DUBLIN CITY UNIVERSITY

SCHOOL OF ELECTRONIC ENGINEERING

Semi-automatic Video Object Segmentation for Multimedia Applications

A thesis submitted for the award of
M. Eng. Degree in Electronic Engineering at
Dublin City University

by
Saman Hemantha Cooray, B. Sc. (Eng)

Supervised by
Dr. Noel O'Connor

June 2003

DECLARATION

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of M. Eng. 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: ..... ID No: 98971298

Saman Hemantha Cooray

Date: 15.06.2003

ACKNOWLEDGEMENTS

First of all, I wish to express my sincere gratitude to Dr. Noel O'Connor for his excellent guidance, affectionate counselling and his encouragement thoughout the course of this research and writing of this thesis. It would not have been possible to complete this work without his guidance and stimulating suggestions.

I would like to thank Dr. Tommy Curran for providing me his kind assistance and other necessary facilities to succeed this work.

I also wish to thank Dr. Sean Marlow and Dr. Noel Murphy for their encouragement and suggestions offered to me throughout this work. My thanks also go to all other members in the VMPG group.

Special thanks then go to my colleague Sean Murphy who used to work in the same lab J119, for his help and many fruitful discussions. I am also grateful to other colleagues who have helpmed me in numerous ways to succeed this work.

My heartiest thanks are offered to my parents and family, who are thousands of miles away at this moment, for their role of encouragement and sacrifice, and for what they are yet to experience in the coming years.

I must give my special thanks to all my Sri Lankan friends for their lovely and entertaining e-mails to keep me in a healthy position. *I can remember that there was a time after April the 27th 1999 I quitely sang some Sri Lankan songs during your absence.*

ABSTRACT

Semi-automatic Video Object Segmentation for Multimedia Applications

Author: Saman Hemantha Cooray

A semi-automatic video object segmentation tool is presented for segmenting both still pictures and image sequences. The approach comprises both automatic segmentation algorithms and manual user interaction. The still image segmentation component is comprised of a conventional spatial segmentation algorithm (Recursive Shortest Spanning Tree (RSST)), a hierarchical segmentation representation method (Binary Partition Tree (BPT)), and user interaction. An initial segmentation partition of homogeneous regions is created using RSST. The BPT technique is then used to merge these regions and hierarchically represent the segmentation in a binary tree. The semantic objects are then manually built by selectively clicking on image regions. A video object-tracking component enables image sequence segmentation, and this subsystem is based on motion estimation, spatial segmentation, object projection, region classification, and user interaction. The motion between the previous frame and the current frame is estimated, and the previous object is then projected onto the current partition. A region classification technique is used to determine which regions in the current partition belong to the projected object. User interaction is allowed for object re-initialisation when the segmentation results become inaccurate. The combination of all these components enables offline video sequence segmentation. The results presented on standard test sequences illustrate the potential use of this system for object-based coding and representation of multimedia.

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

1.1 Overview

In this chapter, a general context of the research undertaken is first described, highlighting the evolution of multimedia technologies in recent years. Current and future trends in multimedia research are briefly described for the sake of justifying the choice of the following research as a Masters project. The rest of the chapter is then devoted to describing the objectives of the research and the structure of the thesis. The main objectives of the research are explained according to their degree of importance in section 1.3. The structure of the thesis is then described in section 1.4.

1.2 Introduction

The conversion of thoughts into words, words into actions, and actions into pictures has brought today's multimedia services to an astonishing level. It is remarkable how much the technology has improved in the field of multimedia, particularly in recent years. People can nowadays consume combined media such as graphics, text, audio, and video, in an interactive manner with many more new functionalities in the future.

The most performance-demanding component of multimedia is undoubtedly video which is addressed throughout this thesis. Video has entered various spheres of modern life, in particular, the current convergence of televisions and computers into new systems, enabling them to be used for entertainment and work at the same time. However, one wouldn't have expected such milestones to come to pass within such a short time. The credit should be given to the researchers and system developers, working in different fields such as computer, telecommunications, and consumer electronics, who have done an excellent job delivering the required services to the end-users efficiently and quickly.

Clearly, the most well known and important application in video is television broadcasting. Television broadcasting was introduced after the Second World War II. It has been developed in various phases, and the most important revolution was the concept of digital transmission of television, i.e. migration from analogue to digital, leading to various applications hitherto impossible. Most importantly, the merging of

video and audio with computers and networks was seen in the mid nineties, leading to the revolutionary concept of multimedia. These issues are currently being further developed by the research community in an attempt to bring the whole world to a state called "virtual reality".

A worldwide penetration of desktop computing in homes and businesses, in recent years, has stimulated many more interactive multimedia applications. These trends have been largely fuelled by the growing success of the Internet, enabling large varieties of different applications. The largest part of the multimedia data stream is, however, occupied by the video information that needs to be transmitted along with other content. It was looking almost impossible to deliver video until recently when new data compression technology and network technology issues were resolved to a certain extent. However, certain complications still remain if streamed video is to be delivered over a data network. This is due to the fact that streaming digital video data to the end-users takes about one to three Megabits per second, which is currently beyond the bandwidth available for personalised user applications.

Whilst the bandwidth constraint remains to be resolved in World Wide Web (WWW) based multimedia applications, television service providers are dealing with the issues of realising two-way communication, if interactivity is to be provided at the end-user terminals. In other words, whatever the transmission media (i.e. cable TV, Satellite TV, terrestrial TV) being used to deliver the service, a back channel is necessary to carry the control signals, if new interactive services are to be brought to home television viewers. Some of the new interactive services could be movies on demand, news on demand, and interactive sports such as personalized sports viewing. In interactive TV services, a phone line is connected to set-top boxes so as to provide a back channel. However, the introduction of these interactive services causes the system to be highly client-server oriented, and it further increases the complexity of the bandwidth and processing power requirements. The Asynchronous Transfer Mode (ATM) protocol and the Integrated Digital Service Network (ISDN) connection network support the transporting of all kinds of service data. Even with ATM, the amount of uncompressed video data is too high to handle, and therefore a mechanism for reducing the amount of data is necessary. To this end, efficient techniques for compressing audio and video, mostly developed within the Motion Picture Experts Group (MPEG) arena, are considered as a viable solution. In this context, MPEG-1, MPEG-2 and MPEG-4 have largely contributed to

resolving the compression issues to facilitate the high bandwidth demanding and interactive services in multimedia applications.

Recently, the provision of multimedia services has been carried over to mobile telephone applications. Due to high bandwidth requirements for these services, currently available mobile communication services haven't proved very successful in providing full services. However, third generation mobile, also known as the Universal Mobile Telecommunication System (UMTS), will support novel services such as multimedia messaging services, games, and virtual tourism to mobile phone users. Nevertheless, due to the current problems such as display, storage and processing limitations of the mobile phones, the success of this innovation is still too early to predict.

Unlike MPEG-1 and MPEG-2, MPEG-4 and MPEG-7 address content based-functionalities to facilitate numerous applications (see chapter 2). These content-based functionalities require some means of effectively describing the audio-visual content. In this context, it is necessary to facilitate easy access to the image data or the content itself if further processing (interaction, manipulation) needs to be performed to efficiently describe and encode them. This task is efficiently achieved by representing the content in the form of objects.

Keeping in mind the importance of rapidly growing multimedia applications, and particularly content-based functionalities, a project on "Semi-Automatic Video Object Segmentation for Multimedia Applications" is undertaken to fulfil a requirement for a research Masters degree.

1.3 Objectives of the Research

The primary objective of this research is to develop a tool for interactive video object segmentation. The main subject of the research work is how to extract semantically meaningful video objects from scenes, in order to facilitate future content-based multimedia applications. In general, the difficulty of extracting a particular object from a scene is application-dependent. Thus, this subject area can be discussed under two categories called "on-line" and "off-line" segmentation. On-line methods are also called unsupervised segmentation (i.e. no user interaction is involved), and off-line

methods are also called supervised (i.e. user interaction is provided) segmentation (see also section 3.2.2). To this end, an attempt is made in our research to determine how difficult or easy it is to extract semantic objects from a given image in terms of the amount of user interaction involved. Furthermore, the task of segmenting video sequences is investigated by incorporating an object-tracking algorithm into the spatial object segmentation tool.

In order to achieve the primary goals, an ancillary objective of the research is to carry out a survey of various current and future trends in multimedia applications. The author's research is targeted at content-based functionalities coming under the auspices of standards such as MPEG-4 and MPEG-7, where video objects have been identified as the core element for content manipulations.

Other research objectives include studying and identifying various segmentation methods both in the spatial and temporal domains. Efforts are also made to identify the suitability of automatic segmentation and semi-automatic segmentation approaches. In this context, automatic spatial segmentation methods are first identified. Secondly, the methods to hierarchically represent the automatically generated results are identified. Video object-tracking algorithms are also studied to identify a suitable method of extending this system to a complete video object segmentation tool.

The final objective of the research can be considered to be identifying future work-areas. In this context, some areas of possible improvements in various components, such as the automatic segmentation process, the manipulation of user interaction, region classification for object tracking, etc., are considered to be important for further study.

In order to achieve the above goals, it is decided to develop the front-end system in a Graphical User Interface (GUI) form. The GUI allows the user to open up image files and to run segmentation algorithms. Further operations such as segmentation tree browsing, region selection, region deletion, region split, object-extraction, etc. are provided in the form of simple buttons, mouse-clicks, mouse-drags, menus, and a slider bar. The entire implementation is carried out in the Java programming language under Microsoft Windows operating system.

1.4 Structure of the Thesis

The thesis is organised as follows: Chapter 2 focuses on still picture and moving picture coding standards. This chapter starts with an introduction to the fundamentals of video coding in section 2.2, outlining the need for data compression and the means of achieving this. Various video formats and colour spaces are described in section 2.3 and 2.4. Block-based video compression and segmentation-based video compression are then addressed in section 2.5; describing the manner in which the segmentation process is utilised in later MPEG standards, such as MPEG-4 and MPEG-7. Subsequently, still image compression standards, such as JPEG and JPEG2000, are described in section 2.6, outlining the evolution of the image coding standards. Moving picture compression and description standards, such as H.261, H.263, MPEG-1, MPEG-2, MPEG-4, and MPEG-7, are described in detail in section 2.7.

In chapter 3, a detailed discussion of still image segmentation is presented, covering most of the theoretical and conceptual aspects of segmentation. Starting with a brief introduction highlighting the need for segmentation in various applications, some basic definitions, such as Regions vs Objects and automatic segmentation vs semi-automatic segmentation, are given in sections 3.2.1 and 3.2.2. Some of the more well-known segmentation methods are then presented in section 3.2.3. In section 3.2.4, segmentation based on a region-merging technique is described, followed by a description of hierarchical segmentation in section 3.2.5. Automatic segmentation algorithms such as morphological watershed and RSST are described in sections 3.3.1 and 3.3.2 respectively. A segmentation representation method, BPT, is described in section 3.4.1.

The fourth chapter is devoted to video object tracking. This chapter is organised in such a way that the main emphasis is on describing various object-tracking algorithms, illustrating their suitability under various scene-constraints. Hence, basic concepts along with some of the tracking difficulties encountered in practice are first described in section 4.1. This is then followed by a detailed discussion of well-known tracking algorithms, such as edge-based and region-based approaches, in section 4.2.

Implementation details are presented in chapter 5. Starting with an overview of the system, design details, illustrated by a system block diagram, are given in section 5.2. The implemented system, broken down into two stages, is further described in sections

5.2 and 5.3 under the headings Initial Object Segmentation and Object Tracking respectively. The GUI through which the user interaction is provided is described in section 5.4. Various graphical components of the system are individually addressed in section 5.5 to describe their importance within this system. The rest of the chapter, presented in sections 5.6 and 5.7, is a description of the software architecture of the system and some concluding remarks.

Experimental results and discussions are presented in Chapter 6. Results obtained for several MPEG-4 test sequences are presented both for spatial object segmentation and image sequence segmentation in sections 6.2 and 6.3 respectively. For spatial object segmentation, results are discussed in the form of both automatically and semi-automatically generated segmentations. Object-tracking results obtained from several experiments, presented in section 6.3, are subject to a detailed discussion in order to evaluate the performance of the tracking algorithm.

Conclusions are drawn in chapter 7. A review of the work carried out within this thesis is given in section 7.2. This is followed by some recommendations for future work, which warrant/require further research and development work.

Finally, publications arising from the research or associated with similar work are listed along with some appendices, which the reader might find useful.

2. STILL PICTURE AND MOVING PICTURE CODING STANDARDS

2.1 Overview

This chapter presents an overview of the most commonly used still image and video compression standards, such as JPEG, JPEG2000, H.261, H.263, MPEG-1, MPEG-2, MPEG-4, and MPEG-7. The chapter starts with an overview of data compression followed by a description of the different video formats and colour spaces, which are commonly used in these standards. Block-based compression and segmentation-based compression aspects are then discussed in section 2.5. Still image compression standards, such as JPEG and JPEG2000, are described in section 2.6. A detailed description of moving picture compression and description standards is then given in section 2.7. Finally, in section 2.8, a discussion reviews these standards, and briefly outlines the use of different standards for various applications.

MPEG standards are generally called audio/video compression standards. However, due to the recent emphasis on content-based functionalities (e.g. MPEG-4 and MPEG-7), a different terminology may be required. In this respect, it would be more appropriate to call them either as *multimedia coding and description standards* or simply as *multimedia standards*.

2.2 Overview of Data Compression

The efficient representation of digital image and video data has been the objective of a great research effort for more than the last two decades. Due to the success of these attempts, the results achieved, in particular the wide set of international standards that have emerged, have provided a valuable basis to enable a large number of applications in several fields, including advanced personal communications (teleconferencing, video-telephony), remote operation (teleworking, distance learning), interactive digital services (multimedia applications, TV program production). In this framework, the growing availability of transmission links, the drastically improved data carrying capacity of computer networks, the enormous progress in digital signal processing, and the development of VLSI technology with application to image and video compression make visual communications more feasible than ever.

In this process, a fundamental role is played by efficient data representations. The idea of data compression is to remove redundant data from the data stream. Only the use of efficient data compression techniques allows reducing this to a rate convenient for cost-effective applications. The precise characteristics of such techniques depend on the available bandwidth, the application requirements, and the allowable complexity or cost of the processing equipment.

Data compression techniques can be divided into two major families called, lossy and lossless. Lossy data compression concedes a certain loss of accuracy in exchange for greatly increased compression. Lossless compression consists of those techniques guaranteed to generate an exact replication of the input data stream after a encode/decode cycle. This type of compression is mandatory when transmitting or storing database records, spreadsheets, or word processing files where a single bad bit could lead to disaster. Lossless compression uses two different types of modelling called, Statistical-based and Dictionary-based. Shannon-Fano, Huffman, and Arithmetic coding algorithms are some of the examples for Statistical based methods, whereas LZ77, LZ78, LZSS and LZW are some of dictionary-based coding methods. A detailed discussion of these data compression techniques is outside the scope of this thesis, but their details can be found in [Nelson and Gailly 1995].

In the television broadcasting industry, availability of video data in digital form and ability to compress them to a data-rate convenient for cost effective transmission have facilitated bringing new services to the television viewers. For example, a digitized raw PAL color picture (720×576) at 8 bits per sample produces 1215 kBytes of data. Transmission of such frames at 25 frames/s requires about 158 Mbit/s bandwidth. This situation is worse for High Definition Television (HDTV) which needs even higher resolution pictures and higher frame rates. Therefore, efficient video data compression is a much-needed requirement in this industry. MPEG-2 targets coding algorithms for HDTV applications at 4-9 Mbit/s [Sadka 2002].

The need for effective data compression is evident in almost all applications where storage and transmission of digital images are involved. For example, 640 pixels × 480 lines VGA colour image generates 921600 bytes of data if each colour channel is coded in one byte (8 bits). This requires about 1.9 minutes for transmission over a 64kbit/s

media without compression [Netravali and Haskell 1995]. In storing such an amount of data, a 700 Mbyte CD can hold only 760 such images.

Luckily, compression of image data without significant degradation of the visual quality is usually possible because images contain high degrees of:

- **Spatial and Temporal redundancy** - This is due to correlation between neighbouring pixels within the frame, and between frames in the case of video sequences;
- **Psychovisual redundancy** - The eye-brain mechanism creates what is known as perceptual redundancy. By exploiting these properties, visibility of impairments caused by the bit reduction can be minimised.

The higher the redundancy, the higher the achievable compression. The performance of any compression system can be evaluated by considering its apparent image quality, data rate, and system cost.

2.3 Different Video Formats

In image processing, digitised video is represented in various different formats. The advantage of the human vision system being more sensitive to changes in luminance than to changes in chrominance was taken into consideration when the CCIR-601 recommendation was created to define digitisation parameters for video in YC_bC_r component 4:2:2 format. In 4:2:2 format, only two samples of C_b and two samples of C_r are used for 4 samples of Y, thereby reducing the horizontal resolution of chrominance components by half. The following video formats were later derived from CCIR-601: Standard Input Format (SIF) for video frames in digital television, Common Intermediate Format (CIF) for frames in video conferencing. For further low bit rate applications, Quarter-CIF (QCIF) has been defined. **Table 1** shows different video formats used in digital video processing. An in-depth discussion of this topic is not provided since it is beyond the scope of this thesis.

Image Format	Resolution (Luminance)	Sub-sampling (YC_bC_r)
CCIR-601 (NTSC)	720×480	4:2:2
CCIR-601 (PAL)	720×576	4:2:2
SIF (NTSC)	360×240	4:2:0
SIF (PAL)	360×288	4:2:0
CIF (NTSC & PAL)	352×288	4:2:0
QCIF (NTSC & PAL)	176×144	4:2:0

Table 1 Different video formats

2.4 Different Colour Spaces

Colour images and videos are usually displayed in a basic colour space model called, RGB. However, in image processing, colour images are presented in a number of different colour-space models for compression and representation purposes. The selection of a particular colour space depends on its effectiveness to the application (video coding, video processing, content representation, etc.) for which the image data is to be used. For this reason, it can often be found that different colour transformations are used in research in order to optimise the performance of the application [Manjunath et al 2001]. Of the several models available, the most important and widely used ones, such as {R, G, B}, {Y, C_b, C_r}, {H, S, V}, {C, M, Y, K}, and {H, M, M, D}, are briefly described below.

RGB: This represents the three primary colours Red (R), Green (G) and Blue (B). This form of colour is mainly used for displaying and capturing purposes.

YC_bC_r: This is the most commonly used ITU standard colour space for compression and transmission purposes in video coding systems. "Y" represents the brightness (luminance) and "C_b", "C_r" represent colour (chrominance). In this model, the brightness component is decoupled from the colour components. YUV is yet another very similar (and interchangeably used) colour space, and is used in the PAL TV

system [Vasudev and Konstantinos 1997]. Both these models are linear transformations from RGB components.

HSV: The HSV colour space is another very commonly used colour model in image retrieval applications, but the transformation associated with it is more complicated than the above ones. This consists of three components called, Hue (H), Saturation (S), and Value (V). This space is also known as HIS where "I" stands for Intensity. "H" and "S" correspond to colour information, while "V" or "I" corresponds to brightness. Hue specifies the pure colour, while S specifies the amount of white light mixed with the pure colour. Once again, colour and intensity are independent as in the YC_bC_r system.

CMYK: This is a somewhat strange acronym to represent Cyan (C), Magenta (M), Yellow (Y), and Black (K). However, it is a rather simpler linear transformation to represent secondary colours, which are also called subtractive colours. This was originally CMY, and a fourth component K was introduced later due to the failure of the system CMY to produce black [Efford 2000]. This is a useful model for printing colour images [Efford 2000].

HMMD: This is a similar and competitive model to HSV. There are four components in this non-linear transformation model, representing Hue (H), Min (M), Max (M), and Distance (D). Min indicates the tint property, giving an idea of how much white colour the image contains, whereas Max indicates the shade property, giving an idea of how much black colour the image contains. Distance indicates how much grey colour the image is composed of. This space may be useful in image retrieval applications, for example, in MPEG-7.

2.5 Block-based Compression and Segmentation-based Compression

In video coding, compression can be achieved either in a block-based manner or in a segmentation-based (nonblock-based) manner. In this section, the two approaches are discussed with regard to their use in compression, also highlighting their role in past and current MPEG standards.

The level of abstraction introduced within these standards enables us to categorise these standards into two areas, revealing that the MPEG standards are now migrating from issues of conventional compression to new issues of content-based functionality. It is well known that MPEG-1 and MPEG-2 standards mostly follow the same coding techniques, despite the fact that they are targeted for different applications. They work on small square blocks of pixels, efficiently performing compression by using motion estimation/compensation along with transform coding and entropy coding techniques. Consequently, these block-based compression approaches cause a type of image distortion called "blocking artefacts" due to imperfect reconstruction. Perfect reconstruction cannot always be guaranteed, particularly at high compression ratios. Blocking artefacts are visible as spatial discontinuities, which are highly undesirable in some applications.

On the other hand, the next generation of multimedia standards such as MPEG-4 and MPEG-7 mostly work on the basis of arbitrary-shaped regions and objects instead of small square blocks. It should be noted that, in MPEG-7, such regions and objects are not used in the context of compression, as the MPEG-7 standard has a different scope from that of earlier standards. MPEG-4 based applications exploit segmentation as a non-normative part of the standard in order to extract and encode arbitrary shaped objects. The shape encoding within MPEG-4 is carried out using a type of bitmap-based coder, which classifies each pixel of the block to be coded as belonging to the object or not. Hence, in MPEG-4, compression between frames is achieved by encompassing features of both segmentation and conventional block-based techniques. A detailed discussion on image and video segmentation is given in chapters 3 and 4.

In a segmentation-based compression approach, since the scene can be separated into several entities, each entity can be encoded according to the requirement of the user. This method of encoding entities separately can also improve the subjective coding efficiency since the coding can be performed according to the user's best interest instead of using one single technique for the entire process. However, an extra cost has to be paid for encoding the shape of the entity compared to a fixed shape square block. One method of shape encoding is known as contour-based coding where the outline or the boundary of the object is coded by using techniques, such as chain coders or polygon approximations. The higher cost involved in contour coding is considered to

be a bottleneck in segmentation-based coding approaches since the contour coding occupies a larger bit-stream compared to that of the texture coding. This highlights the fact that segmentation-based compression techniques need a powerful shape-coding tool.

2.6 Still Image Compression Standards

Still image compression standards were developed for coding single frame colour images. Committees such as ISO/ITU Joint Photographic Experts Group (JPEG) and ISO/ITU Joint Bi-level Image Group (JBIG) have contributed to this task over the last few decades. The term "Joint" arises because of the collaboration of the two groups, ISO and ITU. In this thesis, only a short description of the JPEG standard is given, in order to outline the historical development of image coding standards. JPEG was published as an international standard in 1992 [Wallace 1992]. In JPEG, compression is achieved either in a lossy or lossless manner. Similarly, there are several modes defined within JPEG called, baseline, lossless, progressive, and hierarchical. Our discussion is only concerned about the most popular mode, i.e. baseline mode, which is based on lossy coding only. Furthermore, the latest JPEG standard, JPEG2000 is also described. Other recent image compression standards, such as JPEG-LS, MPEG-4 Visual Texture Coding (VTC), and Portable Network Graphics (PNG), are considered to be outside the context of our discussion.

2.6.1 ISO/IEC JPEG Coding

JPEG can be considered to be one of the most successful and well-recognised generic image compression standards to come into being. Due to the success of its performance, it has been used in wide range of transmission and storage applications to date. This image format is widely used for digital photography, desktop publishing, the Internet, and so on. In JPEG, transform coding is based on the Discrete Cosine Transform (DCT), which is used to convert image data from the time domain to the frequency domain. The DCT is also an orthogonal transformation technique, in the sense that its inverse transformation produces the exact replica of its input data. Compression can be achieved due to its ability to pack most of energy into a few coefficients, enabling other DCT coefficients to be discarded without significant quality loss. The DCT is widely used by other well-established standards, such as H.261, H.263, MPEG-1, MPEG-2, and MPEG-4. The actual compression ratio obtainable from

this standard can vary from 100:1 to 2:1 depending on the specific application and encoder/decoder complexity [Rao and Hwang 1996].

Although colour conversion is part of the redundancy removal process, it is not part of the JPEG standard. JPEG handles colours as separate components in commonly used colour spaces, such as RGB, YC_bC_r, and CMYK. For each separate colour component, the full image is decomposed into 8×8 blocks of pixels, which form the input to the DCT. Typically, in the 8×8 blocks, the pixel values vary slowly, and hence the energy is of low spatial frequency. Thus, a 2D DCT is quite capable of concentrating the energy into a few coefficients, thereby facilitating a high degree of compression. The block diagram of the JPEG coding process is shown in **Figure 2-1**.

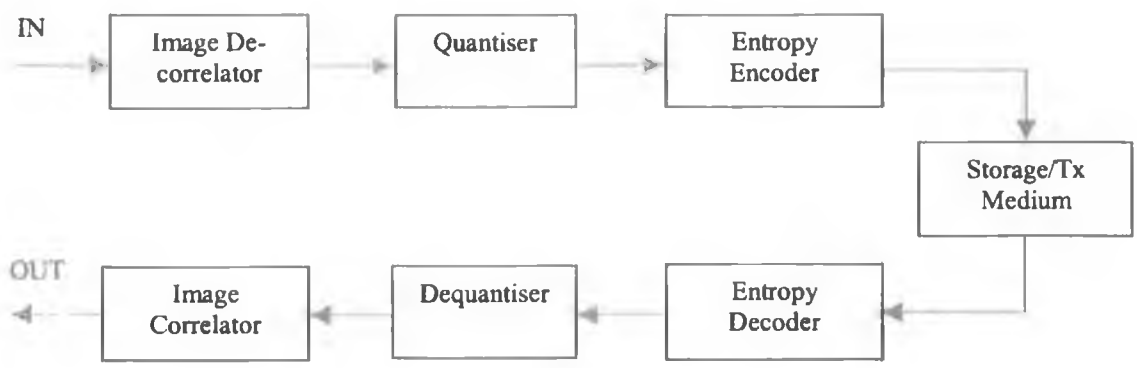


Figure 2-1 JPEG Coding Model

The operation of the JPEG algorithm can be divided into three basic stages:

- The removal of spatial redundancy by means of the DCT, i.e. Decorrelation;
- The quantisation of the DCT coefficients using weighting functions optimised for the human visual system;
- The encoding of the data to minimise the entropy of the quantised DCT coefficients using the Huffman variable length coding.

The image decorrelator is the first step in the JPEG coding method. This process generates the same number of coefficients as pixels, but the upper left corner coefficient is now called the DC coefficient while the rest are called the AC

coefficients. The AC coefficients represent increasing horizontal and vertical spatial frequencies. An ideal transform should completely decorrelate the data in a block, i.e. it should pack most amount of energy in the fewest number of coefficients. The DCT has been found to be extremely efficient for highly correlated data. Depending on the uniformity of the block, the number of non-zero AC coefficients tends to vary, and hence the overall compression. However, if the block is perfectly uniform, only the DC coefficient will be non-zero.

The second step is the quantisation of the frequency coefficients. The input to the DCT consists of eight bit pixel values, but the coefficient values can range from a low of -1,024 to a high of 1,023, occupying eleven bits. The action used to reduce the number of bits required for storage of the DCT matrix is referred to as "Quantisation". The JPEG algorithm implements quantisation by using a quantisation matrix. For every element position in the DCT matrix, a corresponding value in the quantisation matrix gives a quantum value ranging from 1 to 255.

Next, the coding model rearranges the quantised frequency coefficients into a zigzag pattern, with the lowest frequency first and the highest frequency last. The reason for using the zigzag pattern is to increase the run-length of zero coefficients found in the block. Since the DC coefficients of subsequent blocks often vary slightly, they are coded as the difference between the quantised DC coefficient of the current block and the quantised DC coefficient of the previous block, which conforms to the simplest form of predictive coding called, Differential Pulse Code Modulation (DPCM). The quantised AC coefficients usually have the property of containing runs of consecutive zeros, which can be exploited by run-length coding to achieve more compression.

The last step in JPEG coding is the entropy coding. The block codes from the DPCM and run-length models can be further compressed using entropy encoding, given a conversion table. Huffman coding is used for this purpose.

Despite its use in large numbers of applications, the JPEG image format has not proved to be useful in certain applications due to some of its drawbacks, such as blocking effects at high compression (or low bit-rates), poor support for error resilience, lack of scalability, no support for region-based or object-based representation of the content,

etc. Taking these issues into consideration has led to the development of the next generation of the image-coding standard called, JPEG2000.

Scalability (which is also explained in section 2.7.4 for MPEG-2) is an important feature that the JPEG2000 standard aims to support. The scalability feature is useful in extracting images of a quality conforming to the application by decoding only a part of the bit stream. This is supported in two scales: quality (SNR) scalability and resolution (spatial) scalability. Resolution scalability allows end-users to extract different sizes of images by enabling a reduction of the time taken for receiving the image. SNR scalability allows extracting images of the same spatial resolution but in different qualities.

2.6.2 ISO/IEC JPEG2000 Coding

A very interesting and promising still image-coding standard, which has recently been published, is described in this section. The superiority of this standard over JPEG is not limited to better performance in terms of subjective image quality and compression efficiency, but it also conveys many other functionalities and features that were only partly covered (or totally impossible) in the existing system. The JPEG2000 standard consists of seven parts (i.e. part I to VII) [Grosbois *et al.* 2001][1]. The image coding system or the baseline algorithm is described in part I of the standard, whereas extensions to the core coding system are described in part II. It can be found that the overall performance of this standard is compared with other recent standards, such as PNG, JPEG-LS and MPEG-4 VTC, in the literature [Christopoulos 2000], [Santa-Cruz *et al.* 2000]. However, such an exhaustive discussion is not given in this thesis.

This coding system is based on the use of wavelet technology rather than on DCT technology. This system is being developed to satisfy current and future multimedia requirements, such as client-server based applications, real time applications, and other applications, which demand high image quality. It is expected that the use of the original JPEG standard will be overtaken by JPEG2000 in the future, in application areas ranging from portable digital video cameras to advanced medical imaging.

[1] It should be noted that some new parts, i.e. part VIII to X, have just been started, and the part VII has been omitted, leaving the standard to contain nine distinct parts. This news is only recently published on the JPEG2000 web site http://www.jpeg.org/JPEG2000.htm.

However, the intention is not to replace the original JPEG standard but to complement it [Christopoulos *et al.* 1999].

The system block diagram of the JPEG2000 remains the same as that illustrated in **Figure 2-1**. There are two main differences between the two systems. First, the image decorrelator is based on forward Discrete Wavelet Transform (DWT) instead forward DCT as used in the original JPEG standard. Second, the entropy coding method in the JPEG2000 system is based on binary arithmetic coding as opposed to Huffman coding in the original JPEG standard. The basic encoding process can be explained as outlined below while the decoding process corresponds to the exact reverse process. As in JPEG, images are first transformed into colour components, and each colour component is decomposed into rectangular blocks of pixels.

The forward DWT algorithm for image decorrelation is applied on basic units of the image components. These basic units are called "tiles" in the JPEG2000 standard. These tiles are different from the 8×8 blocks used in the JPEG standard, and it is allowed to have arbitrary size of tiles, up to and including the entire image [Christopoulos *et al.* 1999]. The wavelet transformation decomposes the tiles into different resolution levels, providing a number of subbands of coefficients. These coefficients are the representation of the horizontal and vertical spatial frequency characteristics of the tile component.

As in the JPEG standard, the second step is the quantisation of the frequency coefficients. In this step, scalar quantisation is used, which finally arranges the quantised coefficients into a rectangular array of "code-blocks".

In the last step, the code-blocks are fed to the arithmetic coding stage to achieve further compression. The coded data is arranged into layers and output as the code-stream in packets.

The JPEG2000 standard provides additional features to support specific applications. Some of the most promising and important features are:

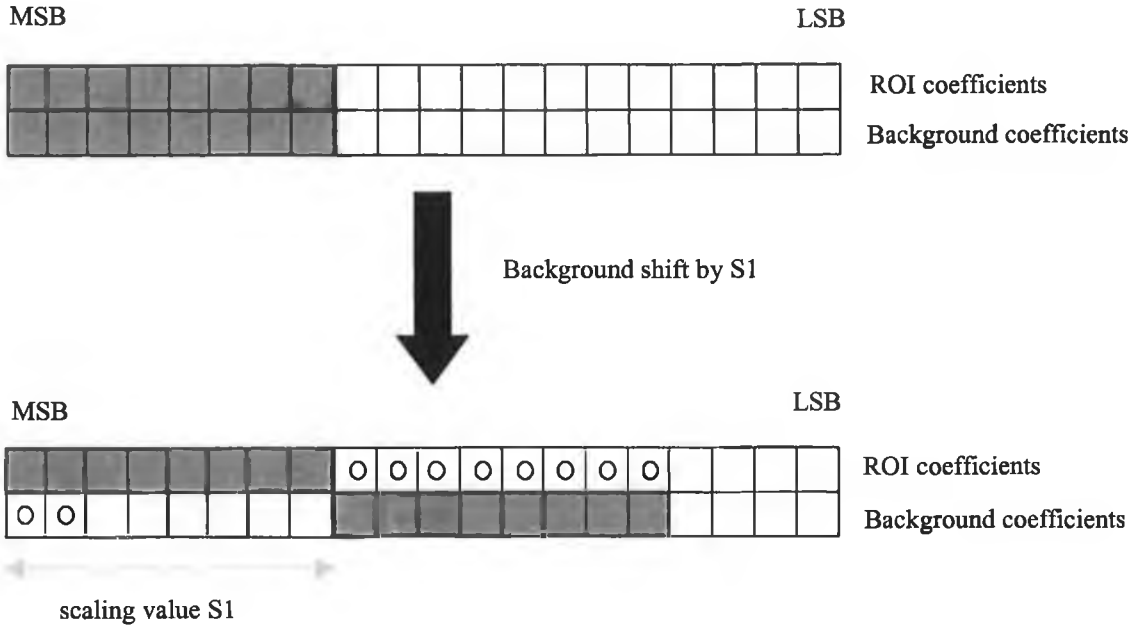
- Ability to provide lossless and lossy compression;

- Ability to specify Regions of Interest (ROI): This allows a user to encode a certain part of the image at better quality or even losslessly, which is very important in high-end applications, such as medial imaging [Christopoulos *et al.* 1999]. In other words, it enables non-uniform distribution of the image quality within the image being coded [Grosbois *et al.* 2001].
- Random code-stream access: The user-defined ROI can be randomly accessed or processed according to user's interest. This is possible due to individual coding of the blocks and packetized structure of the code-stream;
- Error resilience: This is achieved by using "resync" markers to make the system reliable in the presence of high transmission error rates;
- The ability to include image security and content metadata: This corresponds to the protection of the image by encryption and watermarking. Inclusion of content metadata is very useful for today's e-commerce applications.

