Content-Based Retrieval of Arbitrarily Shaped Video Objects in the Uncompressed and Compressed Domains

by

BERNA EROL

MASc. (Electrical and Computer Engineering), University of British Columbia, 1998

B.Sc. (Control and Computer Engineering), Istanbul Technical University, 1994

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENT FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

September 2001

Department of Electrical and Computer Engineering

The University of British Columbia

Vancouver, Canada

Date: December 10th, 2001

Abstract

Advancements in video object segmentation technology and the availability of efficient object-based video representations, such as MPEG-4 [1], have resulted in the increased availability of arbitrarily shaped digital video content. While this enables many exciting applications, the process of locating and accessing a desired video sequence can still be challenging because of the large volume of data associated with even compressed video.

This dissertation proposes generic methods for the retrieval of arbitrarily shaped video objects in the MPEG-4 compressed domain, using their shape, local motion, and color content. Considering that a one-minute long video sequence may contain more than 1,500 frames, summarization of video content is necessary as a first step to efficiently retrieve video. Therefore, we first suggest a method for the summarization of arbitrarily shaped video objects. This is achieved by selecting the temporal instants of video objects –based on their compressed domain shape information– that efficiently represent the objects' salient content.

Next, we propose to extend some well-proven still shape retrieval techniques to retrieve video objects in the compressed domain. We compute the Fourier and ART (Angular Radial Transform) descriptors on the shape approximations obtained from the

MPEG-4 shape coding modes. We also present a method to compute the shape distances between two video objects based on these still shape features.

Unlike in the case of still objects, one of the key features that describe a video object is motion. Classification of video objects by their local motion is addressed in this thesis by presenting three new motion descriptors. These descriptors are computed based on the shape deformations of arbitrarily shaped video, and assume no prior knowledge about the video content.

Color is one of the most widely used low level features in content-based retrieval. In this thesis, we also study efficient color content matching of arbitrarily shaped video, and in particular, color histogram computation in the MPEG-4 compressed domain.

Our experimental results demonstrate that our techniques enable effective and low complexity content-based retrieval. Employing MPEG-4 compressed domain information not only obviates the need for full decompression of the bit stream, hence yielding substantial computational savings, but also allows our techniques to be more robust to segmentation errors.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ACLM	Angular Circular Local Motion
ANMRR	Average Normalized Modified Retrieval Rank
ART	Angular radial Transform
AVO	Audio-Visual Object
BVOP	Bi-directionally coded Video Object Plane
CIBR	Content Based Image Retrieval
DC	Direct Current (The first DCT coefficient)
DCT	Discrete Cosine Transform
DDL	Description Definition Language
FD	Fourier Descriptor
HSV	Hue, Saturation, Value
IDCT	Inverse Discrete Cosine Transform
IVOP	Intra coded Video Object Plane
JPEG	Joint Photographic Experts Group
LPE	Low Pass Extrapolation
MPEG	Moving Picture Experts Group

MTM	Mathematical Transform to Munsell
NMRR	Normalized Modified Retrieval Rank
pdf	probability density function
PVOP	Predictively coded Video Object Plane
QBIC	Query By Image Content
sec	second
VLC	Variable Length Coding
VM	Verification Model
VO	Video Object
VOP	Video Object Plane
XM	eXperimentation Model

Acknowledgments

First of all, I would like to thank Dr. Faouzi Kossentini for his very valuable supervision. My research would not have become a reality without his guidance, encouragement, and commitment. I would also like to express my gratitude to my supervisory committee members Dr. Rabab Ward, Dr. Mabo Ito, Dr. Hussein Alnuweiri, and Dr. Sid Fels, and to my university examiners Robert Woodham, Dr. Jim Little, Dr. Ian Cumming for their valuable feedback and constructive suggestions.

I would like to thank all the members of Signal Processing and Multimedia Group for their friendship and for sharing their valuable technical knowledge with me. I would particularly like to acknowledge my colleagues and friends Adriana Dumitras, Anthony Joch, Shahram Shirani, Michael Gallant, Geoffrey Lefebvre, and Parvin Mousavi for their support and help with the subjective testing. I would also like to extend special thanks to Niranjan Damera-Venkata, Dave Tompkins, Wade Schwartzkopf, and Guner Arslan for making my visit to the University of Texas in Austin most enjoyable. I wish to express my gratitude to Dr. Alen Docef and our lab administrator Mehran Azimi for making life easier with their great technical support.

I would like to thank Arda, for his continuous support and confidence in me and to Carey, for being a great support when writing my thesis and for all his help with proof reading.

Last, I would like to thank my parents and my sisters for being there for me during the hardest times, despite the fact they are at the other end of the world. I would like to dedicate this thesis to the memory of my father, Metin Culcu, who passed away in March 2001, and whom I know is very proud of me right now and smiling down from Heaven.

BERNA EROL

The University of British Columbia

December 2001

Chapter 1

Introduction

"Discovery consists in seeing what everyone else has seen and thinking what no one else has thought."

Albert Szent-Gyorgi (1893-1986)

The availability of low-cost visual data capturing devices has resulted in the creation of visual data at an ever increasing rate. Visual data, even in a compressed format, requires significant storage space. In the past, storage devices were quite expensive and visual databases were quite limited. Nowadays, the availability of affordable storage devices (less than $3 per gigabyte as of 2001) enables the storage of large amounts of visual data. As users have access to large amounts of content available from many sources, such as digital archives, personal and professional databases, the World Wide Web and broadcast data streams, managing this large amount of information becomes very difficult. Considering that the value of the information often depends on how easily it can be found, retrieved, and accessed, there is a need for tools for efficient classification, identification, and retrieval of this information. Because digital video in the uncompressed form has very large storage requirements (50 gigabytes for 30 minutes NTSC resolution video), digital video is mostly available in the compressed forms, such as JPEG [2], JPEG 2000 [3], MPEG-1/2/4 [4][5][1], and H.263 [6]. Therefore, retrieval techniques that enable access and search capabilities directly in the compressed form, without requiring the decompression of the data, is greatly needed.

One common technique for accessing visual data is to associate the visual data with keywords and use text-based search engines to access it. Although this technique has some advantages, such as capturing the semantic meaning, it may not be feasible for large databases since it requires human annotation, which is time consuming and costly. Also, subjective interpretation of the same content by different people may cause some inconsistencies. A picture can mean different things to different people. As Keister put it, "It is not so much that a picture is worth a thousand words, for many fewer words can describe a still picture for most retrieval purposes, the issue has more to do with the fact that those words vary from one person to another" [7].

The alternative to relying on annotation for retrieval is characterizing the visual data with some primitive features that are inherent to visual data itself, such as color, shape, texture, and motion. This technology is referred to as content-based image retrieval (CBIR). Retrieval based on low level visual features can automate the visual data management process and overcome the problem of language mismatch caused by human annotation. Consequently, CBIR plays a crucial role in applications across diverse fields, such as medicine, publishing, design, education, entertainment, crime prevention, architecture, real estate, and broadcasting. One problem associated with CBIR, however, is that attaching semantic meanings to objects using primitive features is very difficult. Nevertheless, in many applications this problem can often be overcome by employing the query by example, where the user presents an image and asks the system to retrieve similar images, instead of entering the query with its semantic meaning.

The problems in content-based visual data retrieval are becoming widely recognized and the search for solutions is an increasingly active area for research and development. Recent research efforts in the content-based access of visual data have led to the development of many commercial and prototype systems [8]-[26] and to the initiation of MPEG-7 [27]-[33], which standardizes a content description interface. An overview of some of the most notable visual retrieval systems is presented in Table 1-1. Though there has been a significant amount of work in the content-based retrieval of still images, the research in video retrieval is still in its infancy and many research areas remain to be explored.

The existing research in the video retrieval area focuses mostly on frame-based video representation, where access to individual objects in a scene is either very limited or non-existent. This is mainly due to the fact that segmentation and identification of objects in a video scene are very difficult and complex tasks. Nevertheless, in the last few years, great advances in this area have led to the development of many state of the art automatic and semi-automatic segmentation algorithms [34]-[39]. Moreover, the most recent MPEG video coding standard, i.e., MPEG-4 [1], offers an object-based representation of video, where the arbitrarily shaped video objects in a frame are accessible individually in the bit stream. Efficient retrieval of arbitrarily shaped video objects is essential to support visual queries, such as when searching for a particular object with a given shape, color, motion, and texture. Besides retrieval, identifying and classifying video object characteristics can be useful in video communication applications as well. For example, after important video objects in a scene are identified, they can be coded with a better accuracy or transmitted in a more robust channel.

Feature type	Name	Developer	Ref
Color	WebSEEk	Columbia University	[8]
Color, texture	BlobWorld	UC, Berkley	[9]
	ImageRover	Boston University	[10]
	WBISS (Wavelet based image indexing and searching)	UC, Stanford	[11]
Color, texture, shape	QBIC (Query by Image Content)	IBM	[12]
	Virage	Virage Inc.	[13]
	PhotoBook	MIT	[14]
	MARS	University of Illinois	[15]
	Nefertiti	National Research Council	[16]
Color, texture, shape, location	VRW (Visual Retrieval Ware)	Excalibur	[17]
	VisualSeek	Columbia University	[18]
Color, motion	JACOB	University of Di Palermo	[19]
	CueVideo	IBM	[20]
Color, texture, shape, location, motion	NeTra-V	UC, Santa Barbara	[21]
	VideoQ	Columbia University	[22]

Table 1-1: A summary of content-based visual retrieval systems.

In this thesis, we propose methods for retrieval of arbitrarily shaped video objects in the MPEG-4 compressed domain using their low level visual content. In particular, we employ the objects' shape, local motion, and color features, since these are the main features that humans use to differentiate between visual objects. The proposed methods employ the MPEG-4 compressed domain framework mostly because of MPEG-4's support of an object-based representation. Employing the MPEG-4 representation also helps to separate the segmentation problem from the retrieval problem, thus providing a better framework for object-based video retrieval than MPEG-1/2 and H.263. To simplify the retrieval problem, we assume that arbitrarily shaped video objects are already segmented and MPEG-4 coded, and camera effects, such as zooming and panning, are already compensated for.

The effectiveness of our proposed retrieval methods are evaluated by comparing their performance with the uncompressed domain counterparts as well as with some well known techniques in the literature. As a retrieval performance measure, we employ Normalized Modified Retrieval Rank (NMRR), a measure used in the core experiments conducted during the MPEG-7 standardization process [40].

The proposed algorithms are found to be computationally efficient. This should be expected since our algorithms perform processing on a low-resolution version of the video object obtained from the bit stream. Finally, although they are based on the MPEG-4 compressed domain representation, our methods would be applicable to any object-based compressed domain representations as long as low-resolution shape and color data can be obtained from the bit stream without needing full decompression.

1.1 Organization of the Thesis

The remainder of this thesis is organized as follows. Since all of our algorithms operate in the MPEG-4 compressed domain, in the next chapter, we first provide an overview of the arbitrarily shaped video object representation in MPEG-4. Considering that an hour of video can contain more than one hundred thousand images, summarization of video content is necessary for efficient retrieval and browsing. This is typically done through the use of key frames in a frame-based retrieval system. In Chapter 3, we address the problem of summarization of arbitrarily shaped video object content. More specifically, we propose two algorithms for the selection of key video object planes (VOPs) that efficiently summarize the salient content of video objects. The algorithms are based on the shape information of video objects in the MPEG-4 compressed domain. The performance comparisons of these algorithms with the key VOP selection methods that exist in the literature and a discussion on their computational complexity are also presented in this chapter.

In Chapter 4, we present efficient methods for the shape retrieval of MPEG-4 video objects. We achieve this by proposing new ways of computing some well known shape retrieval features directly in the compressed domain as well as proposing some new shape features that are inherently based on the MPEG-4 representation of shape information.

We also address the issues of compact representation and efficient quantization of these shape features. Moreover, we propose a new method for measuring the shape similarity between two video objects, which is based on comparing the still shape features of the key video object planes of video objects. Our experimental results show that the proposed compressed domain shape retrieval techniques offer excellent computational savings, more than two orders of magnitude in computation time in some cases, with only a small degradation in the retrieval performance compared to the uncompressed domain methods.

Motion features play an important role in video retrieval. Chapter 5 first discusses the different types of motion that exist in video sequences, and then gives an overview of the existing motion retrieval techniques. We next propose three new motion features that describe the local motion of the video object within its bounding box. The three proposed descriptors are rotation and scale invariant, and based on the angular and circular area variances of the video object, the variances of the Angular Radial Transform coefficients, and the variances of the Fourier coefficients. The proposed descriptors can be derived directly from the MPEG-4 compressed domain. Tradeoffs associated with each of these descriptors are also presented in this chapter. Our experiments demonstrate that the ranking obtained by querying with the proposed local motion features closely match the human ranking.

Color histograms are among the most widely used visual feature representations in content-based retrieval. When processing the image/video data in the JPEG/MPEG compressed domains, the DC coefficients are used commonly to form color histograms without fully decompressing the image or the video bit stream. In Chapter 6, we address the issues arising from the adaptation of DC based color histograms for the arbitrarily shaped video objects in the MPEG-4 framework. More specifically, we discuss the color space selection, quantization, and histogram computation in consideration of the specific characteristics of the MPEG-4 video objects. The conversions between different color spaces employed in this chapter are given in Appendix C. Chroma keying is one of the most popular methods used to obtain arbitrarily shaped video objects. In some cases, it is possible that the chroma key value of the background results in an erroneous computation of the color histogram. In Chapter 6, we also suggest a method for detecting and reducing these potential color artifacts. The experimental results show that substantial retrieval performance improvements are achieved by employing the proposed method in the presence of such artifacts. Finally, Chapter 7 summarizes the research contributions of this dissertation and suggests some future research directions.

Chapter 2

Background: MPEG-4 Object-Based Representation

The MPEG-4 video coding standard enables higher interactivity with the visual content by allowing access to the individual objects in a video sequence [1][41]-[47]. Furthermore, MPEG-4 supports insertion of user data into the bit stream allowing the transmission of content related information, such as some precomputed features, along with the actual video data. These properties make MPEG-4 very well suited for content representation in multimedia databases.

In this chapter, we first give a brief introduction to the MPEG-4 standard. Video object coding is then described in detail by presenting MPEG-4 texture and shape coding techniques. It is important to note that this chapter does not aim to provide a complete overview to video coding. Rather, we describe the MPEG-4 video coding tools that are different than those of MPEG-1/2 [48] and H.263 [6] putting a special emphasis on the parts of MPEG-4 that we utilize for our compressed domain algorithms.

