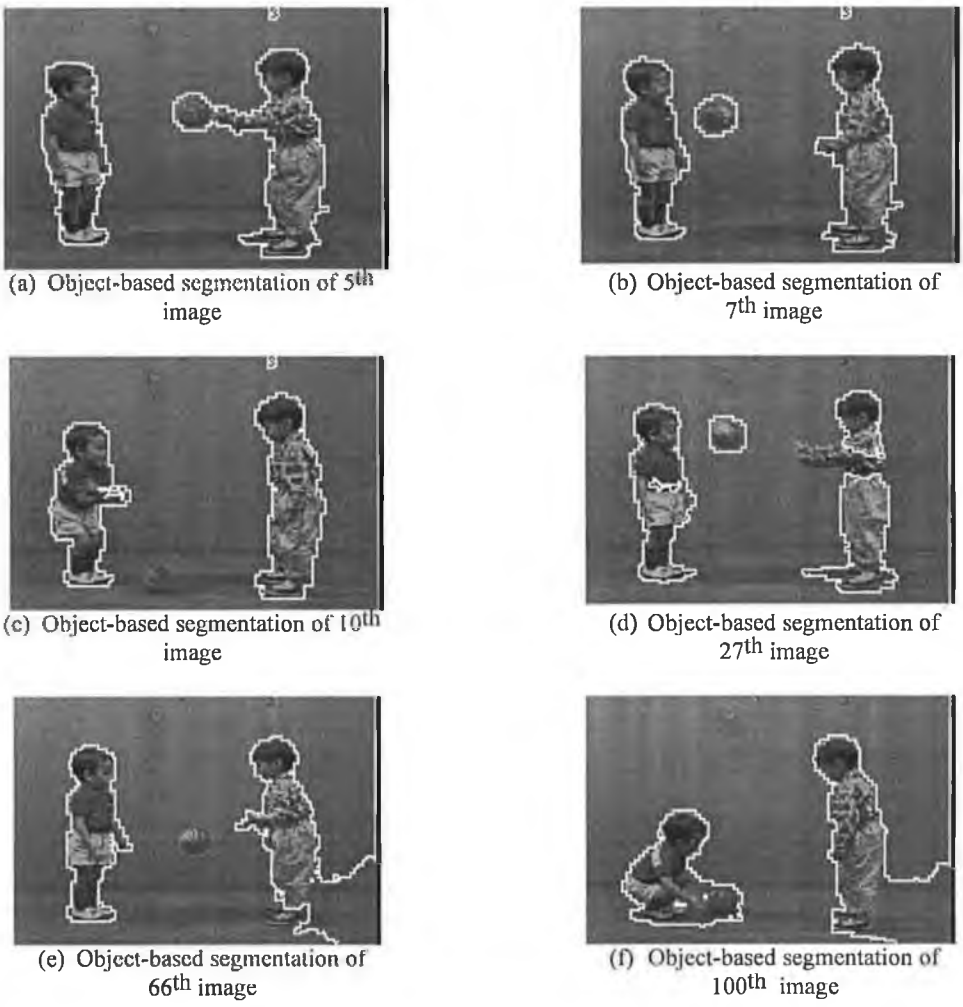
(c) Tracked foreground object in 50th image

(d) Tracked foreground object in 100th image

Figure 9.14 - Tracked foreground object of Weather test sequence

The tracked object-based segmentations of the Kids sequence, considering three objects, (corresponding to the background and each child), are presented in Figure 9.15. This sequence is very challenging for the tracking process. The motion in the sequence (the moving ball and the children attempting to catch it) is very fast. In order to be able to estimate this motion, the parameters of Table 9.1 had to be adjusted for this sequence. The search range was doubled for the first two levels of the estimation hierarchy. The tracking algorithm performs very well for a short time. The flying ball is tracked in the first part of the sequence (Figure 9.15(a) and (b)) and is only subsequently lost when it reaches its destination (it falls sharply and the motion estimation algorithm fails to detect this, see Figure 9.15(c)). The ball, whilst tracked, receives the same label as the child on the right, because the ball was considered part of this object in the first image. When the motion of the ball is not captured by the estimation algorithm, the ball does not appear in the motion compensated mode segmentation of this object. Thus, the pixels of the ball are included in the mode initialisation process for the background object, and the ball is classified to the background. The ball remains classified to the background until it approaches the child on the left (Figure 9.15(d)), at which point it is classified to this child object because it is more similar to this object than to the background (both the ball and the child's torso are red). As the sequence progresses, the

algorithm has difficulty tracking the child on the right. This object initially has part of the background assigned to it: the black striped artefact on the right (this happens since the background has only one mode and the child's hair is black). The object then performs very fast and complicated motion and eventually becomes spatially connected to this part of the background (the child's leg touches the edge of the image). The motion compensated mode segmentation of the child then becomes contaminated with information from the real background, and the object never becomes disconnected from the black stripe. However, these segmentation results are very encouraging considering the challenging nature of this sequence.



(a) Object-based segmentation of 5th image

(b) Object-based segmentation of 7th image

(c) Object-based segmentation of 10th image

(d) Object-based segmentation of 27th image

(e) Object-based segmentation of 66th image

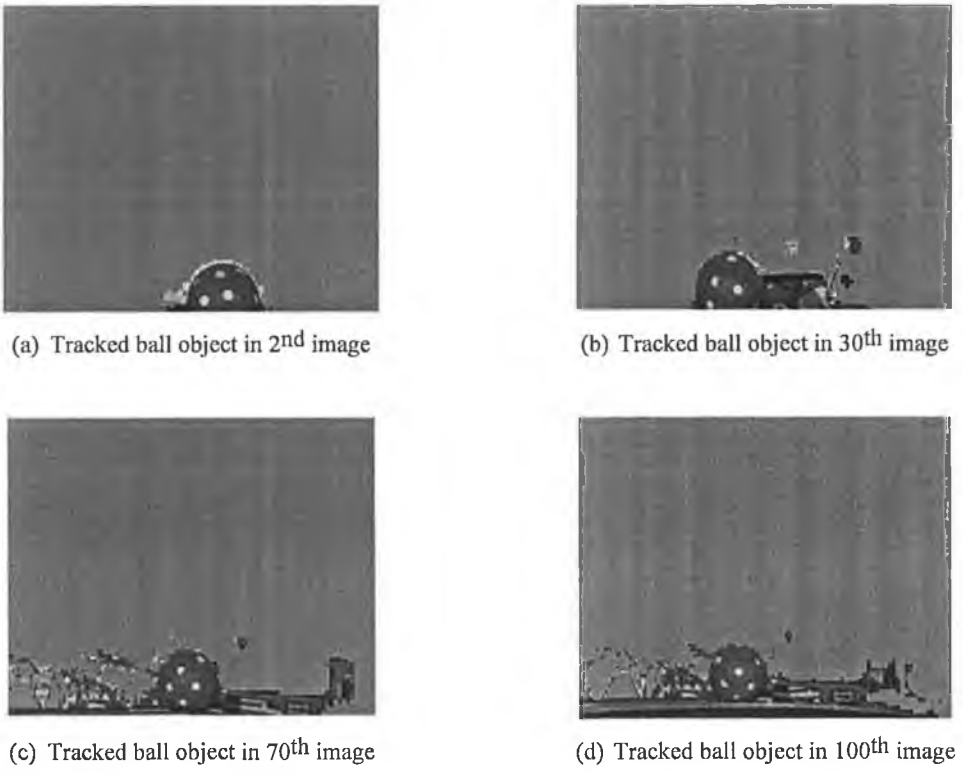
(f) Object-based segmentation of 100th image

Figure 9.15 - Tracked object-based segmentation of Kids test sequence

9.6.3 Failure of the Tracking Process

The failure of the tracking process to accurately segment objects in a sequence is illustrated in Figure 9.16. This sequence represents the worst case scenario for the

tracking algorithm. The segmentation of the object in the first image, whilst reasonably accurate, contains part of the background: the "halo" effect around the ball (see Figure 9.6(h)). Furthermore, the ball is a small object with respect to the size of the scene and its characterisation, which is thus based on a small number of pixels, is particularly susceptible to contamination of parameter estimates. To make matters worse, the ball moves with a rotational motion which cannot be estimated correctly with the relatively simple block-based motion estimation scheme employed. Furthermore, due to camera motion, a new region appears in the scene (the surface below the ball) and is misclassified to the ball object. The result is that the tracking accuracy deteriorates as the sequence progresses.