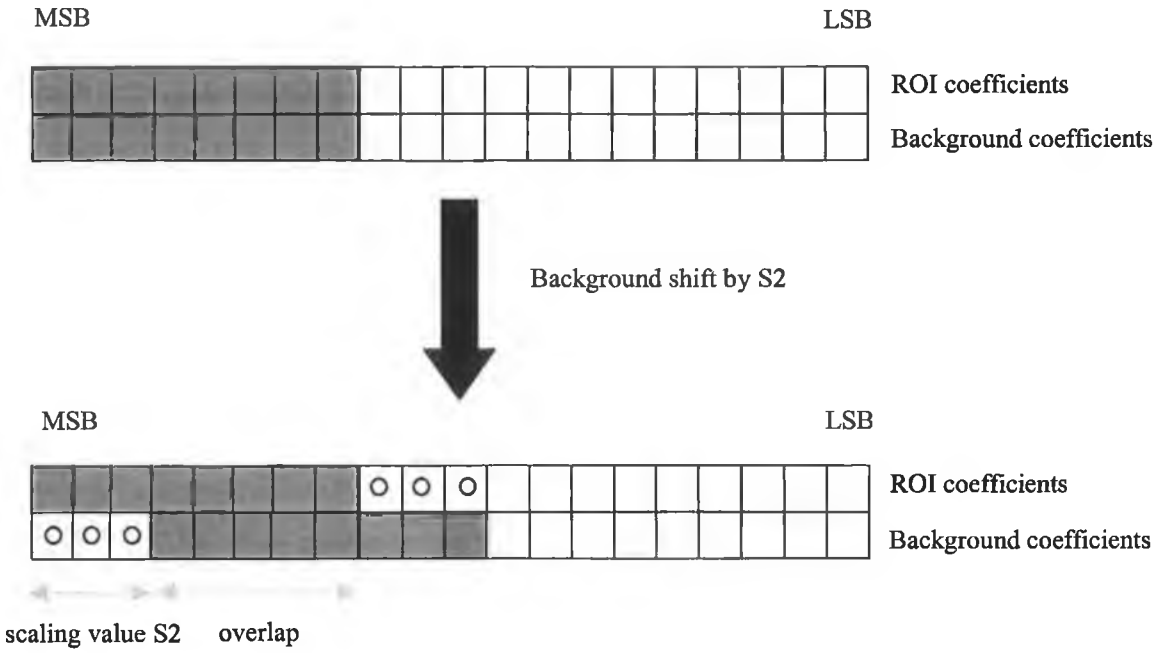
In JPEG2000, ROI coding can be achieved in two ways: ROI Maxshift and ROI scaling. The ROI Maxshift method, defined in the part I of the standard, enables ROI shape coding by downshifting the background coefficients below the ROI coefficients according to a scaling value (S1). This is shown in Figure 2-2a. The scaling value is chosen in such a way that the smallest non-zero ROI coefficient becomes larger than the largest non-zero background coefficient, thereby avoiding an overlap between the ROI coefficients and background coefficients [Boliek *et al.* 2000a][2]. Therefore, no ROI mask is required to separate the ROI coefficients and background coefficients at the decoder end. This is because ROI coefficients can be separated from the background coefficients by comparing them with a threshold, which is derived from the scaling value encoded in the codestream. A very important advantage of this method is that it enables coding of any arbitrary shape region in the image. One disadvantage of this method over the ROI scaling method is that downshifting the background coefficients with the Maxshift scaling value increases the number of bit-planes, and hence more data to encode (i.e. high overhead) [Grosbois *et al.* 2001]. The ROI scaling method, on the other hand, is defined in the part II of the standard, and is known to be a generic method for ROI coding [Grosbois *et al.* 2001]. It is generic because any scaling value can be used to shift the background coefficients, allowing an overlap between the ROI

[2] It is important to note that the selection of the Maxshift scaling value is a normative part of the standard.

coefficients and background coefficients. As depicted in Fig 2-2b, these coefficients are positioned by downshifting the background coefficients towards the least significant bit-planes according to a scaling value (S2) [Grosbois *et al.* 2001]. The disadvantage of this method is that a bit mask needs to be derived to define, in each subband, what the ROI coefficients are, and these bit masks need to be transmitted to explicitly define the shape of ROI. This is because it is not otherwise possible to distinguish ROI coefficients from background coefficients at the decoder. The high cost involved in the bit mask encoding process imposes the constraint that the ROIs be a combination of rectangular and elliptical regions only [Grosbois *et al.* 2001]. Also, this type of ROI coding requires a JPEG2000 part II decoder at the receiver side. A detailed description of these ROI coding methods can be found in [Boliek *et al.* 2000a] and [Boliek *et al.* 2000b].



a. ROI Maxshift operation



b. ROI scaling operation

Figure 2-2 ROI coding

2.7 Moving Picture Compression and Description Standards

The Motion Picture Experts Group (MPEG) was established in 1988, with the mandate to develop standards for coded representation of moving pictures and audio. Within the initial phases of this framework, various algorithms were developed to compress moving pictures for efficient storage and transmission on various digital media.

The MPEG-1 and MPEG-2 audio and video coding standards have attracted much attention over the last several years while an increasing number of hardware and software implementations of these standards are becoming commercially available. MPEG-1 has made a remarkable impact on audio/visual CD-ROM applications with various implementations in both software and hardware. The MPEG-2 standard forms the basic element of existing and future digital TV chip sets, and has been adopted for emerging HDTV applications. Another recent MPEG standard, MPEG-4, is targeted for content-based multimedia applications. The recently born MPEG standard, MPEG-7, is to provide a *Multimedia Content Description Interface* that will further help to develop computer-based multimedia applications and e-commerce applications.

Our discussion is not limited to MPEG. Standards such as H.261 and H.263, which are recommended by ITU-T for low bit-rate videoconferencing applications, are also considered for discussion. Some of the earlier MPEG standards borrowed technologies from JPEG, and more directly from H.261. This means that the MPEG coding systems exploit most of the methodologies of video conferencing standards as well as some other new techniques, most notably motion compensated interpolation.

A video sequence can be considered as a sequence of still images to be coded individually. In practice, temporal redundancy cannot be exploited if image frames in a video sequence are coded individually. In video coding, there are two main modes called, intra-frame and inter-frame. Intra-frame method follows the same techniques as JPEG. More compression is then achieved in the inter-frame mode by exploiting temporal redundancy available between frames.

The use of temporal and spatial redundancies in video compression involves both lossy and lossless transformations. Video coding is generally based on Group of Pictures (GOP) which consist of three types of frames called, I-frame, P-frame and B-frame (see section 2.7.3). I-frames are self-contained and compressed in a similar manner to JPEG. P-frames and B-frames are compressed using motion compensated prediction or interpolation between two reference frames. Therefore, coding a video sequence generally incorporates spatial compression (transform coding, quantisation and entropy coding) performed on the first frame of the GOP, calculation of motion vectors for every 16×16 block (macroblock) of P and B-frames, and compression on the block-difference calculated between the actual and predicted blocks for every P and B-frames (similar to spatial compression mentioned above). On the other hand, the decoding process or reconstruction of a video sequence basically corresponds to decoding the I-frame, motion vectors and block-difference information of each GOP, which were coded at the encoder side. Reconstruction of the P and B-frames at the receiver-end involves motion compensating each macroblock using the decoded motion vectors.

The following discussion describes the use of various redundancies inherent in the natural image sequences, and how they can be utilized to suit different application requirements.

2.7.1 ITU-T H.261

The ITU-T expert group on visual telephony produced the H.261 standard in 1990. It deals with a video codec (encoder and decoder) for audio-visual services at $p \times 64$ kbit/s. It should be noted that H.261 is not the first video conferencing standard for coding digital video. The first video conferencing standard, H.120, has been overshadowed by the H.261 standard due to its better quality and compression efficiency. The H.120 standard was developed by a smaller research group, COST 211 [Whybray *et al.* 1997], at data rates close to 2 Mbit/s. The system was based on Conditional Replenishment Coding (CRC) [Netravali and Haskell 1995]. CRC was found to be appropriate at that time for low bit rate video conferencing and videophone applications, due to the fact that the activity between two successive frames was relatively small. This is due to the stationary cameras used and slowly changing scenes encountered in those applications. However, our discussion is mainly devoted to the more advanced H.261 and H.263 that were the later standards targeted for the above applications.

The H.261 standard is a superior technology to the H.120 standard, and it was a basis for H.263 and some MPEG standards. The data rate within H.261 can be set to vary in integer multiples of 64 kbit/s, according to the well-known expression $p \times 64$. The parameter "p" can range from 1 to 30, providing a minimum of 64 kbit/s and a maximum of 1920 kbit/s data rates. However, the selection of the number (1, 2, 3 and so on) depends on the application rate/distortion requirement. The video formats used in H.261 are CIF and QCIF. A block diagram of the H.261 coding system, extracted from [Whybray *et al.* 1997], is shown in **Figure 2-3**.

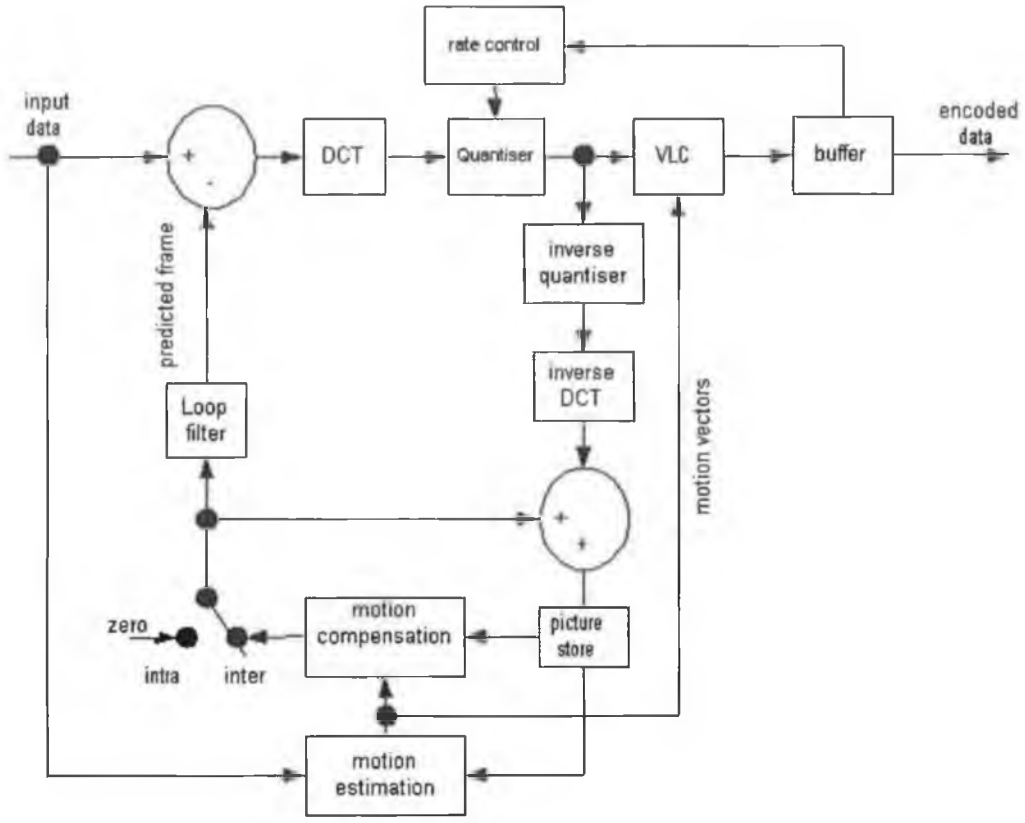


Figure 2-3 H.261 coding system

The three main components in the system are DCT for transform coding, Huffman coding for Variable Length Coding (VLC), and motion estimation/compensation for predictive coding. When compared with the JPEG system, the main difference is only due to the presence of the third component, i.e. predictive coding, which is used to exploit the inter-frame redundancy, while intra-frame redundancy is exploited using the first two components. The input picture data is fed to the subtracter in an 8×8 block of pixels, and then to the DCT, as in JPEG. A coding unit always consists of four 8×8 blocks of luminance, one 8×8 block of U, and one 8×8 block of V, which is referred to as a "macroblock" in video coding.

In H.261, only two types of pictures, namely Intra (I) and Predicted (P), are used. P-pictures are generated using the motion estimation and compensation forward predictive coding method. Prediction of the future frame is facilitated by the decoding module, which is incorporated in the feedback loop of the encoder. This feedback module consists of the inverse DCT (IDCT), and the inverse quantiser (IQ) but not the VLC. This is because only the DCT and quantiser processes are lossy, whereas VLC is

not. The prediction method based on motion estimation is very costly in terms of computations, and is close to 60% of the overall computational load [Vasudev and Konstantinos 1997]. The motion estimation process is, however, a non-normative part of the standard. Addressing more issues on motion estimation research is generally considered to be valuable in order to improve its overall efficiency.

The system functionality can be briefly described as follows. The first picture in the sequence is always coded as intra, and the actual pixels are subject to DCT, quantisation, run length, and VLC coding. In inter-frame mode, the difference between the actual frame and the motion compensated frame is coded using the same techniques. In each picture, the DCT coded and quantised blocks are subject to its reverse transformation, and the previous picture is regenerated by adding the inverse transformed data to the compensated block. These blocks are organised in the picture store (memory) for the subsequent motion estimation and compensation processes to follow. For this, the current picture is fed to the motion estimation block. Motion Vectors (MVs), calculated in the motion estimation process, are fed to the motion compensation block and the VLC block. The motion-compensated macroblocks are then fed to the loop filter for removing unwanted high frequency noise, which, in turn, reduces the visibility of the blocking effects. The MVs fed to the VLC are entropy coded and sent in the coded datastream for the decoder to determine the correct location of the prediction of the current macroblock. A constant bit-rate of the coded bitsrteam can be maintained by monitoring the state of the output buffer and using the rate control feedback loop, which, in turn, controls the quantisation process. This process sets relevant quantisation levels for different macroblocks depending on the amount of data being generated in the coding process. In the encoder, macroblocks can be encoded as INTRA or INTER. The ability to change these modes in between enables to minimise the propagation of coding errors in the encoding chain.

H.261 has several interesting characteristics:

- It defines only the decoder while ensuring the encoder to be compatible with the decoder. For example, past implementations have shown that motion estimation/compensation, the rate control feed back mechanism, the loop filter, pre-processing, and post-processing are open to variations [Vasudev and Konstantinos 1997];

- Because H.261 is designed for real-time communications, it uses only the closest previous frame for prediction to reduce the delay;
- It attempts to balance the hardware complexities of the encoder and the decoder since they are both necessary for real-time videophone applications. Other coding schemes such as Vector Quantisation (VQ) may have a rather simple decoder but a very complex encoder;
- H.261 is a compromise between coding performance, real time requirement, implementation complexity, and system robustness.

2.7.2 ITU-T H.263

The H.263 standard, which has borrowed many technologies from H.261 and MPEG-1 (see section 2.7.3), was standardised in 1995. The major difference between H.261 and H.263 lies in the motion estimation process. Due to many similarities present between the two, the same basic encoder block diagram of H.261, shown in **Figure 2-3**, can be used for our discussion. In this discussion, attention is given to extensions and new features of this standard.

In parallel to the advent of faster modems, the H.263 standard was developed to support low bit-rate communication over Public Switched Telephone Network (PSTN) below 64 kbit/s. Applications are targeted for low delay bi-directional or unidirectional visual communications, such as videophone, videoconferencing, and mobile telephone, at bit-rates up to 64 kbit/s. In accordance with the recommendation, half-pixel precision motion estimation is required for H.263 [Matsuo and Fujimoto 1997] at higher compression. With half-pixel precision MVs, more accurate motion compensation results can be obtained, which results in less prediction errors to be coded, thus making video coding more efficient. Thus, H.263 requires greater processing power than H.261. The following are some of the important deviations [Rao and Hwang 1996] of the H.263 reference model from that of the H.261 reference model:

- Various video formats: It is allowed to use any of the sub-QCIF, QCIF, CIF, 4CIF, 16CIF picture formats;

- Advanced prediction mode: In this mode, half pixel motion estimation, median-based MV prediction, 4 Motion MVs per macroblock, and overlapped block MC are allowed;
- Unrestricted MV mode: With this, motion vectors are allowed to point outside the picture boundaries by extrapolating out the boundary pixels. This feature is useful in case of moving cameras or scenes;
- Syntax-based Arithmetic Coding (SAC) mode: This is a substitute for Huffman coding. This helps for further reduction of bit rates, and hence better coding efficiency;
- PB- frames mode: In this mode, a bi-directional predicted (B) frame is derived from the MVs of the previous P-frame and the next P-frame. The B-frame and the next P-frame are then combined and coded as one unit;
- Weighted quantiser matrix for P-blocks: Unlike H.261, it is possible to change the quantiser on a per macroblock basis but only to the two nearest finer or coarser levels;
- No loop filter: Use of bi-linear interpolations in fractional resolution (half pixel in H.263) motion compensation process satisfies the needs of filtering to reduce the blocking effects;
- 3-D VLC: Unlike H.261 where 2-D VLC (Run, Level) is used, H.263 uses 3-D VLC (Last coefficient-Run-Level) for coding the transform coefficients.

2.7.3 ISO/IEC MPEG-1

MPEG-1 was standardised in 1992, and was the first MPEG standard to code progressively scanned (non-interlaced) video. The main target application for MPEG-1 was audio-visual digital storage media, such as CD-ROM, Video-CD, and CD-I (Compact Disc Interactive), for computer-based multimedia applications at data rates up to 1.5 Mbit/s. It was designed to accommodate several requirements including higher compression, editing, random access, fast forward/reverse searches, etc. As mentioned earlier, MPEG-1 technology was very much based on JPEG and H.261, but it has extra features for supporting the above applications that were not possible with H.261. The basic functional block diagram, shown in **Figure 2-3,** can be considered to

be the same for MPEG video encoding, in spite of the inclusion of more sophisticated processing.

In video coding, MPEG takes advantage of the same two characteristics of video as H.261 and H.263. The high degree of commonality between pictures and the usually predictable nature of movement enable the use of hybrid model of DPCM and transform coding. The basic scheme of MPEG-1 is to predict motion from frame to frame in the temporal direction, and then to use the DCT to organise the redundancy in the spatial directions. The DCT technique is applied on 8×8 blocks while motion prediction in the luminance channel is performed on 16×16 blocks. Similar to other previously discussed standards, the MPEG-1 standard does not specify the encoding process, thereby enabling some flexibility for encoder designers to effectively deal with target applications.

MPEG-1 coding creates four types of pictures I, P, B and D. I-pictures are completely self contained and analogous to ordinary JPEG pictures, but they serve as a reference frame for future P and B frames. I-pictures are the ones that are best suited for random access, despite the disadvantage that they occupy a larger part of the full bit-stream due to less compression. P-pictures and B-pictures, though they contain fewer bits, are functional supersets of I-pictures. P-pictures contain forward prediction, and serve as the reference frame for future P or B frames. B-pictures contain both forward and backward prediction, but they are not used as reference frames for further predictions. DC coded pictures (D-pictures) are similar to I-pictures, but only the DC coefficients from the DCT process are included. The main use of D-pictures is for fast searches, and they are not combined with other pictures [Vasudev and Konstantinos 1997]. Of the four picture types, B-pictures provide the highest degree of compression [Vasudev and Konstantinos 1997], but the expense of more frame memory space, increased picture delay, and more complex processing [Whybray *et al.* 1997] can limit the use of B-pictures for certain applications where the cost and delay are of prime importance. In MPEG coding, three main pictures are organised according to a structure called Group of Pictures (GOP), which serves as a basic access unit. A GOP is generally defined using two parameters called, N and M. N defines the distance between two I-pictures, which are access points or entry points for random access, whereas M defines the distance between consecutive P-pictures. Generally, larger GOPs are more complex,

but because they take better advantage of video characteristics they are more efficient. Shorter GOPs are, on the other hand, much easier to implement, but they consume more space.

MPEG-1 has the following important characteristics:

- It addresses the requirements of a number of computer-based multimedia applications, in particular with random access functionality, which is available at approximately every half a second (due to the nature of the GOP structure within MPEG-1), good quality, and efficient compression;
- It supports picture sizes up to 4096×4096 pixels for many frame rates. However, many of the MPEG-1 applications use SIF picture format. For 16×16 motion compensation, the horizontal size of SIF image (360) must be multiples of 16. For this reason, the horizontal image size has to be either reduced by 8 (4 in each direction) pixels or extended by 8 pixels. When 8 horizontal pixels are discarded, the SIF image size becomes 352×288, and this is known as significant pixel area for SIF [Vasudev and Konstantinos 1997];
- use of D-pictures for fast video searching;
- half pixel resolution motion compensation;
- only Huffman coding is used for VLC.

2.7.4 ISO/IEC MPEG-2

The second phase of MPEG, which was standardised in 1994, addresses high quality, generic (application independent) coding of both interlaced and progressive video including HDTV. MPEG-2 follows a similar method of coding to MPEG-1, but includes extensions to cover a wider range of data rates. The primary application areas targeted by MPEG-2 were mainly digital video broadcast and digital versatile disk (DVD) at data rates of 4-15 Mbit/s, but later developments revealed the possibility of more applications, such as transmission of video over ATM networks and HDTV, at data rates up to 30 Mbit/s [Puri and Chen 2000].

For TV broadcast applications, it was essential to have coding techniques to support interlaced scanning type applications in addition to what MPEG-1 supported. MPEG-2

supports this functionality using additional prediction modes. These prediction modes are categorised into two parts called, frame (interlaced) prediction and field (progressive) prediction. This is because the input picture sequence can be a collection of field pictures or frame pictures. In addition to the two basic modes of field and frame prediction, two more modes are used for improving the prediction efficiency. These are 16×8 motion compensation and dual prime motion compensation [Rao and Hwang 1996]. In 16×8 mode, each16×16 macroblock is considered as a collection of an upper 16×8 segment and a lower 16×8 segment, and these two segments are compensated separately. As a result, it generates two MVs per 16×16 macroblock of P-pictures and four MVs per 16×16 macroblock of B-pictures [Vasudev and Konstantinos 1997]. The dual prime motion estimation can be used for field-based prediction or frame-based prediction. However, it is used only for forward prediction, and has been included in the MPEG-2 standard, in particular, for low delay applications [Vasudev and Konstantinos 1997]. In this mode, a motion vector and a differential offset motion vector are transmitted. In this context, for field pictures, two MVs are calculated from two reference fields. A combination of these two vectors gives the final prediction. Two MVs are derived from this data, and they are used to form two predictions from two reference fields, which are again combined to form the final prediction. In the case of frame pictures, four field predictions are calculated, which are finally combined to form the final two predictions [Vasudev and Konstantinos 1997].

MPEG-2 defines two notions called, Profiles and Levels. A Profile defines functionalities by separating the entire bit-stream syntax into subsets, whereas a Level defines the resolution (spatial/temporal) of the picture. The decoder of a certain profile/level parameter determines its level of capability to decode the bit-stream. An MPEG-2 decoder is backward compatible with a MPEG-1 and H.261 decoder, however, the opposite, i.e. the reverse compatibility, is achieved via the bit-stream scalability feature of MPEG-2 [Vasudev and Konstantinos 1997]. The idea behind scalability is to represent the coded bit-stream in different layers, facilitating decoders of different complexities to be able to decode the bit-stream at various resolutions (spatial and temporal) and bit rates (image qualities). This is an important feature in MPEG-2, in order to become a generic video coding method with wide range of bit rates. Less complex or low-resolution decoders should be able to decode a part of the bit-stream with which the users would be satisfied. For this reason, MPEG-2 decoders

can be made cost effective, in particular, when the intended application does not require the full set of functionalities of the MPEG-2 bit-stream. Another use of scalability is video data transmission in ATM and Internet Protocol (IP) networks, where the Quality of Service (QoS) is guaranteed despite the crucial network bandwidth allocation issues [Whybray *et al.* 1997]. For example, scalability is useful in multicasting environments on IP networks, whereby audio-visual data transmission with the required quality is guaranteed in the presence of various levels of network congestion by dynamically changing the layers [Whybray *et al.* 1997]. Applications requiring timely arrival of data with acceptable quality would also benefit from this feature [Whybray *et al.* 1997]. Bit-stream scalability is defined in four basic modes, namely data partitioning, Signal to Noise Ratio (SNR) scalability, spatial scalability, and temporal scalability [Rao and Hwang 1996].

The following is a summary of the important deviations of MPEG-2 from that of MPEG-1:

- Support for both interlaced and non-interlaced video;
- Support for various sub-sampling schemes such as 4:2:0, 4:2:2 and 4:4:4;
- Provision of a new scalable bit-stream syntax;
- Efficient picture prediction using frame-based and field-based approaches;
- Support for alternate scan method in addition to zigzag;
- D-pictures are not allowed.

2.7.5 ISO/IEC MPEG-4

The previously described standards, such as H.261, H.263, MPEG-1, and MPEG-2, were based on how image data can be efficiently compressed using block-based schemes. However, it was MPEG-4 which initiated the concept of content-based manipulation, allowing traditional video coding system to move from frame-based to object-based coding. Recently, growing computational power to enable more complex data processing, and industry demand have further stimulated the research community to broaden the scope of their initial thoughts. Acknowledging industry demand and the availability of adequate technology, MPEG has developed two new standards: MPEG-4 and MPEG-7. MPEG-4, an object-based audio-visual standard, has led to a common

platform for a diverse set of applications, ranging from computer-based multimedia applications to wireless communication multimedia applications. Some of the main functionalities targeted within MPEG-4 are higher compression efficiency, content-based interactivity, and universal access of multimedia content [Fleury *et al.* 1998]. Content-based interactivity is achieved by separately coding the content in the form of objects. The objects are coded with shape and texture schemes, and the use of efficient coding techniques also facilitates improved compression, which is undoubtedly important for many applications. Use of an object-based composition approach enables support of 2D arbitrary shaped natural video objects as well as synthetic video objects, which is useful for computer graphics applications, such as the generation of virtual scenes. In other words, this enables Virtual Reality (VR), due to the ability of mixing real and synthetic data in the form of 3D graphics and animations.

The method of coding both synthetic and natural video is known as hybrid coding in the terminology of MPEG-4. Nevertheless, the following discussion is based only on the coding of natural video. In MPEG-4, the full range of coding of audio-visual objects at different bit rates is divided into two parts called, Very Low Bit-rate Video (VLBV) and High bit-rate tools. VLBV includes the tools for applications at bit rates 5-64 kbit/s, whereas high bit-rate tools are for applications of 64 kbit/s-10 Mbit/s [Koenen 2001]. However, the purpose of our discussion is to outline general coding aspects, highlighting some extra functionalities the standard supports. The core element of the MPEG-4 coding system is defined as the Video Object (VO), which is any semantically meaningful, arbitrary shape of video content, characterised by intrinsic properties such as shape, texture, and motion. The temporal evolution of a particular VO over a video sequence can be represented by a set of Video Object Planes (VOPs), which can be characterised by texture and shape. Compared to block-based coding, the most crucial and important aspect of MPEG-4 is the shape coding. **Figure 2-4** shows a basic block diagram of the MPEG-4 encoder [Vasudev and Konstantinos 1997] while the decoding process, which is not shown in the picture, is exactly the inverse of the encoder. In fact, the shape coding process can be disabled if the encoder decides not to include shape information at some stages, for example, if a VOP is a rectangular frame constituting the entire image. The coding process, depicted in the figure, shows an example of encoding only one object, though it is possible to encode multiple VOPs in some situations. Moreover, user interaction which could be provided at the encoder side as well as at the decoder side is not shown in the figure.

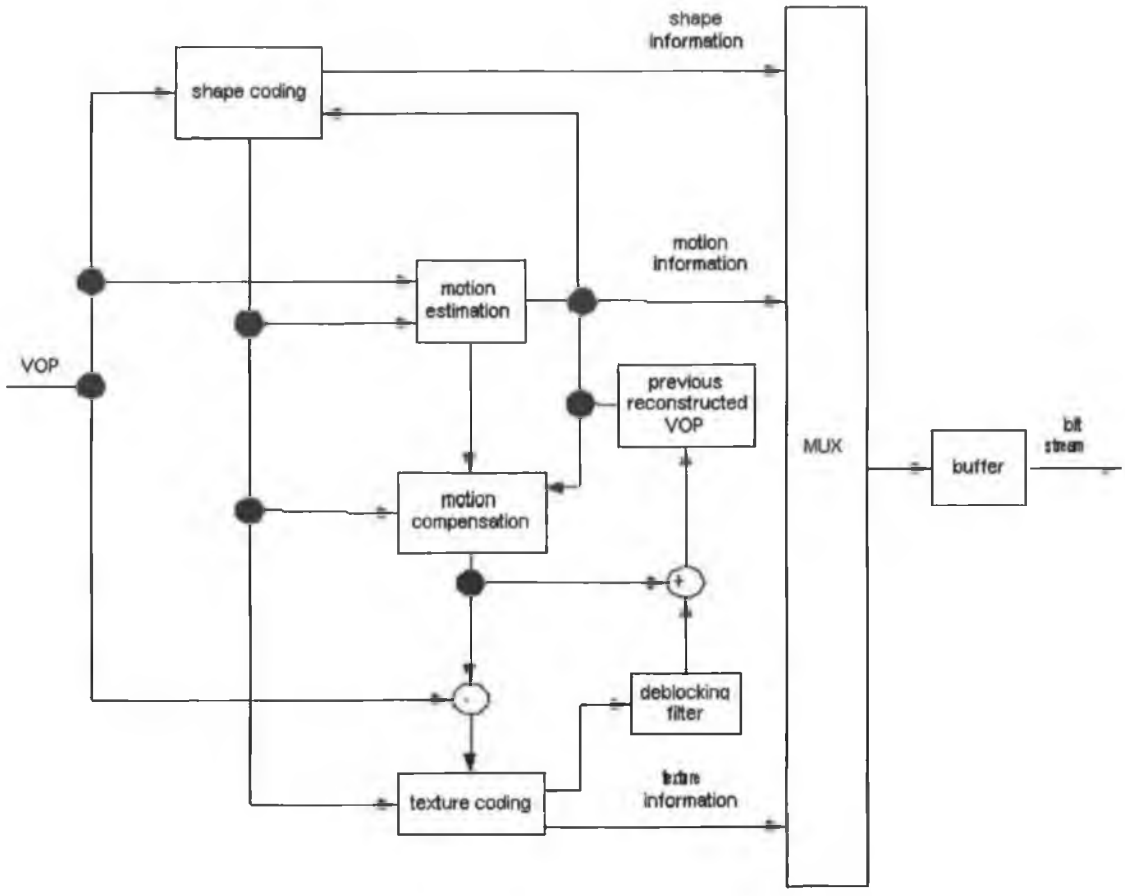


Figure 2-4 MPEG-4 Encoder

The input to the encoder is a VOP as opposed to a square block of data used in the previous standards. The method of generating VOPs, which is usually achieved through segmentation, is not defined in the standard, but is left open to competition to give room for the most competitive or efficient algorithms. Each VOP is processed in 16×16 macroblocks, leaving the entire coding chain less complex, and also providing a certain level of compatibility with previous standards [Fleury *et al.* 1998].

The two main blocks of the encoder can be considered as shape coding and texture coding. The shape of a VOP is represented by what is called an "α-plane", which defines the pixels inside an object (usually represented as grey level 255) and pixels outside an object (usually represented as grey level 0) of the VOP using a simple binary label mask. In addition, a set of transparency values of the pixels of the VOP can be utilised for shape coding purposes. The texture information is generated using the

colour properties (usually Y, U and V) of the VOP. For these schemes, motion estimation and compensation tools help to temporally predict the VOP in the upcoming frames. In this case, the coding of a VOP can be based on any of the following modes [Pascal *et al.* 1998]:

- Intra VOP (I-VOP): VOP is independently encoded;
- Forward Predicted VOP (P-VOP): the current VOP is predicted using a previously decoded VOP;
- Bi-directional Interpolated VOP (B-VOP): the current VOP is predicted using previous and future VOPs.