2.1 Overview of the MPEG-4 Standard

MPEG-4 achieves object-based representation by defining audio-visual objects and coding them into separate bit stream segments. An audio-visual object (AVO) consists of a visual object component, an audio object component, or a combination of these components. Some examples of AVOs include a sound recorded with a microphone, a speech synthesized from text, a person recorded by a video camera, or a 3D image with text overlay. MPEG-4 also supports the composition of a set of audiovisual objects into a scene –referred to as an audiovisual scene. An example of an audiovisual scene composed of natural and synthetic audio and visual objects is presented in Figure 2.1.

MPEG-4 provides functionalities to make it possible to change the position of the AVOs, delete them or make them visible, or manipulate them in a number of ways depending on the nature of their characteristics. For example, if it is a visual object, the user can zoom and rotate it. If it is an audio object, the user can change its pitch, as well as his/her listening point. Also, the quality, spatial and temporal resolutions of the individual AVOs can be modified. For example, in a mobile video telephony application, the user can request a higher frame rate and/or spatial resolution for the talking person than those of the background objects.

The MPEG-4 standard is composed of several parts, including the systems part [49] that addresses issues such as the multiplexing and composition of audiovisual data, the audio part [50] that describes the decoding of the audio data, and the visual part [1] that describes the decoding of the visual data. In this chapter, we focus on the description of the visual part of MPEG-4.

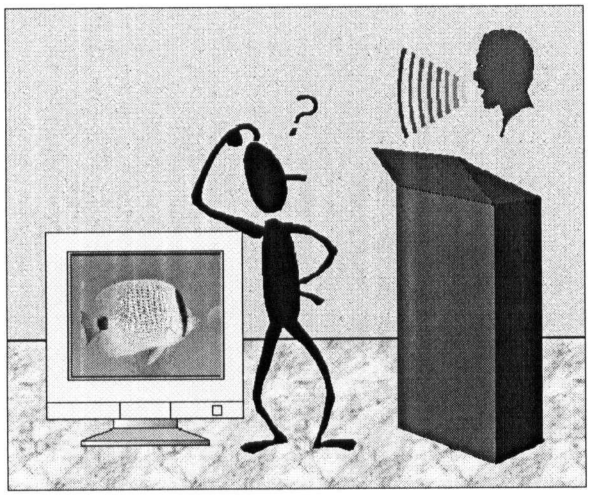


Figure 2.1: Example of an audio-visual scene.

2.2 MPEG-4 Visual Coding Standard

The MPEG-4 visual coding standard aims at providing standardized core processing elements that allow efficient storage, transmission, and manipulation of visual data [1][41]. Different representations and compression algorithms may offer optimum solutions for different applications, bit rates, and formats. Therefore, MPEG-4 provides four different types of coding tools: Video object coding for coding of a natural and/or synthetic originated, rectangular or arbitrarily shaped video object, mesh object coding for coding of a visual object represented with a mesh structure, model-based coding for coding of a synthetic representation and animation of the human face and body, and still texture coding for wavelet coding of still textures.

Since the MPEG-4 syntax is designed to be very generic, and includes many tools to enable a wide variety of applications, the implementation of a decoder that supports the full syntax will most often be impractical. Therefore, MPEG-4 defines a number of subsets of the syntax, referred to as "profiles", each targeting a specific group of applications. Coding of arbitrarily shaped video is supported in "core profile" of MPEG-4, which is the focus of this thesis.

2.2.1 Arbitrarily shaped video coding

A video object (VO) is an arbitrarily shaped video segment that has a semantic meaning. In the MPEG-4 framework, similar to the frame concept in MPEG-1/2 [48], a temporal instance of a video object is called a video object plane (VOP). MPEG-4 allows object-based access to not only the video objects, but also VOPs. Each VOP is defined by its bounding box (the tightest rectangle around the video object), its texture (luminance and chrominance values), and its shape. Coding of a video object involves shape coding, motion compensated prediction to reduce temporal redundancies, and DCT based texture coding of the motion compensated prediction error data to reduce spatial redundancies as illustrated in Figure 2.2.

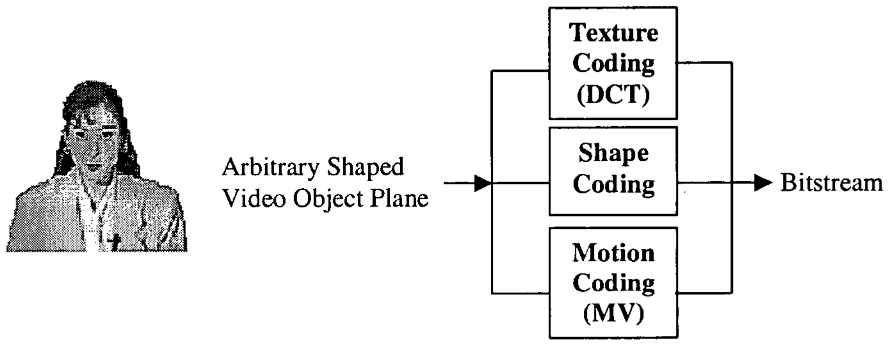


Figure 2.2: MPEG-4 coding of arbitrarily shaped video objects.

Video object coding is performed at the macroblock level. VOPs are divided into macroblocks, such that they are represented with the minimum number of macroblocks within a bounding rectangle. Similar to MPEG-1 and MPEG-2, MPEG-4 supports intra coded (I), temporally predicted (P), and bi-directionally predicted (B) VOPs, all of which are illustrated in Figure 2.3.

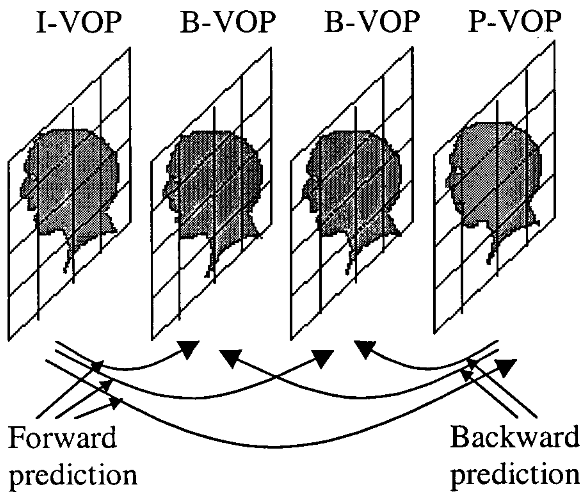


Figure 2.3: VOP prediction in MPEG-4.

Figure 2.4 shows the basic VOP encoder structure. The encoder consists of two main parts: 1) a hybrid of a motion compensated predictor and a DCT based texture coder, and 2) a shape coder. In the first part, motion estimation and compensation is performed (except for IVOPs) on the texture data. Then, the difference between the predicted data and the original texture data is DCT coded and quantized, followed by variable length coding (VLC). Motion information is also encoded using VLCs. Then, the VOP is reconstructed as in the decoder, that is, by applying inverse quantization and inverse DCT (IDCT), and adding the resulting data to the motion compensated prediction data. The resulting VOP is then used for the prediction of future VOPs.

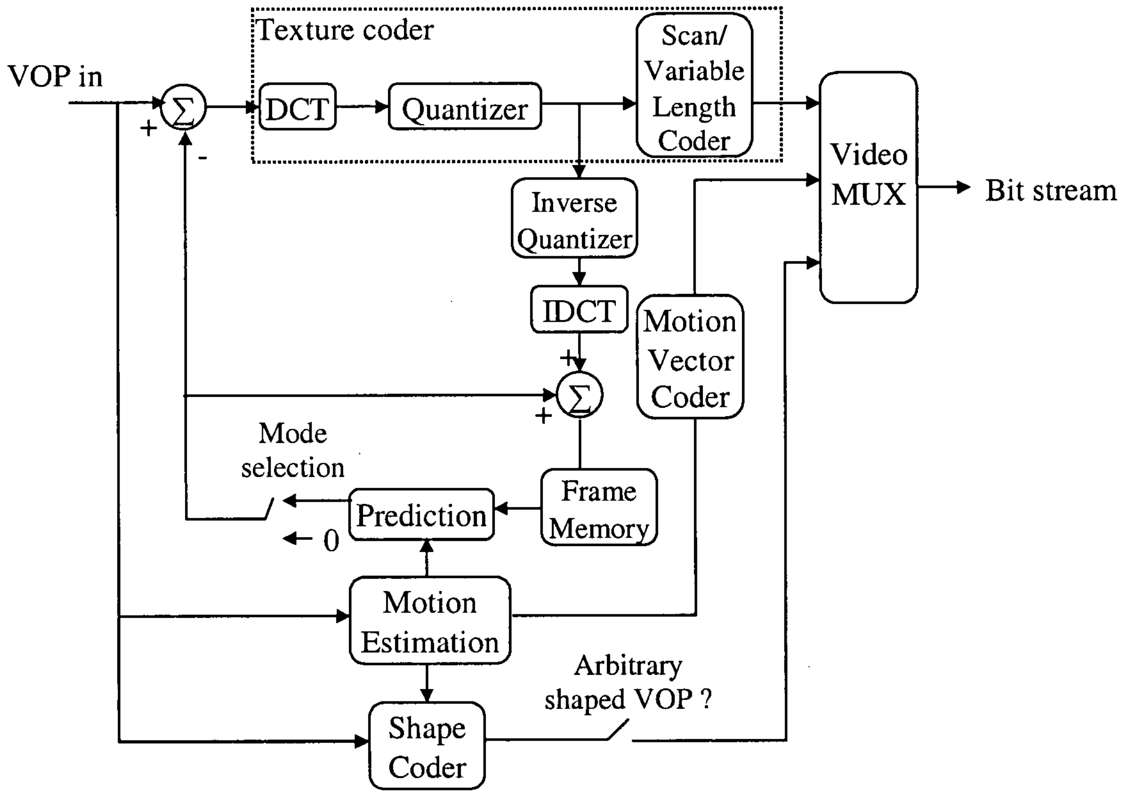
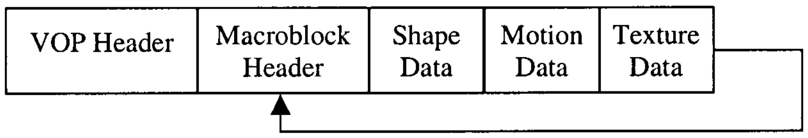


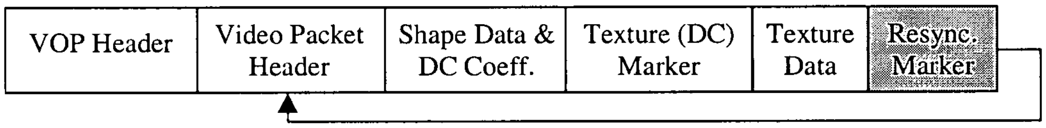
Figure 2.4: A basic block diagram of MPEG-4 video coder.

The shape coder encodes the binary shape and the transparency information of the object. Since the shape of a VOP may not change significantly between the consecutive VOPs, predictive coding can be employed to reduce temporal redundancies. Thus, motion estimation and compensation are also performed for the shape of the object. Finally, motion, texture, and shape information is multiplexed with the headers to form the coded VOP bit stream. At the decoder end, the VOP is reconstructed by combining motion, texture, and shape data decoded from the bit stream.

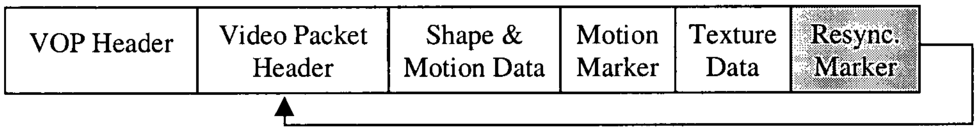
The bit stream is structured so that the texture, motion, and shape information are interleaved at the macroblock level as shown in Figure 2.5.a. MPEG-4 offers error resilience tools that change the bit stream structure so that it is less error prone [51][52]. If "data partitioning" and "resynchronization markers" error resilience modes are enabled, macroblocks in a VOP are grouped into data packets separated by resynchronization markers. This allows greater random access to the bit stream. In the data packets of IVOPs, the shape information is separated from the texture data using the "dc marker". In PVOPs data packets, the motion and shape data are separated from the texture data using the "motion marker". These cases are illustrated in Figure 2.5.b and Figure 2.5.c. If these modes are enabled in the bit stream, it is possible to extract only the shape data without parsing the texture information.



(a)



(b)



(c)

Figure 2.5: Various bit stream structures in MPEG-4 (a) without error resilience options, with data partitioning and resynchronization markers enabled in (b) IVOPs and (c) PVOPs.

2.2.1.1 Texture coding

Intra blocks, as well as motion compensation prediction error blocks, are texture coded. VOPs are divided into macroblocks as illustrated in Figure 2.6, each macroblock consisting of four 8×8 luminance and two 8×8 chrominance blocks. The macroblocks that are completely inside the arbitrarily shaped VOP are coded using a technique very similar to the technique used in H.263 [6], i.e. DCT, quantization, followed by VLC. The blocks that do not belong to the VOP are not coded. If a macroblock lies on the boundary of an arbitrarily shaped VOP, first the pixels that are outside the VOP are padded, and then coded with the same technique used to code the inside blocks. Padding is employed to make the block pixels more uniform, in order to obtain more zero coefficients after DCT and quantization. For inter blocks, the region that is outside the VOP is padded with zeros. In intra blocks, padding is performed by employing the Low Pass Extrapolation (LPE) padding technique described in the MPEG-4 Verification Model (VM) [53] as follows. First, the mean value, μ, of the pixels that are located in the object region is computed. Then the value μ is assigned to each pixel that is located outside the video object. Last, a low pass filter, $f(i,j) = [f(i,j-1)+f(i-1,j)+f(i,j+1)+f(i+1,j)]/4$, is applied to all the block pixels in raster scan order. It is important to note that LPE technique is not part of the standard, but it is a suggested technique in the MPEG-4 VM in order to improve the coding efficiency.

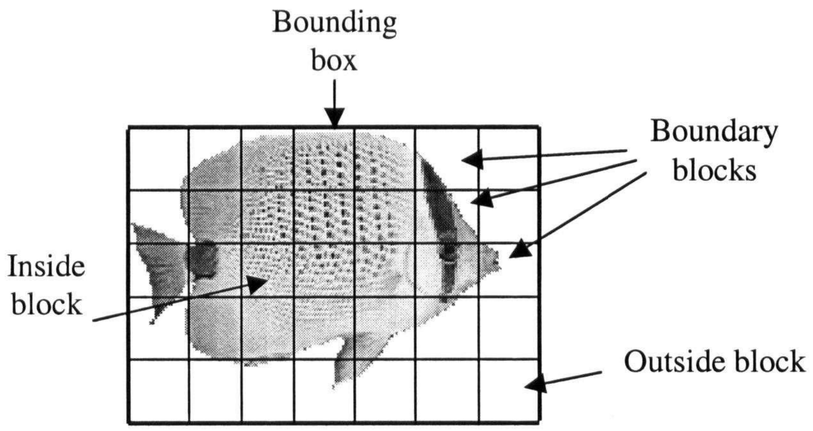


Figure 2.6: Texture of a VOP in MPEG-4.