(a) Tracked ball object in 2nd image

(b) Tracked ball object in 30th image

(c) Tracked ball object in 70th image

(d) Tracked ball object in 100th image

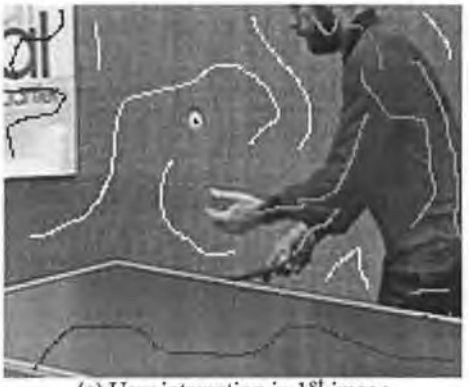
Figure 9.16 - Tracked ball object in the Mobile test sequence

9.6.4 Comparison with *Chalom and Bove's* Approach

As stated previously, the author's scheme addresses the potential limitations of the approach of *Chalom and Bove*. Augmentation of training data for initial parameter estimation, and the use of all available information in the parameter refinement process, lead to very appropriate object PDF models for segmentation (using a small number of

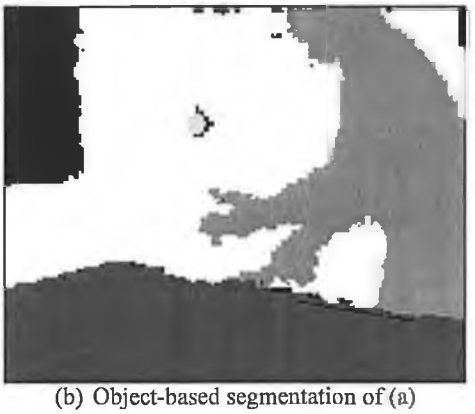
features) regardless of scene type. In this way, the author's scheme can be applied to a wider range of scenes and objects than that of *Chalom and Bove* (who only present results for two sequences, which are similar in nature). Furthermore, using the author's scheme, successful object segmentation and tracking can be achieved without the optic flow motion estimation procedure necessary with *Chalom and Bove's* approach, whilst at the same time accounting for the appearance or disappearance of object regions throughout a sequence. The enhanced performance of the author's scheme is illustrated in Figure 9.17 below, where the same publicly available test sequence as used by *Chalom and Bove* is segmented using the author's approach.

In Figure 9.17 the Table Tennis test sequence is segmented, starting from the same temporal reference as reported by *Chalom and Bove* (see "Simulation Results" in [32]). Whilst the *exact* same user interaction could not be applied, the user scribbles drawn by the author in Figure 9.17(a) attempt to mimic as closely as possible those of *Chalom and Bove.* In fact, one scribble is omitted by the author, corresponding to the second scribble on the table in [32]. This was omitted because otherwise, the augmentation process includes part of the background in the training data (due to the fact that the white stripe around the table does not appear in the watershed segmentation). However, this scribble is difficult to perform and is actually not necessary in the author's approach in order to achieve a very accurate segmentation. The resultant object segmentations (presented in the same format as used by *Chalom and Bove*) are illustrated in Figure 9.17(b)-(e). Comparing these results to those presented in [32], it may seem, at first glance, that the results are almost identical and that the author's scheme provides little extra benefit. However, these results are considered superior for two reasons. Firstly, the author's results are derived using a subset of the features used by *Chalom and Bove*, corresponding simply to luminance, chrominance, and spatial features (which are almost trivial to compute). Thus, the computationally burdensome optic flow, and local texture features of *Chalom and Bove* are not employed and yet almost the exact same highly accurate result is achieved. This again underlines the improved object modelling approach of the author. Secondly, the motion estimation scheme used by the author is very rudimentary (a three-step block matching approach) and yet the tracked results are almost exactly the same, again using a smaller subset of the features used by *Chalom and Bove*.

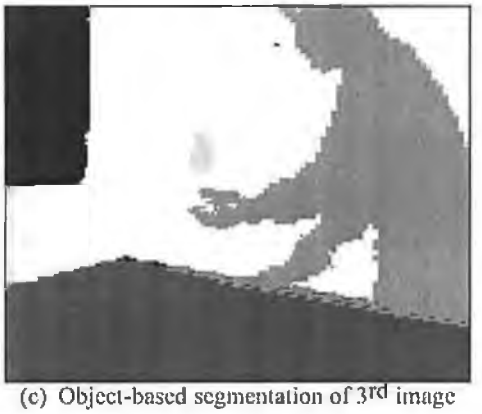


(a) User interaction in 1st image

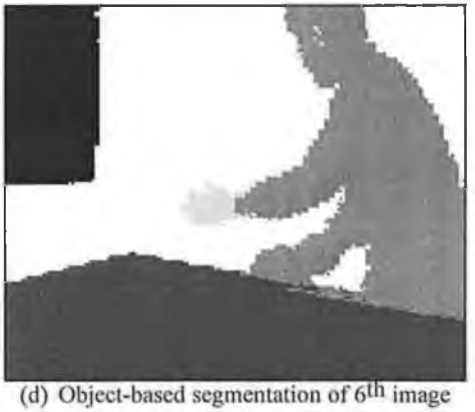
Notes:
(i) number of modes for table = 2
(ii) number of modes for wall = 1
(iii) number of modes for poster = 2
(iv) number of modes for man = 3
(v) number of modes for ball = 1



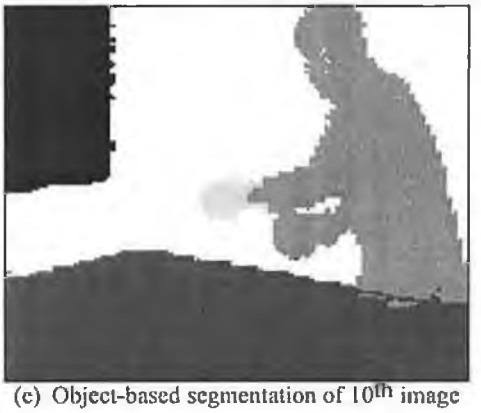
(b) Object-based segmentation of (a)



(c) Object-based segmentation of 3rd image



(d) Object-based segmentation of 6th image



(e) Object-based segmentation of 10th image

Figure 9.17 - Segmentation results using the same test conditions as *Chalom and Bove*

9.7 Discussion

Overall, the performance of the author's segmentation approach is extremely encouraging. For most of the test sequences considered, it is possible to accurately segment a variety of semantic objects in the first image of a sequence. The amount of user interaction required is minimal, and it is easy to perform (similar to that required by *Chalom and Bove's* approach or the morphological approach of chapter six). The

scheme is flexible, allowing different types of objects to be segmented (further object segmentations for a variety of object types using different image resolutions/formats are presented in Appendix B). Furthermore, the amount of subsequent interaction required for pixel-exact segmentations is kept to a minimum, due to the segmentation accuracy obtained. Thus, the scheme meets all the requirements on a supervised segmentation approach outlined in chapter six. It is therefore concluded that this approach is ideal for the purposes of creating arbitrarily-shaped VOPs for future off-line MPEG-4 applications.

The author's scheme is a modified and enhanced version of the approach of *Chalom and Bove*. Comparable segmentation results are obtained using only a subset of the features used by *Chalom and Bove,* corresponding to the subset easiest to compute. By a simple extension of user interaction in a non-critical manner, the successive application of a number of different EM algorithms is avoided. The problem of sparse training data is addressed via automatic augmentation. By using all available observations to derive parameter estimates, object PDF models more closely reflect the nature of the object, which leads to a more generic segmentation approach which can be applied to a wider class of scene types. The potentially problematic tracking approaches of *Chalom and Bove* are replaced by a tracking process which ensures temporal coherence, and which accounts for the disappearance and appearance (at least to some degree!) of object regions.