Initially, the α-plane is divided into Binary Alpha Blocks (BAB) of 16×16 pixels. The encoding process of BABs is based on Context-based Arithmetic Encoding (CAE) [Brady *et al.* 1997] and motion estimation/compensation schemes. Motion estimation/compensation is useful only for P-VOPs and B-VOPs, and the conventional motion estimation process is applied only on the BABs that are completely inside the VOP under study. BABs that are partially covered by the VOP are estimated using the modified block matching technique [Pascal *et al.* 1998]. When the reference BAB lies on the VOP boundary, more efficient estimation is carried out using a repetitive padding techniques [Pascal *et al.* 1998].

For texture coding, VOP texture is divided into 16×16 macroblocks, which generally consist of four 8×8 luminance blocks and two 8×8 chrominance blocks. The texture to be coded depends on the type of the VOP (I, P or B), in the sense that the texture of I-VOPs corresponds to the actual pixel values of the VOP, whereas P-VOP and B-VOP correspond to the residual data after the prediction process. Texture coding is based on 8×8 blocks of pixels undergoing DCT transform, Quantisation, Zigzag scanning, run length coding, and entropy coding. However, this is only the case for macroblocks that lie fully inside the VOP. For other macroblocks, the texture coding process is based on one of the two techniques called, repetitive padding followed by DCT or Shape Adaptive DCT (SA-DCT) [Pascal *et al.* 1998]. The coding of the transparency information of the shape coding process follows a similar process to texture coding. Finally, the information generated from shape coding, texture coding, and motion estimation are multiplexed to combine them into one single bit-stream.

Some of the important functionalities provided in MPEG-4 are: scalable coding (both spatially and temporally), error resilience, data rate control, etc. Furthermore, similar to the MPEG-2 coding standard, MPEG-4 has included the Profiles and Levels scheme. The full standard has been defined in two versions, i.e. version 1 which was standardised in 1999, and version 2 which was standardised in 2000 [Koenen 2001]. The backward compatible Version 2 [Koenen 2001] is a new profile, which adds more tools on the existing version 1.

2.7.6 ISO/IEC MPEG-7

Due to the availability of vast amounts of multimedia information (mainly audio and visual) in compressed digital forms from various different sources over the world, and increasing demands from numerous users who want to use them, the issue of search and retrieval of multimedia content has recently received much attention. This multimedia content is scattered over various places such as digital news archives, digital movie archives, the World Wide Web (WWW), digital libraries, commercial product databases, etc., causing them to be extremely difficult to find. Previous mechanisms of searching any information in the WWW was based only on textual information using text search engines. In this case, the queries are based on keywords or phrases, which are purely text based. Current technologies allow users to search for pictures using low-level characteristics like colour, texture, and shape information in today's multimedia databases [Vasudev and Konstantinos 1997]. MPEG-7, which is also known as the Multimedia Content Description Interface, will extend these limited capabilities towards various content types with additional and meaningful high level features. To search for multimedia data requires that multimedia search engines can store, parse and interpret descriptions of the content. It was clear that these descriptions had to be represented in a standardised form if they were to be accurately and efficiently searched. MPEG-7 defines sets of Descriptors (Ds), Description Schemes (DSs), a Description Definition Language (DLL), and coding schemes for commonly describing multimedia content [Salembier 2001]. A descriptor whose syntax and semantics provide the descriptions of a particular feature of the content is a very important component of MPEG-7. A DS is a way of defining the structure and semantics of its members, which may be both Ds and DSs. The DDL is the language used to allow for creation of new DSs and possibly Ds, thus allowing new extensions and modifications [Koenen 1999]. The DDL in MPEG-7 is based on a textual XML language, which is

suitable for operations like editing, searching, and filtering [Salembier 2001]. The coding schemes mainly correspond to efficient storage and transmission mechanisms.

MPEG-7 became an international standard in September 2001. It is hoped that the application developments based on the standardised tools will target a wide range of potential applications, such as virtual tourism, infotainment, journalism, educational services, investigation services, medical services, geographical information systems, interactive games and so on. **Figure 2-5** shows a basic block diagram of the MPEG-7 system.

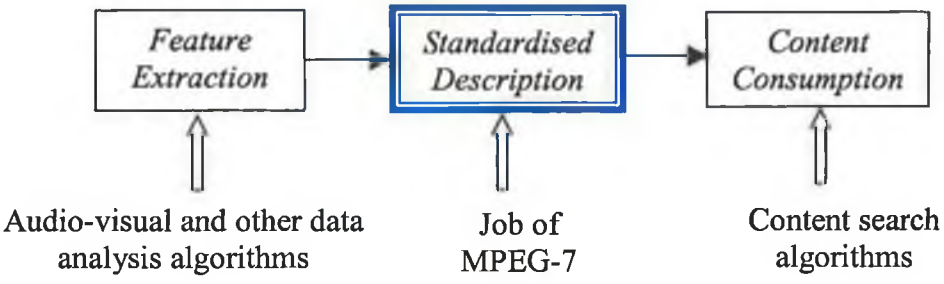


Figure 2-5 MPEG-7 system

MPEG-7 only standardises the description but not the methods to generate and retrieve the description. The description can be from sources such as audio, video, graphics, speech, text, or any combinations of these. The overall system can be represented in three basic blocks: Feature Extraction at the content producers' end, storage of Standardised Description, and Content Consumption at the users' end. Hence, the job of MPEG-7 is to work as an interface for the description of multimedia content between the feature extraction side and the search engine side. However, feature extraction and search tools are outside the scope of the MPEG-7 standard, and are left open to content creators for similar reasons as in other standards.

The feature extraction process corresponds to extracting Ds from the multimedia content that are of interest to the user. Most of the time, a feature has to be defined using more than one descriptor [Puri and Chen 2000]. The low-level features of visual information can be in the form of colour, shape, texture, and motion. MPEG-7 has introduced eight colour descriptors, five shape descriptors, three texture descriptors, and four motion descriptors [Salembier 2001]. In addition, a FaceRecognition

descriptor has also been introduced for the purpose of dealing with facial queries [Salembier 2001]. Thus, in the context of feature extraction from image and video, segmentation plays an important role in describing the content in low-level detail. Furthermore, semantic video object segmentation enables describing the content in high-level detail. Hence, image analysis methods and tools for automatic extraction of descriptors are required in order to be able to efficiently describe the content. For generic applications, user interaction also plays an important role at the higher levels, resulting in the description extraction process becoming a semi-automatic process. For example, a scene or a particular object in a video sequence could be described as an event of a walking and also talking person, using visual and audio characteristics, and user interaction. It is very important, however, to make sure that the amount of user interaction provided at this stage is minimal if the system efficiency is to be well maintained. In some MPEG-7 applications, the extraction of descriptions is not limited to low-level features and high-level features, but includes additional information such as recording date and location in case of a recorded material, storage format, coding methods (in the case of a storage material), name of the director and the title in case of production material, and so on.

The functionality of the search engine is to find the best possible matches of description using the query data. Practically, queries made by users will be at different level of complexities, and the feature extraction system should therefore be able to describe the content at different resolutions.

The data streaming will be taken care of by the Binary format representation for MPEG-7 (BiM). This is due to the fact that textual based XML is verbose, less error resilient in the presence of transmission errors, and difficult to stream [Salembier 2001].

Overall, the degree of success of the standard will largely depend on how intelligent and efficient the extraction algorithms used to describe the multimedia content are. In this context, it is interesting to see some of the future challenges of MPEG-7 in visual analysis, pointed out in [Salembier 2001]. Of those, generation of region of interest, which is more generally a segmentation issue, is a relatively difficult problem, though most of the low-level descriptors can be extracted automatically. Similarly, linking between low-level descriptions and high-level descriptions, which is extremely useful

for matching the queries and retrieval of the correct content, is still an open issue for future research. Furthermore, the selection of best Ds and DSs for a particular application is also an important issue.

2.8 Discussion

The increasing use of multimedia has not only stimulated the MPEG standards to evolve from compression based to description based but also still image compression systems to move from conventional block-based compression techniques to region-based compression, facilitating many new functionalities. JPEG, whose main goal was to provide efficient methods of compression for storage and transmission of images, has passed several phases of development. To that end, it has already developed the JPEG2000 standard with added features and functionalities, providing better performances for current and emerging multimedia applications.

The H.263 standard, which came about after H.261, introduced the ways and means to enable low bit-rate conversational applications over PSTN networks. This was possible mainly due to more efficient compression techniques. Both these standards offer two important features compared to other video compression standards, such as MPEG-1 and MPEG-2. First, they do not cause long delays, as they are mainly intended for bi-directional video communication. Second, they comply with low cost VLSI implementations to some degree, which is very important for commercialisation of videophone and teleconferencing equipment.

MPEG standards, on the other hand, have moved from a frame-based approach to an object-based approach in which the content is described in object form. This is because of the limited access to the content provided by frame-based approaches, whilst the ability to access, manipulate and identify the multimedia content in today's new applications is extremely desirable. Representing the content using object-based coding has enabled the above functionalities, and supports much richer applications in the fields of multimedia broadcasting, personal communication, gaming, database retrieval, advanced surveillance and many more.

All MPEG-standards are generic, in the sense that the encoder operations are not defined within the standard but only the syntax of the bit-stream (the decoding

process), making the standards application-independent. MPEG-1 is the standard supporting storage and transmission of moving pictures and audio at 1.5Mb/s. This has enabled applications in the areas of CD-ROM, DVD, Digital Audio Broadcasting (DAB), etc. MPEG-2, on the other hand, is the standard for digital television including HDTV. The issue of carrying more audio and video channels, particularly over satellites and cable transmission media, where the overall cost factor is largely dependent on the physical transmission bandwidth, has been resolved to a great extent by this standard. However, it is MPEG-4, which has produced a real multimedia representation standard for creation of rich, more interactive content to satisfy current and future demands. With MPEG-7, it was time for MPEG to address the issues of making the information of the multimedia content available for enabling search and retrieval using advanced audio-visual indexing and search tools. These issues were identified as the standardised description of content materials that can be defined using descriptors, description schemes, and a description definition language. With the possibility of content accessibility, tasks such as education services, commercial services, entertainment services, communication services, etc. can be enhanced.

3. IMAGE SEGMENTATION

3.1 Overview

This chapter is devoted to a discussion of image (still picture) segmentation. Image segmentation is a very important pre-processing step in video signal processing applications, and has been therefore subject to extensive research. In general, segmentation may need to be performed either in an automatic manner or a semi-automatic manner depending on the application requirements. It should be noted that our discussion, in this chapter, is mostly concerned with region-based automatic segmentation methods. Attention is also paid to a method of segmentation representation: Binary Partition Tree. The rest of the chapter is then devoted to present other important aspects of image segmentation. Automatic region-based segmentation methods such as Morphological Watershed and Recursive Shortest Spanning Tree are discussed in section 3.3. Binary Partition Tree, which is one of the main components in our system, is described as a method of hierarchical segmentation representation in section 3.4. However, this discussion is not aimed at video segmentation. A detailed discussion about video object-tracking techniques can be found in the next chapter.

3.2 General Description of Segmentation

Segmentation is an essential pre-processing component in current and future content-based multimedia standards, such as MPEG-4 [Koenen 2001] and MPEG-7 [Martinez 2001]. As described in the previous chapter, MPEG-4 encoding is based on video objects/regions for efficient and meaningful image coding, and MPEG-7 is also partly object-based for description representation. Image segmentation is not only applicable to the above cases but also to a wide range of applications in the areas, such as remote sensing (identification of land resources, water resources, terrain, clouds from satellite derived images), video surveillance (traffic monitoring in motorways, identification of enemy targets in military operations), medical imaging (detection and measurement of bones, tissues), industrial inspection (automated quality assurance of products, detection of defects in the products) and so on. **Figure 3-1** shows an example of image segmentation, represented in two forms: regions and contours.

a b c

Figure 3-1 Example of image segmentation (a) original image (b) segmented image represented in mean grey-level regions (c) segmented image represented as contours being superimposed on the original image

In image processing, segmentation is defined as a method to subdivide or partition an image into distinct parts according to user-defined criterion/ criteria. For any segmentation method to be productive, those distinct parts that are generated should be meaningful, and should correspond to distinct objects or features in the image. The purpose of carrying out segmentation can mainly be divided into two parts: firstly, the need to manipulate image pixels for correcting defects or enhancing colour, shape properties of images, which involves purely low-level image analysis; secondly, the need to manipulate image content in the form of semantic objects, which involves high-level processing. However, segmentation is an ill-posed problem, and it is solvable only if the application requirements are properly known. Therefore, semi-automatic segmentation can be vitally useful for general semantic segmentation.

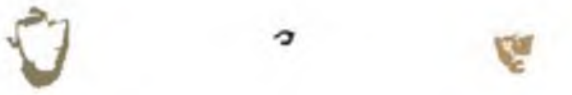
3.2.1 Regions vs Objects

In region-based segmentation, regions and objects are two important and distinct entities. A region is a particular cluster of pixels that an automatic segmentation algorithm can group into one unit, based on pre-defined homogeneity conditions. An object, on the other hand, is a semantically meaningful entity in the real world. In object segmentation, an object can consist of a single region or multiple regions, and the process of grouping regions into an object involves an understanding of the semantics present in the scene [Castagno *et al.* 1998]. These video objects are also referred to as Semantic Video Objects (SVO) in the literature [Gatica-Perez *et al.* 2000], and these two terms have been interchangeably used in the video segmentation field to represent real world entities. Some of the examples of SVOs are "window",

"ball", "face", "jacket", "car", "television", "face", "torso", etc. with which the users may wish to interact. **Figure 3-2** and **Figure 3-3** show examples of objects and regions respectively.

ball child face Original images

Figure 3-2 Examples of semantic objects



region1 region2 region3

Figure 3-3 Examples of automatically extracted regions

As can be seen in **Figure 3-3**, a region is only a set of connected pixels that may not correspond to a real entity. Objects shown in **Figure 3-2**, on the other hand, are comprised of several homogenous regions, and represent semantically meaningful entities. From these examples, it can be seen that a task of defining an object with different levels of detail (say a child with dark hair, brown shirt, black trouser, and white shoes) could be desirable for facilitating object-based querying and retrieval.

3.2.2 Automatic vs Semi-automatic

Automatic algorithms are typically capable of generating different granularities of segmentation, but may fail to produce semantic object segmentations. Generally, the behaviour of each of these algorithms is application-dependent, in the sense that results

can differ due to image complexities, required overall efficiency mainly in terms of speed and accuracy (i.e. required granularity, contour accuracy). This has paved the way for more intelligent image processing, but research so far has proved that human intelligence would be required, in addition to computer intelligence, for these requirements to be achieved. In other words, the use of automatic approaches is limited in some situations, thus leading to semi-automatic approaches.

The two approaches, i.e. automatic and semi-automatic, could be used to support different applications. In automatic approaches, no user interaction is involved, and the segmentation is typically performed in real-time or online. This method of segmentation is useful for real-time applications, such as video surveillance, videoconference, videophone, etc. On the other hand, in semi-automatic approaches, user interaction is provided, and the segmentation is performed off-line (non-realtime). This method of segmentation is useful for more complex applications, such as semantic objects generation, mapping between low-level features and high-level features in video indexing applications, etc.

Although automatic segmentation methods can be fast and efficient to use, the need for efficient object-based segmentation for current and emerging multimedia applications has stimulated research into semi-automatic segmentation methods. For example, the "Chroma-keying" technique, which is used in television studios to create an overlay effect in the pictures, is an alternative way to segment the foreground objects using an artificial background. In this method, the artificial background, which is a normally blue screen, is removed while care is taken not to use blue-colour objects in the foreground. Hence, this process involves colour segmentation to remove blue colour regions from the scene.

In object-based coding and object-based description representation systems, the need to segment scenes into objects is mostly handled using semi-automatic approaches. However, video surveillance types of applications are an exceptional case when dealing with a known background or a known scene. The major reason to use semi-automatic methods is due to the inability of computers to understand the semantics of the real-world entities. Nevertheless, human interaction can be used only if off-line processing can be tolerated by the application. The use of semi-automatic approaches becomes extremely useful, especially when handling segmentation of complex scenes.

3.2.3 How Image Segmentation is achieved

Distinct properties present in images are vitally useful for segmenting still images. Different segmentation algorithms attempt to use these inherent properties in distinct manners in order to produce results to the best possible accuracy. Homogeneity criteria based on image properties such as grey level, colour, texture, or any of these combinations are used for this task [O'Connor and Marlow 1998], [Salembier and Marques 1997], [Salembier and Marques 1999], [Castagno *et al.* 1998]. Although segmentation can be performed on grey scale images more easily, use of other spectral properties, such as colour, can greatly help to enhance the performance of segmentation. Additionally, the use of connectivity or adjacency properties of pixels is a useful attribute. In general, image segmentation can be performed using edge-based techniques, region-based techniques, and model-based techniques.

In edge-based image segmentation, information about edges is used to find distinct parts in the image. Edge detection techniques are used to find edges in still images. There are numerous techniques to perform edge detection, e.g. Canny edge detector [Efford 2000] and SUSAN [Smith and Brady 1997]. Generally, edge detection techniques alone cannot produce accurate segmentation results since they are liable to degrade in the presence of varying image qualities. Therefore, supplementary processing is usually needed in order to relate the extracted edges to region boundaries in the image. Edge linking is used as a method to find continuous image borders. For example, Hough Transform can be used to determine the pixels that lie on a curve of known shape [Gonzalez and Woods R. 1992].

In region-based segmentation, partitioning of an image into regions is carried out by classifying sets of pixels that are coherent. There are two ways of carrying out region-based segmentation. These are known as "Region Growing by Pixel Aggregation" and "Region Splitting and Merging".

In region growing methods, a set of seed points is first defined. The region growing process then starts by merging neighbouring pixels with the relevant seed point, according to a similarity-matching criterion based on grey level, texture, or colour. This process ensures that the regions being generated are disjoint, uniform, and connected. The number of seed points in the set decides the maximum number of regions in the final segmentation. However, this simple approach suffers from two drawbacks, one

being the difficulty of choosing the seed points (how many and which points?), and the other being the properties with which to categorise them into respective regions [Gonzalez and Woods 1992]. For these reasons, it fails to guarantee that every pixel in the image must be in a region, and that the set of regions should be the union of the entire image. Thus, it naturally leads to a need for a more generic region-merging algorithm. One such algorithm is described in section 3.2.4.

In the Region Splitting and Merging method, combining both splitting and merging processes generates homogenous regions. The simple way of doing this is to first subdivide the original image into arbitrary number of regions, and then to decide to further split or merge until the required accuracy is obtained. A more common approach to this is to divide the image first into quadrants, and then to employ a decision-making criterion for further split or merge of any of quadrants until all regions are uniform or the desired number of regions is satisfied. This process, therefore, corresponds to a top-down segmentation process, and may follow a quad-tree structure where the root node of the tree represents the whole image, and each non-leaf node has 4 child nodes. In this method, creation of identical and tiny regions is avoided by provision of the merging step during splitting. It should be noted that this method is likely to produce jagged regions due to the use of block-like regions defined according to a quad-tree.

Image segmentation can also be carried out using model-based techniques. In these approaches, image data is encapsulated in models with the intention of fitting them to the best possible accuracy. The models can be defined using a set of parameters. Some well-known methods are related with Markov Random Field (MRF) models and Bayesian models. Both these methods are successfully used in image segmentation. Image regions and texture can be defined using MRF models, and are usually represented using Gaussian MRF [Manjunath and Chellappa 1991]. Any prior information available is combined in the model's general form. Probability theory is regarded as the way for dealing with these incomplete models. Bayesian analysis can be used to construct hierarchical image models comprising MRF models at lower levels and a process to regularise the segmentation at higher levels. Maximum Likelihood (ML) or the Maximum a Posterior (MAP) criterion is usually used to estimate the model parameters.

3.2.4 Segmentation by Generic Region Merging

In generic merging algorithms, the difficulties encountered in the region-growing techniques (discussed above) are overcome, and the entire system is made efficient by introducing more intelligent criteria. In this manner, the segmentation process is controlled by defining criteria, to stop the region growing process, to decide the order of merging, and to define the model to represent resulting regions. These criteria are known as *merging order*, *merging criterion* and *region model* [Garrido and Salembier 1998], and are defined below:

- merging order: defines the order in which the region-links should be processed to determine the sequence of merging, which is a function of two candidate regions to be merged. Parameters such as region's size, grey level, colour, and shape can be used to calculate the distance between two regions. The two regions of the lowest cost would then become the candidate regions to merge;
- merging criterion: defines a region growing termination criterion by deciding whether two candidate regions should be merged or not. This helps to avoid unnecessary merging of regions. Two possible measures for this are Peak Signal to Noise Ratio (PSNR) and the number of regions the user may wish to achieve;
- region model: defines how to represent a resulting region. When two regions are merged, the size of the resulting region should be the sum of the two regions' sizes (since regions are disjoint). If two colour regions are merged, the mean colour could be considered as the best option to represent the resulting region.

Unlike the region growing method discussed in section 3.2.3, this method considers a set of regions as an initial partition, and employs a merging process to obtain a final segmentation. When the merging process starts from the original image, the individual pixels are assumed to be regions. Regions' adjacency relationships should be maintained during the process of region merging.

3.2.5 Hierarchical Segmentation

The method of "Hierarchical Segmentation" has been investigated in this research. This type of approach produces a set of image segmentations, which successively reduce the complexity of the image or the scene. The overall objective of hierarchically representing segmentation is to achieve a level of detail or a segmentation resolution

which is of interest to the user, thus enabling the system to be effective in many different cases, i.e. general purpose. The hierarchy can be constructed according to a pre-defined criterion/criteria, for example, motion in the case of video sequences, colour or grey-level in the case of still images or any combination of these. In the hierarchy, two neighbouring levels can differ according to a pre-defined criterion/criteria. The upper layers in the hierarchy use more global information, whereas more and more local information is introduced at the lower levels of the hierarchy. Applications such as selective compression for efficient encoding in a rate/distortion manner, transmission of remotely sensed multi-spectral imageries, semantic access and composition, etc. would benefit from this type of segmentation. **Figure 3-4** shows an example of hierarchically represented segmentations.

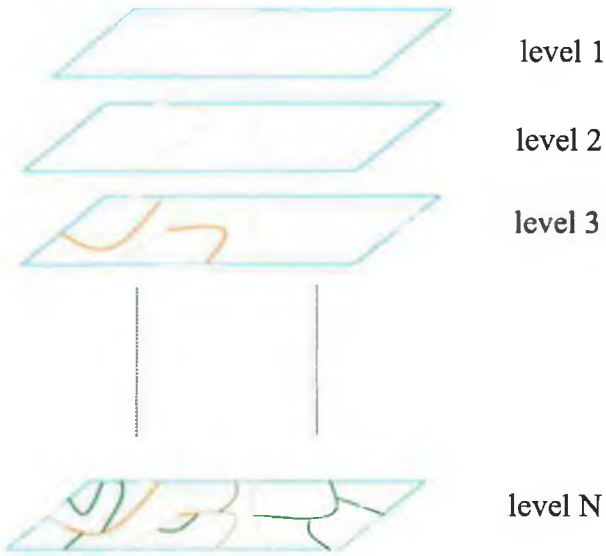


Figure 3-4 Hierarchical segmentation representation

The hierarchy construction process can be either top-down or bottom-up. Moreover, the relationship between two neighbouring levels can differ by any number of regions. In the case of a binary structure, the upper limit of this number is 2.

It is important to note that the selection of an appropriate segmentation from the hierarchical set is a difficult task. Most of the time, selecting a particular segmentation, by combining a number of segmentations, would be more desirable. In this context, user interaction may play a vital role in order to decide which segmentations are to be chosen for the intended application. The following automatic segmentation algorithms, described in section 3.3, are based on region-based hierarchical segmentation.

3.3 Automatic Segmentation Algorithms

Some of the most well-known automatic segmentation algorithms are morphological watershed [Marcotegui *et al.* 1999], recursive shortest spanning tree [Morris *et al.* 1986], pyramidal region growing [Brazakovic and Neskovic 1993], and colour clustering [Palacious and Hedley 1994]. In this section, only watershed and recursive shortest spanning tree are discussed. The binary partition tree technique, which is used as a segmentation representation method in our system, is discussed in the next section.

3.3.1 Watershed Algorithm

The Watershed algorithm, which is a segmentation method by region growing, has evolved from mathematical morphology. A large number of morphological operations have been defined which significantly contribute to the field of image analysis. A detailed description of these morphological operations can be found in [Vincent 1993], [Salembier *et al.* 1996]. In our discussion, only the watershed segmentation technique is addressed. Nevertheless, the morphological filtering techniques which are not discussed in the thesis, can be extremely useful in performing segmentation. These morphological tools have been found to be very useful for bringing the watershed algorithm to its best level of performance [Vincent and Soille 1991].

In watershed segmentation, greyscale image data is considered to be a topographic surface where altitudes of the surface are represented by the pixels' grey level. By flooding this surface from its minima and by preventing water coming from other minima, the image can be partitioned into different sets called, catchment basins and watershed lines. By applying the same process on the gradient image, homogenous grey level regions, which correspond to the catchment basins, can be obtained. However, this method of segmentation produces over-segmentation due to the presence of noise and other local irregularities in the gradient image. This drawback can be overcome by using markers [Vincent and Soille 1991].

Watershed segmentation is comprised of three main stages, namely simplification, marker extraction, and decision [Gatica-Perez *et al.* 2000]. The purpose of the first step is to simplify the image by removing unwanted noise and small details from the image while preserving important image contours. Thus, this process helps to avoid over-segmentation (since the watershed algorithm is extremely sensitive to noise). In most cases, this task is achieved by using a morphological open-close filter by

reconstruction, with a specified size where the size of a structuring element decides the amount of information to be removed. This process produces flat regions, and hence makes the segmentation easier to perform. The marker extraction step, on the other hand, is used to generate the initial regions or to define homogenous areas that contain a set of connected pixels in the simplified image. Markers are generated by identifying the interior of the image regions but not necessarily by identifying the precise region boundaries. The identified markers are assigned unique labels, thereby defining the number of regions in the segmentation. However, the methods of marker extraction are not in the context of our discussion, despite the fact that it is a very important step in the system. The main focused subject in this discussion is the watershed segmentation, which is defined by the third step of the system, i.e. decision. This step allows defining the exact boundaries of the regions identified in the second step by an immersion process, which floods the whole image from the extracted markers. Therefore, this step is also called marker-controlled segmentation.

There are three important parameters to be defined in the watershed method called, minima, catchment basins, and watershed lines. These are graphically shown in **Figure 3-5**. Minima represent the lowest altitudes of the topographic surface where the water can first come through when immersed (assuming there are pierced holes in each minimum). In image terms, a minimum is a connected set of pixels where the grey level of this set is lower (darker) than that of its neighbours. Starting from minima, and gradually immersing the surface into water, the image is partitioned into areas called, catchment basins. In order to prevent water flowing from one catchment basin to the next, dams are built on the surface where the water would merge. These dams are referred to as watersheds. When the immersion process is complete, all the regional minima are surrounded by dams that separate all the catchment basins. These catchment basins are the regions, and dams or watersheds are the region boundaries.

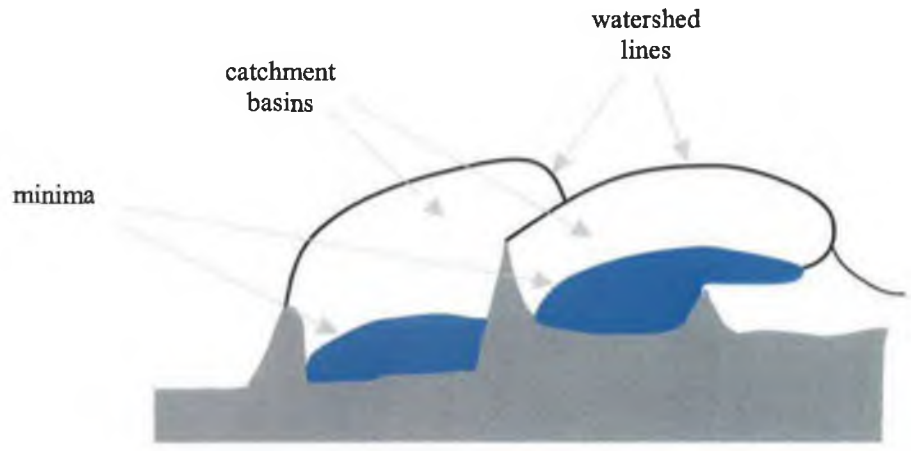


Figure 3-5 Illustration of watershed immersion

In some implementations, attempts have been made to improve algorithmic performance in terms of speed and accuracy. All these attempts have been focused on the immersion techniques and the pixel scanning methods. To this end, sorting of pixels by their grey values in the increasing order and fast breadth-first scanning of the plateaus according to First-In-First-Out (FIFO) data structure, have been used [Vincent and Soille 1991]. Moreover, a hierarchical queue method along with FIFO data structure has been efficiently exploited in [Salembier and Pardas 1994]. This process also relies on a double ordering scheme, in the sense that a point at a particular altitude is flooded only after the points of lower altitude, and points of closer proximity to some flooded points are flooded before the points that are far away.

It should be noted that the watershed algorithm generally produces segmentations with an unpredictable number of regions, which depends on the size of the simplification filter and complexity of the image. This causes either under-segmentation when the homogeneity is defined with a higher tolerance margin, or over-segmentation when the homogeneity is defined with a smaller tolerance margin, causing some difficulties to generate a particular scale of resolution. Although over-segmentation can be useful for semantic object identification, excessive details in the segmentation can make user interactive object segmentation a cumbersome process [Cooray *et al.* 2001].

3.3.2 Recursive Shortest Spanning Tree (RSST)

The RSST algorithm has evolved from graph theory [Morris *et al.* 1986]. Graph theory has been used for image analysis purposes by efficiently exploiting not only the intensity properties but also the spatial properties of pixels/regions. Being a hierarchical method of segmentation, RSST can generate a segmentation of any level of granularity, ranging from the finest (i.e. single pixel level) to the coarsest level (i.e. one single region). It has also been reported that RSST is one of the most efficient automatic segmentation algorithms [Tuncel and Onural 2000], [Morris *et al.* 1986]. In this method, the segmentation granularity is pre-defined by externally specifying the number of regions in the segmentation. In other words, as it was presented in the original algorithm, the merging criterion is set according to number of regions. Thus, the region-growing process terminates at given value, resulting in a partition with a user-specified number of regions. The original algorithm can be found in [Morris *et al.* 1986]. It should be noted that the RSST method can be used for both colour and motion segmentation, as described in [Tuncel and Onural 2000].

Initially, each pixel in a 2D image is considered to be a region, and is mapped onto a node of a graph thereby creating a set of nodes. This set contains a number of regions, which is equal to the number of pixels in the image. Each node represents a region, while a link is initially created using 4-adjacent regions. An example of this type of graph is shown in **Figure 3-6** where pixels or nodes are linked with 4-way connectivity.

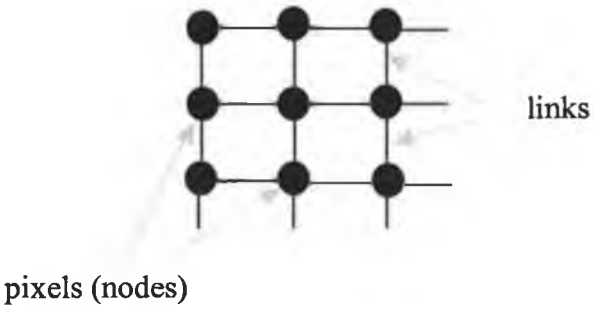


Figure 3-6 Mapped image with 4-way connectivity

For each link, a link-cost or a distance measure between two corresponding regions is calculated using its luminance, chrominance, and spatial size information. Equation 3.1 shown below is used to calculate the above distance measure, and is used to define the merging order.

$$d(R_i, R_j) = \left\{ [Y(R_i) - Y(R_j)]^2 + [U(R_i) - U(R_j)]^2 + [V(R_i) - V(R_j)]^2 \right\} \times \frac{N(R_i) \times N(R_j)}{N(R_i) + N(R_j)} \quad (3.1)$$

where R_i and R_j are two candidate regions, and Y(R), U(R), V(R) represent their mean luminance and chrominance values over all the pixels in the region R. N(R) represents the number of pixels in the region. The above distance measure (equation 3.1) forces regions of similar colour to merge first with a bias factor against merging larger regions. **Figure 3-7** shows a weighted graph where nodes and links have their node weights ($N(R_i)$,$Y(R_i)$, $U(R_i)$ and $V(R_i)$) and link weights ($d(R_i,R_j)$) associated with them.

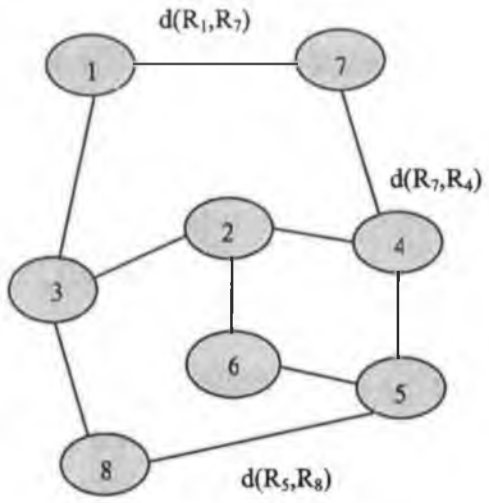


Figure 3-7 Graph of nodes being generated during merging process

The two regions that correspond to the lowest link-cost are merged first, and the link is then removed from the graph. This allows constructing a spanning tree of the initial graph. Repeating the same process reduces the number of regions to the user specified value. Due to this spanning, the affected links are updated with new region nodes, and hence new link-costs with global information. The colour and area information of the newly formed region is calculated, and the new region is represented using mean colour values (Y_{mean}, U_{mean} and V_{mean}) and the sum of the number of pixels in the two regions. The final segmentation partition is obtained by mapping the graph of a required number of nodes to an image with different labels assigned to different regions.