Another difference between texture coding in MPEG-4 and that of H.263 is that the DC and AC coefficients in MPEG-4 intra blocks can be predictively coded. That is, the DC coefficient is predicted from the DC coefficients of either the left or the above block based on the following rule.

```
If ( | $dc_A - dc_B$ | < | $dc_B - dc_C$ |)
        predict from the above block (block C)
else
        predict from the left block (block A)
```

where the dc_A, dc_B, and dc_C are the DC coefficients of the blocks A, B, and C, as illustrated in Figure 2.7. Based on the prediction direction, the reconstructed DC value of the current block, dc_X, is computed as follows,

```
if (predict from block C)
        $dc_X = edc_X + dc_C$
else
        $dc_X = edc_X + dc_A$
```

where edc_X is the prediction error of the DC value of the current block. Also, if the above block is selected for prediction, the AC coefficients of the first row of the current block are predicted using that of the above block, or alternatively, if the left block is selected for prediction, the AC coefficients of the first column of the current block is predicted using that of the block to the left.

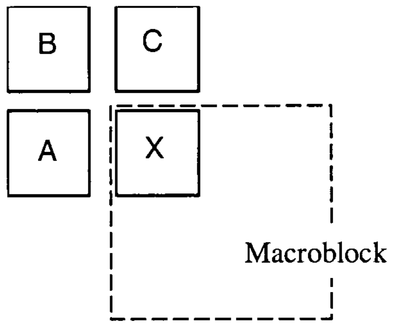


Figure 2.7: Prediction of intra coded macroblocks in MPEG-4.

2.2.1.2 Shape coding

The shape of a VOP is described by a binary alpha plane, which indicates whether or not a pixel belongs to a VOP. A binary alpha plane is divided into 16×16 blocks as illustrated in Figure 2.8. The shape data associated with each of these 16×16 blocks are transmitted in the bit stream, along with the texture information that corresponds to the same area. Three shape coding modes are possible for IVOPs: 1) Transparent, where all pixels in a 16×16 block fall outside the object, 2) Opaque, where all pixels in a 16×16 block are located inside the object, and 3) Intra, where the pixels in a 16×16 block are at the boundary of the object. In intra shape coding, the pixels inside the boundary blocks are raster order scanned and the corresponding binary shape data is context-based arithmetic coded. Lossy shape coding is achieved by subsampling of the binary alpha plane by a factor of 2 or 4 prior to arithmetic encoding.

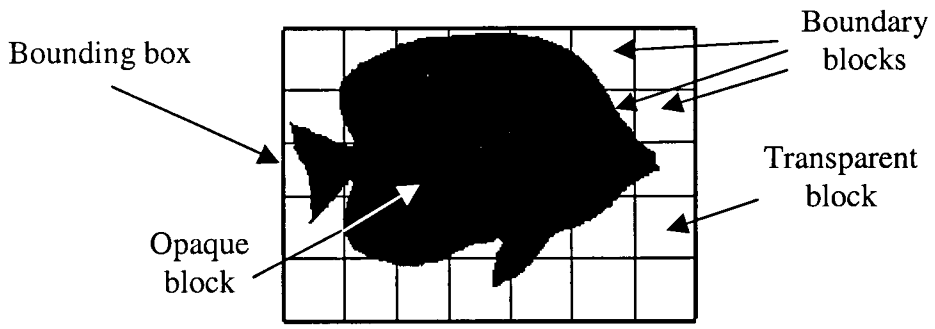


Figure 2.8: Shape of a VOP in MPEG-4.

Seven shape coding modes are supported for the P and BVOPs, as presented in Table 2-1. The transparent, opaque, and intra coding modes are the same as in the IVOP case. The four additional inter shape coding modes involve the transmission of motion vectors and additional update (prediction error) information. In P and BVOPs, the intra shape coding mode is employed only if the current boundary block cannot be efficiently predicted. In inter shape coding, the boundary block is first predicted from the temporally previous or future VOP (depending on the VOP type) and then the difference between the current and the predicted shape blocks is context-based arithmetic coded. The shape motion vectors are also coded predictively using the motion vectors of the surrounding texture and shape blocks.

Coding mode	Coding type	Used in
0	MVDs=0 & no inter update	P and BVOPs
1	MVDs!=0 & no inter update	P and BVOPs
2	Transparent	I, P and BVOPs
3	Opaque	I, P, and BVOPs
4	Intra coded	I, P, and BVOPs
5	MVDs=0 & inter coded	P and BVOPs
6	MVDs!=0 & inter coded	P and BVOPs

Table 2-1: Shape coding modes in MPEG-4.

2.3 Summary

In this chapter, we presented an overview to the visual part of the MPEG-4 standard. We emphasized the coding tools used for arbitrarily shaped video object coding, which are supported in the MPEG-4 core profile. MPEG-4 does not address the problem of object segmentation; nonetheless, it isolates the segmentation problem from the retrieval problem by enabling access to the individual objects in a video scene. Therefore, it provides an excellent framework for our compressed domain retrieval algorithms.

Chapter 3

Video Object Summarization

"If you would be pungent, be brief."

Robert Southey (1774 - 1843)

Considering that digital video is a collection of still images, it is possible to employ still image retrieval techniques for video retrieval. Nevertheless, a direct application of such techniques to video is not trivial considering that a typical 30-minute video sequence may contain more than fifty thousand frames. Therefore, summarization of the video content is often necessary. In a typical digital video retrieval framework, a video sequence is represented by a hierarchical structure, where video sequences are divided into shots, which represent a continuous action in time and space, and then each shot is represented with one or several key frames as shown in Figure 3.1. Employing this representation, the salient content of hours of video can be summarized by a number of key frames as illustrated in Figure 3.2. These key frames can be used for efficient and fast browsing of video sequences as well as retrieval using still image features such as shape, texture, and color.

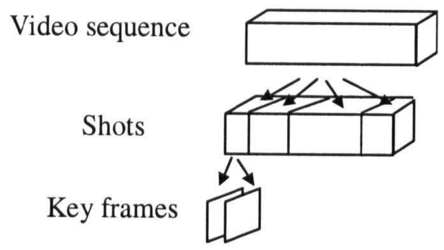


Figure 3.1: Hierarchical representation of a video sequence in a frame-based framework.

A number of methods have been suggested for key frame selection. Some of these algorithms are applied to uncompressed video, and they involve comparing color and motion histograms, computing pixel differences, and performing edge tracking [54][55]. Other algorithms involve operations in the compressed domain (e.g., MPEG-1/2) and take into account the texture coding modes (intra, inter, etc.), motion vectors, and the significant changes in DC coefficients to detect shot boundaries and to select key frames [56]-[59].

Figure 3.2: An example of a frame-based video represented by several key frames.

While key frames provide a summary of the video content, they cannot provide an accurate description of the individual objects within a video scene. In an object-based framework, similar to key frames, key Video Object Planes (VOPs) can be used for the efficient summarization of the video object content. Figure 3.3 shows an example of three key VOPs that summarize a 300-frame video object sequence. Key VOPs can be useful for a number of applications as follows:

- Efficient browsing of the video object sequences, for example summarizing the important actions of a person in a surveillance video.
- Retrieval of video objects by using still image features, for example searching of a video object with a particular color and shape on the Internet.
- Efficient representation of the video object content. For example, in a video telephony application where the bandwidth is limited, only the key VOPs that show the important actions of the speaker can be transmitted instead of the whole video sequence.

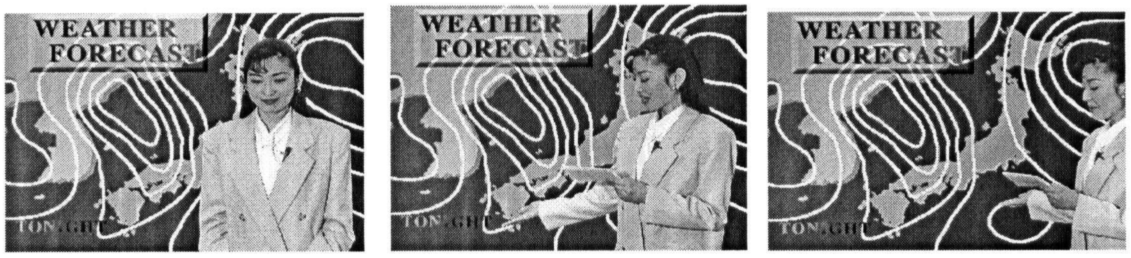


Figure 3.3: Summarization of the foreground video object content with several key VOPs.

Unlike in key frame selection, very little work has been reported on key VOP selection. Gunsel et al. proposed that the motion of the video object and its uncompressed shape data be used for temporal segmentation of video objects and key VOP selection [60]. However, such an algorithm is very computationally intensive, often making key VOP selection unpractical. Ferman et al. suggested an algorithm that uses the texture coding modes in the MPEG-4 compressed domain to extract the key VOPs [61]. Their proposed algorithm employs the percentage of intra coded macroblocks as a measure for significant change in the content. Although the algorithm is simple, the difficult problem of threshold selection has not been addressed. Moreover, the accuracy of using the percentage of intra coded macroblocks is too low for the effective selection of key VOPs.

In this chapter, we propose a new and efficient key VOP selection method that can be computed based on two different distance measures. The proposed method utilizes the shape information of the video objects that can be obtained directly in the MPEG-4 compressed domain. In the following sections, we present the motivation, implementation, experimental results, and the computational requirements of our proposed method.

3.1 Key VOP Selection Based on Shape Information

Typically, key VOPs should be selected such that they reflect significant changes in the shape, color, and texture content of a video object. Using the shape content of a video object for key VOP selection has some advantages over using the color and/or the texture content. First, the texture and color of a video object remain generally consistent during a video object's lifespan. Actually, this fact is used in many spatio-temporal segmentation algorithms for video object segmentation [35]. The shape of a video object, however, may vary significantly due to the object's movement, structure (e.g., articulated, elastic), occlusion, etc. Therefore, a significant change in the content of a video object is more likely to be detected if the object's shape is considered. Second, using the shape of a video object instead of its color or texture is potentially more computationally efficient when processing in the compressed domain. The MPEG-4 bit stream structure is designed such that it is not possible to decode the texture information without having to decode the shape information [1]. On the other hand, the shape information can be extracted from the bit stream without having to decode the texture information when data partitioning is employed. In such a case, extracting the shape information from the bit stream requires very few operations.

Because of the above reasons, our proposed key VOP selection method is based on the shape content of video objects. The main flow of the proposed algorithm is presented in Figure 3.4. Here, the first VOP of a video object is defined as a key VOP, then the distance between the key VOP candidate and the most recent key VOP is computed. If the distance is larger than a threshold then a new key VOP is defined. We employ the Hamming and the Hausdorff distance measures to find the distance between two shapes. The Hamming distance measures the point-by-point difference between two shapes, whereas the Hausdorff distance measures the largest distance between the contours of two shapes [62].

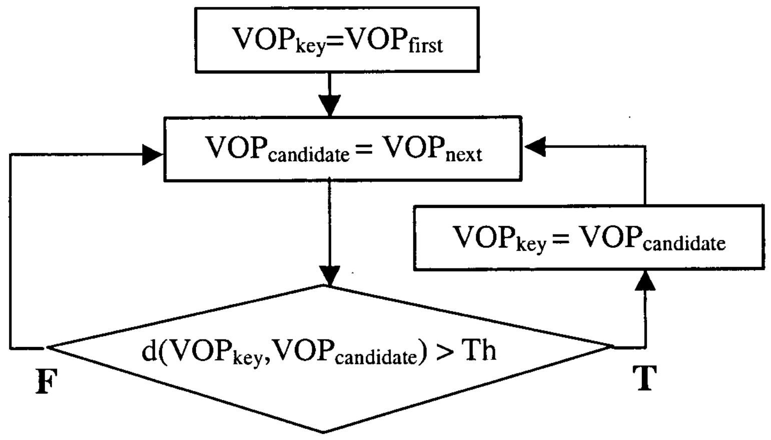


Figure 3.4: Flowgraph of the proposed key VOP selection algorithm.

The computations of the Hamming and Hausdorff distances on the uncompressed shape masks of video objects are straightforward. The Hamming distance is equal to the number of different pixels between the two shape masks and the Hausdorff distance is equal to the maximum distance of two shape contours that are obtained from the two shape masks. In the MPEG-4 compressed domain, these distances are computed based on the shape approximations derived from the shape coding modes as described in the next section. The shape coding modes can be extracted from the MPEG-4 bit stream without full decoding of the bit stream. Besides saving computations, this approximation also makes the proposed algorithms less dependent on how lossy the shape information is coded.

3.1.1 Shape approximation in the MPEG-4 domain using I, P, and B VOPs

Here, we propose that the shape and the boundary of the video object plane be approximated by employing MPEG-4 shape coding modes. As presented previously in Section 2.2, three shape coding modes are possible in the MPEG-4 IVOPs: transparent, opaque, and intra. In our VOP shape approximation, each 16×16 shape block is represented with one value indicating if the block is inside, outside, or at the border of the video object. The coding mode of a shape block in an IVOP directly gives the shape approximation value, i.e., opaque coded blocks are inside, transparent coded blocks are outside, and the intra coded blocks are at the boundary of the video object shape. The boundary of the video object can also be obtained easily by defining the intra coded shape blocks of IVOPs as contour points. Using this approximation, a sub-sampled (by a factor of 16) shape map of a video object plane can be extracted from the MPEG-4 IVOPs without full decompression. This property makes IVOPs ideal key VOP candidates for our key VOP selection algorithms. Nevertheless, there may be MPEG-4 bit streams that do not have periodic IVOPs or the temporal distance between consecutive IVOPs may be very large. In such cases, it may be necessary to consider P and BVOPs as key VOP candidates as well. However, in P and BVOPs, the shape blocks are coded predictively. Therefore, it may not be possible to determine whether a shape block is an inside, outside, or boundary block, without fully decoding and reconstructing the shape information. To address this problem, we next propose a method that allows the use of the same approximation (i.e., inside, outside, or boundary) of a shape block in P and BVOPs.