The limitations of the author's approach are mainly in the tracking process. Clearly, this is very sensitive to contamination of parameter estimates with inappropriate data. A very accurate segmentation of the required object in the first image in the sequence is necessary for accurate tracking. This limitation could be addressed by allowing user interaction on the results of the segmentation process, in order to edit the object segmentation and obtain a more accurate result for tracking (a proposal for the exact form that this interaction could take is presented in the next chapter). This interaction can be considered to be an enhanced form of the post-processing described in this chapter. It is clear that the effect of post-processing on the segmentation result should be reflected in the object PDF models. For the post-processing described in this chapter, the removal of disconnected regions should also be carried out in the mode

segmentation, and the pixels of the removed regions should be assigned to another object. The object to which these pixels should be assigned could be decided based on evaluating each object's PDF for these pixels, and assigning them as an extra mode to the object with the highest probability. If no suitable object is available, then these pixels should be classified as *between object* outliers and included in the refinement process outlined in the next chapter. An alternative solution, is to actually incorporate the effect of post-processing into the segmentation process itself. In fact, this is why spatial co-ordinates are included in the feature vectors. The objective is to obtain spatially homogeneous region type segmentations, and hence, spatially homogeneous objects. However, it is clear that the effect of other features can override the spatial considerations. To avoid this, the classification process of the segmentation algorithm could be further spatially constrained. For example, rather than assigning a pixel to an object based on the probability calculated by evaluating each object's PDF, this assignment could be carried out by weighting the calculated probability with knowledge of the probable classification of neighbouring pixels (in a manner similar to the spatial constraints employed by *Brady and O'Connor* in [39]).

Another limitation of the tracking process is the motion estimation algorithm employed. The block-based scheme investigated here is not reliable when applied to generic scene types. It does not account for the complicated motions (such as rotation) which an object can exhibit. To address this limitation, more sophisticated models should be used. To this end, the scheme presented by *Brady and O'Connor* in [39] (see section 7.5) could be integrated with the scheme described in this chapter[31]. The tracking scheme described in this chapter, without motion estimation, could be used to derive approximate initial segmentations of two consecutive images in a sequence. The motion modelling approach of *Brady and O'Connor* could then be applied, constrained by the object partitions of these segmentations. This should result in very good motion model estimates, which when used to project the objects' mode segmentations, should improve tracking.

The test sequences used in this chapter are considered to be representative of the type of input an MPEG-4 encoder could expect. As such, it can be claimed that for sequences in

31 This is not investigated here but targeted as a direction for future research.

which the segmented objects can be tracked, this approach is a powerful tool for creating arbitrarily-shaped VOPs for off-line MPEG-4 applications. In the next and final chapter, the approach is considered in the framework of a complete VOP creation environment, and directions for future research are identified.

10. CONCLUSION

10.1 A Brief Review

In chapter two, the development of video compression technology in recent years is outlined. The fundamental techniques for reducing spatial and temporal redundancy in a digital video signal (transform coding and predictive coding respectively) are introduced. These techniques are normally applied as block-based tools using the DCT and motion estimation/compensation. The use of these tools in international video compression standards for different applications (H.261, H.263, MPEG-1 and MPEG-2) is briefly described. It is explained how block-based compression can lead to visually disturbing blocking artefacts, particularly at lower bit rates.

Chapter three reviews the investigation of segmentation-based compression by the research community. These approaches were first investigated as a means of obtaining efficient compression whilst avoiding block-based artefacts. The approaches were also attractive as a means of providing content-based functionalities. Segmentation-based compression can be carried out by OOASC or by a region-based approach. SIMOC (an OOASC approach) was eventually abandoned as a compression scheme due to limitations such as (i) its inappropriate source model, (ii) the inefficient compression of model failure regions, (iii) the lack of an INTRA coding scheme (and indeed, the effect of INTRA encoding on segmentation performance) and (iv) the inherent delay associated with OOASC. MORPHECO (which is a region-based approach) does not suffer from such limitations, except the inherent analysis delay. However, in order to provide content-based functionalities, it is necessary for the segmentation used to reflect semantic meaning, which is not the case in a morphological approach. Nor is this the case in SIMOC, due to the limitations referred to above. Furthermore, the analysis (segmentation) and encoding steps in both SIMOC and MORPHECO are tightly coupled (i.e. compression efficiency depends on the performance of the segmentation step).

Chapter four describes the new MPEG-4 compression standard which incorporates the advantages of both segmentation-based and block-based compression. MPEG-4 specifies a compressed representation of rectangular and arbitrarily-shaped VOPs. A sequence of arbitrarily-shaped VOPs represents a semantic video object (VO) present in the scene. A VOP is encoded using a block-based shape coding tool to compress the alpha channel, in conjunction with modified existing block-based tools to compress image data. MPEG-4 does not standardise a segmentation algorithm in order to create arbitrarily-shaped VOPs. Rather, an efficient compression scheme is defined which is independent of the segmentation process (unlike SIMOC and MORPHECO). As such, MPEG-4 can avail of future advances in the field of segmentation. It also leaves open a competitive aspect to the standard which is essential for ensuring its success. An accurate, robust and flexible method of creating arbitrarily-shaped VOPs will be a distinguishing factor for many future MPEG-4 products in the market place. MPEG-7, a new ISO work item aimed at developing a multimedia content description standard, is also briefly introduced in this chapter. It is conjectured that segmentation will have an important role to play in extracting object features, so that they can be described by MPEG-7.

Three *unsupervised* segmentation processes suitable for creating arbitrarily-shaped VOPs, are described in chapter five. Since segmentation in this thesis is discussed in terms of semantic object segmentation, unsupervised region-based approaches are not considered. Such approaches generally have no means of grouping image regions to form objects. The underlying assumption of unsupervised object-based schemes is that any motion in the scene (excluding camera motion) is due to the presence of moving semantic objects. The change detection-based approach of *Mech and Wollborn* [23][24] addresses many of the limitations of change detection as employed in SIMOC. The polynomial motion modelling approach of *Wang and Adelson* [25][26][27] uses affine motion models to ensure that object motions more complicated than simple translation can be segmented. The automatic segmentation functionality of the COST 211ter AM is in part based on the scheme of *Alatan et al* [29] which uses a region-based segmentation tool (RSST) in order to refine the results of motion segmentation (which itself is derived using RSST) via a rule-based processor. All three approaches, however, are restricted by the fact that semantic meaning in a scene cannot be defined

automatically. This is a fundamental limitation of unsupervised segmentation when considered in an MPEG-4 VOP creation context. However, considering segmentation performance, and the fact that no user intervention is required at any stage of the segmentation process, these approaches are suitable for creating arbitrarily-shaped VOPs for real-time MPEG-4 applications which do not require accurate segmentations of objects at all times (e.g. video surveillance).

As explained in chapter six, there exists a class of MPEG-4 applications (e.g. content production or editing) which require accurate segmentations of entire semantic objects. However, in these (normally) off-line segmentation processes user interaction can be introduced to define semantic meaning in a scene, and thus relieve the segmentation process of this ill-posed problem. This is termed *supervised* segmentation. Ideally, user interaction should only be applied to the first image in the sequence, with automatic segmentation and tracking of the defined objects thereafter. It is further proposed that this initial interaction should itself not be excessive. The idea is to roughly mark objects to be segmented and to allow an automatic segmentation process to define their exact shape. Allowing user interaction in the segmentation process means that region-based techniques can be employed. The user may interact with a region-based segmentation process (or interact with the results of such a process) in order to build a semantic object segmentation.

Three supervised segmentation approaches, suitable for creating arbitrarily-shaped VOPs for off-line MPEG-4 applications, are described in chapter six. The reported segmentation results of these schemes are comparable, but those of *Chalom and Bove* [32] are the most promising. The approaches can be distinguished on the basis of user interaction. The approach of *Steudel and Glesner* [30][31] potentially requires significantly more interaction than that of *Chalom and Bove* or the proposed morphological approach. Whilst the morphological approach necessitates the least amount of user interaction, the tracking results of *Chalom and Bove* are superior to the morphological tracking results of *Marqués and Molina* [33]. This undoubtedly due to *Chalom and Bove's* approach of modelling objects as multivariate multimodal PDFs, which allows a good characterisation of objects as a basis for segmentation. However, there exist limitations with the approach of *Chalom and Bove*: the training data used to

derive PDF parameter estimates is very sparse, a number of successive applications of EM algorithms (see below) is required to obtain these estimates, and the tracking mechanism (tracking training data or just using original parameter estimates) may be problematic in complicated scene types. However, these limitations are addressed by the author's further investigations where the advantages of the approach *of Chalom and Bove* (namely the methods of user interaction and object modelling) are used in an enhanced scheme which avoids these limitations.