As stated in [Tuncel and Onural 2000], one challenging problem in using this algorithm is the optimum number of regions if it has to be decided automatically. Generally, this

number is image and application dependent. A PSNR value can be introduced as another option to replace this number. It can be calculated between the current segmentation image and the original image. Hence, by keeping track of the variation of segmentation quality during the merging process a desired segmentation can be obtained. However, none of these measures can guarantee a required level of detail in the segmentation in all cases.

This RSST algorithm can be described using the following steps.

```
Initialisation step:

Link all the pixels with 4-way connectivity
No. of regions = No. of pixels in the image
Colour values of regions (Y,U,V)=colour values of pixels (Y,U,V)
Size of region=1


Loop:

while (no. of regions>specified no. of regions){
      calculate link-costs of updated links
      merge the two closest regions
      update the new regions (colour and size)
      update the links
      decrement no. of regions
      }
```

3.4 Segmentation Representation

In any interactive object segmentation system, an efficient segmentation representation is very important in order to allow efficient browsing and manipulation of the segmented content. Since the author's work is based on interactive object segmentation, a method of hierarchically representing segmentation is found to be a useful component to be built into the system. The Binary Partition Tree technique, not specifically presented to be a segmentation representation method, and having capability to deliver many more functionalities, is described in this section as a segmentation representation algorithm.

3.4.1 Binary Partition Tree (BPT)

The BPT technique is one of the best examples for hierarchically representing segmentation, enabling a process ideal for selecting an appropriate segmentation from a set of segmentation results. It has been proposed that a wide range of applications such as filtering, segmentation, information retrieval, and visual browsing can be supported

using the BPT technique [Salembier and Garrido 2000]. With arbitrary shaped regions represented using this binary tree and due to its simplicity, efficient visual-content searching and browsing can be supported for interactive multimedia applications.

The BPT creation, which is based on a region-merging process, starts from a given initial partition. The regions belonging to the initial partition are represented in the leaves of the binary tree. The rest of the remaining nodes of the tree, i.e. non-leaf, correspond to the regions created by the merging process. The merging of two regions at a time is carried out according to a defined merging order (see equation 3.1) by calculating distances between two regions (a link) and finding the minimum among all the links. While region merging is taking place, "Father-Children" relationship between regions is recorded. Merging sequence of all the regions (i.e. which two regions were merged first, second, third, etc.) is also recorded. The root node of the tree is assigned the level no. 1 (see Figure 3-8). Assignment of level numbers to all others regions is carried out according to the sequence they were merged. Starting from the root node and using the father-children relationship, each region is assigned a level number, which is one level higher than its father-node level. A segmentation at a particular resolution can then be represented by associating all the regions corresponding to a particular level (i.e. any level number in the tree) and all other leaf nodes (if any) whose level numbers are lower than the interesting level number. This process, in turn, creates a hierarchical segmentation representation of the original image. For example, the finest segmentation of the image corresponds to the two regions at the lowest level (level no. 9) and all other leaf nodes represented in blue colour (see Figure 3-8).

In BPT, the merging criterion always remains fixed since the merging of regions continues until one single region is obtained, i.e. the root node, leaving the total number of regions in the tree equal to only one less than double the initial number of regions. For example, there are a total number of $(2 \times N - 1)$ regions for a given initial segmentation of N regions. The binary tree with this set of regions represents the image at different scales of resolution, facilitating the user to obtain a particular level of detail of the image. In order to create a BPT, a colour homogeneity criterion, or both colour and motion homogeneity criteria can be used [Salembier and Garrido 2000]. **Figure 3-8**

shows an example of a BPT with 37 regions and 9 levels for an initial partition of 19 regions. The tree describes regions and their spatial relationships within the image.

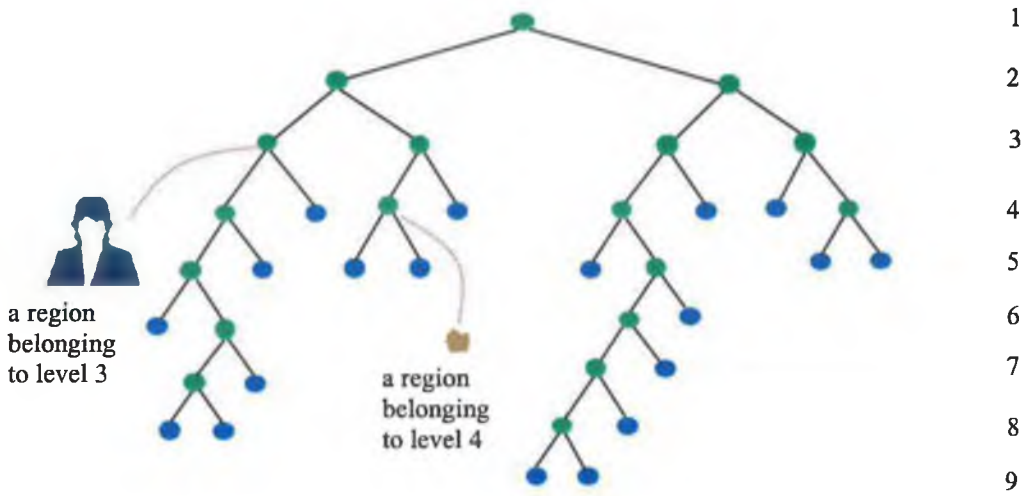


Figure 3-8 A BPT with 9 levels

The blue nodes correspond to the regions in the initial partition, whereas the green nodes represent the merged regions. The numbers shown on the right are different partition levels of the tree, which are used to represent the image at different partition levels. Each node in the tree contains information such as spatial location, spatial area, and colour. Other geometry descriptors, such as shape and rotation, are extremely useful for information retrieval applications, and could be calculated and attached to these tree nodes [Garrido *et al.* 1999]. However, such methods are not used within this work since there is no need to use them in this context of object segmentation.

The full algorithm can be further described in the following way.

```
Initialisation step:
      Link all the regions of the initial partition
      No. of regions = No. of regions in the initial partition
      Colour values of regions    (Y,U,V) = colour values of initial
      regions (Y,U,V)
      Size of region=size of initial region


Loop:
while (no. of regions!=1){
      calculate link-costs of all the links
      merge the two closest regions
      update the new regions (colour and size)
      update the links
      decrement no. of regions
      }
```

Some implementation details of this algorithm can be found in chapter 5.

3.5 Discussion

Image segmentation has been under extensive research over the last couple of decades. Despite the fact that there are a large number of techniques available for this task, segmentation still remains an unsolved problem. In general, image segmentation can be primarily divided into three groups: segmentation by edge detection, segmentation by region detection, and segmentation by model-based techniques. In region-based segmentation, the detection of regions is carried out using two approaches: bottom-up and top-down. Bottom-up segmentation starts from a fine level of segmentation, and gradually builds up larger regions by merging two similar regions at a time. Top-down methods, on the other hand, start from the entire image as one region, and progressively split (or both split and merge) until results of the required granularity are obtained. When the region-based segmentation methods are compared, the bottom-up approaches tend to be more accurate than the top-down approaches since the top-down approaches are likely to produce blocky regions. The main problem of top-down approaches with the quad-tree concept arises due to the difficulties of defining a suitable criterion to split coarsely segmented regions semantically [Park and Ra 1998]. Segmentation by edge detection is not widely used because of its inability to produce accurate segmentation results. A hybrid system using edge detection and region growing proposed in [Fan *et al.* 2001] presents a segmentation technique to combine edge and region features. This type of a new automatic segmentation algorithm is claimed to have produced more accurate segmentation results [Fan *et al.* 2001]. On the other hand, model-based segmentation techniques are generally used to model image texture using probability theory. These techniques are generally known to be computationally intensive.

Among the segmentation algorithms discussed above, the watershed algorithm is the one that has been most used in the literature in a wide range of applications. This is mainly due to the fact that it is one of the most powerful techniques (when used with other morphological operations) to perform well even on highly textured images. For example, morphological operations have been found extremely useful in carrying out image analysis on medical images where image texture analysis is important. However, watershed is a computationally heavy algorithm to implement, and hence requires high

processing power and efficient implementation techniques. Equally, its level of segmentation resolution is very difficult to control since the ultimate number of regions or the segmentation quality is dependent on the simplification filter size.

On the other hand, RSST is a relatively simple algorithm to implement. Therefore, it possesses lower computational cost, as the implementation can be efficiently carried out by modifying only a few links at each step, and by keeping track of the best possible candidate regions to be merged at each step. Furthermore, it is a hierarchical technique, capable of producing segmentations with different scale of resolution in a single run.

The BPT technique is very useful for representing segmentation in a hierarchical form. Simply due to its binary tree structure, high-speed searching and sorting of data, elimination of unwanted parts of data, pruning of tree, etc. can be efficiently facilitated at tree refinement and browsing time. In image search and retrieval applications, describing image content hierarchically, and representing it at different scales of resolution is very important. This is because, the level of resolution at which the images should be described is difficult to predefine since the queries made by the users can range from simple ones to more complicated types. Using images in hierarchical form with the BPT technique allows simplifying complex images, which otherwise cause a large amount of data to be processed.

4. VIDEO OBJECT TRACKING

4.1 Overview

In the previous chapter, image segmentation is discussed. The main objective of this chapter is to give an overview of various existing object-tracking algorithms that are used in video segmentation. As compared to the algorithms explained in the previous chapter, the main difference in these algorithms is the use of motion features since the segmentation is performed on moving pictures. Having identified the widely used algorithms proposed by the research community, this chapter is organised to present the following: firstly, an introduction to video object tracking is given by outlining some of the tracking problems that various algorithms try to resolve; secondly, several tracking algorithms are reviewed. These algorithms are categorised into two types: edge-based and region-based. Although no work has been carried out by the author on edge-based, the intention of this discussion is to give a broad description of well-known tracking algorithms. Finally, a discussion is presented, highlighting the performances of the various approaches. A performance comparison of these algorithms is so far not publicly available to the author's knowledge.

4.2 Introduction

The new concept "Video Object Plane (VOP)", introduced in MPEG-4, aims to represent semantic objects corresponding to meaningful entities in video sequences. Video analysis approaches based on image regions and objects, as opposed to the pixels and blocks used in MPEG-1 and MPEG-2, are demanded by new multimedia standards, such as MPEG-4 and MPEG-7. Thus, video object segmentation has become a very important pre-processing step in video analysis. However, these semantic video objects have proved difficult to extract and to define for generic applications [Marques *et al.* 1996]. Video object tracking refers to the process of tracking a particular video object (initially defined either manually or automatically) over a given video sequence. Various recently published approaches to video object tracking have shown that this task is under extensive research [Gatica-Perez *et al.* 1999], [Gueziec 2002], [Kruse *et al.* 1999], [Loutas *et al.* 2001], [Marques and Llach 1998], [Pateux 2000]. It still remains as one of the most interesting and challenging tasks in video processing.

Video object tracking is used in numerous applications in order to understand where and how any interesting objects appear in the subsequent frames of a video sequence. Some of these applications are video surveillance, videophone applications, news program applications, virtual reality applications, web-based applications (multimedia data search and retrieval applications), telecommunication applications (selective encoding of objects for efficient transmission and storage), etc. Nevertheless, this task is not easy to perform in all cases of real applications where real-time performance and accuracy are important factors. For example, in the case of video surveillance, it may not be too difficult to perform tracking since the scene may not be very complex. The main reason for this is that cameras may not undergo motion in these types of scenarios. Nevertheless, in most other applications, cameras can introduce different effects such as zoom, rotation, etc. making the object-tracking task more difficult to carry out.

Figure 4-1 shows three examples of moving objects to be tracked in two frames of a sequence. A ball with different colours, shown in **Figure 4-1a,** is undergoing a simple translational motion, showing a simple example of object tracking due to uniform shape and motion. **Figure 4-1b** shows a child with changing shapes as the child is bending down in the second frame, which is a more difficult case to handle than a simple translation. A similar situation is shown in **Figure 4-1c,** with a face of changing shape and view, illustrating some practical causes of difficulties in object tracking. However, other causes such as camera motion, occlusion, shading effects, etc. which add even more difficulties into the tracking process are not shown in this example.

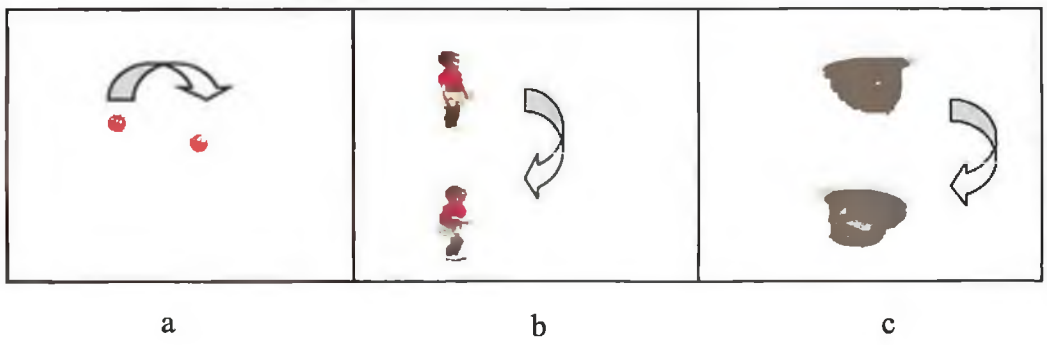


Figure 4-1 Examples of moving objects for tracking

Although motion is a powerful clue to be used in carrying out this type of task, motion itself hasn't been found to be enough to locate the objects of interest and to satisfy the

degree of accuracy required for most practical applications. Thus, spatial properties of images have been found to be useful to be combined with motion features [Zillani and Cavallaro 1999]. It can be found that the watershed algorithm is the most widely used algorithm in recent research for exploiting spatial properties of images [Gatica-Perez *et al.* 1999], [Gu and Lee 1997], [Marcotegui *et al.* 1999], [Marques *et al.* 1996], [Marques and Llach 1998]. Alternatively, other automatic spatial segmentation methods, such as RSST [Tuncel and Onural 2000], [Alatan *et al.* 1997], region split and merge [Vigus *et al.* 2001], K-Means [Kompatsiaris and Strintzis 1999], LabelMinMax [Gu and Lee 1998b], Minimum Description Length (MDL) [Brady and O'Connor 1996], [Pateux 2000], have also been used in the literature in carrying out automatic spatial segmentation.

More specifically, in the context of MPEG-4, video object tracking has come under extensive research in recent years [Castagno *et al.* 1998], [Gatica-Perez, *et al.* 1999], [Gu and Lee 1997], [Marcotegui *et al.* 1999], [Marques *et al.* 1996], [Marques and Llach 1998], [O'Connor and Marlow 1998], [Tuncel and Onural 2000]. Typically, the first stage of any tracking algorithm is object segmentation performed on the first image of a sequence. The object segmentation in the first image can be performed in an automatic or a semi-automatic manner (i.e. areas/regions of interest defined by the user) [Marques *et al.* 1996]. Similarly, user interaction may be necessary during the tracking process since it cannot always guarantee to be a fully automatic process due to various ambiguities present in the scenes [Marcotegui *et al.* 1999].

Object tracking is the problem of detecting the object segmented at the first frame (or on any other reference frame) in the rest of the sequence. Ideally, a tracking algorithm should be able to track any type of moving object within the scene. These objects can be non-rigid or rigid, undergoing any kind of movements in any kind of scenes. Some of the tracking ambiguities such as occlusion, camera motion, shading effect, and deformation are discussed below.

Occlusion: This is one of the most fundamental problems in video analysis, causing complications for motion estimation and object tracking, due to the creation of covered and uncovered areas. This problematic issue in object tracking has been addressed, and some solutions have been presented in [Loutas *et al.* 2001], [Toklu *et al.* 2000]. This

occlusion problem is graphically illustrated in **Figure 4-2**. Some of the best practical examples of this type of problem are in sports, such as basketball and football. The problem arises when the ball is covered and uncovered by the players. A straightforward solution to resolve this problem is the use of multiple cameras. This solution may, however, suit for surveillance type of applications, but may not be adopted for some other applications. Hence, some signal processing intelligence is required for detecting and correcting these occlusions. One of the ways of reducing inaccuracies in this type of problem is by using bi-directional motion estimation/compensation in the motion estimation step.

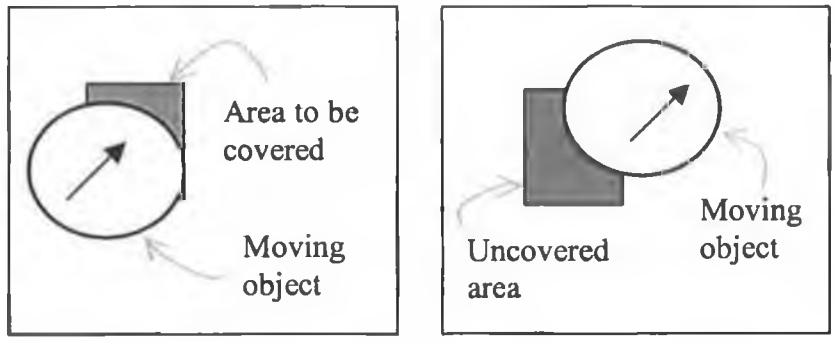


Figure 4-2 Occlusion problem

Camera motion: In most real videos, there can be scenes with moving cameras. These camera movements cause some difficulties in object tracking. In order to overcome this situation, camera motions can be globally estimated and compensated using camera motion detection approaches. This helps to simplify the scene for the tracking process, as claimed in [Moscheni *et al.* 1996]. Furthermore, the affine motion model is known to be a suitable technique to use for handling camera motions, such as translation, rotation, zoom, etc. [Pateux 2000].

Shading effect: This is a type of problem which occurs when the intensity properties of the scene changes in time. However, by using the motion and shape features, this type of problem can be simplified as the object of the same shape is still moving despite its changing illumination. In sports applications such as football, rugby, etc. a ball may be overlapped with the shadow of a player or the ball may be moving in a shadow area caused by the stadium, etc.

Deformation: Generally, video objects can undergo any type of motion with any type of shape changes. These objects with changing shapes are known as non-rigid objects (e.g. humans with moving arms, heads, etc). These shape changes correspond to object deformations, and create difficulties in object tracking. In this context, tracking of non-rigid objects becomes more difficult than rigid objects.

Figure 4-3 shows an example of a generic region-based object-tracking process, illustrating how object segmentation should actually work over a sequence[3]. In this particular example, what the user is initially provided are only the object mask at the 0^{th} frame and all other original images (10^{th}, 20^{th}, 30^{th} and 40^{th}). Therefore, the job of tracking process is to identify binary object masks in the 10^{th}, 20^{th}, 30^{th} and 40^{th} frames for the given object.

[3] Note that these segmented pictures are some of the results obtained using the author's tool.

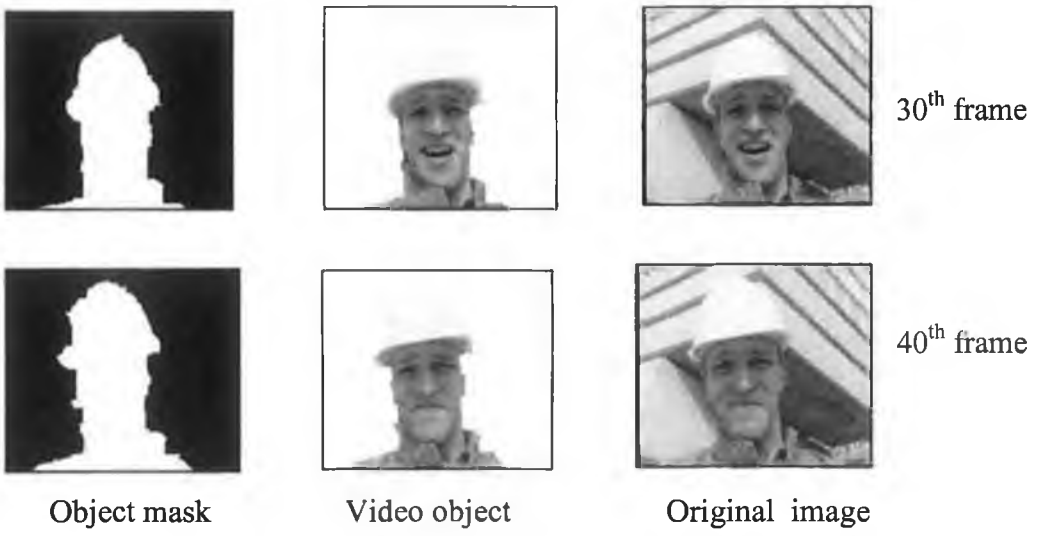


Figure 4-3 Example of video object tracking

4.3 Tracking Algorithms

In this section, different existing tracking algorithms are described. There are two main approaches identified for our discussion. These are edge-based tracking and region-based tracking. It can be seen in the remainder of this chapter that these tracking algorithms exploit various image-features to carry out the object-tracking task. However, it is not possible at the moment to give a firm comparison of their performances, but the emphasis is mainly aimed towards a discussion of their different capabilities in handling various causes of tracking ambiguities. Generally, most of these methods still require some user assistance, particularly in the initialisation phase of the algorithm. Active contours (also known as snakes) can be considered as a particular case of interest to edge-based tracking algorithms. Therefore, a description of active contours is given in section 4.3.1.1.

4.3.1 Edge-based Tracking Algorithms

Edge-based tracking algorithms, also known as contour-based tracking algorithms, consider image discontinuities as the main cue for tracking objects. For this purpose, contour-based tracking algorithms attempt to make use of object contour properties in both the spatial and temporal domain. Thus, this method is primarily based on edge detection and contour-based motion estimation techniques. The edge detection step carries out the necessary tasks to find the actual location of the relevant contours while

the motion estimation model is used to support contour prediction in the next frames. It can be found that a substantial amount of research has been carried out on this topic [Kruse *et al.* 1999], [Zaletelj and Tasic 2001], [Techmer 2001].

In comparison to the other available techniques for object tracking, contour tracking algorithms are known to be efficient in carrying out video object segmentation under various constraints. For example, these methods are capable of handing prominent events such as object occlusions, camera movements, object deformations, in a simpler manner.

Initially, an outline of the object has to be provided (usually in the first frame of the sequence). The boundaries of moving objects can be automatically estimated using the intensity difference between successive frames under restricted environments. Edges can be detected by using Canny operators, as carried out in [Meier and Ngan 1998]. Given this contour, the contour-tracking algorithm then tries to track the contour in the subsequent frames according to the evolution of the curve. To perform this task, a transformation has to be used to match the contour parts according to their distance values. This can be achieved by using a distance transformation such as Hausdorff distance transformation, as used in [Meier and Ngan 1998], which has contributed substantially towards efficient target identifications. This transformation is a measure of two sets of points, which represent a model and an image [Huttenlocher *et al.* 1993].

Figure 4-4 illustrates the required matching to be performed on contours to identify them in the next frames [Techmer 2001]. In this figure, $c(k)$ represents the reference contour and $c'(k)$ represents the current contour to be matched. **Figure 4-4b** illustrates a more difficult and also a more realistic formulation of this problem.

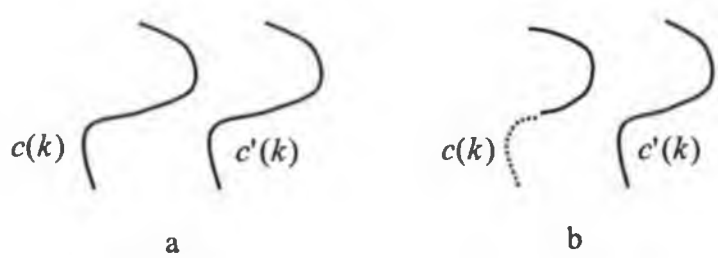


Figure 4-4 Contour matching

In [Techmer 2001] and [Zaletelj and Tasic 2001], the contour matching transformation is based on a simple translation motion model and affine motion model.

In [Kruse *et al.* 1999], the object-tracking problem is considered using two stages: object contour tracking and Kalman filtering. The use of Kalman filtering helps to perform a good prediction. However, our discussion is not aimed at details of these edge detection and motion prediction techniques. The author has not carried out research on these topics during the course of this work on object segmentation.

4.3.1.1 Active Contours (Snakes) Tracking Algorithms

Active contours, also known as snakes, developed by Kass M, *et al.* in 1987, constitute a major research area in video object segmentation [Cohen 1991], [Goldenberg *et al.* 2001], [Kass *et al.* 1987], [Molloy and Whelan 2000]. These snakes are a special case of deformable image models, and are based on minimising energy along a curve according to some criteria. These curves or splines are active shapes, responding and moving according to energy values within an image. Hence, the name "active" has been given because of its dynamic behaviour, always trying to minimise energy along the desired path or the contour. The main objective of this method of segmentation is to detect moving objects with changing shapes, which are represented as changing energies. In practice, objects of similar recognisable shapes are not exactly the same. Some objects, like lips, for example, change over a time period. To detect movements, the splines assigned to these type of objects should be able to deform and stretch to fit local minima accordingly as the shape changes. Since it is only local minima that are used, which is not a global feature, an initial location must be provided for the algorithm. This initialisation step typically produces a set of control points. Given this set of control points, the active contour technique minimizes the snake's energy (i.e. the energy along the boundary) by weighting several internal and external energies. Internal forces guarantee a smooth boundary curve while external forces bring the boundary curve towards the edges of the object. These initialisation shapes can be simple shapes, such as an oval shaped contour (say for approximating a human face object) in the image or any other approximation. **Figure 4-5** shows an example of how the snake model works.

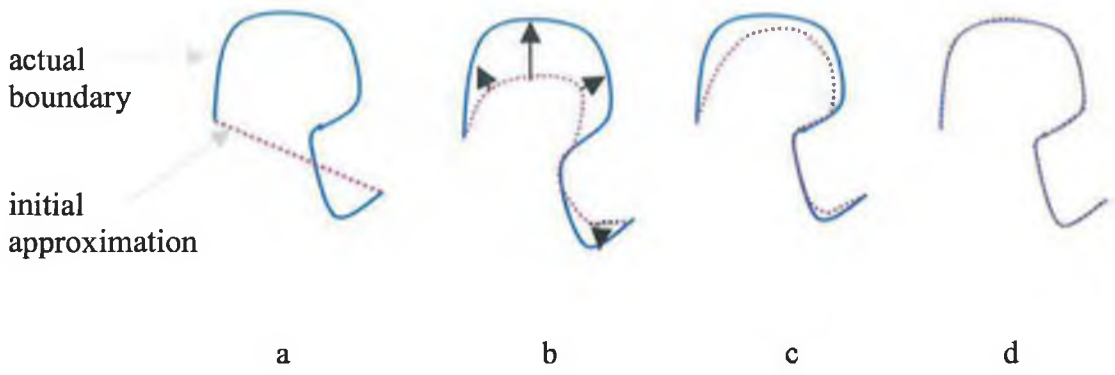


Figure 4-5 Snakes model

The initial approximation to the snake and the true boundary are shown in **Figure 4-5a**. In this example, the snake is approximated by a straight line. However, this is a user interaction step, which only provides an approximate shape and a starting point for the snake to adjust to its true desired boundary. **Figure 4-5b** and **Figure 4-5c** show the iteration process of the snake being pulled towards its true boundary. The energy-minimised snake is then shown in **Figure 4-5d**.

In the active contour method, a snake is defined as an energy function. The energy level of the spline is affected by the gradient value of the image and internal forces, such as the curvature and the continuity of the spline. In other words, the energy value of the snake is dependent on the shape and the location in the image. It is also important to note that the snake is a linear model, which makes it suitable for fast object segmentation [Goldenberg *et al.* 2001]. An in-depth discussion of the snake model can be found in [Blake and Isard 1998], [Kass *et al.* 1987].

An alternative method to snakes, known as fast geodesic active contours, has been introduced in [Goldenberg *et al.* 2001]. This approach provides both a geometric model and energy minimisation functionality, thereby facilitating more advantages over previous snake methods. Nevertheless, it is stated that non-linearity property of this method is a disadvantage over the snake model [Goldenberg *et al.* 2001].

In the active-mesh tracking system presented in [Molloy and Whelan 2000], automatic initialization techniques have been taken into consideration. This method has used

feature points extracted from the image as the vertices to initialise the mesh. It is important to note that the snake algorithm is a computationally simple one compared to some of the other image processing algorithms. Nevertheless, this algorithm is not a perfect algorithm for object tracking due to the presence of various instabilities. However, there are some improved algorithms based on the snake model for more accurately tracking objects in video sequences [Cohen 1991], [Goldenberg *et al.* 2001], [Molloy and Whelan 2000].

4.3.2 Region-based Tracking Algorithms

Region-based tracking algorithms attempt to perform segmentation by combining motion features and spatial features of image sequences. There is quite a number of research publications, which concentrate on region-based tracking algorithms, and some of them can be found in [Marques and Llach 1998], [Marques and Llach 1998], [Gatica-Perez *et al.* 1999], [Gu and Lee 1998a], [Wang 1998]. In this context, the spatial features of images are exploited through region-based spatial colour segmentation, whereas the motion features are exploited through motion estimation techniques. The research carried out by the author on semi-automatic object segmentation is based on this type of region-based tracking method.

In any generic region-based object segmentation, a VO can be comprised of one or more automatically segmented regions. In region-based object tracking methods, object tracking corresponds to identifying the correct set of regions from a given image partition of regions. This process is generally facilitated by defining an object of interest in the first frame and creating a segmentation partition of a set of regions for each subsequent frame of the sequence. In this context, it is clear that the main component of this type of tracking is the spatial segmentation from which the regions are drawn. Thus, the accuracy of the video object segmentation can be highly dependent on the accuracy of the spatial segmentation. In addition, motion information, which is a powerful cue to be used in carrying out object tracking, also contributes to a certain degree of the full process. The motion estimation can either be based on block matching algorithms or affine motion model, the latter capable of producing more robust results. Considering these issues, it is understood that these region-based tracking techniques are equivalent to the methods combined with partition projection

and region classification [Gatica-Perez *et al.* 1999], [Gu and Lee 1998a], [Marques and Llach 1998].

4.3.2.1 Partition Projection Method

In this stage, a spatially segmented image, also known as the image partition, is projected onto the next frame, producing an approximation of the next partition. For this purpose, all the regions extracted from the previous partition (P_{t-1}) are used as markers to relate the previous partition with the data of the current image (I_t), in order to construct the final partition for I_t. Therefore, this process doesn't introduce any new regions in the current partition since the regions are only projected from the previous partition [Marques and Llach 1998].

In order to incorporate the motion involved in this process, the estimation of motion between the two frames I_t and I_{t-1} is necessary, which can be done using a block-matching/polynomial motion model. In the case of block matching, which can be either backward or forward estimation, the motion estimation technique can be constrained by the previous segmentation (P_{t-1}) by considering the best-matched blocks in conjunction with the regions of the partition. If a compensated block belongs to more than one region, the block size can then be reduced by splitting that block into smaller blocks [Marques 1996]. With this motion information, the previous partition and the image can be compensated to find the object of interest.

These steps, however, do not produce accurate results in the presence of fast motion, occlusion etc., mainly due to the simplicity of the motion model. Therefore, the compensated partition may not represent a real partition, containing disconnected regions with the same label. These factors lead to the introduction of post-processing steps in the new partition to correct the boundaries of the objects [Pateux 2000]. Alternatively, these inaccuracies are resolved in [Marques 1996] by using the watershed algorithm, as a second stage, to propagate markers. Thus, the current partition is built by propagating the markers of each region of the compensated partition both in the compensated image and the current image [Marques 1996], [Wang 1998]. However, there are still some problems with this type partition projection method, and these issues are discussed in [Marques 1996].

4.3.2.2 Region Classification

Region classification is an important component in region-based tracking techniques. Prior to region classification, it is necessary to perform the motion estimation between two frames (i.e. current frame and previous frame). Motion estimation is generally performed on regions (i.e. region-based motion estimation) rather than on blocks. Of these two methods, region-based motion estimation produces slightly more accurate results [Marques and Llach 1998]. Spatial segmentation is performed on each frame of the sequence to make the current partition available for the tracking algorithm. For example, it can be seen in [Gu and Lee 1998b] that the system is comprised of region-based pre-processing, region extraction, region-based motion estimation, region-based classification, and region post-processing functional blocks. Given motion information, it is possible to compensate the previous partition or the current partition depending on the manner motion estimation is carried out (i.e. backward or forward), leading the region classification process to be based on backward or forward motion.

In the case of backward projection, based on the compensated partition (more specifically, a set of compensated regions) and the previous video object mask/ masks, a decision is made for every compensated region to estimate which video object it belongs to. Similarly, in the case of the forward projection method, based on the compensated previous video mask(s) and the current partition, a decision is made to decide which compensated mask should contain which regions in the current partition. The decision can be made based on a majority criterion, which however, does not guarantee perfect results in all cases. The remedy for this drawback of the system is to introduce a post-processing step to remove errors caused from the classification process. This step can involve categorisation of certain regions to be uncertainty areas for further consideration, utilisation of morphological filtering techniques for correcting object boundaries either by removing or filling in certain regions, etc. In particular, more accurate and reliable results can be obtained if the decision is made by first categorising the regions into three parts called, background, foreground and uncertainty areas. These uncertainty areas are then processed to correctly identify their relationship with the object of interest. Therefore, the region classification process is a very important stage in this type of tracking method. One of the drawbacks of our implemented system is that the uncertainty areas are not defined; instead each region is considered either to be a background or foreground.