In MPEG-4, the shape of a P and BVOP is coded by using one of the seven possible coding modes that are summarized in Section 2.2.1.2. If the coding mode of a shape block in a P or BVOP is opaque, transparent, or intra, then the shape block is inside, outside, or at the boundary of the video object, respectively. If the coding mode of a shape block is one of the four inter modes, then it is not possible to indicate in an accurate way that the shape block belongs to inside, outside, or the boundary of the video object, without decoding the shape information of the reference and predicted shape blocks, and reconstructing the predicted shape block. However, we can predict where the shape block is located by considering each possible combination of the shape coding modes of the reference and predicted shape blocks, as summarized in Table 3-1. Our prediction rules are based on the following observations. If a shape motion vector and/or some update information is coded for a predicted shape block, then regardless of the shape coding mode of the reference block, the predicted shape block is very likely to be located at the boundary of the video object. If neither motion vectors nor update information is coded, then the shape block type will be expected to be exactly the same as that of the reference block. However, since the shape motion vector is coded predictively, this may not be always true. Therefore, this inter mode requires further analysis. The MPEG-4 variable length coding tables used for the shape coding modes have been constructed such that, if both the reference and the predicted block are opaque, then it is most efficient to transmit the predicted block in inter mode with no motion vector and update information. Hence, if the reference block is opaque and the predicted block is inter coded with no update information being sent to the decoder, then the current block is likely to be an opaque block. On the other hand, if both the reference and the predicted blocks are transparent, then it is most efficient to transmit the current block as transparent. Therefore, if the shape of a predicted block is inter coded with no motion vector and update information, and its reference block is transmitted as transparent, then it is very likely that the predicted block belongs to the boundary of the video object. When reconstructing the approximated shape information of P and BVOPs, first, the approximation rules summarized in Table 3-1 are applied to each PVOP in a group of pictures so that all the inter modes of the PVOPs are mapped to one of the transparent (outside), opaque (inside), and intra (boundary) coding modes. Then, the same rules are applied to the BVOPs to approximate them from their reference I or PVOPs. If a predicted block does not have a corresponding reference block in its reference VOP, then we apply the copy rule of MPEG-4 [1]. That is, if the number of lines (respectively columns) is larger in the current VOP than in the reference VOP, the bottom line (respectively rightmost column) is replicated as many times as needed in the reference VOP such that all blocks in the predicted VOP have corresponding blocks in the reference VOP.

Block coding mode in the reference VOP	Block coding mode in the predicted VOP	Approximated block coding mode
transparent, opaque, intra	transparent	transparent (outside)
transparent, opaque, intra	opaque	opaque (inside)
transparent, opaque, intra	intra	intra (boundary)
transparent, opaque, intra	MVDs!=0 & no inter update	intra (boundary)
transparent, opaque, intra	MVDs=0 & inter coded	intra (boundary)
transparent, opaque, intra	MVDs!=0 & inter coded	intra (boundary)
transparent	MVDs=0 & no inter update	intra (boundary)
opaque	MVDs=0 & no inter update	opaque (inside)
intra	MVDs=0 & no inter update	intra (boundary)

Table 3-1: Approximation of the shape coding modes for P and BVOPs.

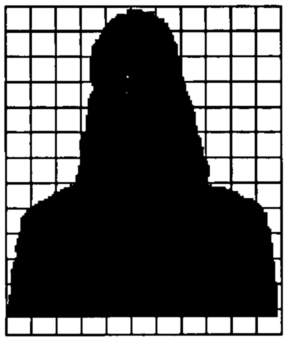
3.1.2 Key VOP selection using the modified Hamming distance

The Hamming distance between two shapes is defined as the number of different pixels between the shapes. In our proposed method, the shape of the VOP is first obtained by using the approximations described in the previous section, and the values "0", "1" and "2" are assigned to the outside, boundary, and inside shape blocks, respectively, as depicted in Figure 3.5. Then, a modified version[1] of the Hamming distance between the two VOPs is computed as follows,

[1] For the Hamming distance measure, the distance between two pixels can be only 0 or 1. The distance between two pixels can here be 0, 1, or 2.

$$d = \sum_{n=0}^{N} \sum_{m=0}^{M} \left| \alpha_{m,n}^{1} - \alpha_{m,n}^{2} \right|, \tag{3.1}$$

where $\alpha^{1}_{m,n}$ is the shape approximation value of the key VOP candidate in the m^th row and n^th column (in number of blocks) of the binary alpha plane, $\alpha^{2}_{m,n}$ is the shape approximation value of the temporally closest key VOP corresponding to the same location, and M and N are the width and height (respectively) of the VOP's bounding box. When the horizontal and/or vertical dimensions of the key VOP candidate are different from those of the key VOP, M and N are assigned to the larger dimensions, and the extended blocks are padded with "0". Because of the "0", "1" and "2" values assigned to outside, boundary, and inside blocks (respectively), the modified Hamming distance is larger when an outside block corresponds to a inside block than when an outside block corresponds to a boundary block, or vice versa.



$\longrightarrow$

0	0	0	0	1	1	1	0	0	0	0
0	0	0	1	1	2	1	0	0	0	0
0	0	0	1	2	2	1	0	0	0	0
0	0	0	1	2	2	1	1	0	0	0
0	0	0	1	2	2	2	1	0	0	0
0	0	1	1	2	2	2	1	0	0	0
0	0	1	2	2	2	2	1	1	0	0
1	1	1	2	2	2	2	2	1	1	1
1	2	2	2	2	2	2	2	2	2	1
1	2	2	2	2	2	2	2	2	2	1
1	2	2	2	2	2	2	2	2	2	1
1	2	2	2	2	2	2	2	2	2	1
1	1	1	1	1	1	1	1	1	1	1

Figure 3.5: Approximation of the shape of an IVOP by using the shape coding modes in MPEG-4. The "0", "1" and "2" values are assigned to the outside (transparent), boundary (intra), and inside (opaque) blocks, respectively.

A problem here is that a slight spatial shift between two very similar shapes may result in a large Hamming distance. Consider the two alpha planes presented in Figure

3.6.a and Figure 3.6.b. Even though the shapes look almost the same, the Hamming distance between the two shapes is very large, as depicted in Figure 3.6.c. The minimum Hamming distance between two shapes can be determined by computing the Hamming distance for every possible alignment of the two shapes. However, this would require a very large number of computations, making the algorithm impractical. Our experiments showed that aligning the mass centers of the two shapes provides a good approximation for the alignment corresponding to the smallest Hamming distance. This is depicted in Figure 3.6.d. Since the actual shape of a VOP is not available without decoding the bit stream, the mass centers are found by using the shape approximations.

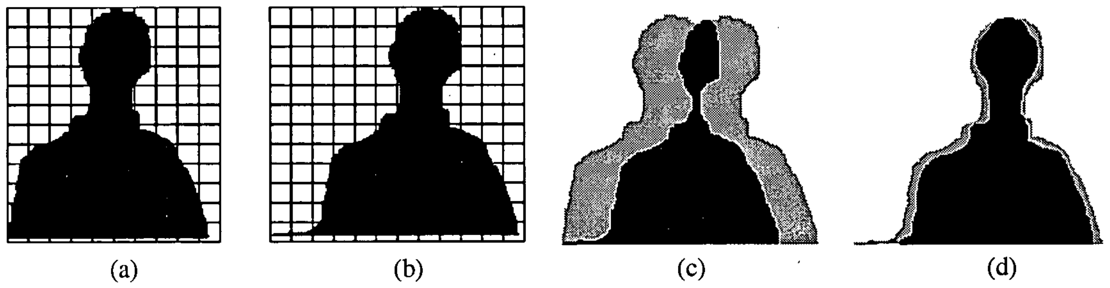


Figure 3.6: (a) Shape of a key VOP, (b) Shape of a key VOP candidate, (c) The large Hamming distance between the two VOPs (shown in gray) caused by the miss-alignment (d) The small Hamming distance between the two VOPs using mass center alignment.

Recall that a new key VOP is selected when the distance between the approximated shape of a key VOP candidate and that of the key VOP is larger than a threshold. The threshold should be adaptive to 1) the activity level and 2) the size of the video object. First, the activity level of a video object needs to be considered because a threshold that is optimized for low activity video objects may result in an erroneous selection of every single key VOP candidate as a key VOP in highly active video objects. Even though it is desired to have more key VOPs for video objects that are more active, the threshold needs to be increased in order to avoid selecting an excessive number of key VOPs for such video objects. Second, the threshold should be selected so as to maintain size invariance. This can be achieved by scaling it with the area of the VOP bounding box. We compute the threshold for each key VOP candidate as follows,

$$T_1 = \lambda_1 \; \phi \; \min(M_1, M_2) \; \min(N_1, N_2), \tag{3.2}$$

where λ_1 is an empirically determined parameter that is constant for all VOPs, ϕ is determined by the activity level of the video object, M_1 and N_1 are the width and height (in number of blocks) of the key VOP (respectively), and M_2 and N_2 are the width and height of the key VOP candidate (respectively). In the cases where the heights and the widths of the current key VOP and key VOP candidate are different, the smaller dimensions are used to determine the area of the VOP. This way, if the dimensions of the current key VOP and the key VOP candidate are significantly different, then the threshold is made small enough so that it is likely to be exceeded.

Since the parameter ϕ depends on the activity level of the video objects, and the video objects may not have uniform activity levels throughout their lifespans, we need to divide the video objects into temporal segments with uniform activity levels. The activity level of a video object can be predicted by monitoring the number of intra coded shape blocks in the P and BVOPs, and defining a new segment when a significant change is detected. The number of intra coded shape blocks, γ, can be obtained from the MPEG-4 bit stream without decoding the shape data. In order to provide size invariance, γ is scaled with the area of the VOP.

The gradient of γ is used to determine the significant variations in γ. We employ a 5-point median filter in order to remove the spikes that correspond to sudden changes in γ with very short duration. This is followed by a 3-point averaging filter to smooth the local changes. Then, the gradient is approximated by

$$\Delta\gamma \cong \gamma[n] - \gamma[n-1] \quad , \tag{3.3}$$

where $\gamma[n]$ and $\gamma[n-1]$ are the numbers of intra coded shape blocks of the current P or BVOP and the temporally previous P or BVOP, respectively. A large gradient value indicates a significant change in γ. Whenever the absolute value of the gradient of γ is above a threshold T_{ts}, a new temporal activity segment is defined. After thresholding, very small temporal segments are combined with the neighboring temporal segments to prevent having an excessive number of temporal segments.

3.1.3 Key VOP selection using the Hausdorff distance

The Hausdorff distance measure can also be used to measure the similarity between two shapes. It is defined as the maxmin function between two sets of points as follows [62],

$$h(A,B) = \max_{a \in A} \{ \min_{b \in B} \{ d(a,b) \} \} , \tag{3.4}$$

where a and b are the points of the sets A and B respectively, and d(a, b) is the Euclidean distance between these points. More specifically, the Hausdorff distance between the sets of points A and B is the maximum distance of the points in set A to the nearest point in set B. The Hausdorff distance is not symmetric, i.e., h(A,B) may not be equal to h(B,A). Therefore, a more general definition of the Hausdorff distance is given by

$$H(A,B) = \max\{h(A,B), h(B,A)\}, \tag{3.5}$$

where h(A,B) and h(B,A) are the Hausdorff distances from A to B, and from B to A, respectively.

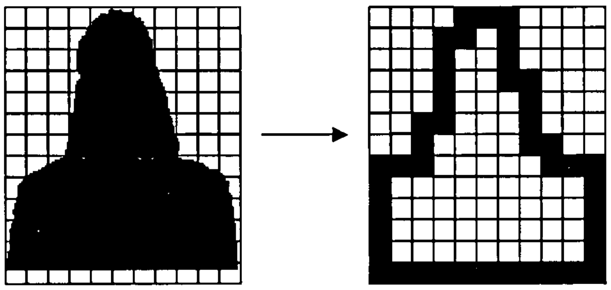


Figure 3.7: Approximation of the shape contour of an IVOP by using the shape coding modes in MPEG-4. The intra coded shape blocks in IVOPs are selected as the contour points.

Similar to the key VOP selection algorithm proposed in the previous section, the first VOP of a video object is declared as a key VOP, and whenever the Hausdorff distance between a key VOP candidate and its temporally closest key VOP is larger than an adaptive threshold, the key VOP candidate is selected as a new key VOP. As in the Hamming distance case, the contours of the key VOP and the key VOP candidate are aligned using their mass centers in order to make the Hausdorff distance invariant of spatial shifts.

Finding the Hausdorff distance between the shape contours of the key VOP and the key VOP candidate involve a large number of Euclidean distance computations.

Moreover, extracting the contours of the VOP requires the decoding of the shape data. In order to avoid these computations, we approximate the contour of a VOP shape from the shape coding modes by defining the boundary shape blocks as the contour points. This is depicted in Figure 3.7. As a result, the number of contour points is significantly reduced, yielding a 16×16 times reduction in computations, besides not needing to decode the shape data.

Unlike the threshold used in the Hamming distance based algorithm, the threshold used in this algorithm does not depend on the activity level of the video object. Our experiments showed that high activity video objects that have large numbers of intra coded shape blocks do not necessarily have large Hausdorff distances between the key VOPs and the key VOP candidates. This should be expected because, unlike the Hamming distance where all the points of the shape affect the distance between two VOPs, the Hausdorff distance is affected by only the two points that have the largest distance, one in the key VOP, and the other one in the key VOP candidate. The threshold, however, still depends on the size of the video object. Since the Hausdorff distance is based on the Euclidean distance, the threshold is scaled by the diagonal length of the VOP bounding box. The threshold is given by

$$T_2 = \lambda_2 \sqrt{\min(M_1, M_2)^2 + \min(N_1, N_2)^2}\,, \qquad (3.6)$$

where λ_2 is a pre-determined scale-factor that is constant for all VOPs, M_1 and N_1 are the width and height (in number of blocks) of the key VOP (respectively) , and M_2 and N_2 are the width and height of the key VOP candidate (respectively). If the widths and heights of the key VOP and the key VOP candidate are different, then the smaller dimensions are selected so as to make it more likely to declare a new key VOP if there is a significant size difference.

3.2 Experimental Results

The proposed key VOP selection algorithms are implemented in C++, and the Microsoft MPEG-4 decoder [63] is used for parsing and partial decoding of the MPEG-4 bit streams to obtain the shape coding modes. In this section, we present our key VOP selection results for three video objects: Hall Monitor, which is a surveillance video sequence, Bream, which is a sequence that shows a fish swimming and turning, and Weather, which is a sequence that shows an anchorwoman presenting the weather forecast. These sequences cover a variety of video objects, from the highly active Hall Monitor to the low motion Weather. The Hall Monitor video sequence was segmented after production, whereas the Bream and Weather video sequences were segmented during production using chroma keying.