The EM algorithm is described in chapter seven as a method of obtaining ML estimates of mixture density parameters in the presence of incomplete data. The object segmentation problem can be considered to be an incomplete data problem: object parameters are required in order to create a segmentation, but these parameters cannot be estimated without a segmentation. The approach of the EM algorithm is to start with appropriate initial parameter estimates and iteratively refine these based on estimated pixel object membership probabilities, to finally converge on the ML parameter estimates. A by-product is the required object-based segmentation. The EM algorithm has been successfully employed in the unsupervised video segmentation approach of *Brady and O'Connor* [39], and the supervised approach of *Chalom and Bove*, referred to above. As such, it represents a powerful segmentation tool, and is used by the author in the modified segmentation approach of *Chalom and Bove* presented in chapter nine.

Two supervised region-based segmentation approaches, developed by the author, are presented in chapter eight. These approaches are developed as they are the basis of the author's modified and enhanced version of the scheme of *Chalom and Bove*. The first approach is based on clustering multidimensional feature vectors. The second approach employs the EM algorithm. The same form of user interaction as employed by *Chalom and Bove* (except that it is region-based in nature) is used in each case. The clustering scheme requires that every region type in the image be indicated by user interaction, otherwise misclassification occurs. The EM-based approach deals with this by detecting outliers which correspond to region types not indicated by the user. As a result, the EM-based approach produces a segmentation of the image into homogeneous regions which reflect the region types indicated by the user. However, construction of a semantic object based on this result will result in an inaccurate segmentation unless outliers are

reassigned appropriately. Interaction with the results of the clustering process can also produce semantic objects. However, in both cases the interaction must be performed very carefully. The semantic object segmentations obtainable with these approaches are not as accurate as those obtainable with the schemes described in chapter six, nor does user interaction meet the necessary requirements. Object tracking is feasible with these approaches but difficult in practice and is therefore not further explored. However, the EM-based approach (which itself is an enhanced form of the clustering approach) can be enhanced using the object PDF model of *Chalom and Bove* in a straightforward manner.

The resultant supervised object-based segmentation approach is presented in chapter nine. It is based on ML estimation of the parameters of multivariate multimodal PDFs (via the EM algorithm of chapter seven), which are used to model semantic objects in the scene. The segmentation process lends itself to object tracking throughout a sequence in a straightforward manner. From the presented results (see sections 9.4 and 9.6), it is clear that very accurate object segmentations can be obtained for still images with very little user interaction (much less interaction than in the region growing approach of *Steudel and Glesner* described in chapter six). This user interaction is also easy to perform, meeting the requirements outlined in chapter six. The tracking scheme performs well in most cases. Limitations such as a rudimentary motion estimation scheme and sensitivity to unsuitable parameter estimates, mean that accurate tracking of objects is not always achieved. However, possible solutions to these limitations are proposed. In most cases, this scheme can produce very accurate object segmentations throughout a sequence, which could be refined with little effort to obtain pixel accurate segmentations. The approach is therefore considered to be an ideal candidate for producing arbitrarily-shaped VOPs for off-line MPEG-4 applications.

Whilst the object PDF model used in the author's approach is that of *Chalom and Bove*, the overall scheme is somewhat different in order to address the limitations associated with *Chalom and Bove's* approach. Sparse training data is avoided using an automatic augmentation process developed in chapter eight, the result of which is a richer data set for the calculation of initial object PDF parameter estimates. The successive application of a number of EM algorithms to calculate these estimates is avoided via a simple extension of user interaction (which is easy to perform, and is non-critical in nature).

Furthermore, the EM algorithm employed by the author uses all available information (corresponding to all pixels in the image) in order to refine object PDF parameters and converge on very suitable models. The potentially problematic tracking of *Chalom and Bove* is addressed by using a scheme developed by the author, which ensures temporal coherence, whilst making allowance for object regions appearing/disappearing in a scene. The overall result is that the author's scheme can be successfully applied to a wider class of scene types. Using the same test sequence as *Chalom and Bove* and almost identical user interaction, the same segmentation and tracking results can be obtained whilst using only a subset of the features employed by *Chalom and Bove* (corresponding to those easiest to compute), and a very simple motion estimation scheme. This indicates that the object PDF models derived by the author are more complete and appropriate for segmentation than those of *Chalom and Bove*.

10.2 Directions for Future Research

10.2.1 A Complete MPEG-4 VOP Creation Environment

In this section, the integration of the segmentation algorithm of chapter nine into a complete interactive environment for producing arbitrarily-shaped VOPs for MPEG-4 applications is proposed. In order for this environment to be really useful for an end user, a number of enhancements are required to the basic algorithm. Some of these are implementation issues whilst some present opportunities for future research. The latter are briefly described here.

The objective of a VOP creation environment is to provide the user with a complete segmentation application which can be tuned based on his/her preferences in order to segment and track objects in a wide variety of scene types. The various high-level processing steps required by such a scheme are presented in Figure 10.1.

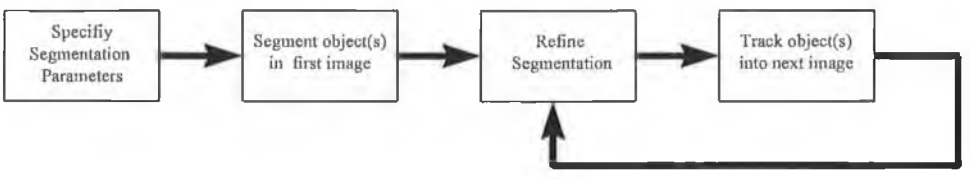


Figure 10.1 - The processing steps of a complete VOP creation environment

The first step is to specify the segmentation parameters. The parameters used in chapter nine constitute a reasonably robust parameter set for most scene types and could be used as appropriate defaults for the non-technical user. However, for the more advanced and technically-aware user, as much flexibility as possible should be allowed in this step. Parameters to be specified could be the input to the segmentation process, or parameters configuring the segmentation process.

The segmentation algorithm of chapter nine uses multiple input information sources, corresponding to intensity, colour, and spatial location, arranged in feature vectors. For certain segmentation problems, when specifying the input to the segmentation process, the user may wish to use a subset of these (e.g. when a grey level image is to be segmented). Alternatively, the user may wish to include different representations of the features (e.g. *RGB* components for colour rather than *YUV*) or to include new features which may be useful for particular images (e.g. texture features based on local grey level co-occurrence matrices). The effect of different feature sets in the segmentation process is an area which requires investigation. The user must also specify the number of modes for each object in this step (the iterative scheme outlined in the next section could be incorporated here).

Parameters configuring the segmentation algorithm could be a specification of the tools to be used in the segmentation process. The watershed algorithm is currently used to obtain a fine region-based partition of the luminance image. However, for certain images the watershed may miss important object/region contours (because they are not strongly in evidence in the luminance component). Thus, the user may wish to choose the region-based segmentation algorithm to be used. Choices could include a watershed in the colour space or the RSST segmentation of *Alatan et al* [29]. The effect of different schemes on segmentation performance should be investigated to ensure that these are viable options. A true expert user may also wish to specify the motion model to be used in tracking. Sometimes a simple translational model will suffice, whilst as outlined in chapter nine, in other cases a more sophisticated model would be more appropriate. The user should be allowed to choose an appropriate motion model based

on observation of the motion in the sequence. The effect of different motion models on the segmentation process should be investigated.

After specifying the input and configuring the segmentation process, the algorithm of chapter nine, suitably adjusted, could be invoked for the first image. The results of this segmentation process may suffice for the user's needs. Alternatively, a more accurate segmentation may be required. It is therefore desirable to allow the user to refine the segmentation results if necessary. The framework in which to perform this refinement is a matter for further research. One scenario is to present each object to the user separately (in a similar manner to the images of Figure 9.3(i) and (j), for example) along with the *between object* outliers (which the user should assign to one of the objects). User refinement of one object segmentation result should then be reflected in the results for other objects. One possibility for the actual refinement mechanism is to allow the user to click on regions within the object segmentation and thereby delete the associated region in the fine region-based segmentation from the object (and add it to another object). Another, perhaps more interesting, possibility is to reapply the segmentation algorithm to the object segmentation results. This process, termed here *scalable segmentation*, could be used to refine an object's segmentation or to further segment an object into sub-objects (e.g. consider creating "hot-spots" within an object for an interactive tele-shopping application). The idea of scalable segmentation is illustrated in Figure 10.2[32]. The concept of scalable segmentation and its potential consequences for the segmentation process should be further investigated.