4.4 Discussion

A general discussion of two tracking methods described is presented in this chapter. Both methods are powerful competitors, each having some distinct features and drawbacks. However, it is outside the scope of this thesis to produce an exhaustive performance comparison.

In order to implement a reliable tracking algorithm, it can be argued that all of the available information should be used. In most of the algorithms discussed above, when motion is calculated between frames, only two consecutive frames are considered. This causes some inaccuracies when the correlation of the object in the two frames is low due to scene ambiguities, such as those mentioned in section 4.2. This suggests that more information from other frames of the sequence should be exploited, when available.

When implementing video object tracking algorithms for real-time applications, the efficiency of the algorithm, in terms of its speed, has become the main concern since the algorithms are generally very computationally intensive. However, it should not degrade the accuracy of the segmentation in the event of faster implementations since both these requirements need to be satisfied in real applications.

In [Zaletelj and Tasic 2001], two approaches, region-based and contour-based are briefly compared. It is mentioned that region-based methods are not accurate in extracting object edges, though it uses full region information to estimate global object motion. On the other hand, contour-based techniques can handle only slow object motion [Zaletelj and Tasic 2001].

In [Vigus *et al.* 2001], it is argued that the edge-based methods are more suitable for images containing man-made objects since these types of images more clearly define the object boundaries as compared to the natural scenes. In this case, region-based methods are more suitable for images from natural scenes where object boundaries are difficult to extract using edge detection.

The efficiency of a generic tracking algorithm, in terms of its speed, is discussed in the region-based object tracking method presented in [Vigus *et al.* 2001]. It is stated that usage of a predictor operator can facilitate fast object tracking since the predictor step

can be incorporated with a spiral search to quickly obtain results. In this paper, the Kalman filter, which is a recursive predictor operator, has been used as the predictor.

In [Techmer 2001], a fast object tracking technique is proposed based on real-time contour-based motion estimation. This technique claims to have produced robust results due to use of contours for motion estimation and object tracking.

In the edge-based tracking methods using active contours, the internal and external forces applied to the deformable image model allow object tracking. The main drawback of this method is the fact that the tracking can only be performed if an initial contour is provided. However, the strengths of this method over the other tracking methods, in particular, the speed of the tracking process to detect fast moving objects is not clear at the moment. Nevertheless, this method is capable of handling both global and local scene deformations.

In [Molloy and Whelan 2000], the active mesh technique is used for tracking applications in unconstrained motion environments, camera motions, multiple objects, and occlusion effects. These actual tracking difficulties are overcome in this approach by adaptive active-mesh and multiple meshes algorithms. Introducing drop and add features for updating the mesh over time is carried out in adaptive active-mesh method, whereas subdividing the entire mesh into subunits and operating them separately with no transfer of information between them is carried out in the multiple meshes algorithm.

Summarising the above, region-based object tracking algorithms can be very computationally intensive, particularly when the segmentation has to be performed under all cases of prominent tracking difficulties. On the other hand, edge-based tracking algorithms may not be the best for tracking fast moving objects and images with less texture properties. An advantage of edge-based tracking is that it is possible to perform object tracking without a prediction step, which is significantly advantageous in moving cameras scenes [Techmer 2001].

5. A SEMI-AUTOMATIC VIDEO OBJECT SEGMENTATION TOOL

5.1 Overview

This chapter sets out the system details of the proposed semi-automatic video object segmentation tool. Various functional blocks of the system are taken into account for this discussion. The discussion starts with the system architecture in section 5.2, outlining its functionalities using a system block diagram. Subsequent discussions give a description on how the problem was structured towards the development of a semi-automatic tool for interactive video object segmentation. Thus, a description of the GUI developed is presented in section 5.3. The initial object segmentation process (which involves RSST, BPT and user interaction) is described in section 5.4. The implemented object tracking technique is then described in section 5.5. An in-depth discussion of the user interactivity features supported within the system is given in section 5.6. Furthermore, software implementation details, describing the functionalities of various classes, are given in section 5.7. Potential improvements that are worthwhile proposing to modify the tool in adapting to different applications or new user- requirements are given in section 5.8. The entire implementation was carried out in the Java programming language under the Microsoft Windows platform using Java Builder as the integrated Development Environment (IDE). As such, it required an implementation of various classes and functions from scratch since there was no pre-existing Java software available for this type of work.

5.2 System Architecture

A block diagram of the system is shown in **Figure 5-1**. The implemented system is comprised of four main blocks called, RSST, BPT, User Interaction, and Object Tracking, which collectively constitute two main functionalities: initial object segmentation and object tracking. The descriptions of initial object segmentation and object tracking techniques are given in sections 5.3 and 5.4 respectively. However, the object-tracking stage needs to be used only when objects need to be extracted from image sequences. In other cases, where only still image segmentation is needed, object segmentation can be achieved using only RSST, BPT, and user interaction. With this flexibility, this system can be considered as a generic tool, suitable for both still image segmentation and video object tracking. For the sake of clarity of results, the output

results, i.e. VOs in **Figure 5-1**, are displayed in the form of original grey-level objects, despite the fact that provision of only an alpha mask would suffice for a real object-tracking application.

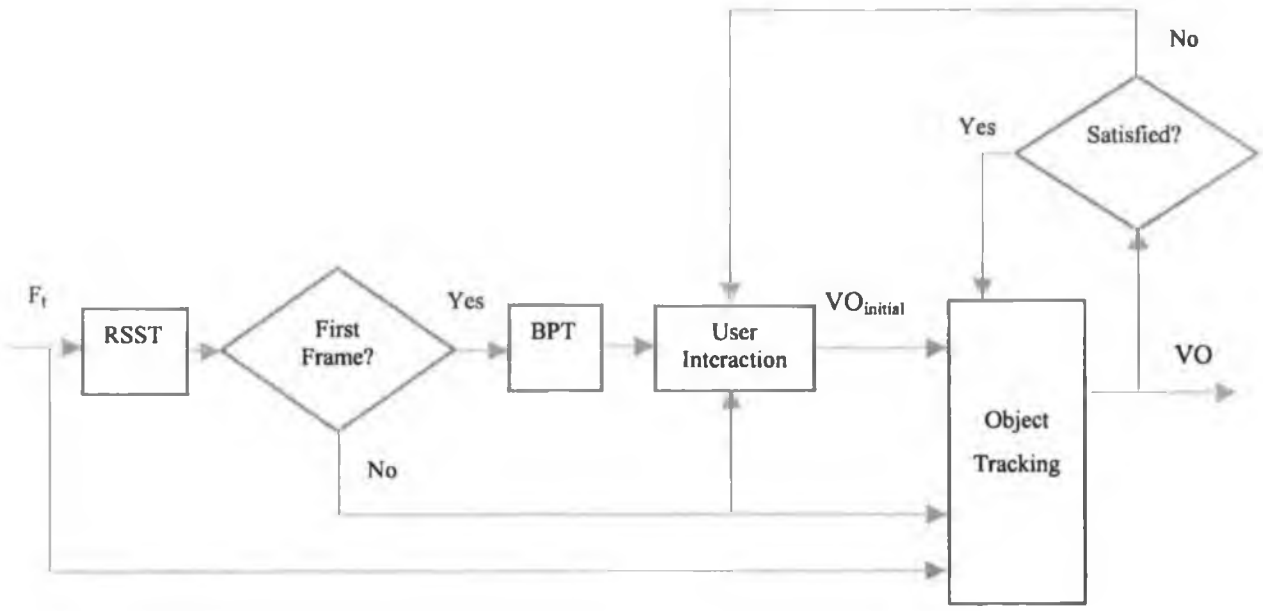


Figure 5-1 System Block Diagram

The overall functionality of the system can be described in the following manner. The first frame of the sequence (or the first image to be spatially segmented), i.e. F_t is fed to the RSST algorithm. This step creates an initial segmentation of a user-specified number of regions, and this parametric value is provided through the Graphical User Interface (GUI) shown in **Figure 5-2**. The BPT is then constructed starting from this initial partition of homogenous regions. At the first frame, a hierarchical representation of the initial segmentation in the form of a binary tree is provided to the user for interactively generating user-specified object/objects. This means that RSST and BPT are followed with user interaction to extract the initial object ($VO_{initial}$) from the first frame of the sequence. More importantly, as can be seen in the above diagram, the BPT technique is utilised only at the first frame of the sequence. This means that the hierarchical segmentation representation is not employed again after the segmentation of the first frame. For a defined object, the automatic tracking process is launched simply by clicking on a button in the GUI. This step feeds subsequent frames of the sequence to the RSST block and the object-tracking block. The object-tracking

component, which is described in section 5.3, mainly consists of spatial segmentation, motion estimation, object projection, and region classification techniques. However, if discrepancies in results occur, there is a need to allow further user guidance, in order to accurately continue object tracking. This step, which needs to be activated only if a major segmentation distortion occurs, is called object re-initialisation in our system. Thus, the status shown in the above diagram "*Satisfied?*" needs to be checked while the tracking process is progressing. An answer "No" to the question stops the tracking process and allows the user to refine the object. This object refinement step facilitates avoiding propagation of errors throughout the sequence. At this stage, the user performs add/remove of small regions to/from the current object-mask, given a segmentation of fine regions. It should be noted that the amount of user interaction involved in the re-initialisation stage is much less than that involved at the first frame. The current object mask is typically corrected with only a few mouse-clicks. The tracking process continues as long as the output results are within an acceptable quality throughout the required length of the sequence the user is interested in. Further details of the individual blocks of this system are presented in the following sections.

5.3 The Graphical User Interface

The interactive video object segmentation tool is enriched by the GUI developed to support the required user interaction. User interaction is provided in conjunction with simple mouse clicks and drags. The key feature of this tool is the distinction between the two notions of segmentation, namely region-based and object-based segmentation. Regions are extracted using the automatic RSST algorithm where no user interaction is involved. On the other hand, semantic objects are extracted through user interaction by manually grouping the relevant regions generated by RSST. This tool allows the user to interact during both the initial segmentation and object tracking processes.

The GUI allows the user to open up image data files from the file menu and save the output results to disk. The user opens up the luminance file of the sequence to be segmented from the file menu, which, in turn, signals the system to open up its respective chrominance files. At present, only PGM file format is supported, though it could easily be extended to other file formats. This process then automatically runs the RSST and BPT algorithms, thus creating the initial segmentation and its hierarchical

representation. When creating the BPT, parameters such as spatial size, and mean colour, are calculated and attached to the respective nodes of the tree.

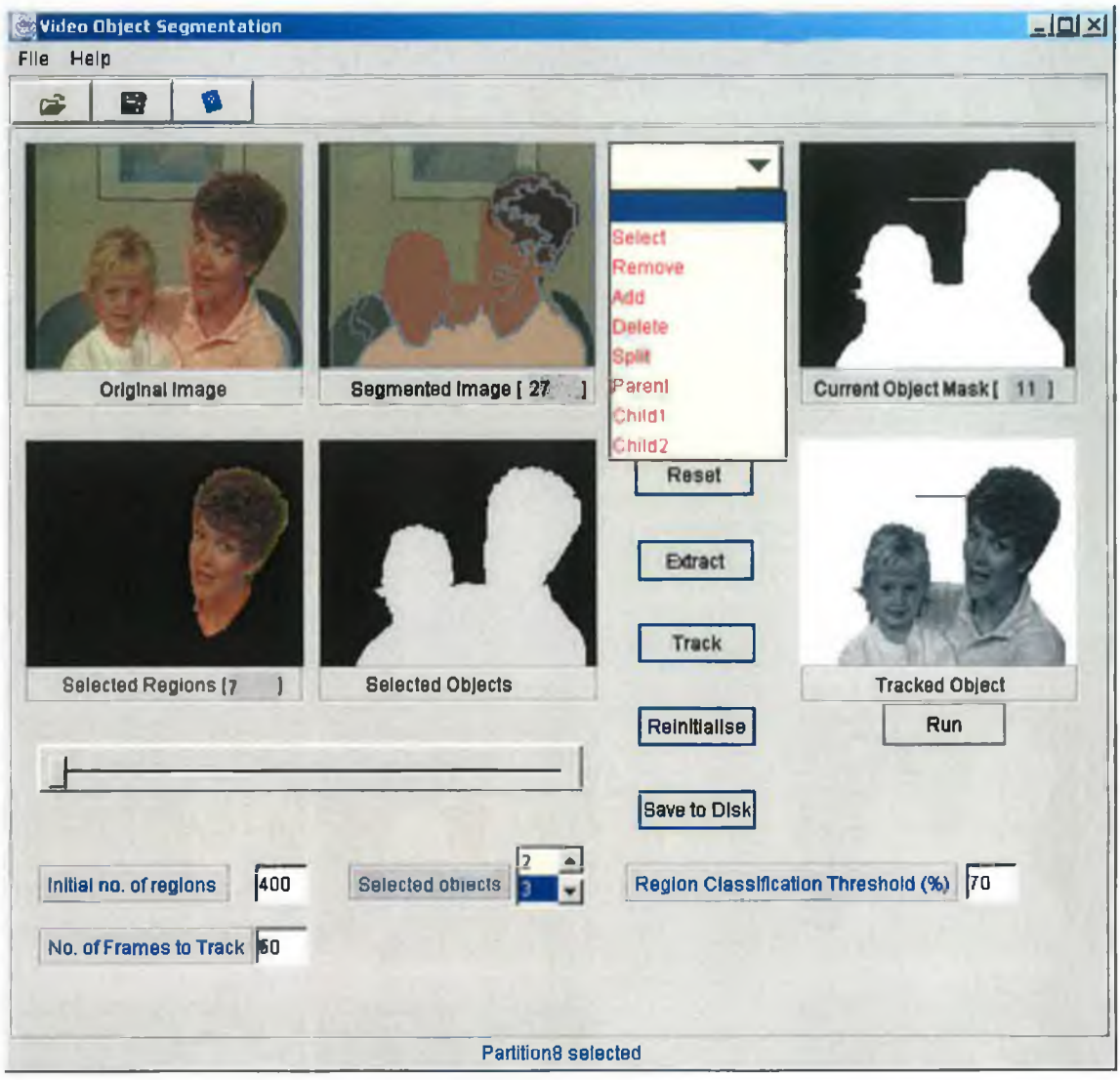


Figure 5-2 A screen-shot of the GUI

The input parameters such as initial number of regions, number of frames in the sequence to track, and threshold parameter values for region classification can be provided through the GUI. Other important region manipulation functionalities such as selecting regions, removing regions, showing child/parent nodes, splitting regions, forcing some regions not to subdivide further during the browsing process, object re-initialisation during tracking, etc. are provided in the form of simple left and right mouse-clicks. Browsing of the binary tree is provided using a slider-bar, which the user

can drag to move around different segmentation partition levels. The implemented GUI system is shown in **Figure 5-2**, with an image of the mother and daughter sequence being displayed.

There are six windows provided in the GUI in addition to other graphical components, such as text-fields, combo-boxes, slider-bars, and buttons. These windows are titled: original image, segmented image, current object mask, selected regions, selected objects, and tracked object (from left to right and top to bottom). The use of these windows is to display the segmentation results while allowing the user to interact with low-level image regions to effectively complete the segmentation process. The top-left window shows the original image on which the segmentation algorithms are applied. This gives the user an idea of what the original image is while allowing the user to visually evaluate its segmentation results.

The second window (the top middle), i.e. the segmented image, shows segmentation results with regions represented in their mean colours. Immediately after running the automatic segmentation algorithms, this window shows the root level of the hierarchical segmentation. Thus, it shows only a rectangular image filled in with mean colour. During the tree browsing process, segmentations with different resolutions are constantly displayed according to the level of the tree the user is browsing at. During the object formation process, the user can click on image regions on this window to select what operation is to be applied to perform interactive object-segmentation, i.e. select, remove, add, delete, show parent, show child1, show child 2 or split. In addition, during the browsing process, a simple mouse-click on any of the regions also allows the user to force a region not to be further subdivided.

The purpose of having the selected-regions window (bottom-left) is to show which regions are being selected by the user during the process of manually grouping homogeneous regions. In case of unnecessary regions being selected, the user is allowed to remove them and select new ones. The regions being selected during this process are shown in their original colours on this window, which more clearly gives an idea of the object being formed.

The bottom-middle window, i.e. selected objects, shows an object mask to indicate which object has been selected by the user. These object masks are needed for saving segmentation results to disk, as an input to the tracking algorithm, etc.

The other two windows, i.e. current object mask and tracked object, are useful during the temporal segmentation. When the tracking process is progressing, these two windows are constantly updated with the current object mask and the tracked object. The current object mask window is used during the re-initialisation phase to allow the user to refine the current object either by adding or removing tiny regions. The sixth window, i.e. tracked object, constantly shows the tracking results; hence the user gets visual feedback of the tracking process as it is progressing. At present, only a grey level image display is supported on this window; however, it is only meant to give an indication of the nature of tracking results. Once the tracking process is over, functionality is provided to animate the tracked segmentation results (using RUN), if necessary. The functionalities of these graphical components are described in section 5.6.

5.4 Initial Object Segmentation using RSST, BPT and User Interaction

In any user assisted object segmentation approach, the most crucial and important step is to generate the required initial object that needs to be tracked in subsequent frames. The first stage of our system facilitates the initialisation of the desired object by providing hierarchically represented segmentation results and user interaction. Two conventional techniques, RSST and BPT (descriptions of these techniques can be found in chapter 3) are employed in this stage. The purpose of using RSST is to automatically generate an initial segmentation of the required granularity. However, other automatic segmentation algorithms, such as morphological watershed as used in [Marcotegui *et al.* 1999], Pyramidal Region growing [Brazakovic and Neskovic 1993], Colour Clustering [Palacious and Hedley 1994], could be used in place of RSST. In our approach, the RSST was chosen mainly due to its simplicity and efficiency [Mulroy 1997]. The BPT, created by a process of iterative binary merging of initial regions, provides the hierarchical region-browsing feature to the user, thereby allowing the object segmentation to happen in a user-friendly manner. To create the binary tree, the RSST output, which is a set of homogenous colour regions, is fed as the input to the

BPT. Both these segmentation techniques are based on a region-merging process. In the case of RSST, the user decides the merging criterion, which is the number of regions of the initial segmentation. However, in the case of BPT, the merging criterion is always fixed since the merging process continues until it reaches one single region. Moreover, a uniform region model is used both in RSST and BPT. Thus, the initial object extraction task is performed using the hierarchically represented homogeneous image regions and user interaction. This user interaction involves manually selecting any interesting image regions and grouping them into objects. A block diagram of the initial object segmentation process is shown in **Figure 5-3**. Various user-interactive features provided in our GUI system are described in section 5.6.

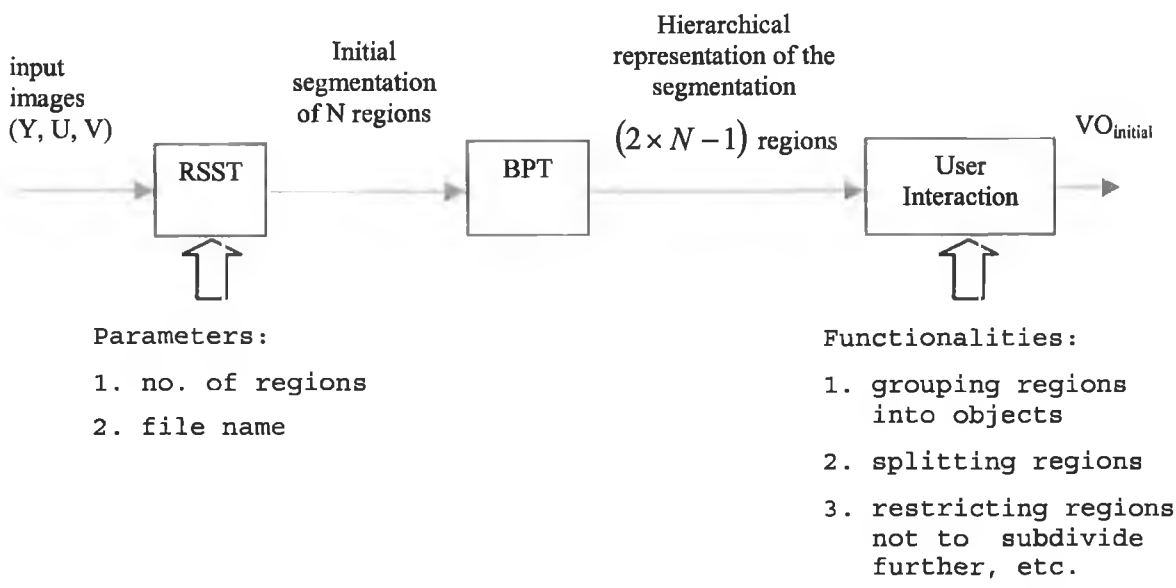


Figure 5-3 Initial Object Segmentation

In the user interaction stage, the efficient tree browsing functionalities are extremely useful in order to effectively select regions from different levels of the hierarchy. The following pseudo code illustrates the established method for browsing, adopted in the BPT technique.

```
current_level=current slider position;

for every region "R" in the tree {
if ( (R->level=current_level) OR (R->level<current_level AND
R->child[0]=null))
show its details;
}
```

where *child[0]* is one of the two child nodes of *R*. Note that *R->child[0]* and *R->child[1]* should be null for all leaf nodes *R*.

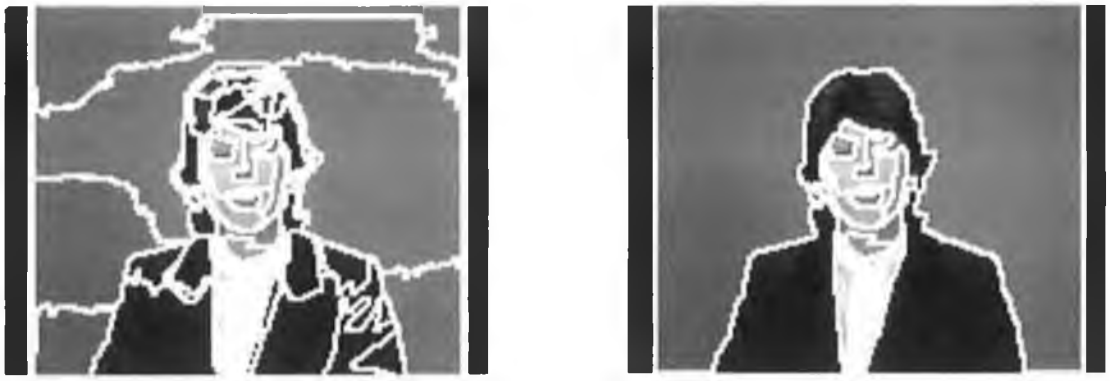
The current position of the slider-bar (tree navigator) is taken as the level of the binary tree. The slider position 1 corresponds to the root node, which represents the entire image with mean colour information. When the slider is moved, details of the corresponding level are displayed by searching the full set of regions in the tree, which has different levels ranging from *level=1* (root node) to *level=n* (highest level). This is accomplished by taking regions' level information as well as their child-parent relationship into account. For example, a request for displaying the regions corresponding to the 5^{th} level in the tree causes it to display all the regions belonging to level 5 and all the other leaf nodes, whose level numbers are lower than the current level (4, 3 and so on).

In order to speed up the tree browsing, a browsing simplification step is introduced in our system. This performs a task of hiding the selected region's child-nodes. When a region is marked not to be further subdivided, its all child-nodes are made inactive (i.e. *R->active=false*) so that only the selected region will be displayed as the user goes down the tree. This is achieved by introducing an additional condition into to the above conditions in order not to show redundant details as per user's request. This corresponds to identifying whether a region's mode is "active" or "not" during the browsing process. The following pseudo code explains the addition of extra steps into the previous conditions.

```
for every region "R" in the tree {
if ( (R->level=current_level AND R->active=true) OR
(R->level<current_level AND R->active=true AND R->child[0]=null)
OR (R->level<current_level AND R->active=true AND
R->child[0]>active=false))
show its details;
}
```

Figure 5-4 shows results from this simplified browsing process. **Figure 5-4a** shows an initial segmentation of 50 regions for an image taken from the "Claire" sequence, and **Figure 5-4b** shows the simplified segmentation of only 16 regions obtained using the simplification technique described above. In this example, more details are shown around the face, whereas background and the body details have been forced to be hidden by the user. Thus, it reduces the number of regions to be selected in the

segmentation, or in other words, results in a fewer number of mouse-clicks in the object segmentation.



a. Spatial segmentation of 50 regions

b. Simplified segmentation while browsing

Figure 5-4 Simplified browsing process

5.5 Object Tracking

Object tracking is the second component of this tool. Extensive research is being carried out in this field due to the increasing demands of MPEG-4 and MPEG-7-based applications. Although various researchers have come up with different approaches, a generic approach for this still remains as a significant challenge.

In our approach, object tracking is performed using region-based spatial segmentation, motion estimation, object projection, and a simple region-classification technique. A forward region tracking approach is used, involving block-based motion estimation followed by object projection and a region-classification method. This, in turn, determines a final set of regions as the tracked object. In this method, the accuracy of the final results depends on the accuracy of the spatial segmentation and motion estimation processes. It is important to note that the hierarchical binary tree is not used at this stage, but only the finer regions obtained from RSST are employed. In order to have a fine object boundary, an over-segmentation of the image is desirable. This requirement is adequately supported by the system since the user can define the required granularity at any stage of the tracking process due to the use of RSST.

a. compensated mask

b. compensated mask after the hole-filling process

c. final segmentation

Figure 5-5 Processing of tracked objects

Once the initial VO is created, and hence a VO mask, the motion between the current frame and the previous frame is calculated. The projection step then uses this motion information to compensate the VO mask.

Motion estimation is based on the full-search (exhaustive search) block-matching algorithm. It was chosen in our approach due to the fact that it is an optimal block-matching algorithm despite being relatively slower than the hierarchical block-matching algorithms, such as logarithmic search, 3-step search, etc. Motion estimation is performed by taking blocks of 8×8 pixels from the previous frame and a search width of 16 pixels in the current frame. Thus, the search area is bounded by a displacement of 8 pixels in each direction. In our approach, Minimum Absolute Difference (MAD) measure is used as the block-matching criterion. Only full-pel precision motion estimation is carried out, however, half-pel precision motion estimation could have helped to improve our motion estimation results further.

On projecting the object mask using the motion vectors, some holes were found to be appearing in the compensated mask. Therefore, a simple hole-filling technique is used to overcome this problem in our system. It is based on morphological dilation and erosion filtering techniques. This effect is shown in **Figure 5-5a**. **Figure 5-5b** shows the compensated mask after applying the hole-filling technique on the result shown in **Figure 5-5a**. Simultaneously, the current frame is input to the RSST algorithm to create a spatial segmentation. The tracking process then takes the compensated object-mask and the spatial segmentation of the current image as the inputs. The final segmentation, shown in **Figure 5-5c**, is obtained by applying the region classification technique,

which is used to classify which regions of the current image correspond to the compensated mask. It is actually this process which makes the final decision to select the candidate regions from the current image, i.e. the background/foreground decision. This technique works simply by counting the number of pixels that are common to the compensated video object mask (VO^c) and to the region being studied, and then determining if that number is higher or lower than the product of a pre-defined threshold and the total number of pixels of the region. Alternatively, the region classification technique can be described as follows (see also **Figure 5-6** and **Figure 5-7**).

a. compensated video object | b. current partition | c. identified regions | d. resulting video object

Figure 5-6 Region classification

Firstly, we need to find how many pixels of each region of the current partition lie inside the compensated Video Object (VO^c) .

i.e. for every pixel p_j in R_i, $C_{IN} = C_{IN} + 1 \; if \; p_j \in VO^c$

$C_{OUT} = C_{OUT} + 1 \; if \; p_j \notin VO^c$

where R_i represents the i^{th} region, p_j represents the j^{th} pixel, C_{IN} is the number of pixels counted inside, C_{OUT} is the number of pixels counted outside.

Secondly, regions are classified to be valid if $C_{IN} \geq (C_{IN} + C_{OUT}) \times THD$ where THD is a pre-defined threshold. The union of those valid regions corresponds to the tracked object.

The overall tracking process can be illustrated using the block diagram shown as **Figure 5-7**.

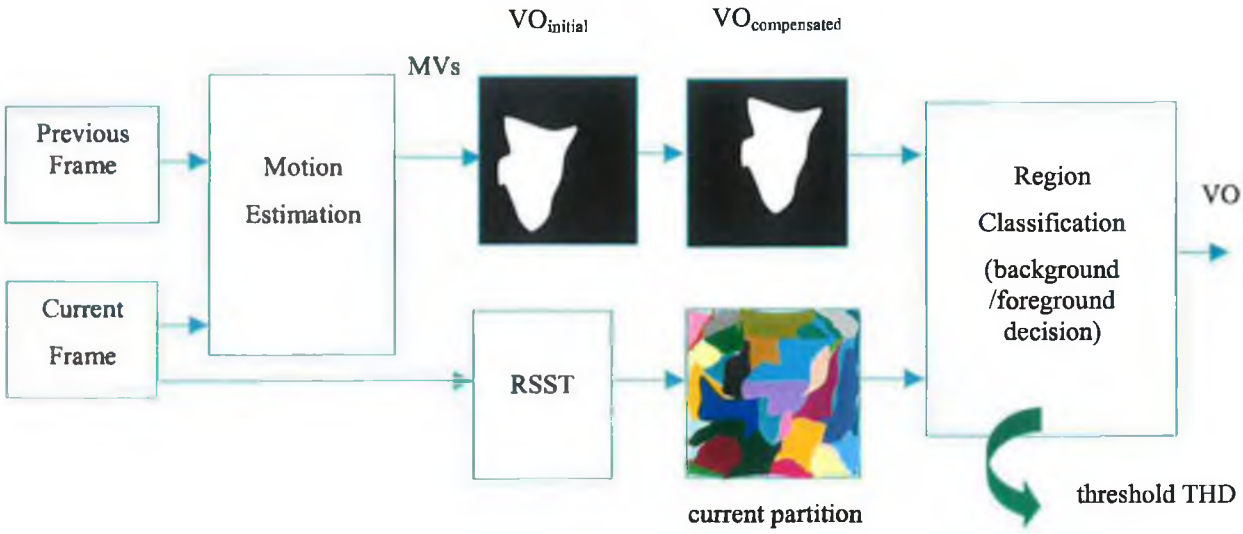


Figure 5-7 Object tracking method

For the tests conducted in our research (see the results discussed in chapter 6), a threshold of 0.7 was chosen by trial and error, i.e. a region from the current partition is considered to be valid if it was covered more than 70% by the VO^c. In our experiments, this parameter proved to be dependent on the segmentation granularity, and can therefore determine the accuracy of the final results[4]. The regions identified from this process finally represent the VO in the current frame. However, the introduction of such a simple threshold criterion can lead to the merging of inaccurate background regions with the foreground object. A detailed discussion of these issues can be found in the next chapter under results and discussion.

5.6 Various features supported to allow interactivity

In this section, various features that are supported to facilitate user interactivity are described. This interaction plays a vital role, particularly in the first stage of the segmentation process. In the second stage where object tracking is used, user interaction is needed for re-initialising the object whenever the tracked segmentation becomes inaccurate. The following sections are devoted to a discussion of events, such as how objects are formed, how the tree browsing process works, how to simplify the

[4] The issues in choosing an appropriate threshold value for region classification are addressed in chapter 6 under results and discussions. In our experiments, a threshold value 70% was found to be an appropriate value, given a spatial segmentation of suitable granularity.

tree browsing process, how to refine regions which are not accurate enough, how object re-initialisation works, etc.

5.6.1 Forming the objects

This is the process of generating spatial objects with the help of user guidance. At this stage, objects are incrementally built by clicking on image regions at various levels in the hierarchy. Since the segmentation is represented in a hierarchical manner, it is possible to browse the tree at various levels of resolution. Consequently, the user can select the best possible region in terms of its accuracy and size, thereby minimising the amount of human interaction or in other words the *number of mouse-clicks*. The larger the size of a region selected to fit into an object, the fewer the number of mouse-clicks required to complete the object formation process. It can be found in chapter 6 that the complexity of the spatial object extraction process is discussed in terms of the number of mouse-clicks used. During the object segmentation process, this number is constantly displayed on the selected-regions window. In order to provide better flexibility, more options such as select, remove, show parent, show child 1, show child 2, and split are provided as simple mouse-clicks to be performed on a selected region.

5.6.2 Browsing the tree

As shown in the GUI, the browsing of different levels of the partition tree is provided using a slider-bar. The relationship between the slider-bar and the tree hierarchy can be described using the example shown in **Figure 5-8**.

In this example, there are four hierarchical levels and five leaf nodes. Level1 corresponds to the root node of the tree (i.e. region R1). Level2 consists of two regions, R2 and R3, which are the child nodes of R1. In level3, there are four regions, R4-R7, whose parent nodes are R2 and R3. The last level has introduced only two new regions R8 and R9, and they are the child nodes of R4.