The threshold used in the Hamming distance based key VOP selection algorithm depends on the parameters λ_1 and ϕ, as well as the dimensions of the video object. While the dimensions of the video object are extracted from the MPEG-4 bit stream, the values of the parameters λ_1 and ϕ are empirically determined. The parameter λ_1 indicates the percentage of the shape area that is allowed to be different before selecting a new key VOP. Selecting a lower value for λ_1 would result in a higher number of key VOPs and vice versa. The change in the number of key VOPs for different values of λ_1 is presented in Figure 3.8 for the Bream video object. Our experiments show that setting the value of the parameter λ_1 to 0.25 and changing the value of the parameter ϕ from 1 to 1.5 depending on the activity level of the video object, as presented in Table 3-2, result in key VOPs that represent efficiently the content of the video objects.

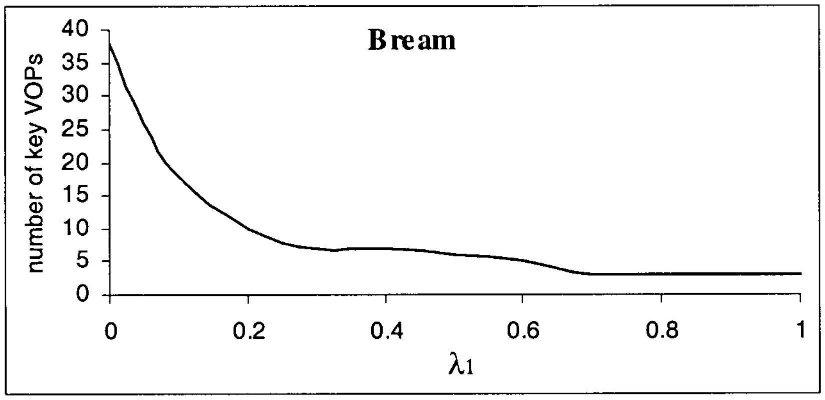


Figure 3.8: The change in the number of key VOPs for different λ_1 values for the Bream video object.

ϕ	Activity level (average percentage of intra coded shape blocks)
1	0% to 29.9%
1.2	30% to 69.9%
1.5	70% to 100%

Table 3-2: Selection of the values for the parameter ϕ depending on the activity level of video objects.

Since the parameter ϕ depends on the activity level of the video object segment, the video objects are divided into temporal segments with uniform activity levels prior to key

VOP selection. The value of the temporal segmentation threshold T_{ts} is set to 0.01. Figure 3.9 shows the change in the percentage of intra coded shape blocks for the Bream video object. The two major peaks of the graph correspond to the two highly active segments of the video object, that is, where the shape of the video object changes rapidly. The temporal segments and their corresponding activity levels for the Weather, Bream, and Hall Monitor video objects are shown in Table 3-3.

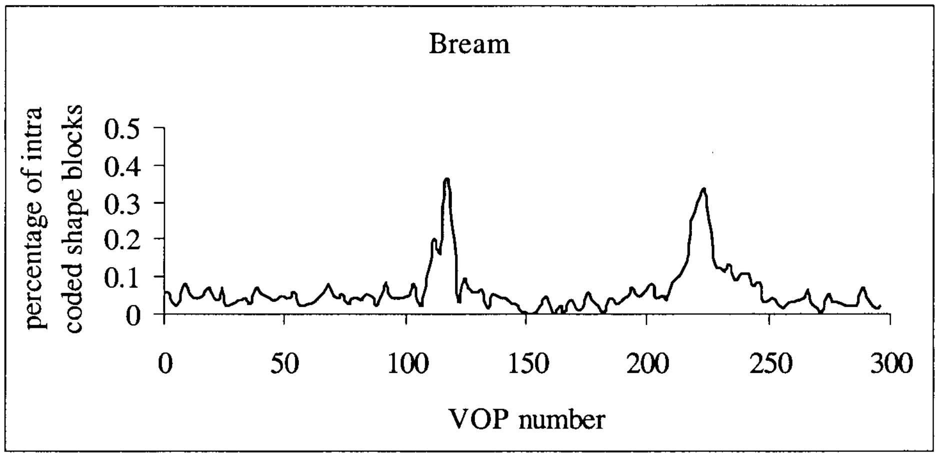


Figure 3.9: The intra coded shape block activity for the Bream video object.

Next, we present our key VOP selection results for video objects that are coded at 15 VOPs per second, following the IPPPIPPP structure, with lossless shape coding, and using a constant quantizer value of 10 for texture. In this set of experiments, only IVOPs are considered as key VOP candidates. We also demonstrate the performance of the proposed algorithms in the case where very lossy coding for shape, i.e. downsampling by four, is employed. Moreover, we present a set of experiments that demonstrate the effects of the use of video object activity level parameter ϕ. In the second set of experiments, we perform key VOP extraction from the IBBBPBBBPBBBP structured MPEG-4 bit streams, where I, P, and BVOPs are also considered as key VOP candidates. In the third set of experiments, we compare our key VOP selection results with the results of the uncompressed domain implementation of our algorithms and other available methods for key VOP selection.

Video object	Segment no	start VOP no	stop VOP no	Activity level (average percentage of intra coded shape blocks)
Weather	0	0	176	0.4%
	1	178	200	5.7%
	2	202	270	1.3%
	3	272	300	5.8%
Bream	0	0	102	4.6%
	1	104	124	15.4%
	2	126	206	3.6%
	3	208	228	18.0%
	4	230	242	11.0%
	5	224	300	3.6%
Hall Monitor	0	6	34	53.3%
	1	36	70	31.5%
	2	72	96	37.1%
	3	98	120	28.2%
	4	122	178	35.8%
	5	180	206	16.9%
	6	208	228	31.92%
	7	230	248	66.3%

Table 3-3. Temporal segments for the Weather, Bream, and Hall Monitor video objects.

3.2.1 Key VOP selection using IVOPs

Figure 3.10, Figure 3.11, and Figure 3.12 show the key VOPs selected for the IPPPIPPPI structured Bream, Weather, and Hall Monitor video object bit streams using the Hamming distance based algorithm and considering the IVOPs as key VOP candidates. The key VOPs extracted using the Hausdorff distance measure, by setting the value of the parameter λ_2 to 0.2, are presented in Figure 3.13, Figure 3.14, and Figure 3.15 for the Bream, Weather, and Hall Monitor video objects, respectively. As seen from the figures, both algorithms select key VOPs that provide a good summarization of the video objects.

The performance of our proposed algorithms has very little dependency on the coding rate of the shape information. In the next experiment, we employ the most lossy coding possible for the MPEG-4 shape information, where the intra coded shape blocks are downsampled by a factor of four. In this case, the selected key VOPs for the Bream video object using the Hamming distance based algorithm are 0, 112, 128, 208, 224, 232, and 240. The key selected VOPs using the Hausdorff distance based algorithm are 0, 112, 136, 200, 224, 232, and 240. These key VOPs are very similar to the ones shown in Figure 3.10 and Figure 3.13, for the Hamming and the Hausdorff distance based algorithms, respectively. Therefore, our proposed algorithms perform similarly when the video object shape is coded losslessly or in the most lossy mode possible.

We next demonstrate the effects of the video object activity level ϕ when determining the threshold for the Hamming distance based algorithm. Since the Bream and Weather video objects have moderate activity levels, as given in Table 3-3, the activity level parameter ϕ in most of the temporal segments is equal to 1. Therefore it does not have any effect on the threshold computation. On the other hand, the activity level of the Hall Monitor video object is high in most of its temporal segments, as shown in Table 3-3. Consequently, the parameter ϕ affects the decision threshold when computing the Hamming distance. If the parameter ϕ is not employed when selecting key VOPs for the Hall Monitor video object, then the selected key VOPs are 6, 22, 62, 86, 94, 134, 150, 166, 174, and 246, as shown in Figure 3.16. Because the Hall Monitor video object is highly active, using the threshold that is not scaled up with the parameter ϕ results in the selection of an excessive number of key VOPs. When these key VOPs are compared to the ones presented in Figure 3.12, which were selected considering the activity level of the video object, it can be seen that they do not improve much the summarization of the salient content of the video object. Therefore, employing the activity level of video objects for key VOP selection prevents the selection of an excessive number of key VOPs for highly active video objects, while yielding a sufficient number of key VOPs that represent efficiently the salient content of a video object.

3.2.2 Key VOP selection using I, P, and BVOPs

In our next experiment, we extract key VOPs from the IBBBPBBBPBBB structured Bream video object bit stream, where not only IVOPs, but also P and BVOP types are considered as key VOP candidates. The selected key VOPs in this case are 0, 112, 120, 128, 208, 224, 232, and 240 using the Hamming distance measure, and 0, 112, 120, 128, 216, 228, 236, and 252 using the Hausdorff distance measure. The key VOPs selected using the Hamming distance measure are identical to those presented in Figure 3.10, where only IVOPs were key VOP candidates. The key VOPs selected using the Hausdorff distance measure are very similar to the VOPs presented in Figure 3.13, although they are not exactly the same. This should be expected because, small prediction errors in P or BVOPs do not affect the Hamming distance significantly (since every block in a VOP is used for measuring the Hamming distance), while small prediction errors that may occur at the edge of P or BVOPs may affect the resulting Hausdorff distance (since the Hausdorff distance is measured between two points).

3.2.3 Comparisons with other methods

For comparison purposes, we also implement our proposed method employing the uncompressed (actual) shape data instead of the approximated shape data. Using the algorithm that is based on the Hamming distance, the selected key VOPs are 0, 240, 256, and 296 for the Weather, 0, 112, 128, 208, 224, 232, and 240 for the Bream, and 6, 22, 54, 102, 142, 190, 230, and 246 for the Hall Monitor video objects. The uncompressed domain version of the Hausdorff distance based algorithm yields the key VOPs 0, 192, and 248 for the Weather, 0, 112, 136, 208, 224, 232, and 248 for the Bream, and 6, 14, 22, 78, 110, 134, 182, and 246 for the Hall Monitor video objects. The key VOPs selected using the shape information in compressed domain are similar to the ones selected using the decompressed shape information. Therefore, processing in the compressed domain becomes very advantageous, since the same performance levels are achieved using 16×16 times less computations and without requiring the decompression of the shape data.

We also compare our key VOP selection algorithms with the compressed domain algorithm proposed by Ferman et al. [61]. Their key VOP selection algorithm is based on the texture coding modes of the PVOPs, and a key VOP is declared whenever the corresponding percentage of intra coded blocks exceeds a threshold. Using this algorithm, the key VOPs selected for the Hall Monitor video object are presented in Figure 3.17. As can be seen from the figure, unlike our key VOP selection algorithms (see Figure 3.12 and Figure 3.15), the algorithm proposed in [61] selects redundant key VOPs (see VOPs number 143, 160, and 175), while also failing to represent some important content changes, more specifically the VOP number 246.

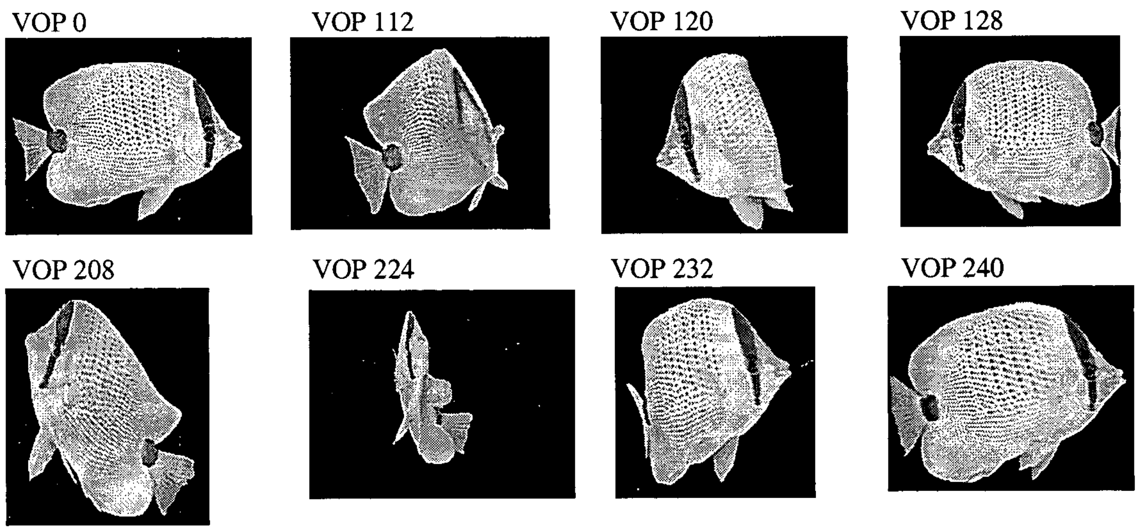


Figure 3.10: The key VOPs selected using the Hamming distance based algorithm for the Bream video object.

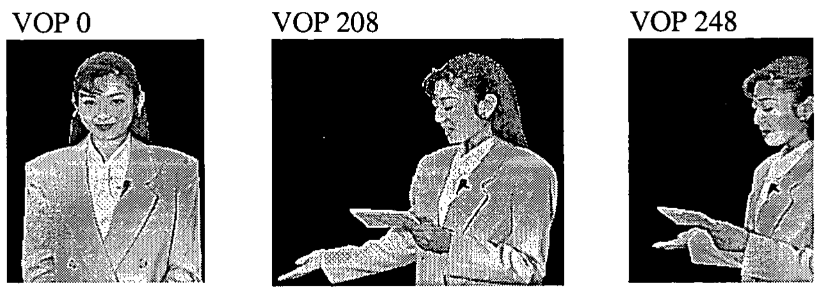


Figure 3.11: The key VOPs selected using the Hamming distance based algorithm for the Weather video object.

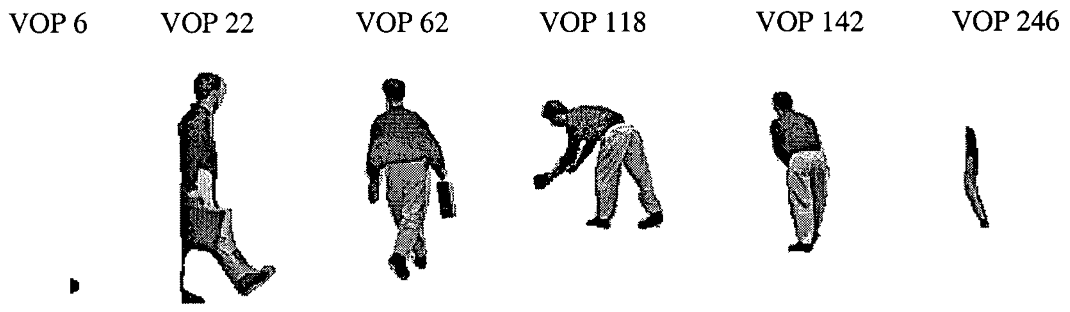


Figure 3.12: The key VOPs selected for the Hall Monitor video object using the Hamming distance based algorithm and with employing a video object activity level (ϕ) dependent threshold.