Given a segmentation of an initial image which meets the users requirements the tracking mechanism of chapter nine could then be employed[33]. After tracking, the user should again be given the choice to refine the tracked segmentation in the same way as above, before proceeding with the segmentation of the next image.

Such a scenario as that outlined above should provide a flexible and powerful environment for creating arbitrarily-shaped VOPs for MPEG-4 encoding. It could be used as a content creation tool for future MPEG-4 applications. It employs powerful

---

32 These preliminary results were obtained using the segmentation algorithm of chapter nine.

33 It is assumed here that the limitations of the tracking scheme are addressed, specifically that the effect of post processing is reflected in the PDF models.

segmentation technology which can be guided by the user to address his/her requirements. From the user's point of view, this corresponds to a powerful image processing tool-box which relieves the drudgery of manual segmentation, normally a time-consuming (and expensive) process which involves outlining by hand the required object in each image of a sequence.

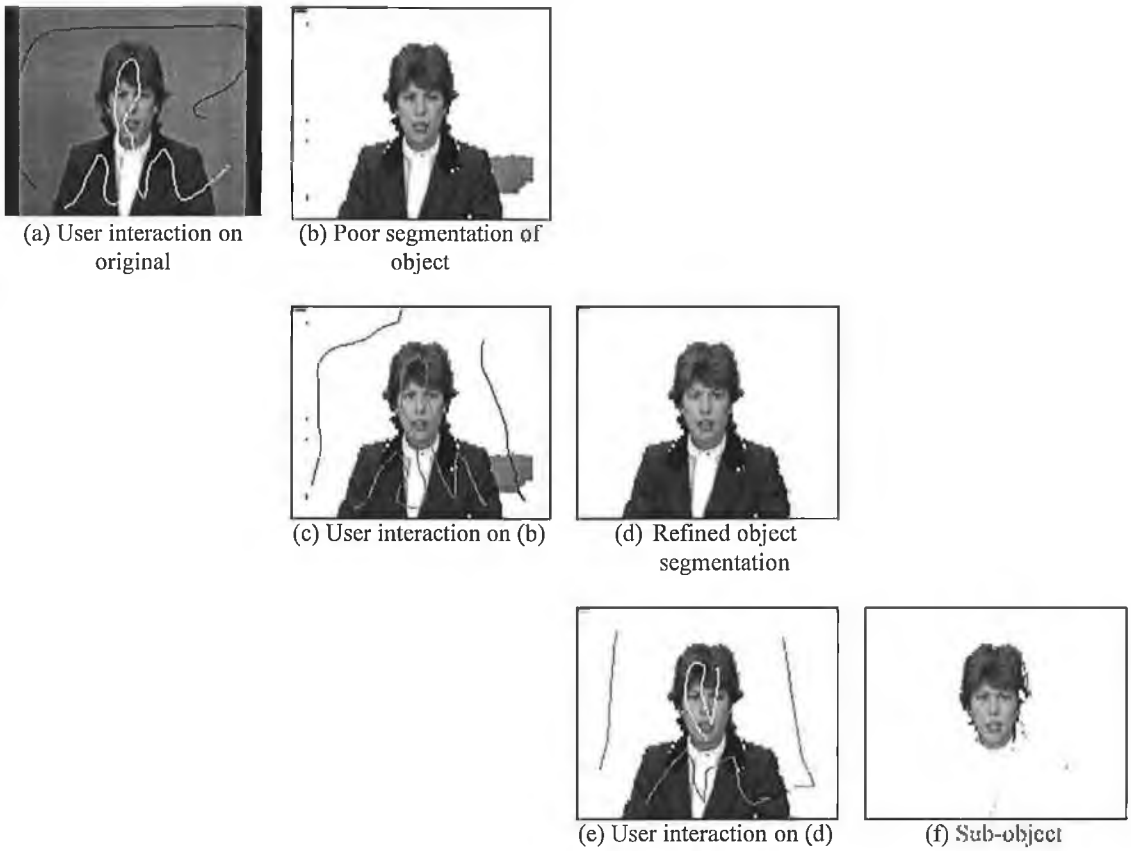


Figure 10.2 - *Scalable segmentation* for refinement and further segmentation

10.2.2 MPEG-7 and Segmentation

As outlined in chapter four, MPEG-7 is a new standardisation effort for audiovisual content description. Visual content will be described based on a number of features extracted automatically, and semi-automatically from the visual data[34]. MPEG-7 will not standardise the method of feature extraction, but will standardise a set of features and appropriate instantiations of these features. Whilst not exclusive to MPEG-4 coded data, MPEG-7 will support the object-based approach of MPEG-4. Low-level object features

34 Of course, some high-level features will be extracted on a completely manual basis (e.g. director, actors, characters, plot synopsis, etc).

such as colour, shape, position, trajectory, composition, etc. may be useful in describing objects. It is clear that segmentation may have an important role to play in MPEG-7.

The segmentation algorithm described in chapter nine may be an appropriate approach to extracting instantiations of low-level MPEG-7 features (instantiations of features are sometimes termed *descriptors*). This is particularly true if the process were to be incorporated into a complete VOP creation environment for MPEG-4 applications as proposed in the previous section. The creation of a VOP immediately addresses features such as the object's shape, location and trajectory in a scene. The descriptors used to represent these features based on a segmentation result is a matter for further research. For example, in order to extract the trajectory of the object in a scene it is necessary to consider the entire sequence of tracked VOPs. The VOPs could be processed in order to extract a polynomial motion model which describes the temporal evolution of the object (this in particular is targeted as a direction for future research).

The output of the segmentation process of chapter nine is not simply a segmentation of the objects present in a scene. The segmentation is based on a characterisation of the objects and as such, the output of the process is also a description of each object present (corresponding to the PDF models converged upon by the iterative process). This description consists of information on the number of region-types making up an object and the colour and spatial characteristics of these region types. This can be considered as addressing features such as colour and composition for an object. It is conjectured that a multimodal multivariate PDF has the potential of being a good starting point for extracting MPEG-7 descriptors for these object features. Again, the exact descriptors to be derived from the PDF is a matter for future research.

As stated in chapter nine, a very good characterisation of an object is not always necessary for segmentation purposes. Rather a characterisation which distinguishes the object from other objects in the scene is required. For MPEG-7 purposes, which will attempt to describe individual objects, this characterisation should be as complete as possible. A more complete characterisation could be obtained by performing an iterative user interaction process. For example, the mode segmentation of the augmented training data based on the initial user input number of modes could be displayed to the user who

may then adjust his/her original estimate to more closely represent the visual data, thus gradually converging on a better overall description of the object.

The underlying objective of MPEG-7 is to make it easier to locate AV data. If an object is to be described using descriptors derived from a PDF, it is necessary to develop a matching criteria in order that searches can be made against stored object descriptions. Developing such a similarity measure is not trivial (e.g. different PDFs may have different features, different numbers of modes, etc.). The development of a suitable similarity metric and an appropriate matching criterion, as well as the generation of good representative parameters in order to form queries, are targeted as important areas for future research.

REFERENCES

[1] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall International Inc., 1989.

[2] K.R. Rao and J.J. Hwang, *Techniques and Standards for Image, Video and Audio Coding*, Prentice-Hall PTR, 1996.

[3] B. Furht, J. Greenberg, and R. Westwater, *Motion Estimation Algorithms for Video Compression*, Kluwer Academic Publishers, 1997.

[4] R. Westwater and B. Furht, *Real-time Video Compression: Techniques and Algorithms,* Kluwer Academic Publishers, 1997.

[5] W. J. Welsh, S. Searby and J. B. Waite, "Model-based Image Coding", British Telecom Technology Journal, Vol. 8, No. 3, pp. 94-106, 1990.

[6] K. Aizawa and H. Harashima, "Model-based Analysis Synthesis Image Coding (MBASIC) System for a Person's Face", Signal Processing: Image Communication, Vol. 1, No. 2, pp 139-152, 1989.

[7] H. G. Musmann, M. Hotter, J. Ostermann, "Object-oriented Analysis Synthesis Coding of Moving Images", Signal Processing: Image Communication 1, pp. 117-138, 1989.