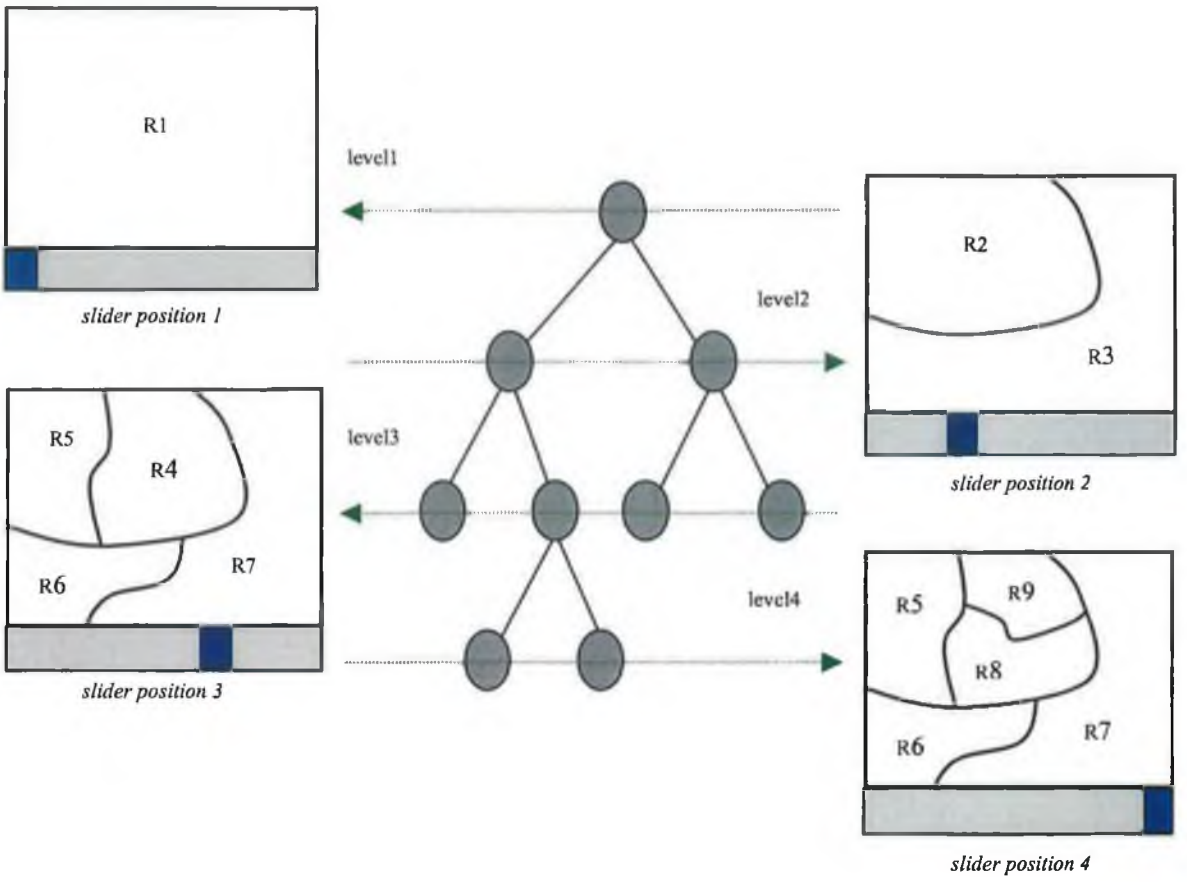


Figure 5-8 Relationship between slider-bar and tree hierarchy

In our implementation, the current level of the binary tree is considered as the current position of the slider-bar. The length of the slider-bar is set equivalent to the number of levels the BPT algorithm produces. Thus, the image can be browsed at different resolution levels by simply changing the position of the slider. More importantly, objects can be formed selectively while performing other supportive functionalities, such as region splitting, browsing simplification, etc. **Figure 5-9** shows an actual case of a segmentation performed on the first frame of the mother and daughter sequence. The initial segmentation consists of 40 regions, which led to a total number of 79 regions and 15 levels in the tree hierarchy.

The partition at level 1 shows only one region, which corresponds to the same shape of the original image (a rectangle) and its mean colour. Level 2 shows an image partition of two regions represented with their mean colour values. This shows that the region at level 1 is now split to 2 regions, which have been attached to the nodes of the tree at level 2. Similarly, the nodes at level 5 correspond to 22 regions. The 15^{th} level, which is

the last level of the tree, shows the initial partition of 40 regions. During the browsing process, it is not mandatory to show all the automatically generated regions, which otherwise make the region selection process slower and more difficult. For example, if the initial partition contains 400 regions, it would then be a tedious task to select regions out of 400 where some regions could even be very small in size. This has led us to introduce an extra functionality in our system, namely tree-browsing simplification (explained in section 5.6.3), in order to hide some of the regions that are not important. In our tool, as some additional information to the user, the corresponding number of regions of the currently displayed image-partition is shown on the segmented-image window during the browsing process.

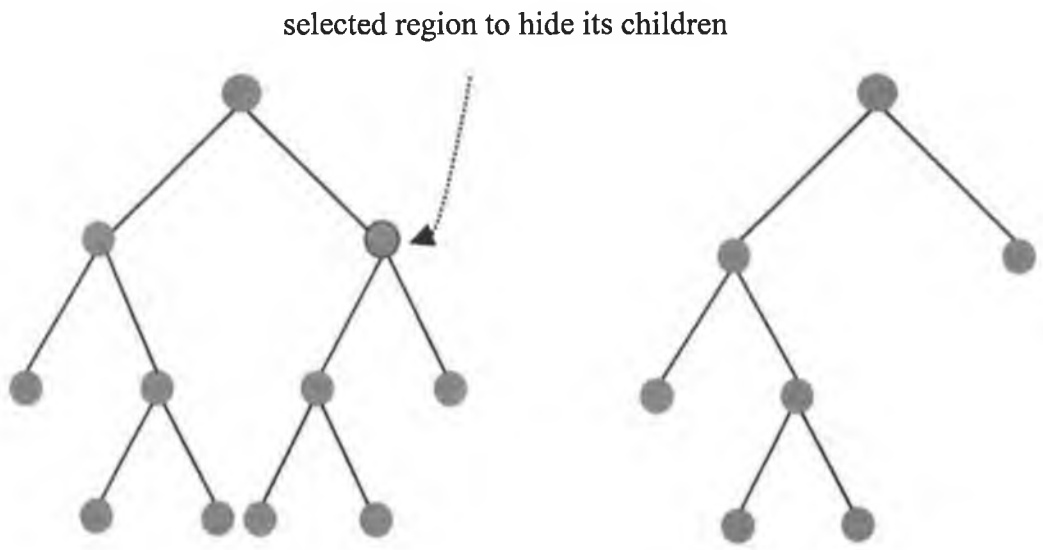
a. level 1 b. level 2 c. level 5 d. level 15

Figure 5-9 Image partitions corresponding to different levels in the hierarchy

5.6.3 Simplification of browsing process

The tree browsing-simplification step introduced in our tool has substantially helped to simplify the overall object-segmentation process. This allows hiding all the child-nodes of any selected region on the segmentation, which very efficiently allows a small number of homogenous regions to be selected in the final segmentation. Hence, the overall object-extraction process becomes easier and faster. More details of this step can be found in section 5.4. In the GUI, this functionality is supported by left mouse-clicks to be performed on the required region whose child nodes are to be hidden (see also GUI shown in **Figure 5-2**). When a particular region is clicked, its contour is marked in a different colour so as to highlight the region not to be further subdivided. These highlighted contours can be seen in **Figure 5-2**, marked on the segmented image. **Figure 5-10** illustrates the browsing-simplification step used in this tool. The hierarchical region-tree created by the automatic segmentation algorithms (i.e. RSST and BPT) is shown in **Figure 5-10a**. The blue node at level 2 in the tree corresponds to the region selected by the user to force its children to be hidden during the browsing

process. The tree browsing resulting from this step is shown in **Figure 5-10b**. It shows that four child-nodes of the marked region have been hidden.



a. Hierarchical binary tree showing 6 leaf nodes and one node (contour marked in blue) whose child nodes are to be hidden during the browsing process

b. Resulting hierarchical binary tree during the browsing process

Figure 5-10 Browsing simplification step

5.6.4 Object Extraction using EXTRACT button

The object formation process is completed when the user presses the EXTRACT button. One single press on this button creates an object mask of the segmentation. Providing an EXTRACT button in the GUI and storing these object masks in a dynamic array facilitate extracting any number of objects from a segmented image. The extracted objects are assigned different labels and displayed in a list, allowing the user to select any of the object/objects to get the final segmentation. In our system, these object masks are displayed in randomly assigned colour values just for clarity.

5.6.5 Save to disk functionality

A functionality is provided to save segmentation results to disk. Segmentation results, such as initial segmentation and extracted objects, can be saved either in grey scale image format or in colour image format. These results can be represented in segmentation label form, region-based form (regions can be represented in mean colour

values or original colour values), or in the form of contours superimposed on the original image.

5.6.6 Reset functionality

Although the reset functionality is not an essential feature to have in the system, it very importantly allows a quick return to the normal states of the segmentation process, when necessary. This is useful if mistakes that occurred during the manual segmentation process need to be quickly corrected. Thus, with this functionality, the speed of the overall segmentation process can be improved. Some of the resetting operations currently supported in this tool are (a) bringing the slider position to its original position and displaying the root node image partition, i.e. level 1 (useful if the browsing needs to be restarted from the 1st level) (b) clearing the selected-regions window with a blank screen (useful if the selected regions did not satisfy the user's intention) (c) bringing the current states of regions to its original states (useful if any region/regions, which are marked for browsing simplification, need to be brought back to their original states)[5]. It should, however, be noted that this reset command does not re-modify the binary tree structure subject to changes during the region splitting process, which is explained in the next section.

5.6.7 Refinement by region splitting- further interaction

In some cases, automatically extracted regions do not fulfil the user's needs in terms of their accuracy. Due to this drawback, a region refinement step is provided to modify the automatically generated hierarchical tree structure. In this approach, spatial regions can be further split to deal with a too coarse or inaccurate spatial segmentation. It is important to note that this feature is used only if the user needs to split regions to describe them in finer details or in a more meaningful manner. Our technique is based on a simple mouse-driven contour marking. This is carried out by drawing a contour of the interesting object on the original image and selecting "split" option on the segmentation. In contrast, it is clearly a process of manual refinement where the spatial accuracy of the newly formed object is completely dependent on how accurate the user is capable of performing freehand drawing.

---

[5] These three operations are automatically launched just with a single mouse-click to the reset button. This is not exactly a functionality of the type undo/redo.

The following technique, which was extracted from the work related to the European ACTS project "DICEMAN", explains how new regions are formed from this splitting process. Once a contour is drawn on the original image (see the GUI), the contour points are stored in a dynamic array. After the user selects which region he/she wants to split on the segmented image, it is then required to identify the pixels of the selected region that lie inside the marked contour. These identified pixels then describe one of the newly formed regions, with the second new region being the collection of pixels identified outside the marked contour. The following technique, illustrated in **Figure 5-11**, is used to identify the correct pixels within the drawn contour. For example, let S1-S5 be the contour points and P_1 be the pixel that needs to be identified as being inside or outside the contour. If P_1 is inside the contour (surrounded by S1-S5) it should then form a closed path, whereby the addition of θ_1, θ_2, θ_3, θ_4 and θ_5 should make 360°. This process is carried out for all the pixels of the selected region so that the entire selected region can then be divided into two regions. The corresponding mean colour values are calculated and attached. The new regions are assigned new labels, and the original region is deleted.

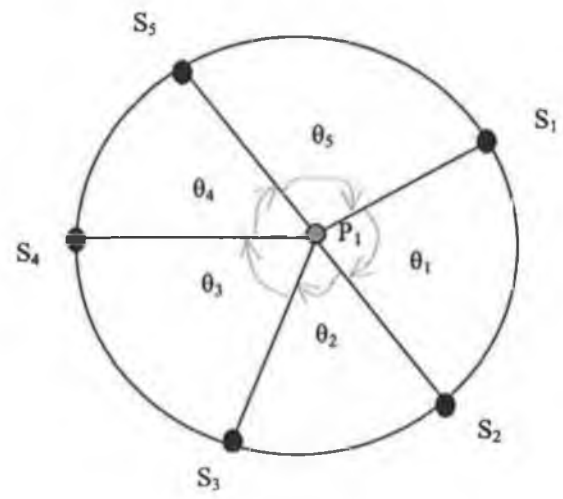


Figure 5-11 Pixel identification for splitting process

Figure 5-12 shows an example of what the results from our splitting technique looks like. The automatically generated partition is shown in **Figure 5-12a**. The splitting technique could be used, for example, in case the user wants to see a region of the mother's face instead of torsos of both mother and daughter at level 2 of the segmentation. The result shown in **Figure 5-12b** is obtained by drawing a contour on

the original image and selecting split option by clicking on the segmented image (i.e. in this case, it is the root node of the tree or the first region of the segmentation). This is an example of a situation where the automatically generated results don't seem to satisfy the user's needs.

a. automatically generated regions

b. manually generated regions from splitting technique

Figure 5-12 results from automatic segmentation and splitting technique

However, it is very important to pay attention to the contour accuracy of such regions, as it certainly demands a more intelligent splitting technique to accurately define the object boundaries. Also, once a manual refinement is done on a region at an early stage of the hierarchy, it destroys the automatically generated hierarchical structure of its children. This is because the correlation between the parent nodes and the child nodes cannot be retained after that level. Thus, it is more appropriate to refine only the leaf nodes of the tree, so the splitting technique does not need to damage the structure of their ancestors.

5.6.8 Track

The tracking algorithm is invoked by pressing the "Track" button. At this stage, it is necessary to have an object-mask defined. Once spatial objects are extracted, it is possible for the user to select any of the objects from the list, which displays objects as different labels. Currently, only single object tracking is supported, but this could easily be extended to multiple object tracking. Once the tracking algorithm is launched, the current frames are fed to the motion estimation block as well as to the RSST block. The entire tracking process, explained in section 5.5, is executed at this stage. The tracking results are written to disk in PGM or in BMP format, if necessary. If the tracked segmentation tends to be inaccurate, the re-initialisation step can be invoked.

5.6.9 Re-initialise

The re-initialisation step is needed when the tracked segmentation starts to be inaccurate. This situation can arise for several reasons. It can be due to object deformations where the object's shape changes during the tracking period. Due to this effect, the object being tracked may not precisely correspond to the initial object. Other situations like object occlusions, camera zooming effects, object disappearances can also cause similar problems. In this semi-automatic segmentation system, an option is provided to the user to manually initiate the re-initialisation process. The user decides to do so when it starts to generate results that are not satisfactory with reference to the initial object. Thus, executing "Reinitialise" stops the currently running processes and waits for human assistance for object-refinement. The level of user interaction at this stage is kept as simple as possible by allowing the user to refine the current object mask, either by adding or deleting small regions generated by RSST. Hence, no hierarchical segmentation representation is used, but a finer segmentation is provided to facilitate accurate object-refinement. It should be noted that the object-tracking step in our tool becomes a completely automatic process if a need for re-initialisation does not arise.

5.6.10 Other text-fields

Several text-fields are incorporated to pass input parameters to the algorithms used. These parametric values need to be input by the user before running the automatic algorithms, i.e. before opening up the file menu, if the provided default values are different from what is required. The first text-field "initial number of regions" is provided to pass a value to the RSST algorithm to specify the number of regions to be available in the initial segmentation. The text-field "number of frames to track" passes a value to the tracking algorithm to specify the length of the image sequence in terms of number of frames. Thus, for example, if the user opens up the 10^{th} frame of a particular sequence, the algorithm will execute the tracking for the specified length starting from the 10^{th} frame. A threshold value for the region classification process can be passed through the region classification threshold text-field. The default value is set to 70% since this value was found to be suitable for most of the experiments carried out during the research.

5.7 Software Implementation

The Java language was chosen as the programming language for our system implementation. The main reason why Java was selected in this work is the provision of the user interactivity feature, which is one of the main concerns in the author's tool. As described in section 5.3, a GUI is the front-end through which the user interacts for carrying out object segmentation. Although the algorithms could have been written in a faster procedural language such as C, and utilised within a java-based GUI using the Java Native Interface (JNI) feature, preference was given to make the system more flexible, adaptable for future changes, and also based on the author's personal interest. The disadvantage of using Java in this system is the speed constraint as Java is obviously not the best language for speedy applications. Due to the platform independence feature of Java, it wouldn't be too difficult to port this tool to any other applications running under different platforms. However, the C++ language, in particular Visual C++ running under the Microsoft Windows platform, which also supports tools for developing GUIs even with a better speed efficiency, can also be considered as an alternative development environment. Furthermore, the KDevelop IDE, which is a C/C++ programming environment for Linux/Unix, is certainly a powerful system to use for this type of GUI-based image analysis applications.

The author has tried out different IDEs. These IDEs were "Visual Café" by Symantec, "VisualAge" by IBM and "JBuilder" by Borland. Due to various difficulties with memory management, manipulation of graphical components and the overall complexity of use, etc., it was finally decided to use JBuilder for our work. It was found that JBuilder undoubtedly made the implementation task easier since it caused a minimal amount of trouble in comparison with the other software tools then available to us. The version of JBuilder used during the implementation work was JBuilder 3.5, though it wasn't the latest release at the time of our implementation. However, the execution speed of the JBuilder IDE was found to be slower than the other two IDEs mentioned above as the entire Jbuilder IDE itself has been written in the Java language. The following are the main classes that are used in our software modules for performing interactive object segmentation.

- **ObjectSegmentation class**

This is the main class in the project. This class is extended from Java JFrame class. Most of the user interactive functionalities such as slider-bars, buttons, combo-boxes,

text-fields and image-panels were implemented within this class. All of the following classes are instantiated within this class, which then allows us to access their data members and methods, when necessary.

- **RSST class**

The methods and data members for creating the RSST are incorporated in this class. Data members such as original image-size, number of links of the SST, number of regions of the SST, and an array of RSST_region classes are defined within this class. All the necessary methods for reading PGM files from the disk, initialising regions' arrays with original colour values, pixel-lists, link-lists, and merging regions are provided. In the function for merging regions, methods are provided to update regions and links during the merging process. Update of regions is done by calculating mean colour values for new regions, and update of links is done by calculating link-costs and rearranging the links in the graph.

- **RSST_link class**

A link in the RSST is defined by using an index number, a data member namely cost, and a 2-element array of two regions to be linked together. These links are subject to new updates and removals from the graph while the merging takes place according to its link-cost.

- **RSST_region class**

A region for RSST is defined with a three-element array for mean colour values (Y, U and V), a data member for area (spatial size), a region link-list, and a pixel link-list. This form of region class is used only during the RSST formation process.

- **BPT class**

The BPT is defined using a set of data members and methods. Most importantly, it contains a dynamic array for Region class. This is because the hierarchical binary tree may be susceptible to structural changes, as the tree needs to be modified during the splitting process (user interaction). Numerous data members such as the total number of regions in the tree, the arrays for region link-cost calculation, merging sequence calculation are included. Methods are provided to initialise the regions of the initial segmentation with their colour, area, and pixel position information followed by some

more methods to merge regions by maintaining a level hierarchy for each region, a track of merging sequence, child-parent relationships, etc.

- **Region class**

A Region class is defined in a slightly different way to that of RSST_region. The main reason for this is to define regions more appropriate for use in the hierarchical binary tree creation process. In this class, a region is defined using an index, a 3-element array for mean colour, an area (spatial size), a level (to specify the level number in the tree), a decision (to specify the status of it whether active or not), a pixel list, a Region class for parent node, and a 2-element array of the Region class for two child nodes.

- **Display class**

The Display class is extended from JPanel class. The purpose of having this class is to enable the use of graphical windows in the GUI. There are six classes of this type in our GUI to display images at various levels of the segmentation process. The main task is to override the paintComponent() method to paint the image panel either with RGB components or grey component so that panels can be repainted, when necessary.

- **Motion class**

The method for motion estimation is provided in this class. Parameters can be passed to the function to change the block-size and search-area. However, these are currently hard-coded, though they could be input either through the GUI or as command line arguments.

5.8 Discussion

The presented semi-automatic segmentation tool is comprised of two main stages. These are spatial object segmentation and object tracking, which collectively form a tool for video object segmentation. The spatial object segmentation task is carried out using both automatic segmentation algorithms and user guidance. Automatic segmentation is based on the RSST algorithm and the BPT technique. The automatically generated results obtained from the RSST are hierarchically represented using the BPT technique. The spatial object segmentation is then completed by provision of user interaction where the user manually extracts the required objects. Despite the fact that there is a relatively high amount of user interaction involved in this

phase of the process, the ability to accurately perform object segmentation on any complex image makes the system suitable for generic applications. Furthermore, numerous functionalities built into the system for efficiently browsing the hierarchically represented segmentations and forming semantic objects have enabled this spatial object-segmentation method to be considered a useful system.

In the object tracking stage, VOPs are generated for a specified length of the image sequence. As described above, this component is comprised of several phases, such as spatial segmentation, motion estimation, object projection, and region-based classification techniques. The object projection step is performed based on the estimated motion information, and therefore its accuracy is largely dependent on the performance of motion estimation. A Region classification technique, on the other hand, is used to finally classify regions that are generated from a region-based spatial segmentation technique into a background or foreground object. Hence, its performance is highly related to the spatial segmentation process.

Despite the fact that this system is fully functional, there are certain areas where some shortcomings are apparent, and should therefore be reconsidered for the sake of improving the overall performance of the tool. The major improvements are required at the second stage, i.e. object tracking, which is a relatively more difficult task than the first.

In this context, an advanced motion estimation technique needs to be identified and integrated into the tool. This would then help to obtain more accurate motion information, thus resulting in more accurate object projections in the tracking stage. However, it is not only this process that could improve system performance. The last phase of the object tracking, which is the region classification step, also plays a big role on the quality of final segmentation results. In addition, further user guidance that may be required during the tracking process needs to be facilitated in an efficient manner. This would mean that in the intermediate stages a minimal amount of user interaction should be provided for object refinement, keeping in mind that user interaction is simple to perform and takes very little time.

The introduction of several possible improvements to the spatial object segmentation could enrich the overall performance. One challenging addition in this regard is to

introduce a single mouse drag to mark the image to segment the intended object, thereby avoiding multiple mouse-clicks. This would, in turn, contribute largely to reduce the user interaction involved in object formation, but certainly requires an elegant approach to perform the task. In addition, a PSNR criterion would be another option with which the user can define the granularity of the initial segmentation. Furthermore, a more advanced technique to refine some of the automatically generated regions, which require further fine details, would be a useful enhancement.

6. EXPERIMENTAL RESULTS AND DISCUSSIONS

6.1 Overview

In previous chapters, system implementation details including various algorithmic details such as still image segmentation and image sequence segmentation are described. This chapter presents the results obtained using the semi-automatic segmentation tool developed during the course of our research. The results are presented for spatial segmentation and image sequence segmentation in sections 6.2 and 6.3 respectively. The main emphasis is on how easy or difficult it is to extract semantic objects from a given image. Thus, the results of the spatial object-segmentation are discussed in terms of "the number of mouse-clicks" required during the process of user interaction. Tracking results obtained from the second stage of the tool, presented in section 6.3, are discussed by considering how the results depend on various parameters that need to be pre-defined. Furthermore, visual similarities and dissimilarities with reference to the original object are discussed in order to evaluate the performance of our tracking algorithm. Nevertheless, a full objective evaluation of these tracking results is outside the scope of this thesis, despite the fact that there are new objective evaluation methods which are emerging, and may prove to be useful [Mech and Marques 2001], [Correia and Pereira 2001]. In section 6.4, an overall discussion of the results is given by highlighting where the errors are significant in the implemented system.

Our experiments were carried out on standard test data sequences. Image sequences, such as "Claire", "Foreman", "Mother and Daughter", "Table Tennis", and "Children", were the data sets used for these experiments. Although a segmentation tool should be able to work on any image sequence, it may produce variable quality results depending on how "intelligent" the tool and how "complex" the sequences to segment are. The test sequences used in our experiments exhibit different characteristics that allow us to experiment with slowly moving objects, fast moving objects, moving objects under camera motions, occlusion effects, etc. As a result, some sequences are easier to segment than others. In our experiments, these data sets are used in the QCIF format. In addition, some object-segmentation experiments were also carried out on non-standard images with different image resolutions. These results are shown in appendix A.

The results presented in the following sections are categorised into two parts, i.e. spatial object segmentation results and object tracking results. In spatial object segmentation results, automatic region-based segmentation and interactive semantic object segmentation are discussed. Object tracking results then correspond to the results obtained from the automatic object-tracking algorithm (the term automatic is valid whenever the re-initialisation step is not used, see section 5.6.9).

6.2 Spatial Object Segmentation Results

As described in chapter 5, spatial object segmentation is performed in two steps. Firstly, an automatic segmentation algorithm, RSST, is used to generate a specified number of regions, known as the initial partition, to facilitate the subsequent object segmentation processes. Secondly, user interaction is provided to build up semantic objects by selectively clicking on homogenous image-regions. In this section, the initial partitions are represented with mean colour of the regions. Consequently, it is difficult to distinguish the original image and the actual segmentation partition at a glance when the number of regions in a partition is high (i.e. the granularity is finer). The extracted objects are shown in their original colour on a white background.

As the interactive object-segmentation process is progressing, the actual number of mouse-clicks is automatically counted and displayed underneath its respective window. The number of mouse-clicks gives an indication of how difficult or easy the object extraction process is. However, a definitive conclusion cannot be reached only from the number of mouse-clicks since the object extraction process substantially depends also on other user-interactive steps, such as tree browsing, region refinement, etc. Nevertheless, the results go some way towards reflecting the success of the prime goal of our work and the use of the implemented system towards object segmentation applications.

The following figures, **Figure 6-1** - **Figure 6-5** show the results of the automatic region-based segmentation performed using the RSST algorithm with the original image as an input to the algorithm. These results correspond to different initial partitions, containing different number of regions specified by the user. Hence, our discussion is based on selection of a suitable value for the initial number of regions. To

this end, values such as 20, 100, 200, 400, 600 and 800 are chosen for comparison purposes of our results.

In **Figure 6-1**, the original image and different initial partitions containing different number of regions are shown for the 1st frame of the "Claire sequence". It can be seen that the segmentation quality is higher when the number of regions in the partition is larger. In this experiment, the original image possesses a simple background and it does not exhibit a highly detailed picture. Therefore, it is easier to produce a user-desired segmentation with a smaller number of regions.

Original image 20 regions 100 regions 200 regions

400 regions 600 regions 800 regions

Figure 6-1 Automatic region-based segmentations for the Claire image

The variation of the segmentation quality with corresponding initial number of regions in the partitions can be noticed, especially with the last four segmentations producing results quite difficult to distinguish from the original image. However, the question of which segmentation is best for further use (such as tracking, describing desired spatial objects, etc.) remains open.

Figure 6-2 shows the original image and different initial partitions for the 1st frame of the "Foreman" sequence. This image contains more fine detail than the above Claire image.

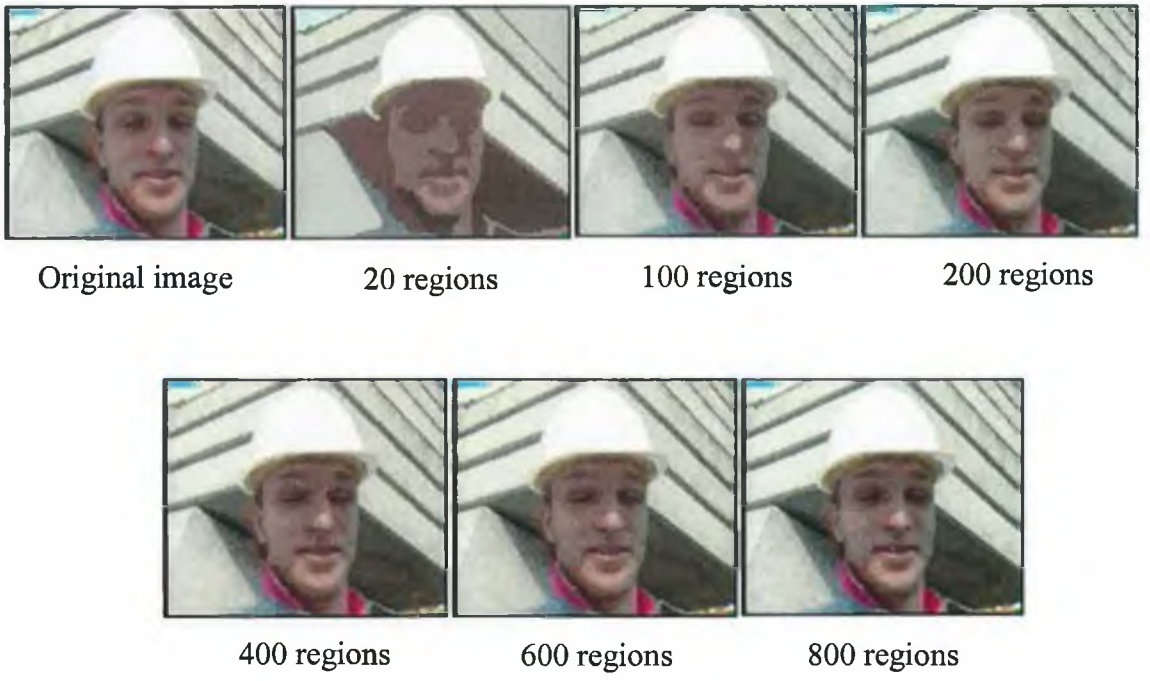


Figure 6-2 Automatic region-based segmentations for the Foreman image

This set of results illustrates the need for a larger number of regions in an initial partition due to the fact that finer segmentation granularity is needed to separately identify different objects in the image. These images show different segmentation-granularities with the last segmentation (i.e. 800 regions) almost identical to the original image.

Figure 6-3 shows the original image and different initial partitions for the 1st frame of the "Mother and Daughter" sequence. This image also contains more fine detail than the above Claire image.

Original image 20 regions 100 regions 200 regions

400 regions 600 regions 800 regions

Figure 6-3 Automatic region-based segmentations for the Mother and Daughter image

In these results, a similar effect can be seen on the first three segmentations. The segmentation results then reach a quality where a human eye cannot easily differentiate the original image when it exceeds 400 regions. Nevertheless, it is difficult to define an exact value, which should be used to achieve the best object segmentation as this task generally depends on the particular object to be segmented. In our experiments, it was noticed that a particular object of interest couldn't always be guaranteed to be free of background regions if the number is not well chosen.

The results shown in **Figure 6-4** correspond to the 60^{th} frame of the "Table tennis" sequence. In this experiment, the 60^{th} frame is chosen to facilitate extraction of an object corresponding to the person playing table tennis.

Original image 20 regions 100 regions 200 regions

400 regions 600 regions 800 regions

Figure 6-4 Automatic region-based segmentations for the Table Tennis image

The first three segmentations can be easily differentiated from the original image, whereas segmentations at 400 regions and above do not show a noticeable difference. However, it was noticed that an accurate object-segmentation of this person could not be achieved at 400 regions, even though, in this example, an initial segmentation of 400 regions shows a perfect segmentation to a human eye.

Figure 6-5 shows a similar set of results. This image shows a special case since it contains multiple distinct objects within the picture. In this experiment, the 13th frame of the "Children" sequence is chosen because of the fact that it provides three useful objects.

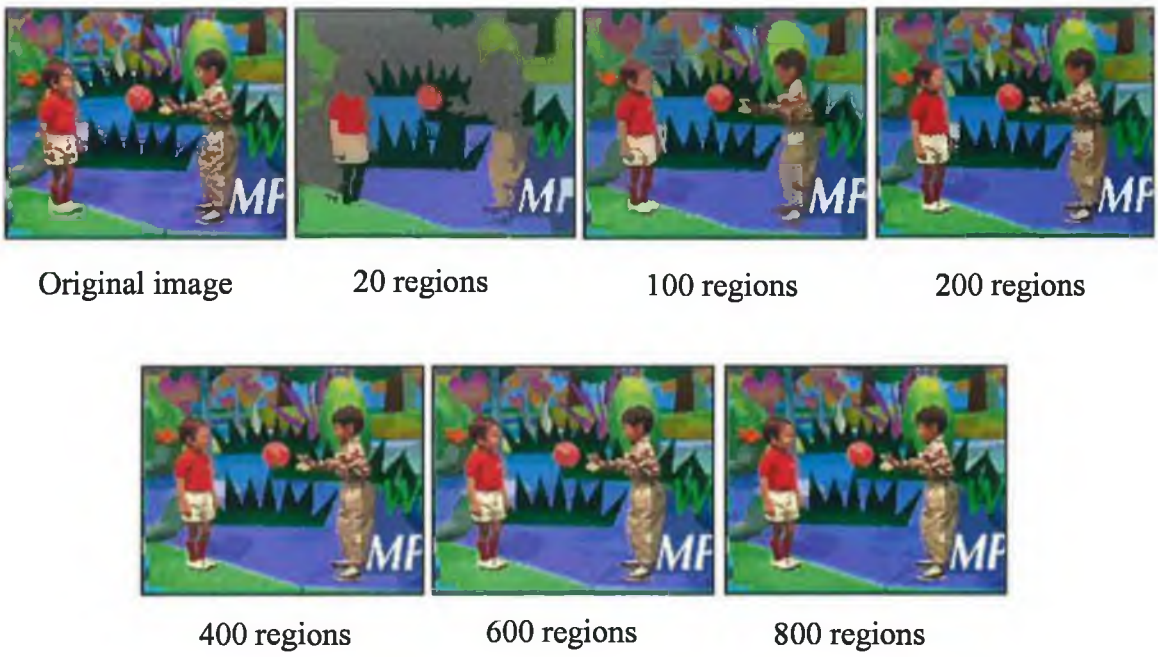
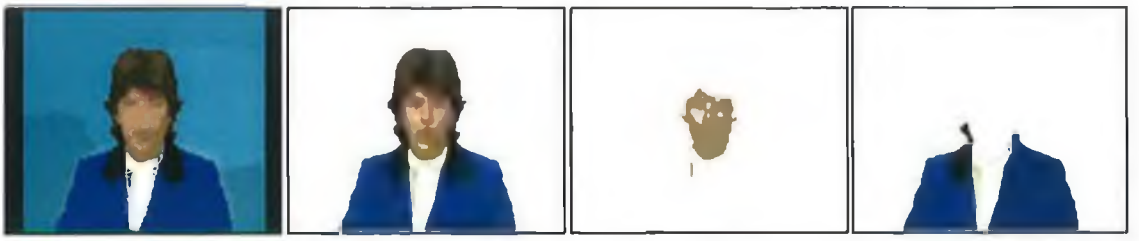


Figure 6-5 Automatic region-based segmentations for the Children image

It was noticed in our experiments that satisfactory object segmentation results are possible only when the number of regions reached a level about 800. Similarly, it can again be argued for different numbers of regions for segmentation of different objects since it is the object that requires a certain number of regions to be present in the initial partition.