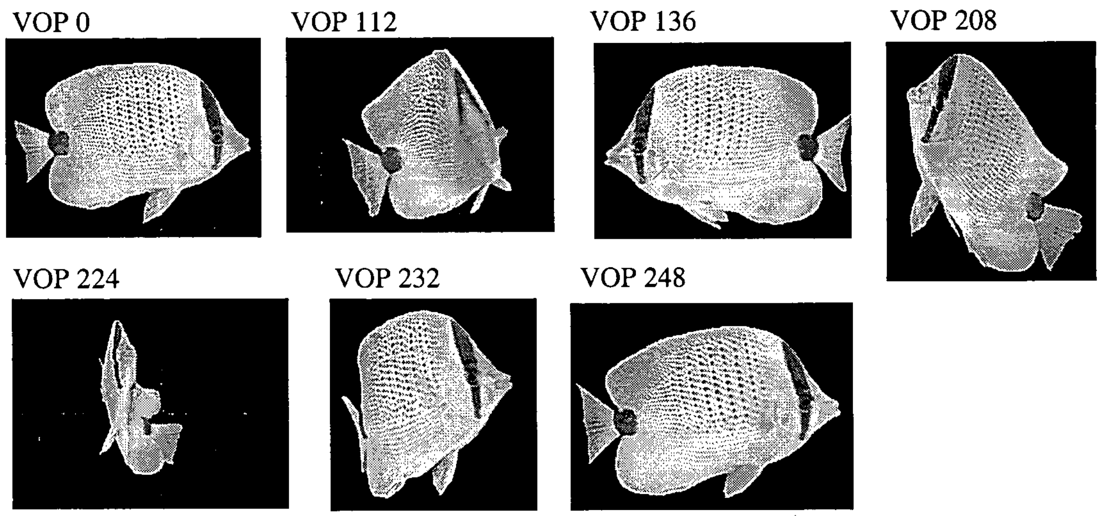


Figure 3.13: The key VOPs selected using the Hausdorff distance based algorithm for the Bream video object.

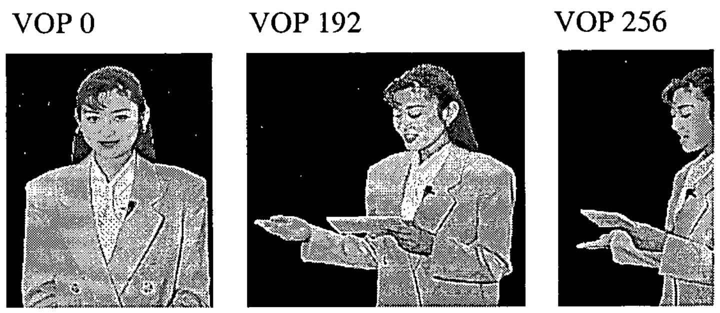


Figure 3.14: The key VOPs selected using the Hausdorff distance based algorithm for the Weather video object.

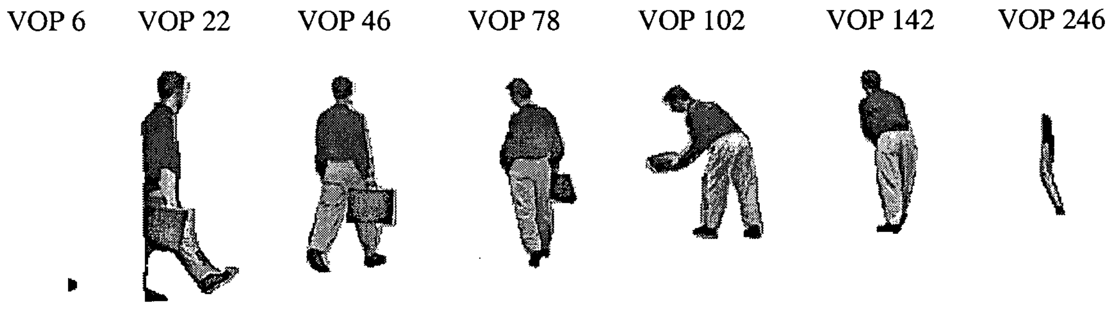


Figure 3.15: The key VOPs selected using the Hausdorff distance based algorithm for the Hall Monitor video object.

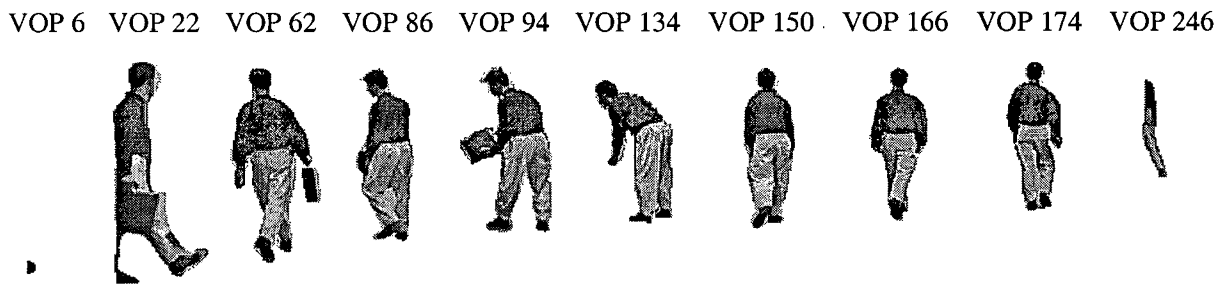


Figure 3.16: The key VOPs selected for the Hall Monitor video object using the Hamming distance based algorithm and without employing a video object activity level (ϕ) dependent threshold.

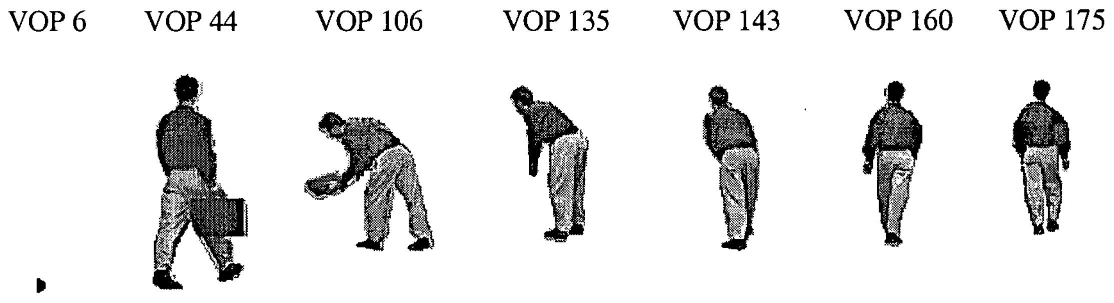


Figure 3.17: The key VOP selection results for the Hall Monitor video object using the algorithm proposed in [61].

3.3 Conclusions

In this chapter, we presented a new method for key VOP selection, using the Hamming and the Hausdorff distance measures, that efficiently summarize the salient content of video objects. As presented in the experimental results, the performance of the Hamming and Hausdorff distance based algorithms are similar. Measuring the Hausdorff distance is more computationally complex than measuring the Hamming distance. Nevertheless, even though the number of operations required is larger, the implementation of the Hausdorff distance based algorithm is simpler, since it does not require dividing the video objects into temporal segments with uniform activity levels. Therefore, depending on the application and available processing resources, either similarity measure one can be used for efficient key VOP selection.

Using the proposed compressed domain shape approximations, the operations required to compute the Hamming and Hausdorff distances are reduced by approximately 16×16 times compared to the uncompressed domain implementations. Also, since the decompression of the shape data is not required, the bit stream processing time is reduced significantly. Besides saving computations, using the shape approximations makes the proposed algorithms less dependent on the segmentation errors and how lossy the shape information is coded.

In this thesis, we employ key VOPs for shape content matching, as presented in Chapter 4, and color content matching, as presented in Chapter 6, of video objects. In these chapters, unless otherwise stated, we utilize the Hausdorff distance based algorithm and select the IVOPs as key VOPs for practical purposes.

Chapter 4

Retrieval of Arbitrarily Shaped Video by using Shape Features

"Hamlet: Do you see yonder cloud that's almost in shape of a camel?
Polonius: By the mass, and 'tis like a camel, indeed.
Hamlet: Methinks it is like a weasel.
Polonius: It is backed like a weasel.
Hamlet: Or like a whale?
Polonius: Very like a whale." Hamlet. Act III, Scene II

William Shakespeare (1564-1616)

Shape matching is an important part of content-based visual data retrieval as there is considerable evidence that humans recognize objects primarily by their shape [64][65]. Retrieval of objects with their shape is a challenging task and much effort has been devoted to finding the shape features and similarity matching techniques that closely resemble human perception [74]-[84]. Consequently, a number of effective shape representations were developed. These shape representation methods can be classified into the following two groups based on the way they describe the shape information [74]:

1. Region based (internal) methods: Features that describe the region of the shape enclosed by a boundary, e.g., area, circularity, eccentricity, moment invariants, and major axis orientation [74]-[76], Zernike moments [67], Angular Radial Transform descriptors [85], etc.
2. Boundary based (external) methods: Features that describe the contour of the object, e.g., Scaled Space representations [68][69], Fourier Descriptors [70]-[72], etc.

One of the most recognized content-based retrieval systems in the literature is QBIC from IBM [12]. In QBIC, the authors employ some global shape features, such as the area, circularity, eccentricity, major axis orientation, and a set of algebraic moment invariants to characterize and match the shape of the objects. In another system, NeTra, Deng and Manjunath propose to represent the shape using curvature, centroid distance, as well as Fourier descriptors [21]. Sclaroff and Pentland present a method in the Photobook system [14], where they define a stiffness matrix that describes how each point in the object is connected to other points. In their system, all shapes are first converted into this matrix and the eigenvectors of the matrix are employed for similarity matching. In another approach, Del Bimbo et al. study shape similarity by elastic matching of users' sketches, where they propose similarity measures that conform to the perception of similarity by humans [78]. In [79], Wang et al. attempt to combine global and local shape features for effective shape retrieval. They employ elongation and compactness features for early elimination of the most dissimilar shapes, and then extract and match the salient points of the boundary of the visual objects. Mokhtarian et al. propose a Curvature Scaled Space shape descriptor that is computed by convolving a parametric representation of the curve with a series of Gaussian functions and extracting the resulting curvature zero crossing points [68]. In one of the most recent retrieval systems, MARS [84], Chakrabarti et al. propose to represent the shape with a bitmap divided into grids and employ an adaptive resolution technique where the resolution of the grid cells varies from one portion of the shape to another depending on weather it improves the quality of the representation. Furthermore, MPEG-7, the MPEG standard for content-based description, employs two shape descriptors; one is a contour based descriptor based on a Curvature Scaled Space representation, and the other one is a region based descriptor based on the Angular Radial Transform [85].

Most of the above features are rotation, translation, and size invariant, and are robust to small physical deformations (e.g., stretch) of the shape. They find use in many applications, including surveillance, chromosome classification, and efficient access to trademark, criminal investigation, photograph, and medical databases. Consequently, there were several attempts to employ some of these still shape features for video retrieval. For example, the VideoQ system [22] supports video retrieval with the shape content besides other content-based features. In VideoQ, the global shape features, i.e. eccentricity, first and second moments, and the area of the video object are employed for efficient shape retrieval. In another retrieval system, NeTra-V, the shape features for each instant of the video objects are found by employing the Fourier descriptors of the curvature, centroid distance, and complex coordinate functions. These features are computed only for the subobjects in the Intra frames and the mean of these shape features is used for efficient video object retrieval. These systems show a degree of success at retrieving video objects with their shape, however they have some shortcomings as summarized below:

- The current systems take frame-based compressed video as input, which needs to be decoded and segmented before accessing the shape information of the video objects in the video sequence.
- In the existing systems, the shape of video objects is represented with one set of features, usually obtained by averaging the features that belong to each temporal instant of the video objects. While such a representation would work well for video objects that have constant shapes during their lifespan, it is insufficient if the shape of a video object changes significantly due to, for example, an object entering to or exiting from the scene, occlusion, or high motion. Therefore, new representation methods and shape distance measures are needed to more accurately match the video objects in consideration of the fact that their shape could vary significantly during their existence.

In this chapter, we address the above problems by:

- Suggesting a method for computing several effective shape features that are derived from the MPEG-4 compressed domain shape information,
- Proposing a technique and a similarity measure to compute the shape distances between video objects.

The rest of this chapter is organized as follows. In the next section, we first discuss the compactness, eccentricity, Fourier, and Angular Radial Transform (ART) descriptors in detail. These shape descriptors are commonly used and are well proven to be effective for shape retrieval [21][22][85]. In Section 4.2, we propose methods to compute these descriptors directly in the MPEG-4 compressed domain. Moreover, we introduce two new MPEG-4 shape descriptors. The efficient quantization and compact representation of these features are also addressed. The effectiveness of these compressed domain features compared to their spatial domain implementations (obtained by fully decompressing the bit stream) is presented at the end of the section. In Section 4.3, we extend our proposed still shape retrieval methods to arbitrarily shaped video object retrieval through the use of key VOPs. Finally, we discuss the performance of our proposed methods in the last section.

4.1 Shape Features

In this section, we give an overview to the computation of some well known shape descriptors.

4.1.1 Global shape features

Global shape features are widely employed by many shape retrieval systems, as they are easy to compute. Even though they do not accurately represent the shape of an object, they are very useful for efficiently eliminating the dissimilar shapes to the query shape in large databases.

Compactness is a global shape feature that indicates how round the region of the shape is, and it is defined by $C = \frac{l^2}{A}$, where l is the perimeter and A is the area of the object's shape.

The "eccentricity" feature defines the elongation of a shape. It has several definitions, but here we employ this feature as defined in [75], $E = \frac{\beta}{\delta}$, where β is the length of the straight line segment joining the two contour points that are farthest from each other and δ is the length of the line that is perpendicular to the major axis and of such length that a box could be formed that just encloses the boundary.

4.1.2 Fourier descriptors

Fourier descriptors are one of the most successful boundary representations as shown in the literature [39][71] and they are commonly used for shape retrieval [15][21]. They are generally computed using the ordered contour of the object's shape. In order to obtain the same number of Fourier descriptors for the query and database objects, the boundary of each object needs to be re-sampled to N samples. The re-sampled contour points are represented as complex numbers in an object-centered coordinate system as follows,

$$z_n = (x_n - x_m) + j(y_n - y_m), \tag{4.1}$$

where x_n and y_n are the coordinates of the contour points, and x_m and y_m are the coordinates of the VOP mass center. Employing a mass-centered coordinate system provides translation invariance. The 1-D Fourier Transform of the contour is then computed as follows [70][72],

$$Z_k = \frac{1}{N} \sum_{n=0}^{N-1} z_n \, e^{-\frac{2j\pi n k}{N}} \qquad 0 \le k < N, \tag{4.2}$$

where N is the number of re-sampled contour points and z_n represents the complex coordinates of the nth contour point. The above equation results in N complex Fourier coefficients and the coefficients around 0 and N-1 provide a coarse representation of the shape where the coefficients around N/2 represent the finer details. Re-ordering of the coefficients is done so that the lower coefficient indices correspond to lower frequencies and the higher coefficient indices correspond to higher frequencies.