[8] M. Hotter, "Object-oriented Analysis-Synthesis Coding Based on Moving Two-Dimensional Objects", Signal Processing: Image Communication 2, pp. 409-428, 1990.

[9] M. Hotter and R. Thoma, "Image Segmentation based on Object-oriented Mapping Parameter Estimation", Signal Processing, Vol. 15, No. 3, pp. 325-334, October 1988.

[10] T. Sikora, "The European COST211ter Activities - Research Towards Advanced Algorithms for Coding Video Signals at Very Low Bit Rates", Proceedings IEEE International Conference on Image Processing, ICIP'96, Vol. 3, pp 667-670, Lausanne, September, 1996.

[11] N. Brady, N. O'Connor, L. Ward (editors), "SIMOC1", COST211ter Simulation Subgroup Document, Sim(95)08, February, 1995.

[12] M. Bierling, "Displacement Estimation by Hierarchical Block Matching", Proceedings 3rd SPIE Symposium on Visual Communications and Image Processing, pp 942-951, Cambridge, MA, November 1988.

[13] I. A. Witten, R. M. Neal, J. G. Cleary, "Arithmetic Coding for Data Compression", Communication of the ACM, Vol. 30, No. 6, pp. 520-540, June 1987.

[14] L. Vincent, "Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms", IEEE Tans. on Image Processing, Vol. 2, No. 2, pp. 176-201, April 1993.

[15] P. Salembier and M. Pardas, "Hierarchical Morphological Segmentation for Image Sequence Coding", IEEE Trans. on Image Processing, Vol. 3, No. 5, pp. 639-651, September 1994.

[16] L. Vincent and P. Soille, "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 13, No. 6, pp. 583-598, June 1991.

[17] N. Brady, N. O'Connor, L. Ward, "Evaluation Report for DCU Object-based Encoder", Internal Technical Report, Video Coding Group, Dublin City University, January 1994.

[18] F. Pereira, "MPEG-4: A New Challenge for the Representation of Audiovisual Information", Proceedings of PCS'96, Melbourne, March, 1996.

[19] R. Koenen, F. Pereira and L. Chiariglione, "MPEG-4: Context and Objectives", Signal Processing: Image Communication, Special Issue on MPEG-4. Part 1: Invited Papers, pp 295-305, Vol. 9 No. 4, May 1997.

[20] N. E. O'Connor, S. A. Winder, "Current Developments in MPEG-4 Video", Proceedings of ADVICE'96, Cambridge, June 1996.

[21] T. Ebrahimi, "MPEG-4 Video Verification Model: A video encoding/decoding algorithm based on content", Signal Processing: Image Communication, Special Issue on MPEG-4. Part 1: Invited Papers, pp. 367-384, Vol. 9, No. 4, May 1997.

[22] R. H. Koenen, "Overview of MPEG-7 Goals and Objectives", Proceedings of WIAMIS'97, pp. 111-116, Louvain-la-Neuve, Belgium, June, 1997.

[23] R. Mech and M. Wollborn, "A Noise Robust Method for Segmentation of Moving Objects in Video Sequences", Proceedings of ICASSP'97, Munich, April 1997.

[24] R. Mech and M. Wollborn, "A Noise Robust Method for 2-D Shape Estimation of Moving Objects in Video Sequences considering a Moving Camera", Proceedings of WIAMIS'97, pp. 57-62, Louvain-la-Neuve, June, 1997.

[25] H. Adelson and J. Y. A. Wang, "Representing Moving Images with Layers", IEEE Transactions on Image Processing, Vol. 3, No. 5, Sept. 1994.

[26] Y. A. Wang and E. H. Adelson, "Layered Representation for Motion Analysis", Proceedings of the IEEE Computer Vision and Pattern Recognition Conference, pp. 361-366, New York, June 1993.

[27] Y. A. Wang and E. H. Adelson, "Spatio-Temporal Segmentation of Video Data", Proceedings of the SPIE: Image and Video Processing II, Vol. 2182, San Jose, February 1994.

[28] COST211ter Group Members, "COST211ter Simulation Subgroup Focus Document Ver 2.0", http://www.teltec.dcu.ie/cost211ter/focus.html, Nov. 1996.

[29] A. Aydin Alatan, E. Tuncel and L. Onural, "A Rule-based Method for Object Segmentation in Video Sequences", Proceedings IEEE International Conference on Image Processing, ICIP'97, Vol. 2, pp. 522-525, Santa Barbara, October, 1997.

[30] A. Steudel and M. Glesner, "Object Tracking in Video Sequences with Fuzzy Contour Refinement", Proceedings of WIAMIS'97, pp. 33-38, Louvain-la-Neuve, June, 1997.

[31] A. Steudel and M. Glesner, "Image Coding with Fuzzy Region-Growing Segmentation", Proceedings IEEE International Conference on Image Processing, ICIP'96, Vol. 2, pp. 955-958, Lausanne, September, 1996.

[32] E. Chalom and V. M. Bove, "Segmentation of an Image Sequence using Multi-dimensional Image Attributes", Proceedings IEEE International Conference on Image Processing, ICIP'96, Vol. 2, pp. 52 5-528, Lausanne, September, 1996.

[33] F. Marqués and C. Molina, "An Object Tracking Technique for Content-based Functionalities", SPIE Visual Communication and Image Processing (VCIP-97), volume 3024, pp. 190-198, San Jose, USA,1997.

[34] C. W. Therrien, *Discrete Random Signals and Statistical Signal Processing*, Ch. 6, pp. 279-336, Prentice Hall Signal Processing Series, 1992.

[35] T. K. Moon, "The Expectation-Maximisation Algorithm", IEEE Signal Processing Magazine, pp. 47-60, Nov. 1996.

[36] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, Ch. 1 & Ch. 2, Wiley Series in Probability and Mathematical Statistics, 1992.

[37] R. A. Redner and H. F. Walker, "Mixture Densities, Maximum Likelihood and the EM Algorithm", SIAM Review, Vol. 26, No. 2, pp. 195-239, April, 1984.

[38] M. Feder and E. Weinstein, "Parameter Estimation of Superimposed Signals Using the EM Algorithm", IEEE Trans. on Acoustics, Speech and Signal Processing, pp. 477-489, Vol. 36, No. 4 , April 1988.

[39] N. Brady and N. O'Connor, "Object Detection and Tracking using an EM-Based Motion Estimation and Segmentation Framework", Proceedings IEEE Conference on Image Processing ICIP'96, Vol. 1, pp. 925-928, Lausanne, September 1996.

[40] C.W. Therrien, *Decision Estimation and Classification*, Ch. 12, pp. 217-219, John Wiley & Sons, 1989.

[41] N. O'Connor, S. Marlow, "Supervised Image Segmentation using EM-based Estimation of Mixture Density Parameters", Proceedings of WIAMIS'97, pp. 27-32, Louvain-la-Neuve, June, 1997.

[42] N. O'Connor, S. Marlow, "Supervised Semantic Object Segmentation and Tracking via EM-based Estimation of Mixture Density Parameters", Proceedings NMBIA'98 (Springer-Verlag) , pp 121-126, Glasgow, July, 1998.