Considering the required granularities of segmentation, spatial object-segmentation is performed on some selected frames of the sequences, such as "Claire", "Foreman", "Mother and Daughter", "Table Tennis" and "Children", with their initial partitions containing 22, 600, 400, 700 and 800 regions respectively. These numbers are chosen in this manner so as to suit the requirements of the tracking process, presented in section 6.3. Given this partition, the BPT generates a browsable binary tree to facilitate interactive object-segmentation. This binary tree contains nearly twice the number of regions of the initial partition. Therefore, the number of mouse-clicks required, or the number of regions selected manually by the user in the segmentation process, should be a comparative measure against the total number of regions in the tree. The following figures, i.e. **Figure 6-6 - Figure 6-10** show the results of the interactive object-segmentation along with the initial partitions used.

Figure 6-6 shows the initial partition and the interactive object-segmentation results for the 1st frame of the "Claire" sequence. The girl, her face, and her torso are chosen as different semantic objects in this experiment. Since this image does not contain any complex background objects, the first semantic object could be extracted with an initial partition of only four regions. If a different object like a face or a torso is needed, the number of regions in the initial partition has then to be increased. It is found that, for an initial partition of 22 regions and hence a binary tree containing 43 regions with nine levels, only two mouse-clicks are required to segment the first object. It takes only 1 mouse-click for the face object while 5 mouse-clicks are required for the torso object.



Initial partition-
22 regions

Claire object-
2 mouse-clicks

Face object-
1 mouse-click

Torso object-
5 mouse-clicks

Figure 6-6 Interactive object segmentation for Claire image

For the Foreman sequence, the man, his helmet, and both helmet and face are chosen as three semantic objects in the second experiment. The initial partition is created with 600 regions, which results in a binary tree of a total number of 1199 homogeneous regions with 25 levels. The first object requires 12 mouse-clicks. The second object, the helmet, requires only 3 mouse-clicks while the third object requires 15 mouse-clicks. The initial partition and the extracted semantic objects are shown in **Figure 6-7**.

Initial partition- 600 regions | Foreman object- 25 mouse-clicks | Face and helmet object – 15 mouse-clicks | Helmet object- 3 mouse-clicks

Figure 6-7 Interactive object segmentation for Foreman image

In the third experiment, the object comprising both mother and daughter is segmented. The results are shown in **Figure 6-8**. In order to have satisfactory fine segmentation, the initial partition is created with only 400 regions, which produces 799 regions and 26 levels in the binary tree. The interactive segmentation process for this object requires 12 mouse-clicks.

Initial partition- 400 regions | Mother and Daughter object- 12 mouse-clicks

Figure 6-8 Interactive object segmentation for Mother and Daughter image

The person playing table tennis is selected as the semantic object for the fourth experiment. The RSST algorithm is run for 700 regions. Hence, the binary tree contains 1399 homogeneous regions and 26 levels. The results show that only 13 mouse-clicks are required to extract the object shown in **Figure 6-9**.

Initial partition- 700 regions | Person object- 13 mouse-clicks

Figure 6-9 Interactive object segmentation for "Table Tennis" image

In the fifth experiment, three semantic objects are chosen. The first two are the two children playing with the ball in the scene. The third object is the ball itself. For this experiment, an initial partition of 800 regions is created using the RSST, hence the BPT provides 1599 homogenous regions with 34 levels.

Initial partition- 800 regions | Right child object - 16 mouse-clicks | Left child object- 15 mouse-clicks | Ball object- 1 mouse-click

Figure 6-10 Interactive object segmentation for "Children" image

To extract the first object, which is the child on the right, 16 mouse-clicks are required. The next child on the left requires 15 mouse-clicks, and the "ball" requires only one mouse-click. **Figure 6-10** shows these results along with the initial partition used for the interactive segmentation.

From these experiments, it is clear that the task of extracting a particular object from a particular image depends on the complexity of the image and the required degree of

accuracy for the object's boundary. When the image is more complex, a larger number of regions should be generated in the initial partition. If the required number is not satisfactory, the regions comprising the extracted object may not accurately define the object's boundary. This can lead to a segmentation result either missing some parts of the object or containing excessive regions from the background. Similarly, the time taken for extracting objects is highly dependent on the image complexity and to what extent the user is familiar with the system. The former is due to the presence of a higher number of regions in the browsable tree as a result of high image complexity. A higher number of regions in the tree also increases the number of levels in the tree, making the tree browsing process more time consuming. The segmentation process can also take more or less time depending on the level of user familiarity with the system. In order to summarise the above results and to present the importance of the degree of user familiarity, the above results are tabulated in **Table 6.1**, with the last column allocated for the time taken by the user to complete the interactive part of the segmentation process. This interactive part involves browsing the binary tree, selecting the desired regions and grouping them into objects. In effect, it requires more time if the region refinement step also needs to be used. However, it should be noted that the total time recorded during the interactive object segmentation process corresponds to the author's experience, though it should ideally be related to an average user.

Type of image	Initial number of regions	Total no. of regions in the tree	No. of levels in the tree	Type of object	No. of mouse- clicks required	Total time taken in sec.
Claire	22	43	9	Claire image	2	16
				Face	1	10
				Torso	5	37
Foreman	600	1199	25	Foreman	25	370
				Helmet	3	55
				Face with helmet	14	150
Mother and Daughter	400	799	26	Mother and daughter	12	175

Table Tennis	700	1399	26	Player	13	390
Children	800	1599	34	Left child	15	215
				Right child	16	310
				Ball	1	8

Table 6.1 A summary of interactive object segmentation

The following figure, **Figure 6-11** presents another set of interactively performed object segmentation results for the same 1st frame of the "mother and daughter" sequence. In this case, the initial segmentation contains only 300 regions, which is different from the one illustrated in **Figure 6-8**. The objective of this experiment is to illustrate why an initial partition of more than 300 regions is required for this sequence if an accurate object-segmentation is required. In object 1, a thin brown coloured line appears across the mother's hair. If this line is to be avoided, it then results in the segmentation of object 2, which is missing an important part of the mother's hair. This situation does not arise in the case of an initial partition with 400 regions, shown in **Figure 6-8**. This is an important issue to be considered for the second stage of our system, i.e. tracking, since this may require a higher number of regions than initially specified. Neglecting this issue may result in a tracked object-segmentation with either falsely identified background regions or missing foreground regions.

Initial partition- 300 regions | Object 1 | Object 2

Figure 6-11 Interactive object segmentation for Mother and Daughter sequence with an initial partition of 300 regions

6.3 Object Tracking Results

Video object tracking results are presented in this section. The test sequences used for these experiments are "Claire", "Foreman", "Mother and Daughter" and "Children", each of which exhibits different characteristics. All four sequences are used in the QCIF format, the first three being 200 frames long and the last one being only 50 frames long. In our discussion, these particular lengths of image sequences are considered to be satisfactory from the results evaluation point of view. Similarly, all four sequences start with the 1st frame, and the tracking is carried out in each frame of the sequence. The hole-filling technique, which is based on morphological opening by reconstruction, uses the same size of structural element for all four sequences. A 9×9 square structuring set is used for this purpose. The results presented below were obtained from a fully automatic tracking process for a given object. However, results could be improved if the automatic process is interrupted with the re-initialisation step provided in the system. For the first three sequences, the input object, i.e. the spatially segmented object at the first frame of the sequence, is the same as the second picture in Fig 6.6, 6.7 and 6.8. For the "Children" sequence, the 1st frame of the sequence is used for the purpose of tracking only one child, which then allows us to test the tracking algorithm for 50 frames[6]. Due to complicated and fast motions present in this sequence, it is only a short period of time that the actual tracking can be performed before it starts to fail. Since it is difficult to present large sets of results in this discussion, results for the "Claire" sequence are shown on every 25th frame, for "Forman" and "Mother and Daughter" sequences, results are presented on every 10th frame, while for the "Children" sequence, results are presented on every 6th frame. This enables us to illustrate more fully the results on the last three sequences since the process of segmenting the "Claire" sequence is much easier than the other three, and hence needs little discussion. From these results, it becomes clear that the results for some of these sequences are not as accurate as they could be. This is not surprising since most of our work was focused on interactive spatial object-segmentation, leaving further work on object tracking to be carried out in the future.

Figure 6-12 shows the automatically tracked objects for the "Claire" sequence. The object tracking is performed by clicking on the "Track" button in the GUI once a

[6] Note that the 13th frame of this sequence was used in the spatial object segmentation experiment (results shown in **Figure 6-10**) for better presentation of the different objects.

spatial object is defined by the user in the first frame. The subsequent region-based spatial segmentation, performed on each frame of the sequence, is based on the same initial number of regions provided in the first stage of the segmentation. In this experiment, only 22 regions are provided. Generally, it is an easy task to track such an object with a simple background, such as that present in this sequence. Since the "Claire" sequence is a video conferencing type of scenario, only a small amount of motion is present during the entire sequence. Moreover, the motion is mostly occurring around the head, and the surrounding background is not moving or changing. The results presented on this type of sequence help to demonstrate the suitability of our system for video conferencing type of applications. The threshold value used for the region classification stage is 0.7.

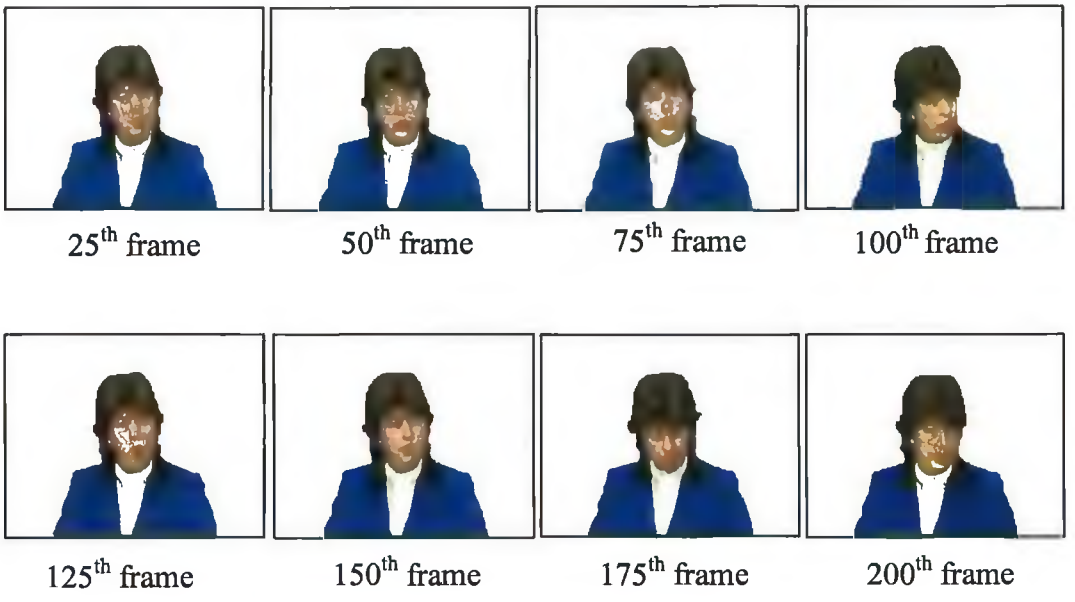


Figure 6-12 Object tracking results for "Claire" sequence

These results do not show any kind of falsely identified background regions or any missing foreground regions throughout the tracking process. Thus, these results give a clear indication that the tracking technique has done a perfect and very promising job on this sequence. Although the results are only displayed up to the 200th frame, our experiments have proven that the same success could be achieved even up to more than the 400th frame. The results presented in **Figure 6-13** correspond to the identified objects for the "Foreman" sequence, which can be considered as one of the more difficult sequences to segment. In this sequence, various effects such as camera

zooming, camera movements, changes in the orientation of the object, object disappearance can be noticed. These effects lead to errors in the motion estimation process, and hence in the entire tracking process.

The tracked objects at every 10^{th} frame of the sequence are shown in this figure. The spatial segmentation is performed on each frame of the sequence for the same number of regions, specified at the first frame. The same threshold value, i.e. 0.7 is used for the region classification process.

10^{th} frame 20^{th} frame 30^{th} frame 40^{th} frame

50^{th} frame 60^{th} frame 70^{th} frame 80^{th} frame

90^{th} frame 100^{th} frame 110^{th} frame 120^{th} frame

130^{th} frame 140^{th} frame 150^{th} frame 160^{th} frame

170th frame 180th frame 190th frame

Figure 6-13 Object tracking results for Foreman sequence

These results are very satisfactory despite the challenging characteristics of the sequence. Although the results are not perfect, the simple approach used, with no user interaction involved during the tracking process, certainly shows the potential of the approach for certain applications. It can be seen that Foreman is moving fast, with his head showing different orientations during the 200-frame length of the sequence. The size of the object, hence the sizes of the regions in the segmentation, are periodically changing due to the zoom in and zoom out effects. This caused some difficulties for our tracking algorithm. Due to these causes, around the 20th, 30th and 130th frames, some falsely identified background regions can be noticed around the helmet region. At the 130th and 140th frames, the identified object contains a background region around the shoulder. Similarly, some missing foreground object parts are also noticeable at the 40th, 80th, 100th, 110th, 120th, 130th, 150th, 170th, etc. around the helmet, whereas at the 160th and 170th, a full face is not appearing. The results illustrate that the algorithm is capable of tracking the "Foreman" object until it disappears at the 190th frame. However, note that this tool currently does not support detection of object-reappearance, were it to occur.

Figure 6-14 shows the object-tracking results for the "Mother and Daughter" sequence. In this sequence, no fast object movements, no camera movements, no zooming effects appear, but an occlusion effect is the major difficulty occurring during the middle of the sequence. This occlusion effect can similarly lead to errors in the motion estimation process. The tracked objects at every 10th frame of the sequence are shown in this figure. The spatial segmentation partitions are created with 400 regions on each frame of the 200-frame long sequence. The same threshold value, i.e. 0.7, is used for the

region classification process. As in the previous experiments, the results correspond to automatic tracking.

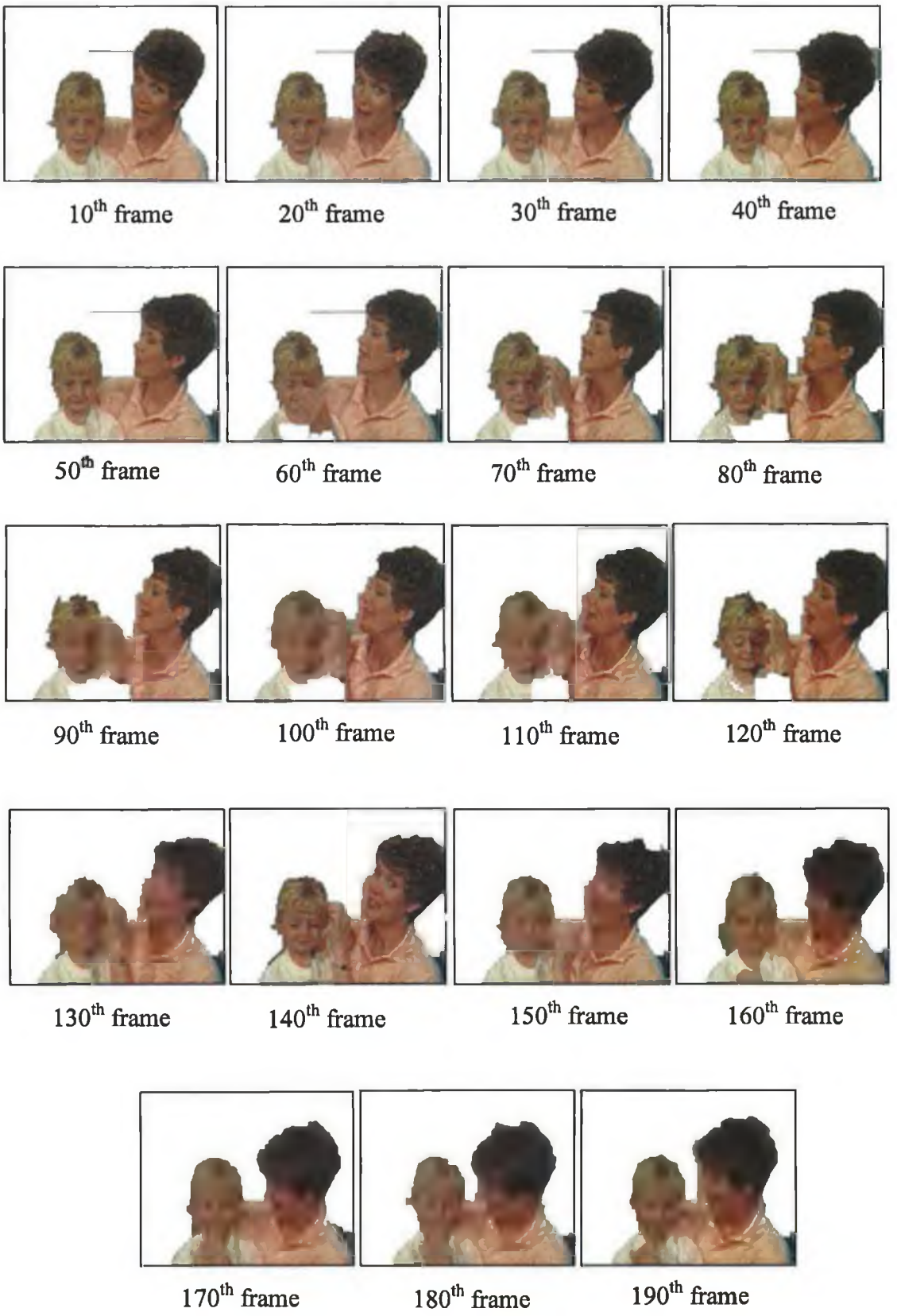


Figure 6-14 Object tracking results for Mother and Daughter sequence

In comparison with the results for the first two sequences, the tracking algorithm has not performed very well for this sequence. The main reason is due to the fact that mother in the scene starts to move her arm causing occlusion. However, the results are still satisfactory. The inability to cope with occlusion, which starts to appear at the 57^{th} frame, has resulted in larger holes present in the segmented object. These effects can be noticed from the 60^{th} frame to the 120^{th} frame. This particular shortcoming of the system necessitates a different approach for correcting the appearances of large holes in the segmentation. The hole-filling technique used in the tool has not been able to cope with such large holes. To this end, solutions can be proposed in three ways. First, as is illustrated in the sequel of this section, a large structuring element size can be used for hole-filling. Second, a bi-directional motion estimation method could be useful for more accurate motion estimation/ compensation, and hence reduce the errors in the entire segmentation process. Third, user interaction can be provided at the start of such errors to redefine the object. It can also be seen that some other errors have occurred during the segmentation process due to falsely identified regions. Some of these errors are due to the insufficient number of regions present in the spatial segmentation to achieve a sufficient granularity, as previously illustrated in **Figure 6-11**. The reason for this is because of varying image content, which is happening over the time period of the sequence. This necessitates a higher number of regions to be present in some of the subsequent frames. This is one of the limitations in our approach: how to perform this task dynamically? Limitations also occur due to the use of a simple region-classification step, which classifies regions as per a "yes or no" decision using a pre-defined threshold. Moreover, this threshold is dependent on the number of regions spatially created in the image.

The tracking results for the "Children" sequence is presented in **Figure 6-15**. In this experiment, only one object is considered, i.e. the left child. In this sequence, object movements such as the children bending and then standing are present, while the text advertisement "Welcome to MPEG World" passes over the objects during the middle of the sequence causing occlusion. Due to the presence of these difficult events, this can be considered as the most difficult sequence to segment in our experiments. The tracked objects at every 6^{th} frame of the sequence are shown in this figure. The spatial segmentation partitions are created with 800 regions on each frame of the 50-frame long sequence. The same threshold value, i.e. 0.7, is used for the region classification

process. Similar to the previous experiments, the object tracking process is completely automatic.

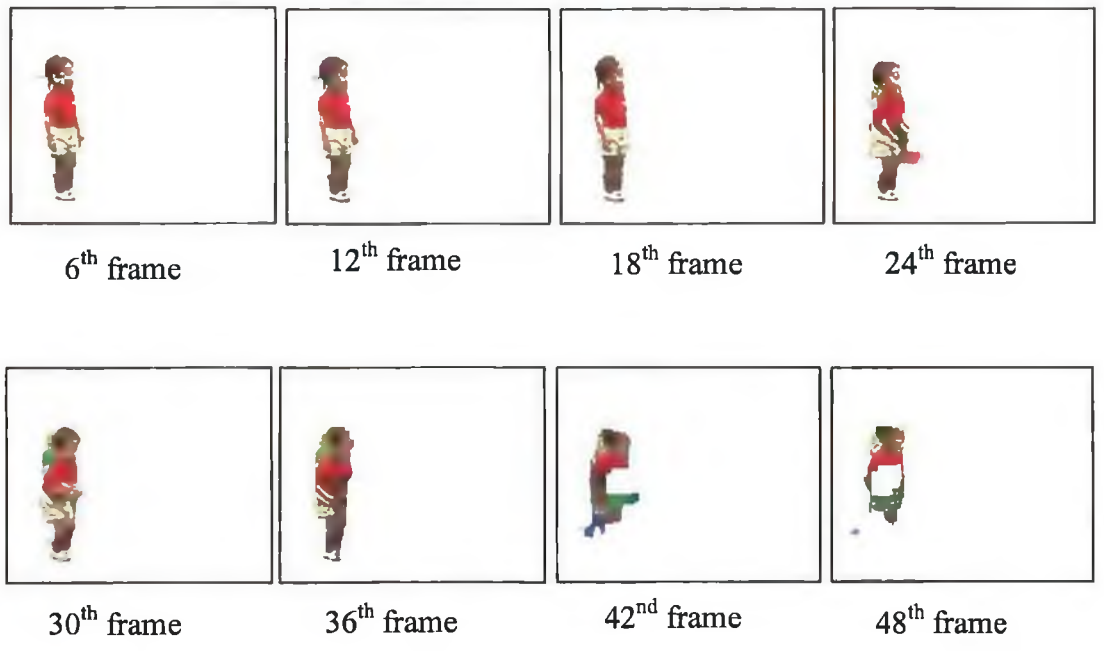


Figure 6-15 Object tracking results for Children sequence

These results again do not show a promising performance of segmentation in the presence of occlusion in a manner similar to the Mother and Daughter sequence. The occlusion occurs when the advertisement text moves and covers the left child, leading to errors in the motion estimation process. However, our algorithm has been able to track the object's basic movements while some errors occur mainly due to the use of the simple region-classification process. Nevertheless, this is certainly a very difficult sequence to segment.

Some important measures to be taken in manipulating our object-tracking method are the assignments of correct parametric values to the number of regions in the spatial segmentation partition and to the region-classification threshold. The objective of the following case studies is to illustrate the effects of these two parameters, and to show how they are crucial to achieve better segmentation results. To this end, only the "Foreman" sequence is considered. This sequence is chosen because it has produced satisfactory results for the values of 600 and 0.7 for these parameters, as previously

illustrated in **Figure 6-13**. In order to keep the illustration aspects short, results are shown only on every 20th frame of the sequence.

Figure 6-16 illustrates the results for the "Forman" sequence with a different value assigned for the region classification threshold. It is set to 0.6 instead of 0.7, but the number of regions in the spatial segmentation partition is kept at 600.

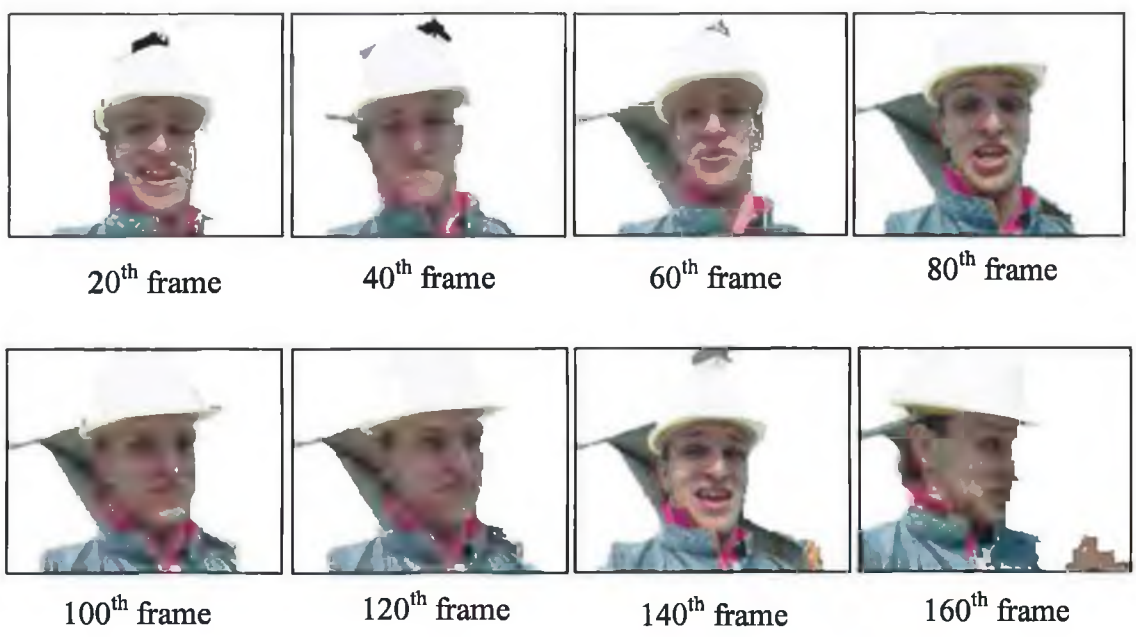


Figure 6-16 Object tracking results for Foreman sequence with 0.6 threshold value

In comparison with the results in **Figure 6-13**, this experiment proves that a smaller threshold value has caused segmentation errors with incorrectly classified regions. It is clear that the foreground object is connected with some of the larger background regions. It is certain from this experiment that the region-classification process has resulted in these errors due to the fact that the chosen region-classification threshold (60%) is too small for this sequence.

The following experiment is to observe the algorithmic behaviour with a higher threshold value of 0.8, keeping the number of regions value still at 600. **Figure 6-17** shows the results obtained from this experiment.

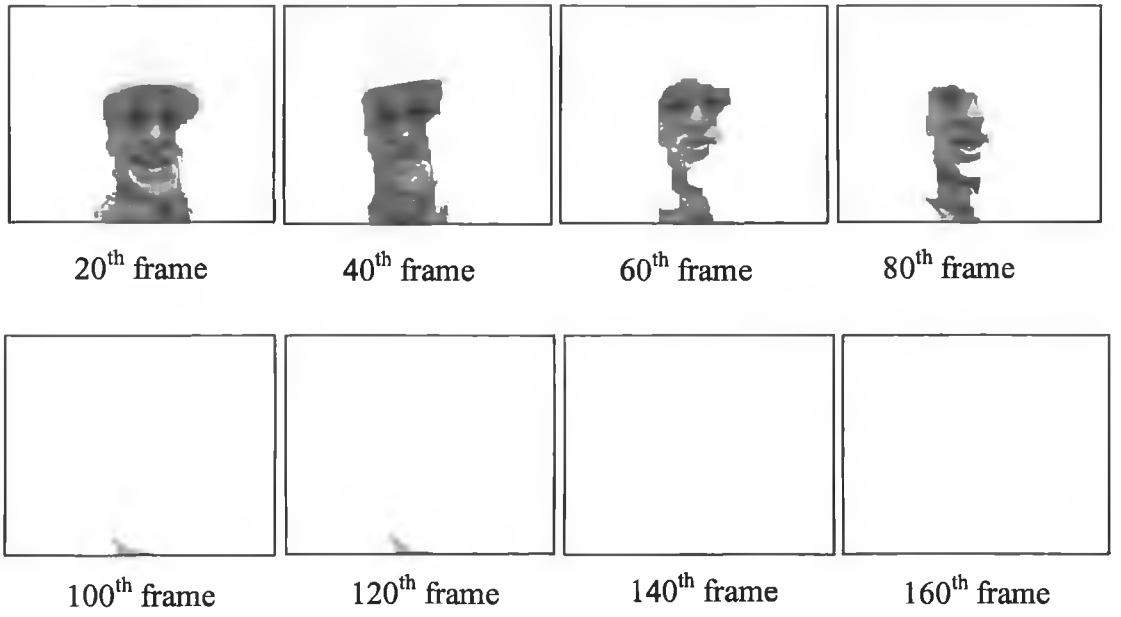


Figure 6-17 Object tracking results for Foreman sequence with 0.8 threshold value

It can be seen that the object being tracked is gradually disappearing as the tracking is progressing, illustrating poor results. Nothing at all is identified at the 160th frame since the algorithm has not been able to keep the correct object intact throughout the sequence. The cause of this situation is due to a use of too high a threshold value in the region-classification process, leading progressively to missing regions. The next case study is performed to observe how the region-classification threshold is dependent on the number of regions in the spatial partition.

Figure 6-18 illustrates the results obtained for the same sequence but for a threshold of 0.8 and 900 regions. These results are similarly shown on every 20th frame of the sequence.

Figure 6-18 Object tracking results for Foreman sequence with 0.8 threshold value and 900 region spatial segmentation

It is clear that results are still not satisfactory. However, the objective of this experiment is to evaluate results under the different parameters described above. This set of results can be claimed to be more accurate than the results shown in **Figure 6-17**, but their quality is still below that of the results shown in **Figure 6-13**. The nature of this behaviour proves that the success of this tracking algorithm is dependent on these two parameters. Hence, the biggest limitation, which still remains unsolved in our system, is how the user can decide the values for these two parameters.

One of the remedial actions for removing large holes present in the segmentation is illustrated below by using larger structuring elements in the hole-filling stage. The "Mother and Daughter" sequence is considered for this case study to evaluate the success of this method over the results presented in **Figure 6-14**. For better elaboration of its performance, only the 60^{th} to 130^{th} frames are used, and the obtained results are presented in **Figure 6-19**. These results too correspond to a fully automatic tracking event, running from the 1^{st} frame to the 200^{th} frame of the sequence, with the same

parametric values used for the number of regions and the threshold (400, 0.7) as in the previous case.

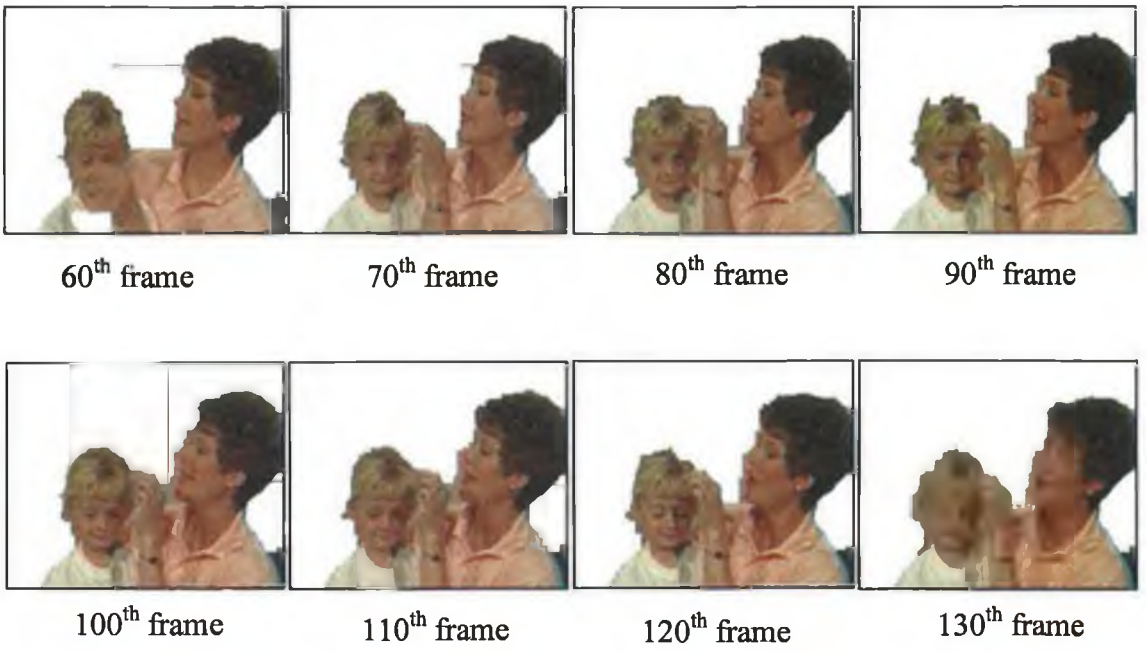


Figure 6-19 Object tracking results for Mother and daughter sequence with 15×15 structuring set for hole-filling

In comparison with the results presented in **Figure 6-14,** tracking performance has been greatly improved. A hole appearing case can be seen only on the 60th frame. The correction was due to the use of a larger structuring set, i.e. 15×15, for morphological filtering. In addition to the previously described two interdependent parameters, this case study proves that final results are also dependent on this particular filtering parameter used in the hole-filling stage.