The first coefficient Z_0 gives the mass center of the closed contour, which in our case is equal to zero since we already employ an object-centered coordinate system. Z_1 gives the radius of the circle with an area equal to that of the shape. Here, we use Z_1 to scale the rest of the Fourier coefficients in order to provide scale invariance. The phase of the Fourier coefficient contains the rotation information of the contour. Using only the amplitude of the scaled Fourier coefficients and ignoring the phase information ensures the rotation invariance of the Fourier descriptors. As a result, the scaled amplitudes of the

M lowest frequency AC Fourier coefficients are employed to form a Fourier descriptor feature vector as follows,

$$\vec{F} = [f_1, f_2, f_3, \ldots, f_M] = \left[\frac{|F_2|}{|F_1|}, \frac{|F_3|}{|F_1|}, \frac{|F_4|}{|F_1|}, \ldots, \frac{|F_{M+1}|}{|F_1|}\right], \tag{4.3}$$

where $F_1 \ldots F_{M+1}$ are the lowest frequency AC coefficients.

4.1.3 ART descriptors

ART descriptors are obtained from the Angular Radial Transform coefficients of the shape region. Employing ART based descriptors is an efficient way to represent shape, as they are easy to extract and match. Furthermore, since the ART descriptors describe the region of a shape, unlike their contour based counterparts such as Curvature Scale-Space and Fourier descriptors, they are capable of representing holes and unconnected regions. Consequently, an ART based representation was recently adopted by MPEG-7 [27]. The definition of the ART transform is given by [85]

$$A_{nm} = \int_0^{2\pi} \int_0^1 V_{nm}(\rho,\theta) f(\rho,\theta) \rho d\rho d\theta, \tag{4.4}$$

where A_{nm} is an ART coefficient of order n and m, $f(\rho,\theta)$ is the binary shape map in polar coordinates, and $V_{nm}(\rho,\theta)$ is the ART basis function, which is separable along the angular and radial directions as follows,

$$V_{nm}(\rho,\theta) = G_m(\theta) R_n(\rho), \tag{4.5}$$

where $G_m(\theta)$ and $R_n(\rho)$ are the angular and radial basis functions respectively, and they are given by

$$G_m(\theta) = \frac{1}{2\pi} e^{jm\theta}, \text{ and } R_n(\rho) = \begin{cases} 1 & n = 0 \\ 2\cos(\pi n\rho) & n \neq 0 \end{cases}. \tag{4.6}$$

The real parts of the 2-D ART basis functions are depicted in Figure 4.1 for eight angular and four radial functions.

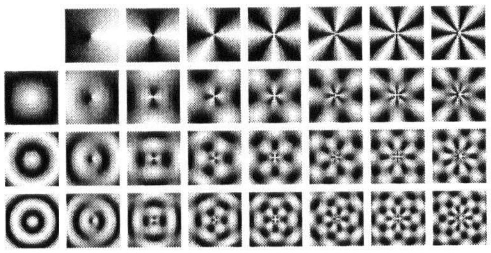


Figure 4.1: The real part of the basis functions of the ART transform for eight angular and four radial functions.

The magnitudes of the ART coefficients are commonly employed as ART descriptors since they are rotation invariant as shown below.

$$A_{nm} = \int_0^{2\pi} \int_0^1 V_{nm}(\rho,\theta) f(\rho,\theta) \rho d\rho d\theta \text{, and} \tag{4.7}$$

$$A_{nm}^{\alpha} = \int_0^{2\pi} \int_0^1 V_{nm}(\rho,\theta) f(\rho,\theta)\, e^{j\alpha} \rho d\rho d\theta \tag{4.8}$$

$$= A_{nm} e^{\alpha j} \Rightarrow \left\| A_{nm}^{\alpha} \right\| = \left\| A_{nm} \right\|. \tag{4.9}$$

4.2 Shape Similarity Matching of Still Video Object Planes in the MPEG-4 Compressed Domain

In this section, we propose techniques for computing the above shape features using the MPEG-4 compressed domain shape information. Recall that a video object is a collection of video object planes. Therefore, before addressing the issues related to shape matching of video objects, we first look into extraction of shape features from the individual video object planes (VOPs).

It is possible to obtain a rough approximation of the VOP shape by decoding only the MPEG-4 shape coding modes. Here, we construct the shape of the Intra coded VOPs (IVOPs) by defining the intra and opaque coded shape blocks as being inside the shape. On the other hand, the contour approximation is obtained by defining the intra and opaque coded blocks that have at least one transparent block (based on 4 connectivity) as boundary. Note that, since each 16×16 block of a binary shape map is represented with one shape coding mode (see Chapter 2), the resulting shape approximation is a downsampled version of the uncompressed shape map by a factor of 16 in both horizontal and vertical directions. Even though this approximation is not very precise, it is sufficient to compute some shape features with a reasonable accuracy. Moreover, it has the following advantages:

1. Since only the shape coding modes are used for the approximation, full decoding of the bit stream and the reconstruction of the shape data is not necessary, saving memory and computation time.
2. Additional computational savings are obtained as the shape features are computed for a downsampled version of the shape by a factor of 16 in both dimensions.
3. Employing a rough approximation of the shape instead of the real shape data makes the shape features more robust to small segmentation errors.

4.2.1 Extraction of the global shape features

In order to compute compactness, the perimeter and the area of an MPEG-4 object needs to be known. However, they cannot be obtained without decompressing the bit stream. We propose to compute the compactness of the shape of a VOP directly in the compressed domain using,

$$C = \frac{\eta^2}{O + 0.5xI}, \tag{4.10}$$

where η is the number of approximated contour points, and O and I are the number of opaque and intra coded shape blocks in a VOP, respectively.

We also compute the eccentricity feature on the approximated shape boundary extracted by the technique described in the previous section. The major and minor axes lengths are found by using this approximated boundary and their ratio is stored as the eccentricity of the shape.

4.2.2 Fourier descriptor extraction

Compressed domain Fourier descriptors are computed using the contour points obtained from the approximated boundary of the VOP. In order to compute the Fourier descriptors, first the extraction of the ordered contour points is needed. This is generally a trivial operation. However, because we employ a downsampled version of the real object contour in our approximation, there are potentially some contour points bordering with more than two other contour points. This would make the ordering of the contour points more challenging as there would be more than one possible contour path to follow.

In order to overcome this problem, we initially upsample the approximated VOP shape by 4, extract the ordered object contour in the clock wise direction based on 8 connectivity, and then downsample the extracted contour points by 4 to their original size.

Re-sampling to N points, in order to obtain the same number of Fourier descriptors for all the database objects, is done by first forming a boundary by linear interpolation using the ordered contour points, then re-sampling the boundary into $N = 2^6 = 64$ equally distanced points so that the Fourier transform can be performed efficiently using the Fast Fourier Transform. After computing the Fourier descriptors as described in Section 4.1.2, the first M Fourier descriptors are employed for shape retrieval. The value of M determines how accurately the shape is represented. Selecting a small M value results in a smaller size feature vector but reduces the retrieval accuracy. On the other hand, selecting a large value for M increases the storage requirements and also potentially increases the retrieval accuracy. However, it can also possibly make the feature vector more sensitive to segmentation errors and errors caused by our shape approximation. These tradeoffs are presented in detail in Section 4.2.6.2.

4.2.3 ART descriptor extraction

Here, we propose to compute ART descriptors using the approximated shape map obtained from the MPEG-4 bit stream by defining the intra and opaque coded shape blocks as being inside the shape. We find the discrete ART coefficients of the shape map as follows. First, the size of the binary shape data is normalized by linear interpolation to a predefined width, W, and height, H, to obtain the size invariant shape map $I(x,y)$. The mass center of the binary shape map is aligned with the center of *I(x,y)*, i.e. *I(W/2, H/2)*. Then, the discrete ART coefficients of the shape map of the video object plane k (VOP_k) are computed by

$$A_{nm}(VOP_k) = \sum_{x=-W/2}^{W/2} \sum_{y=-H/2}^{H/2} V_{nm}\left(\arctan\frac{y}{x}, \sqrt{x^2+y^2}\right) I_{VOP_k}\left(x+\frac{W}{2}, y+\frac{H}{2}\right), \quad (4.11)$$

where $V_{nm}(\rho,\theta)$ is the ART basis function as given in Equation (4.5) and $I_{VOP_k}(x,y)$ is the shape map of VOP_k.

The mass center of a VOP in the compressed domain is computed as follows,

$$\begin{bmatrix} m_x \\ m_y \end{bmatrix} = \frac{1}{O+0.5xI}\left\{\sum\begin{bmatrix} X_O \\ Y_O \end{bmatrix} + \frac{1}{2}\sum\begin{bmatrix} X_I \\ Y_I \end{bmatrix}\right\}, \quad (4.12)$$

where O and I are the number of opaque and intra shape coded blocks, X_O, Y_O and X_I, Y_I are the x and y coordinates of opaque and intra coded blocks, respectively. Because of the alignment of the object center with the center of *I(x,y)*, the ART coefficients are translation invariant. Interpolation of the shape to a predefined width and height prior to the transform makes the resulting coefficients scale invariant as well. Rotation invariance is ensured by employing the magnitude of the coefficients as shown in Section 4.1.3.

In the MPEG-7 eXperimentation Model (XM) document [85], the suggested numbers of angular and radial functions for the ART descriptors are 12 and 3, respectively. These numbers are optimized mostly for shape data in the spatial domain, which typically has higher resolution than our shape approximation maps. Selecting small numbers for these parameters reduces the computational requirements significantly. This problem is addressed in more detail in Section 4.2.6.3.

4.2.4 Other compressed domain descriptors

One of the main advantages of compressed domain processing is that it is possible to make use of some information about the data that were already extracted during the encoding process and made available in the bit stream. Motivated by that, we propose two new shape features that are directly based on the MPEG-4 representation of the shape information. The first feature, pruned intra density, ID_P, gives a measure of the shape boundary complexity and is defined by $ID_P = \frac{I_p}{\sqrt{w^2 + h^2}}$, where I_p is the number of intra coded shape blocks that need to be pruned from the VOP shape to obtain a closed contour, and w and h are the width and height of the VOP bounding box, respectively. This feature takes a large value if the shape boundary has a lot of irregularities and branches.

The second compressed domain feature, inside intra density, ID_I, is defined as $ID_I = \frac{I_i}{\sqrt{w^2 + h^2}}$, where I_i is the number of intra coded shape blocks that do not have any transparent coded neighboring blocks based on four-point connectivity, and w and h are the width and height of the VOP bounding box, respectively. A large value of ID_I is an indication of the presence of holes in the VOP shape.

4.2.5 Feature normalization

Normalization of the features to a predefined range is necessary when employing more than one feature since the dynamic range of the computed feature values could change from one shape feature to the other drastically. A straightforward way of normalization is to scale the feature values with the dynamic range of the values of that particular feature. However, in that case, if there are some feature values that are significantly different than the rest of the values, then a very small range would likely to be allocated to represent most of the data. For example, if the obtained feature values are {0, 1, 2, 4.5, 5, 500}, when each value is scaled with the dynamic range of these values, i.e. 500, a very small range of values would represent most of the data. In order to overcome this problem, we consider the statistical distributions of the feature values and perform normalization accordingly.

Figure 4.2 shows the discrete probability distribution of the compactness and eccentricity values. Figure 4.3 illustrates the probability distribution of the pruned intra density and inside intra density features. The probability distributions of the Fourier and ART descriptors are presented in Figure 4.4 and Figure 4.5, respectively. The compactness and eccentricity feature values show the characteristics of Gaussian distribution whereas the rest of the feature values are better modeled with an exponential distribution.

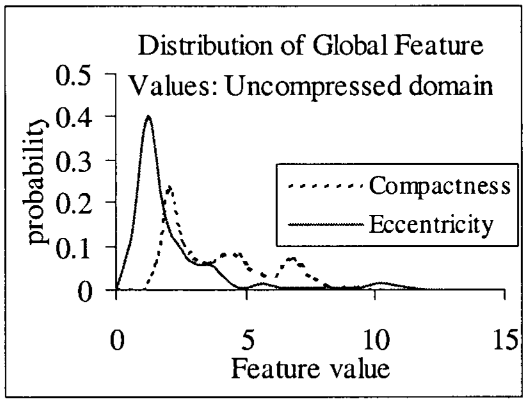
For a Gaussian source with mean μ and standard deviation σ, the normalization is achieved by $x' = \frac{x-\mu}{m\sigma}$. The value of m affects the probability of x' being in the range of [-1 1]. In particular, for m=1 this probability is 68%, for m=2 it is 95%, and for m=3 it is 99%. We employ m=3 for to normalize compactness and eccentricity features.

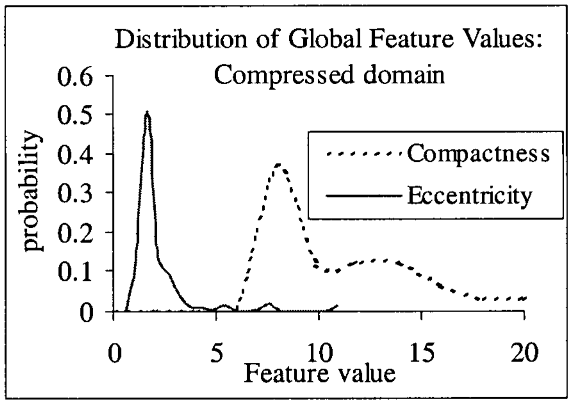
The probability distributions of rest of the features can be represented by exponential distribution function, $\alpha e^{-\alpha x}$, where α is a constant and the mean and standard deviation are equal to $1/\alpha$. In this case, the probability of the feature value being in the range of $0<x<2\sigma$ is

$$P(0 < x <= 2\sigma) \overset{\sigma=1/\alpha}{=} \int_0^{2/\alpha} \alpha\, e^{-\alpha\, x}\, dx = -e^{-\alpha x_{i+1}} \Big|_0^{2/\alpha} = -e^{-\alpha \frac{2}{\alpha}} + 1 \cong 0.86, \qquad (4.13)$$

and the normalization to the [-1 1] range is performed by $x' = \dfrac{x-\sigma}{\sigma}$.