GLOSSARY

alpha plane	image channel specifying the degree of transparency of each pixel in an image
AM	Analysis Model: an evolving description of the COST 211ter common image/video analysis system
arithmetic encoding	an efficient method of encoding data given the probability distribution of the data
BAB	Binary Alpha Block: a subset of pixels from a binary alpha plane (consisting of 16×16 pixels)
CAE	Context-based Arithmetic Encoding: an efficient method for compressing binary image data
CD-ROM	Compact-Disc/Read Only Memory: a high density storage media used for removable read-only memory on a computer system
chroma-key	(also blue-screen) a studio process whereby the shape of a foreground object is directly recoverable from the filmed footage
CIF	Common Intermediate Format: picture format with three components corresponding to a luminance (Y) component and two colour difference (U and V) components (the picture size is 352 pixels per line by 288 lines)
COST 211ter	a European collaborative research group working on aspects of video compression and analysis
DCT	Discrete Cosine Transform: a reversible transform used to achieve redundancy reduction in image compression systems
DPCM	Differential Pulse Code Modulation: a predictive coding method incorporating a feedback loop to avoid error propagation
DSM	Digital Storage Media: any media for storing digitised audio-visual information (e.g. CD-ROM)
EM	Expectation-Maximisation: a robust iterative procedure for obtaining ML estimates in the presence of incomplete data

HDTV	High Definition Television: television services incorporating a widened aspect ratio, increased picture resolution and CD quality audio
INTRA compression	compression of an image by reducing spatial redundancies
INTER compression	compression of an image by reducing temporal redundancies
ISDN	Integrated Services Digital Network: a public digital network intended to replace PSTN while also adding new and improved services
ISO	International Standards Organisation
ITU-T	International Telecommunications Union
MAP	maximum a posteriori: hypothesis testing whereby an entity is assigned to the most probable of a number of available groups based on evaluating each group's PDF
macroblock	a subset of an image on which compression tools are applied (consisting of a 16×16 block of luminance pixels and two 8×8 blocks of chrominance pixels)
MC	Motion Compensation: a method of obtaining a prediction for an image based on estimated motion (see Motion Estimation)
MDL	Minimum Description Length principle: a means of defining an optimisation criterion for model fitting problems where no bound on the complexity of the model is specified
ME	Motion Estimation: the process of estimating the motion present between two successive images in a video sequence (in it's simplest form it is carried out in a block-wise fashion)
ML	Maximum Likelihood: a method of parameter estimation based on a choice of the most likely parameter values according to a fixed set of observations
MPEG	Motion Picture Experts Group: an ISO working group dealing with the standardisation of coding methods for multimedia data
OOASC	Object-Oriented Analysis Synthesis Coding: an approach to video compression based on estimating the parameters of objects present in the scene and synthesising the object based on these parameters
PDF	Probability Distribution (Density) Function

PSTN	Public Services Telephone Network: the conventional analogue telephone network
QCIF	Quarter CIF: picture format which is ¼ the size of CIF
QSIF	Quarter SIF: picture format which is ¼ the size of SIF
RLC	Run-Length Coding: a method of forming coding events based on the number of sequential occurrences of data items in an ordered scan
RSST	Recursive Shortest Spanning Tree: a hierarchical (multivariate) segmentation technique based on successively merging regions in a fine partition
segmentation	the process of grouping pixels in an image according to a common characteristic
SIF	an image format similar to CIF
SIMOC	Simulation Model for Object-based Coding: an OOASC scheme developed by COST 211ter
VLC	Variable Length Coding: an entropy encoding technique whereby more frequently occurring coding events are given shorter code words
VO	Video Object: the MPEG-4 term for a semantic object present in a scene
VOP	Video Object Plane: the MPEG-4 representation of a semantic object at a given time instant corresponding to an image consisting of Y, U and V components and an alpha plane which defines the object's segmentation
watershed	a segmentation technique based on applying immersion simulations to a morphological gradient

APPENDIX A: USEFUL DERIVATIONS

Derivation necessary for Equation 7-5 and Equation 7-14

$$\frac{\partial}{\partial m}\left((x-m)^T A(x-m)\right) = -A(x-m) - A^T(x-m)$$

where x, m are $k \times 1$ vectors, and A is a symmetric $k \times k$ matrix.

Proof:

$$e = (x-m)^T A(x-m)$$

$$= \left[(x_1 - m_1)\ldots(x_k - m_k)\right]\begin{bmatrix} a_{11} & \cdots & a_{1k} \\ \vdots & \ddots & \vdots \\ a_{k1} & \cdots & a_{kk} \end{bmatrix}\begin{bmatrix} (x_1 - m_1) \\ \vdots \\ (x_k - m_k) \end{bmatrix}$$

$$= \left[(x_1 - m_1)\ldots(x_k - m_k)\right]\begin{bmatrix} \sum_{j=1}^{k} a_{1j}(x_j - m_j) \\ \vdots \\ \sum_{j=1}^{k} a_{kj}(x_j - m_j) \end{bmatrix}$$

$$= \sum_{i=1}^{k}(x_i - m_i)\sum_{j=1}^{k} a_{ij}(x_j - m_j)$$

Consider $\frac{\partial e}{\partial m_k}$, (i.e. row k of vector $\frac{\partial e}{\partial m}$):

$$\frac{\partial e}{\partial m_k}\left\{(x_k - m_k)\sum_{j=1}^{k} a_{kj}(x_j - m_j)\right\} + \frac{\partial e}{\partial m_k}\left\{\sum_{i=1}^{k}(x_i - m_i)a_{ik}(x_k - m_k)\right\}$$

$$= -\sum_{j=1}^{k} a_{kj}(x_j - m_j) - \sum_{i=1}^{k}(x_i - m_i)a_{ik}$$

$$= -\sum_{j=1}^{k} a_{kj}(x_j - m_j) - \sum_{i=1}^{k} a_{ki}{}^T(x_i - m_i)$$

$$= -\left(\begin{array}{c}\text{row } k \text{ of} \\ A(x-m)\end{array}\right) - \left(\begin{array}{c}\text{row } k \text{ of} \\ A^T(x-m)\end{array}\right)$$

$$\Rightarrow \frac{\partial e}{\partial m} = \frac{\partial}{\partial m}\left((x-m)^T A(x-m)\right) = -A(x-m) - A^T(x-m)$$

Example 1: ML Estimation of a Univariate Gaussian PDF

The PDF of the *random variable* takes the form:

$$p(x|\theta)=\frac{1}{\sqrt{2\pi\theta_2}}\exp\left[-\frac{(x-\theta_1)^2}{2\theta_2}\right]$$

which is completely defined by the parameters $\theta=\begin{bmatrix}\theta_1 & \theta_2\end{bmatrix}^T$, where $\theta_1=m$ is the mean and $\theta_2=\sigma^2$ is the variance of the distribution.

Considering N independent observations, the likelihood function can be written as:

$$l_x(\theta)=p(x_1,\cdots,x_N|\theta)=\frac{1}{(2\pi)^{\frac{N}{2}}(\theta_2)^{\frac{N}{2}}}\exp\left[-\frac{\sum_{j=1}^{N}(x_j-\theta_1)^2}{2\theta_2}\right]$$

Accordingly, the log likelihood function can be written as:

$$L_x(\theta)=\log p(x_1,\cdots,x_N|\theta)=-\frac{N}{2}\log(2\pi)-\frac{N}{2}\log(\theta_2)-\frac{\sum_{j=1}^{N}(x_j-\theta_1)^2}{2\theta_2}$$

To obtain θ_1^{ML} set $\frac{\partial}{\partial\theta_1}L_x(\theta)=0$:

$$\Rightarrow-\frac{2\sum_{j=1}^{N}(x_j-\theta_1)}{2\theta_2}(-1)=\frac{\sum_{j=1}^{N}(x_j-\theta_1)}{\theta_2}=0$$

$$\Rightarrow\sum_{j=1}^{N}(x_j-\theta_1)=0$$

$$\theta_1^{ML}=m^{ML}=\frac{1}{N}\sum_{j=1}^{N}x_j$$

To obtain θ_2^{ML}, set $\frac{\partial}{\partial\theta_2}L_x(\theta)=0$

$$\Rightarrow-\frac{N}{2\theta_2}+\frac{1}{2\theta_2^2}\sum_{j=1}^{N}(x_j-\theta_1)^2=0$$

$$\theta_2^{ML}=\left(\sigma^2\right)^{ML}=\frac{1}{N}\sum_{j=1}^{N}(x_j-\theta_1)^2$$

Example 2: ML Estimation of Mixtures of Univariate Gaussian PDFs

The i^{th} PDF in the mixture is written as:

$$p_i(x|\theta_i) = \frac{1}{\sqrt{2\pi\theta_{i_2}}} \exp\left[\frac{(x-\theta_{i_1})^2}{2\theta_{i_2}}\right]$$

which is completely defined by the parameters $\theta_i = \begin{bmatrix}\theta_{i_1} & \theta_{i_2}\end{bmatrix}^T$, where $\theta_{i_1} = m_i$ is the mean of group G_i, and $\theta_{i_2} = \sigma_i^2$ is the variance of group G_i. The maximising expression then becomes (see chapter seven):

$$\sum_{i=1}^{K}\sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial\theta_i}\left[-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\theta_{i_2}) - \frac{(x_j-\theta_{i_1})^2}{2\theta_{i_2}}\right] = 0$$