6.4 Discussion

Results were presented for object segmentation both in the spatial and temporal domain. The spatial object-segmentation stage is more or less a complete implementation compared to the object-tracking stage. This means that a considerable amount of work is left as further improvement to be incorporated into the object-tracking system. The most important feature of the spatial object-segmentation stage is the user interactivity. The feature of mouse clicking certainly adds benefits as it helps to produce a more accurate segmentation, despite the fact that it is a time consuming

and cumbersome task to manipulate in some cases. Ideally, for MPEG-4 and MPEG-7 types of applications, this challenging user interaction can be preferable if it can be restricted to only one "mouse drag" on the desired object. If so, it naturally needs more intelligence to be incorporated into the supervised segmentation approaches.

In the tracking process, the results exhibit various distortions at the boundaries. Estimation of the motion results in errors due to shading effects, zooming effects, deformation effects, occlusion problems, etc. The main limitation of our tracking technique is its dependency on many parameters. As was illustrated in the results of the previous section, tracking results are highly dependent on parameters such as number of regions in the spatial segmentation, the region-classification threshold, and the filter size used in the hole-filling step. Moreover, two parameters, i.e. the initial number of regions and the region-classification threshold, act as interdependent parameters. The difficulty of these issues arises when the user has no knowledge of the sequence to be segmented.

The factors causing most of the errors are due to two main reasons. First, the motion estimation process needs to be improved since the block-based motion estimation does not provide accurate results in all cases. Use of a more advanced motion model, such as affine/polynomial motion model, could help to improve the overall performance of our system. Second, the simple region-classification technique used in this system causes certain errors. Classifying regions by a "yes or no" decision is too simple, and can cause substantial errors in the segmentation process. Therefore, this step could also be replaced with a more sophisticated technique if further accuracy needs to be achieved.

If the performance of the tracking technique is to be completely assessed, it is then necessary to discuss the temporal coherency of the tracked objects throughout the sequence using an objective evaluation criterion. However, an objective quality measure step is not considered in our discussion due to the wide scope it introduces within the context of this thesis. Since our work was mostly concentrated on the interactive spatial object-segmentation, the second stage (i.e., object tracking) is still open for some more improvements.

7. CONCLUSION

7.1 Overview

The results and discussions presented in the previous chapter have produced some valuable inputs to ongoing research on semantic video object segmentation. This chapter is concerned with some conclusions that can be drawn within the context of our research. In this respect, this chapter is organised in the following manner. First, a review of the entire work carried out is outlined. This review consists of a summary of the research undertaken, and how the problem has been addressed. Secondly, some recommendations for future research towards semantic video object segmentation are presented. In particular, some potential research work that can be considered as extensions to the current system are highlighted.

7.2 Review of the work carried out

Our research work dealt with video object segmentation of both still images and image sequences, the latter being also referred to as object tracking in this thesis. This research is conducted both on automatic segmentation techniques and manual segmentation techniques. A combination of these two methods has enabled us to work towards an object segmentation tool in the form of a semi-automatic object segmentation approach. Our main emphasis in this research is on how user interaction can be performed on generic object-segmentation tasks. Keeping the required user interactivity and the overall system flexibility in mind, the entire system is developed in the Java programming language.

This system is designed in two stages: spatial object segmentation and object tracking. It was decided to perform the spatial object segmentation in a supervised manner in order to make the system suitable for generic applications. It uses a combination of a conventional segmentation algorithm, RSST, BPT, and user interaction. The object tracking process can be either supervised or unsupervised depending on the accuracy of the tracked segmentation results. In chapter 6, the effectiveness of our approach is shown both in the spatial and temporal domains.

The first phase of the system is developed as a user-friendly object-based still image segmentation tool. This stage consists of running two automatic algorithms to generate

a hierarchical binary tree structure of homogeneous regions with a pre-defined granularity. User guidance is provided at this stage to help the computer in forming objects that are otherwise not recognisable to it. In this phase, the task requiring most time is user interaction, which is used to selectively group regions into objects. A region correction/refinement step is also provided using a region-splitting technique. In the current system, the accuracy of the manually created regions is dependent on the accuracy of the user drawn contour.

The second phase of the system is developed as an extension to the first phase to facilitate video segmentation. The object-tracking algorithm developed for this purpose is based on a region-based tracking technique (see section 4.3.2), which allows identifying a pre-defined object in subsequent frames of the sequence. When the object tracking is in progress, the tracked results are constantly displayed in a separate window in order to present results to the user for better judgment of the tracking performance. The system is designed for listening to interrupt events so as to stop currently running tracking processes, if necessary. Hence, during the tracking process, the user has the flexibility to re-initialise the object and restart the tracking process if the tracked results begin to show errors. However, in order to improve the stability of this tracking system, there are certain aspects which need to be further addressed, which otherwise would generate variable quality results for different users. The main reason behind this lack of stability is the use of many dependent parameters that need to be pre-defined. In this respect, some of the solutions that could be useful to improve the system-performance are described in section 7.3. The object-tracking process is also relatively slow in our system due to the presence of intensive computations and the chosen programming language.

Overall, the effectiveness of our system is due to several reasons. Since the number of regions in the initial segmentation is specified by the user, it is straightforward to generate an initial segmentation partition of the required granularity. Similarly, RSST is an efficient and relatively easy algorithm to implement. The BPT is also a relatively simple approach to segmentation representation. Introducing it into this work makes the region-browsing process fast and efficient. In general, identifying an interesting semantic object in a video sequence is heuristic and is a challenging task. However, the relatively simple tracking technique used in this tool is capable of overcoming some of the difficult tracking obstacles due to use of a region-based approach.

The results illustrated in chapter 6 demonstrate the usefulness of our tool for MPEG-4 and MPEG-7 type applications. For MPEG-4 video encoding, the tracking functionality of this tool enables to produce a video sequence in the form of a set of VOPs. In the context of MPEG-7, the tool is useful for describing still pictures by using our interactive spatial segmentation approach. The feature of user interactivity adopted within this tool facilitates the users in extracting any number of objects in a single run of automatic algorithms. This is due to the fact that our system performs the automatic region-based segmentation first, and the user then manually carries out object extraction. Thus, it allows extraction of any number of objects as per users interest. Furthermore, the second component of the system, i.e. tracking, could also be used in the future for extracting motion features with certain additions to the current system.

The above summarised research work along with the background reading carried out during the entire period of the research is documented in seven main chapters in this thesis. Of these seven, the first chapter is devoted to an introductory description of the research with sections describing the research objectives and the thesis structure. The second chapter focuses on a study of still image and moving picture coding standards. A review of such different standards has provided us not only a better understanding of the evolution of the MPEG standards but also why the research undertaken is so important in the context of later standards, such as in MPEG-4 and MPEG-7. Some important aspects of block-based compression and segmentation-based compression, colour spaces, and video formats are also described. Moving more towards the actual research area, image segmentation is presented in chapter 3. Starting with simple notions such as regions vs objects, automatic vs semi-automatic segmentation, basic segmentation techniques are first presented. These then follow more specific descriptions of region-based segmentation techniques, such as watershed and RSST. The Binary Partition Tree technique is discussed as a hierarchical segmentation representation method. The first stage of the implemented system is primarily covered by the concepts discussed within this chapter. The theoretical aspects of object tracking techniques are presented in chapter 4. The object tracking is the second component of our system for carrying out segmentation of video. In this chapter, the main tracking techniques, such as edge-based and region-based, are described in order to compare region-based tracking approaches with other available approaches. In the fifth chapter, the proposed video object segmentation tool is presented. The system overview is first described. The GUI is presented in section 5.3. The two main components of the

system, i.e. initial object segmentation and object tracking, are then described. The subsequent discussions in this chapter focus on the interactive features supported within the GUI and the software implementation modules. The sixth chapter is devoted to presenting the results obtained from our experiments. The various sets of results are presented along with discussions in order to demonstrate the effectiveness of the implemented system. Hence, some of the advantages and disadvantages of our system are clearly remarked while presenting some of the necessary improvements to be incorporated in the future.

7.3 Recommendations for future work

In this section, some recommendations for future work are presented. Some of these recommendations are short-term implementation issues whilst others need to be considered in a longer-term project. Future work is concerned with the idea of bringing the current system to a more robust and accurate semantic video object segmentation system, keeping user interaction issues also in mind.

The short-term implementation tasks can be categorised as:

- Extend the single object tracking approach to a multiple object tracking approach;
- Support of advanced region splitting technique, in case automatically generated results need to be corrected;
- Allow the user to choose merging criterion based on the number of regions or PSNR in creating initial partitions.

The long-term research can be categorised as:

- Carrying out a complementary mouse-clicking feature with a simple mouse-scribble given a region-based segmentation;
- Representing segmented data with suitable visual descriptors;
- Implementing more advanced motion estimation models;
- Investigating advanced region-classification methods.

At present, the segmentation tool facilitates only single object tracking. Nevertheless, in practice, it is necessary to perform multiple object tracking. It is, however, a straightforward alteration in the software to facilitate this feature since it requires only keeping track of all the interesting objects - similar to what is performed on a single object with slight changes in some parts of the software modules.

In the current system, manual corrections to automatically generated results are achieved by splitting a selected region according to a user-drawn contour. This method does not always fulfil the requirement of region splitting with the required boundary accuracy, as accurate mouse drawing is, in most cases, very difficult and requires a substantial amount of effort on behalf of the user. Thus, an automatic technique is necessary to properly split a user-selected region into two whilst reducing the workload of the user. Two approaches are possible: the first being an algorithm supported by an approximate user-drawn contour for initiating the splitting process, and the second being an approach to perform the task entirely automatically by considering distinct features within the region of interest. In the first method, given an approximate contour, it can be performed by defining "IN" and "OUT" boundaries as a search area to find the correct object boundary, as carried out in [Gu and Lee 1998]. In the second method, if contours were not to be introduced, it would also be possible to split a particular region into two finer regions by applying the RSST algorithm in a slightly different manner to that of its conventional implementation. This can be done by constraining the RSST algorithm to produce only two regions (i.e. initial number of regions should be set to 2).

The merging criterion used in a bottom-up segmentation method can be based on either number of regions or a PSNR measure. Due to the use of RSST, which in its original algorithm operates based on the number of regions; our system is also currently based on the same merging criterion. As a result, it is sometimes difficult to ensure the segmentation quality according to a desired application. However, RSST could define the merging criterion in terms of a PSNR quality criterion. This can be introduced as an additional option to the user. However, this slightly increases the computational complexity of the overall algorithm since it requires keeping track of the segmentation quality changes during the process of region merging. With the availability of this option, the user can select an appropriate merging criterion as he/she wishes before starting the segmentation process.

The main form of user interaction involved in the spatial segmentation process is based on mouse-clicks. Depending on the number of regions, which constitute the object of interest, a certain number of mouse-clicks needs to be made during the object formation process. This number can sometimes be unnecessarily large since incorrect mouse-clicks may happen in the process of region selection, depending on how familiar the user is with the system and how complex the segmented image is. However, the lower limit of this number is always equal to the number of distinct regions in the object. The results shown in **Table 6.1** correspond to a case of the minimum number discussed above (mistakenly selected regions are not counted). Therefore, one of the goals of our research is how this minimum number can be reduced. In certain instances, the number would be large as illustrated in **Table 6.1**, but in other instances, it would realistically be an appropriate approach in a semi-automatic segmentation environment. Nevertheless, in order to be able to set the goals for generic and real-time applications, it would be necessary to have much simpler user interaction, undoubtedly at an extra cost of computation. To this end, a more user-friendly solution could be to have only a single scribble (in case of a single object extraction), leaving the computer to perform the rest. A simpler approach to this task is to provide two labels with two scribbles (foreground and background) to carry out object segmentation, involving little extra user interaction. However, if such a feature is to be introduced in our system, more research needs to be carried out in order to find a suitable metric for grouping regions from pre-segmented images. Such a metric could initially be based on a simple threshold-operation/classification technique by combining regions' colour features and spatial features. This could further lead to use of more advanced techniques based on the Expectation Maximisation (EM) algorithm [Feder and Weinstein 1988], [O'Connor and Marlow 1998], albeit with an increase of computation load.

According to the MPEG-7 normative tools, a semi-automatic segmentation system needs to describe the content using relevant descriptors. Both low-level descriptors (mostly extracted automatically) and high-level descriptors (mostly extracted manually) are used to describe a piece of multimedia content. In our system, representing information in low-level descriptors is straightforward, however, incorporation of high-level descriptors for semantically describing objects and events requires further work. The details of the colour and texture descriptors, shape descriptors, and motion descriptors, which have already been standardised as low-level visual descriptors, can

be found in [Manjunath *et al.* 2001], [Bober 2001], [Jeannin and Divakaran 2001] respectively.

The motion estimation model certainly plays an important role in the overall accuracy of the segmentation. Thus, it is necessary to compare more advanced motion models with block matching motion algorithms. It is believed that the affine/polynomial motion model can provide more accurate results in the presence of not only translation but also rotation, divergence, and shear. In this respect, it can be expected that the entire segmentation tool can be made more robust if such a motion model can be introduced to the system.

Another noteworthy aspect of future work required can be mentioned as some in-depth research on region-classification techniques to be used in the object-tracking step. Some of the results illustrated in the previous chapter have proved that the region-classification stage is certainly in need of more improvements. Instead of reaching a yes/no decision straightaway it would be the best to reconsider any uncertain regions for better accuracy of results.

REFERENCES

[Alatan *et al.* 1997] S. A. Alatan, E. Tuncel, and L. Onural, "A Rule-based Method for Object Segmentation in Video Sequences," International Conference on Image Processing, 1997.

[Blake and Isard 1998] A. Blake and M. Isard, Active Contours, Springer-Verlag New York, 1998, ISBN: 3540762175.

[Bober 2001] M. Bober, "MPEG-7 Visual Shape Descriptors," IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 6, pp. 716-719, 2001.

[Boliek *et al.* 2000a] M. Boliek, C. Christopoulos, and E. Majani, "JPEG 2000 Part I Final Committee Draft Version 1.0," ISO/IEC JTC1/SC29 WG1 N1646R, March 2000.

[Boliek *et al.* 2000b] M. Boliek, E. Majani, J. S. Houchin, J. Kasner, and M. L. Carlander, "JPEG 2000 Part II Final Committee Draft", ISO/IEC JTC1/SC29 WG1 N2000," December 2000.

[Brady and O'Connor 1996] 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.

[Brady *et al.* 1997] N. Brady, F. Bossen and N. Murphy, "Arithmetic Encoding of 2D Shape Sequences," Proc. IEEE International Conference on Image Processing (ICIP'98), Oct. 1997.

[Brazakovic and Neskovic 1993] D. Brazakovic and M. Neskovic, "Mammogram Screening using Multiresolution-based Image Segmentation," International journal of pattern recognition and artificial intelligence, vol. 7, no. 6, pp. 1437-1459.

[Castagno *et al.* 1998] R. Castagno, T. Ebrahimi and M. Kunt, "Video Segmentation based on Multiple Features for Interactive Multimedia Applications," IEEE

Transactions on Circuits and Systems for Video Technology, vol. 8, no. 5, pp. 562-571, September 1998.

[Christopoulos *et al.* 1999] C. Christopoulos, J. Askelof and M. Larsson, "Efficient Method for Encoding Regions of Interest in the Upcoming JPEG2000 Still Image Coding Standard," IEEE Signal Processing Letters, vol. 7, no. 9, pp. 247-249, September 1999.

[Christopoulos *et al.* 2000] C. Christopoulos, A. Skodras, and T. Ebrahimi, "TheJPEG2000 still Image Coding System: An Overview," IEEE Transactions on Consumer Electronics, vol. 46, no. 4, pp. 1103-1127, November 2000.

[Cohen 1991] L. D. Cohen, "On Active Contour Models and Balloons," Computer Vision Graphics and Image Processing: Image Understanding, pp. 211-218, August 1991.

[Cooray *et al.* 2001] S. Cooray, O'Connor N., Marlow S., Murphy N., and Curran T., "Hierarchical Semi-automatic Video Object Segmentation for Multimedia Applications," SPIE ITCom'01, August 2001.

[Correia and Pereira 2001] P. Correia and F. Pereira, "Standalone Objective Evaluation of Segmentation Quality," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS'01), May 2001.

[Doulamis *et al.* 1999] N. D.Doulamis, A. D. Doulamis, K. Ntalianis and S. D. Kollias, "Intelligent Techniques for Image Sequence Analysis: Towards Semantic Video Object Segmentation," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), pp. 21-24, June 1999.

[Efford 2000] Efford Nick, Digital Image Processing: Practical Introduction using Java, Pearson Education Limited, 2000, ISBN 0-201-59623-7.

[Fan *et al.* 2001] J. Fan, D. K. Y. Yau, A. K. Elmagarmid, W. G. Aref, "Automatic Image Segmentation by Integrating Colour Edge Extraction and Seeded Region Growing," IEEE Transactions on Image Processing, vol. 10, no. 10, pp. 1454-1466, October 2001.

[Feder and Weinstein 1988] M. Feder and E. Weinstein, "Parameter Estimation of Superimposed Signals using the EM Algorithm," IEEE Transactions on Accoustic, Speech, and Signal Processing, vol. 36, no. 4, pp.477-489, April 1988.

[Fleury *et al.* 1998] P. Fleury, S. Bhattacharjee, L. Piron, T. Ebrahimi and M. Kunt, "MPEG-4 Video Verification Model: A Solution for Interactive Multimedia Applications," SPIE Journal of Electronic Imaging, vol. 7, no. 3, pp. 502-515, July 1998.

[Garrido and Salembier 1998] L. Garrido and P. Salembier, "Region-based Analysis of Video Sequences with a General Merging Algorithm," In IX European Signal Processing Conference (EUSIPCO), vol. 3, pp. 1693-1696, September 1998.

[Garrido *et al.* 1999] L. Garrido, P. Salembier and J. R. Cases, "Representing and Retrieving Regions using Binary Partition Tree," IEEE Int. Conference on Image Processing (ICIP), October 25-28, 1999.

[Gatica-Perez *et al.* 1999] D. Gatica-Perez, M. Sun and C. Gu, "Semantic Video Object Extraction based on Backward Tracking of Multivaluaed Watershed," International Conference on Image Processing (ICIP), pp. 145-149, October 1999.

[Gatica-Perez *et al.* 2000] D. Gatica-Perez, C. Gu and M. Sun, "Semantic Video Object Extraction using Four-band Watershed and Partition Lattice Operators," IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 5, pp. 603-618, May 2001.

[Goldenberg *et al.* 2001] R. Goldenberg, R. Kimmel, E. Rivlin and M. Rudzsky, "Fast Geodesic Active Contours," IEEE Transactions on Image Processing, vol. 10, no. 10, pp. 1467-1475, October 2001.

[Gonzalez and Woods 1992] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Addison-wisely publishing company, 1992, ISBN 0-201-50803-6.

[Gu and Lee 1997] C. Gu and M. Lee, "Semantic Video Object Segmentation and Tracking using Mathematical Morphology and Perspective Motion Model," International Conference on Image Processing (ICIP), vol. II, pp. 514-517, 1997.

[Gu and Lee 1998a] C. Gu and M. C. Lee, "Semiautomatic Segmentation and Tracking of Semantic Video Objects," IEEE Transactions on Circuits and Systems for Video Technology, vol. 8, no. 5, pp. 572-584, September 1998.

[Gu and Lee 1998b] C. Gu and M. Lee, "Semantic Video Object Tracking Using Region-based Classification," International Conference on Image Processing (ICIP'98), pp. 643-647, 1998.

[Gueziec 2002] A. Gueziec, "Tracking pitches for Broadcast Television," IEEE Computer- Human Interaction, pp. 38-43, March 2002.

[Grosbois *et al.* 2001] R. Grosbois, D. Santa-Cruz and T. Ebrahimi, "New Approach to JPEG 2000 Compliant Region of Interest coding," Proc. of the SPIE 46th annual meeting, Applications of Digital Image Processing XXIV, July-August 2001.

[Herve 1997] B. Herve, Digital Television MPEG-1, MPEG-2 and Principles of the DVB System, New York, NY 10158-0012, 1997, ISBN 0 340 69190 5.

[Hunter 2001] J. Hunter, "An overview of the MPEG-7 Description Definition Language (DDL)," IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 6, pp. 765-772, 2001.

[Huttenlocher *et al.* 1993] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing Images using the Hausdorff Distance," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, no. 9, September 1993.

[Jeannin and Divakaran 2001] S. Jeannin and A. Divakaran, "MPEG-7 Visual Motion Descriptors," IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 6, pp. 720-724, 2001.

[Kass *et al.* 1987] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active Contour Models," International Journal of Computer Vision, pp. 321-331, 1987.

[Koenen 2001] R. Koenen, "Overview of the MPEG-4 standard," ISO/IEC JTC1/SC29/WG11 N4030, March 2001.

[Kompatsiaris and Strintzis 1999] I. Kompatsiaris and M. G. Strintzis, "Higher Order Segmentation and Tracking of Objects for Visualization of Videoconference Image Sequences," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), 1999.

[Kruse *et al.* 1999] S. Kruse, A. Graffunder and S. Askar, "A New Tracking Scheme for Semi-Automatic Video Object Segmentation," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), pp. 93-96, June 1999.

[Loutas *et al.* 2001] E. Loutas, K. Diamantaras and I. Pitas, "Occlusion Resistant Object Tracking," International Conference on Image Processing (ICIP), pp. 65-68,October 2001.

[Manjunath *et al.* 2001] B. S. Manjunath, J. Ohm, V. V.Vasudevan, A. Yamada, "Color and Texture Descriptors," IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 6, pp. 703-715, 2001.

[Manjunath and Chellappa 1991] B. S. Manjunath and R. Chellappa, "Unsupervised Texture Segmentation using Markov Random Field Models," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 5, pp. 478-482, May 1991.

[Marcotegui *et al.* 1999] B. Marcotegui, F. Zanoguera, P. Correia, R. Rosa, F. Marques, R. Mech and M. Wollborn, "A Video Object Generation Tool Allowing Friendly User Interaction," International Conference on Image Processing (ICIP), October 1999.

[Marques and Llach 1998] F. Marques and J. Llach, "Tracking of generic objects for video object generation," International Conference on Image Processing (ICIP'98), pp. 628-632, 1998.

[Marques and Molina 1997] F. Marques and C. Molina, "Object Tracking for Content-based Functionalities," SPIE Visual Conference on Image Processing (VCIP), vol. 3024, pp. 190-198, 1997.

[Marques 1996] F. Marques, "Temporal Stability in Sequence Segmentation using the Watershed Algorithm," Mathematical Morphology and its Applications to Image and Signal Processing, pp.321-328, May 1996.

[Marques *et al.* 1996] F. Marques, B. Margotegui, F. Meyer, "Tracking Areas of Interest for Content-based Functionalities in Segmentation-based Video Schemes," IEEE International Conference on Acoustic, Speech and Signal Processing, vol. 2, pp. 1224-1227, May 1996.

[Martinez 2001] J. M. Martinez, "Overview of the MPEG-7 standard," ISO/IEC JTC1/SC29/WG11 N4509, December 2001.

[Matsuo and Fujimoto 1997] M. Matsuo, H. Fujimoto, "A Programmable Video Codec System for Low Bit rate Communication," IEEE Transactions on Consumer Electronics, vol. 43, no. 3, pp. 903-910, August 1997.

[Mech and Marques 2001] R. Mech and F. Marques, "Objective Evaluation Criteria for 2D-shape Estimation Results of Moving Objects," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS'01), May 2001.

[Meier and Ngan 1998] T. Meier and K. N. Ngan, "Video Object Plane Extraction for Content-based Functionalities in MPEG-4," International Workshop on Very Low Bitrate Video Coding (VLBV'98), pp. 121-124, October 1998.

[Morier *et al.*1997] F. Morier, J. Benois-Pineau, D. Barba, and H. Sanson, "Hierarchical Segmentation of Video Content", Picture Coding Symposium, ITG-Fachberichte, Issue 143, pp. 271-275, 1997.

[Morris *et al.* 1986] O. J. Morris, M. J. Lee and A.G. Constantinides, "Graph Theory for Image Analysis: An Approach Based on the Shortest Spanning Tree," IEE Proceedings, vol. 133, no. 2, pp. 146-152, April 1986.

[Moscheni *et al.* 1996] F. Moscheni, F. Dufaux, and M. Kunt, "Object Tracking Based on Temporal and Spatial Information," International Conference on Speech and Signal processing, vol. 4, pp. 1914-1917, May 1996.

[Mulroy 1997] J. Mulroy, "Video Content Extraction: Review of Current Automatic Segmentation Algorithms," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), June 1997.

[Molloy and Whelan 2000] D. Molloy and P. Whelan, "Active Meshes," Pattern Recognition Letters, 21(12), pp. 1071-1080, 2000.

[Nelson and Gailly 1995] M. Nelson and J. Gailly, The Data Compression Book- 2nd Edition, M and T Books, New York, 1995, ISBN 1-55851-434-1.

[Netravali and Haskell 1995] A.N. Netravali and B.G Haskell, Digital Pictures: Representation, Compression and Standards- 2nd Edition, Holmdel, NJ, 1995.

[Ngan *et al.* 1999] K. N. Ngan, T. Meier and D. Chai, Advanced Video Coding Principles and Techniques, Elsevier Science, 1999, ISBN 0 444 82667 X.

[O'Connor and Marlow 1998] 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.

[O'Connor *et al.* 1997] N. O'Connor, N. Brady, S. Marlow, "Supervised Image Segmentation using EM-Based Estimation of Mixture Density Parameters," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), June 1997.

[Palacious and Hedley 1994] M. Palacious and M. Hedley, "3D Colour Histogram Clustering using Scale Space," Proceedings of the APRA Colour Imaging Workshop, Canberra, pp.141-144, 1994.

[Park and Ra 1998] H. S. Park and J. B. Ra, "Homogeneous Region Merging Approach for Image Segmentation Preserving Semantic Object Contours," Proc. of International Workshop on Very Low Bit rate Video Coding, pp. 149-152, Oct. 8-9, 1998.

[Pateux 2000] S. Pateux, "Tracking of Video Objects using a Backward Projection Technique," SPIE Visual Conference on Image Processing (VCIP), pp. 1107-114, June 2000.

[Peak *et al.* 1999] S. Peak, A. Benitez and S. Chang, "Self-Describing Schemes for Interoperable MPEG-7 Multimedia Content Descriptions," Symposium on Electronic Imaging: Visual Communication and Image Processing, IST/SPIE, January 1999.

[Puri and Chen 2000] A. Puri, T. Chen, Multimedia Systems, Standards and Networks, New York, Basel, 2000, ISBN 0-8247-9303-X.

[Rao and Hwang 1996] K. R. Rao and J. J. Hwang, Techniques and Standards for Image, Video and Audio Coding, Prentice Hall PTR, New Jersey 07458, 1996.

[Sadka 2002] Abdul H. sadka, Compressed Video Communications, John Wiley & Sons Ltd., 2002, ISBN: 0 470 843128.

[Salembier 2001] P. Salembier, "Overview of the MPEG-7 Standard and of Future Challenges for Visual Information Analysis," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS'01), pp. 75-82, May 2001.

[Salembier and Garrido 2000] P. Salembier and L. Garrido, "Binary Partition Tree as an Efficient Representation for Image Processing, Segmentation and Information Retrieval," IEEE Transactions on Image Processing, vol. 9, no. 4, pp. 561-576, April 2000.

[Salembier and Marques 1997] P. Salembier and F. Marques, "Segmentation-based Video Coding System Allowing Manipulation of Objects," IEEE Transactions on Circuits and Systems for Video Technology, vol.7, no.1, February 1997.

[Salembier and Marques 1999] P. Salembier and F. Marques, "Region-based Representations of Image and Video: Segmentation Tools for Multimedia Services," IEEE Transactions on Circuits and Systems for Video Technology, vol. 9, no. 8, pp. 1147-1169, December 1999.

[Salembier and Pardas 1994] P. Salembier and M. Pardas, "Hierarchical Morphological Segmentation for Image Sequence Coding," IEEE Transactions on Image Processing, vol. 3, no. 5, September 1994.

[Salembier *et al.* 1996] P. Salembier, P. Brigger, J. R. Casas, and M. Pardas, "Morphological Operators for Image and Video Compression," IEEE Transactions on Image Processing, vol. 5, no. 6, June 1996.

[Santa-Cruz *et al.* 2000] D. Santa-Cruz, T. Ebrahimi, J. Askelof, M. Larsson and C. Christopoulos, "JPEG 2000 Still Image Coding versus Other Standards," Proc. of the SPIE's 45th annual meeting, Applications of Digital Image Processing XXIII, vol. 4115, pp. 446-454, Jul 30- Aug. 4, 2000.

[Siggelkow and Grigat 1996] S. Siggelkow and R. Grigat," Segmentation of Image Sequences for Object Oriented Coding," ICIP'96, vol.2, pp. 447-480, Lausanne, Switzerland, Sept 1996.

[Smith and Brady 1997] S. M. Smith and J. M. Brady, "SASAN- A New Approach to Low Level Image Processing," International Journal of Computer Vision, pp. 45-78, May 1997.

[Sonika *et al.* 1999] M. Sonika, V. Hlavac and R. Boyle, Image Processing Analysis, and Machine Vision, PWS publishing, ISBN 0-534-95393-X.

[Techmer 2001] A. Techmer, "Contour-based Motion Estimation and Object Tracking for Real-time Applications," IEEE Int. Conference on Image Processing (ICIP), pp. 648-651, October 2001.

[Toklu *et al.* 2000] C. Toklu, A. Teklap and A. T. Erdem, "Semi-automatic Video Object Segmentation in the Presence of Occlusion," IEEE Transactions on Circuits and Systems for Video Technology, vol.10, no.4, June 2000.

[Tuncel and Onural 2000] E. Tuncel and L. Onural, "Utilization of the Recursive Shortest Spanning Tree Algorithm for Video-Object Segmentation by 2-D Affine Motion Modeling," IEEE Transactions on Circuits and Systems for Video Technology, vol. 10, no. 5, pp. 776-781, August 2000.

[Vasudev and Konstantinos 1997] B. Vasudev and K. Konstantinos, Image and Video Compression Standards: Algorithms and Architectures, Kluwer Academic Publishers, 1997, ISBN 0-7923-9952-8.

[Vigus *et al.* 2001] S. A. Vigus, D. R. Bull and C. N. Canagarajah, "Video Object Tracking using Region Split and Merge and a Kalman Filter Tracking Algorithm," International Conference on Image Processing, pp. 650-653, October 2001.

[Vincent 1993] L. Vincent, "Morphological Greyscale Reconstruction in Image Analysis: Applications and Efficient Algorithms," IEEE Transactions on Image Processing, pp. 176-201, April 1993.

[Vincent and Soille 1991] L. Vincent and P. Soille, "Watersheds in Digital Spaces: An Efficient Algorithm based on Immersion Simulations," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 6, June 1991.

[Wang 1998] D. Wang, "Unsupervised Video Segmentation based on Watersheds and Temporal Tracking," IEEE Transactions on Circuits and Systems for Video Technology, vol. 8, no. 5, pp. 539-546, 1998.

[Wallace 1992] G. K. Wallace, "The JPEG Still Picture Compression Standard," IEEE Transaction on Consumer Electronics, vol. 38, no.1, February 1992.

[Whybray *et al.* 1997] M. W. Whybray, P. G. Morrison, P. J. Mulroy, "Video Coding-techniques, standards and applications," BT Journal, no. 4, October 1997.

[Zaletelj and Tasic 2001] J. Zaletelj and J. F. Tasic, "Video Object Segmentation based on Edge Tracking," IEEE Int. Conference on Image Processing (ICIP), pp.813-816, October 2001.

[Zillani and Cavallaro 1999] F. Zillani and A. Cavallaro, "Image Analysis for Video Surveillance based on Spatial Regularization of a Statistical Change Detection," Proc. 10th Int. Conference on Image Analysis and Processing, pp. 1108-1111, 1999.

APPENDIX A – SOME SPATIAL SEGMENTATION RESULTS ON NON-STANDARD IMAGES

Image size: 140X168

Original image | Initial segmentation- 200 regions | Extracted object 1- 1 mouse-click | Extracted object 2- 3 mouse-clicks

Image size: 176X134

Original image | Initial segmentation - 200 regions | Extracted object - 9 mouse-clicks

Image size: 176X142

Original image | Initial segmentation - 20 regions | Extracted object - 2 mouse-clicks

PUBLICATIONS ARISING FROM THIS WORK

1. Saman Cooray, Noel O'Connor, Sean Marlow, Noel Murphy and Thomas Curran, "Semi-automatic Video Object Segmentation using Recursive Shortest Spanning Tree and Binary Partition Tree", Workshop on Image Analysis for Multimedia Services (WIAMIS), pp. 15-18, Tampere, Finland, 16-17 May 2001.

2. Saman Cooray, Noel O'Connor, Sean Marlow, Noel Murphy and Thomas Curran, "Hierarchical Semi-automatic Video Object Segmentation for Multimedia Applications", ITCom'01, SPIE, USA, 19-24 August 2001.