To obtain $\theta_{i_1}^{ML.}$:

$$\sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial\theta_{i_1}}\left[-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\theta_{i_2}) - \frac{(x_j-\theta_{i_1})^2}{2\theta_{i_2}}\right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij} \frac{(x_j-\theta_{i_1})}{\theta_{i_2}} = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij}(x_j-\theta_{i_1}) = 0$$

$$\theta_{i_1}^{ML.} = m_i^{ML.} = \frac{\sum_{j=1}^{N} \tau_{ij} x_j}{\sum_{j=1}^{N} \tau_{ij}}$$

To obtain $\theta_{i_2}^{ML.}$:

$$\sum_{j=1}^{N} \tau_{ij} \frac{\partial}{\partial\theta_{i_2}}\left[-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\theta_{i_2}) - \frac{(x_j-\theta_{i_1})^2}{2\theta_{i_2}}\right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij}\left[-\frac{1}{2\theta_{i_2}} + \frac{(x_j-\theta_{i_1})^2}{2\theta_{i_2}^2}\right] = 0$$

$$\Rightarrow \sum_{j=1}^{N}\left[-\tau_{ij}\theta_{i_2} + \tau_{ij}(x_j-\theta_{i_1})^2\right] = 0$$

$$\Rightarrow \sum_{j=1}^{N} \tau_{ij} \theta_{i_2} - \sum_{j=1}^{N} \tau_{ij} (x_j - \theta_{i_1})^2 = 0$$

$$\theta_{i_2}^{ML} = (\sigma_i^2)^{ML} = \frac{\sum_{j=1}^{N} \tau_{ij} (x_j - \theta_{i_1})^2}{\sum_{j=1}^{N} \tau_{ij}}$$

APPENDIX B: EXTENDED RESULTS

In this section, a selection of segmentation results obtained using the object-based scheme of chapter nine applied to various image resolutions/formats are presented. Two MPEG-4 test sequences were used to generate these results: Mother and Daughter (CIF) and Fish (SIF). Also presented is a segmentation of a JPEG test image of resolution 522 $\times$ 404. The resultant segmentations are scaled down so that they can be arranged appropriately in the document.

The only segmentation parameters changed from those outlined in chapter nine in order to generate these results were associated with motion estimation. For both Fish and Mother and Daughter the motion estimation search range was doubled in order to accommodate these larger image resolutions.

Mother and Daughter Test Sequence (CIF)

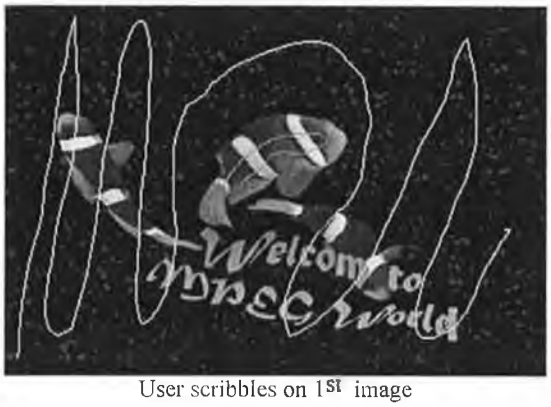
User scribbles on 1st image
mother and daughter - 6 modes
background - 4 modes

Mother and daughter segmented in 1st image

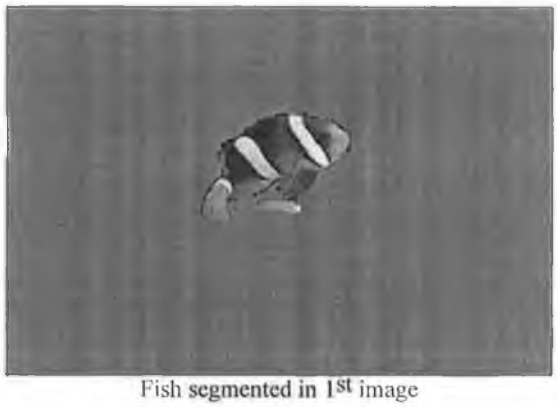
Mother and daughter segmented in 41st image

Mother and daughter segmented in 91st image

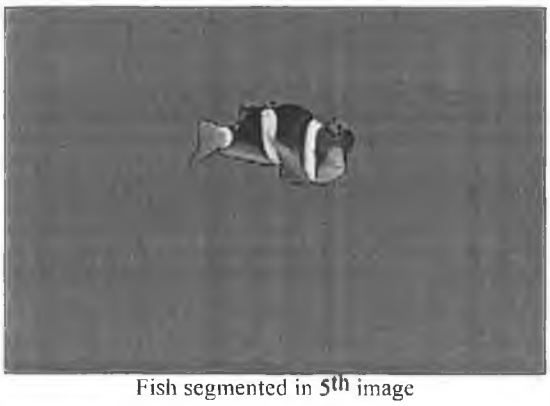
Fish Test Sequence (SIF)



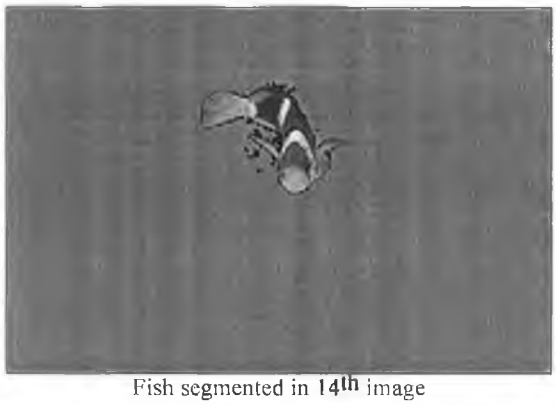
User scribbles on 1st image
fish - 3 modes
background - 5 modes



Fish segmented in 1st image

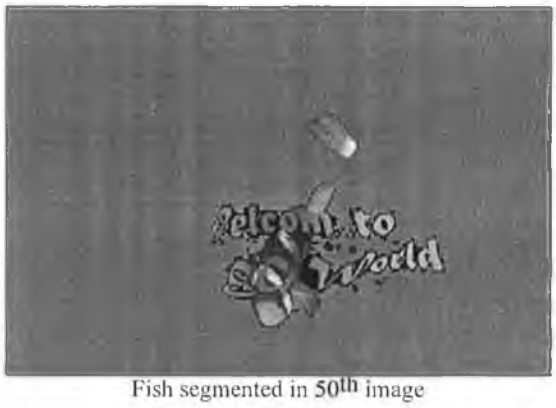


Fish segmented in 5th image

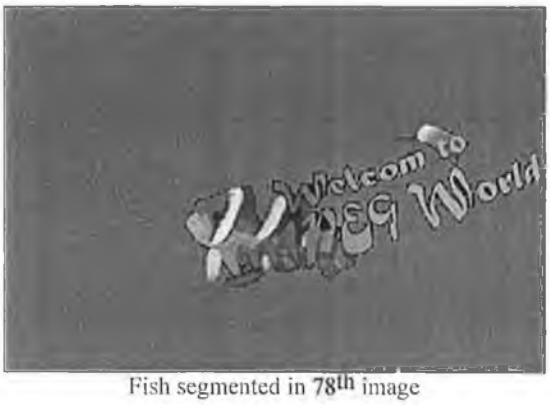


Fish segmented in 14th image

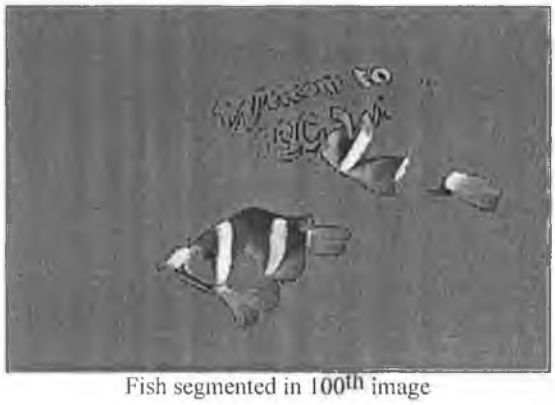
Fish segmented in 47th image



Fish segmented in 50th image



Fish segmented in 78th image



Fish segmented in 100th image

JPEG Test Image

User scribbles on original image
foreground object - 4 modes
background object - 3 modes

Segmentation of foreground object

Segmentation of background object