The mean and standard deviation values needed for normalization of our shape features are given in Table 4-1. As can be seen from the figures and the table, the distribution, mean, and standard deviation of the compressed domain features are very similar to those of the uncompressed domain features.





(a) **(b)**

Figure 4.2: Discrete probability distributions for the (a) uncompressed and (b) compressed domain global shape features.

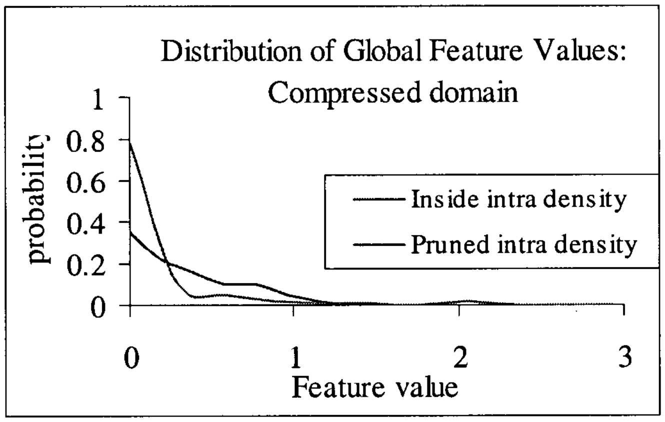


Figure 4.3: Discrete probability distribution for the compressed domain specific shape features.

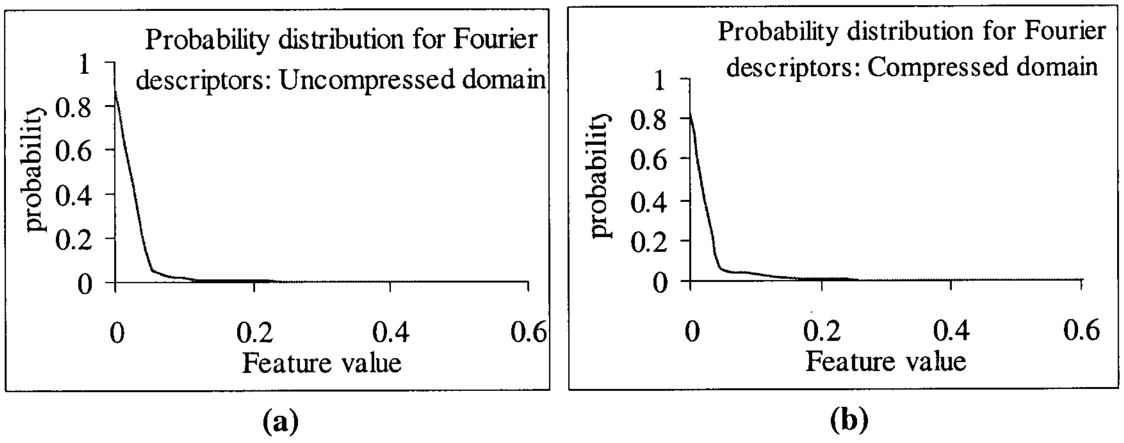
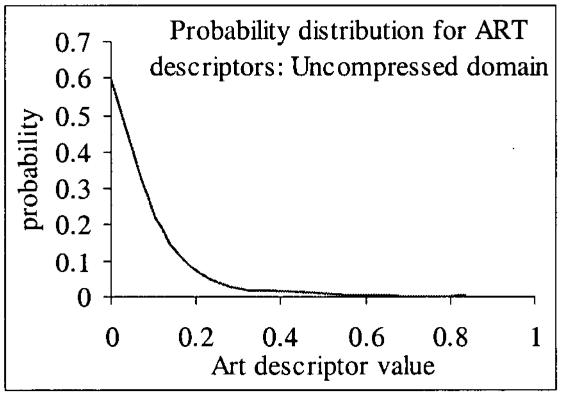
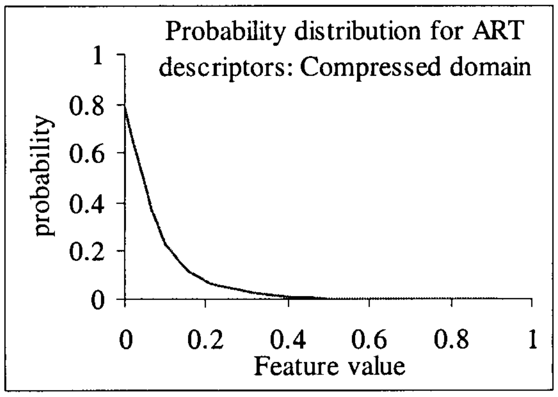


Figure 4.4: Discrete probability distribution for the Fourier descriptors in the (a) uncompressed and (b) compressed domains.



(a)



(b)

Figure 4.5: Discrete probability distribution for the ART descriptors in the (a) uncompressed and (b) compressed domains.

Shape feature	Domain	Mean	Standard deviation
Compactness	uncompressed	3.965	1.943
Eccentricity	uncompressed	2.281	2.095
Fourier descriptors	uncompressed	0.040	0.086
ART descriptors	uncompressed	0.095	0.121
Compactness	compressed	11.96	6.406
Eccentricity	compressed	1.996	1.69
Inside intra density	compressed	0.394	0.346
Pruned intra density	compressed	0.244	0.424
Fourier descriptors	compressed	0.035	0.077
ART descriptors	compressed	0.095	0.1

Table 4-1: Mean and standard deviation values of the all shape features.

4.2.6 Retrieval of the video object planes

In this section, we provide some experimental results that show the performance of our proposed shape descriptors in the case of individual video object plane retrieval. The computation times presented here are obtained by using a Pentium 233 MHz. computer with 128 MBs of memory. Our database is formed by encoding 20 video objects with two or three different spatial resolutions. The contents of these video objects are shown in Appendix B. The resulting database contains more than 50 MPEG-4 video object bit streams with more than 2000 Intra coded video object planes. The results presented here, unless otherwise specified, are obtained by averaging the retrieval results for the four VOP queries shown in Figure 4.6. As can be seen from Figure 4.6, these query VOPs cover a good variation of shapes. For each given query, the ground truth VOPs (i.e. the most similar VOPs) are marked manually. Then the results of our retrieval algorithm are compared to these ground truth VOPs.

We employ the Normalized Modified Retrieval Rank (NMRR) measure to evaluate the performance of the compressed domain Fourier and ART descriptors. The NMRR measures how many of the correct items are retrieved as well as how highly they are ranked among the retrieved items for a given query. ANMRR is the average NMRR over a set of queries. The NMRR and ANMRR values are in the range of [0, 1]. Lower values represent a better retrieval rate. The specific formulas of these measures can be found in Appendix A.

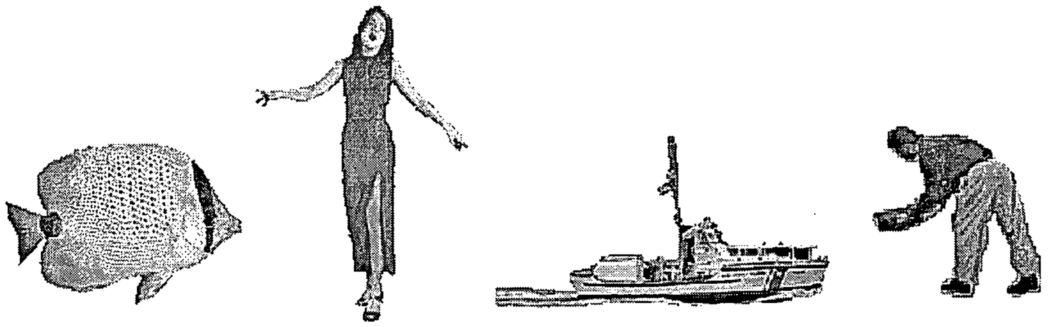
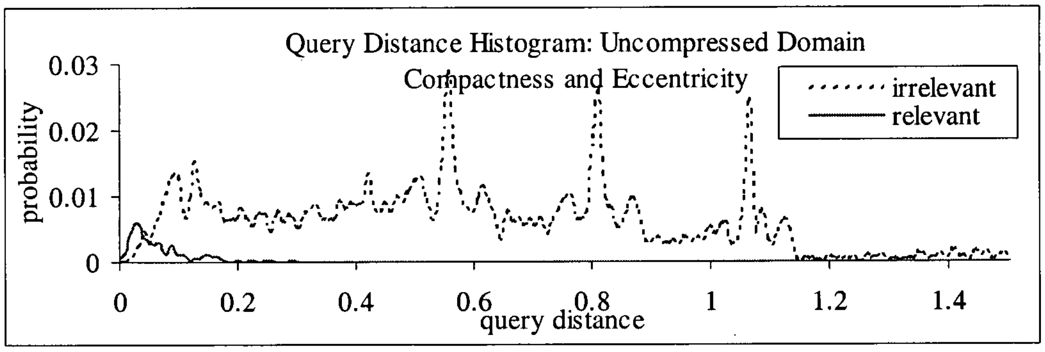


Figure 4.6: The query video object planes.

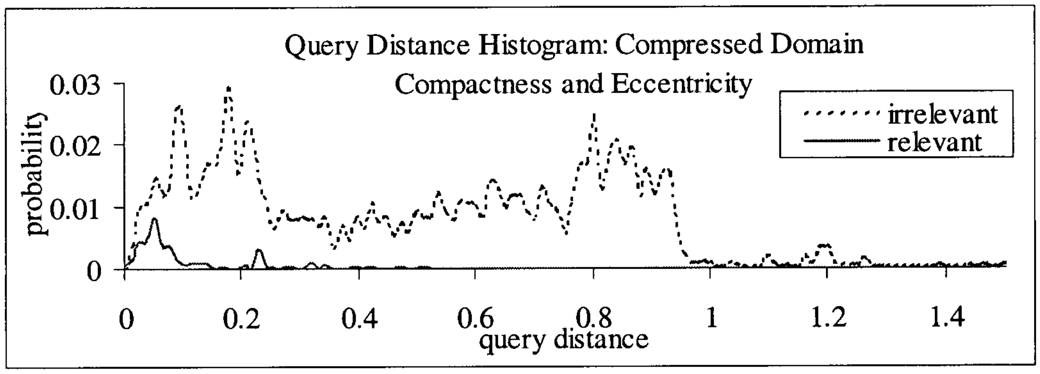
4.2.6.1 Retrieval with the global shape features

We first employ the global shape features to eliminate the most irrelevant shapes and refine the query results by employing the Fourier and/or the ART descriptors. Recall that we propose to use two global shape features, compactness and eccentricity. In this case, our global feature vector is $\vec{R} = [f_c \ f_e]$, where f_c and f_e are the normalized feature values of compactness and eccentricity, respectively. Alternatively, we can also employ the feature vector $\vec{R} = [f_i \ f_p]$, where f_i and f_p are the normalized inside intra density and pruned intra density feature values, respectively. The distance between the query feature vector, $\vec{R}_q$, and the database feature vector, $\vec{R}_d$, is computed by employing the L2 norm $d = \left\| \vec{R}_q - \vec{R}_d \right\|$ (i.e. Euclidean distance). If the distance between the query feature vector, $\vec{R}_q$, and the database feature vector, $\vec{R}_d$, is larger than a threshold, T_g, then the database VOP is concluded to be irrelevant. If the distance is smaller than T_g, then more accurate and more computationally complex features are employed for further refinement.

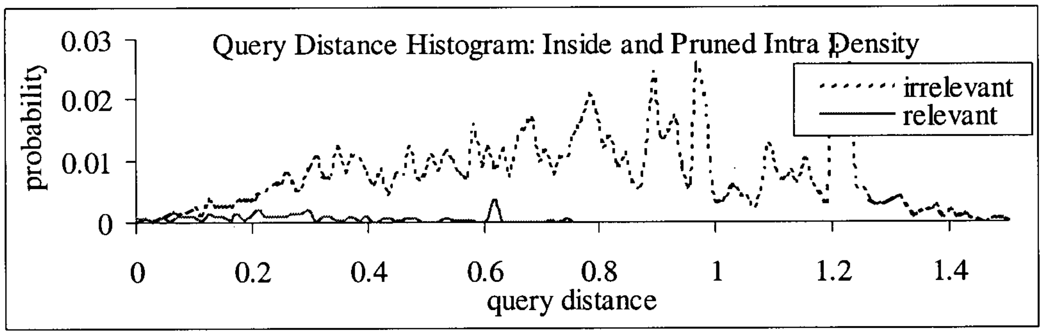
Figure 4.7 shows the distribution of the relevant and irrelevant database items versus the query distance for various global feature vectors. The number of relevant and irrelevant database items is found by performing queries with the four VOPs shown in Figure 4.6. The effectiveness of a global feature vector is determined by its ability to eliminate a large number of irrelevant database items without eliminating the relevant database items. The threshold T_g is selected such that it is small enough to save a significant amount of computations by marking most of the database items as irrelevant and at the same time it is large enough to prevent erroneous elimination of the relevant database items. The percentage of the eliminated irrelevant database items versus erroneously eliminated relevant items for different T_g values is presented in Table 4-2. Although it depends on the computation/accuracy requirements of the application, here we select T_g such that the probability of erroneously excluding a relevant item is less than 5%. Therefore, in average, we eliminate 88%, 64%, and 52% of the database items by employing $[f_c\ f_e]$ in the uncompressed domain and the compressed domain, and $[f_i\ f_p]$ in the compressed domain, respectively. Computational time required to process one VOP employing eccentricity and compactness features in uncompressed domain is 67 msec. Using our compressed domain compactness and eccentricity descriptors, the computational time is greatly reduced to 0.4 msec. and when inside intra density and pruned intra density features are employed, computation time is reduced to 0.3 msec.



(a)



(b)



(c)

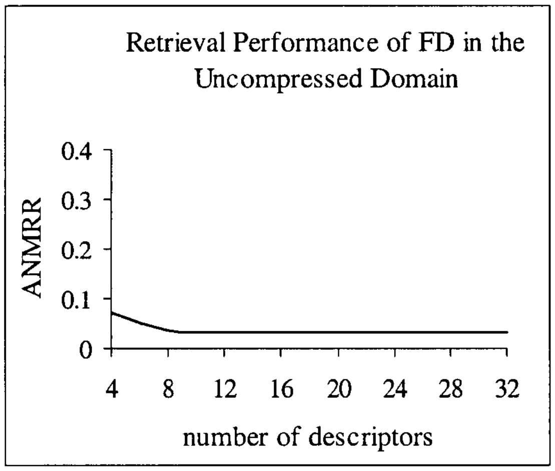
Figure 4.7: The distance histograms obtained by employing (a) compactness and eccentricity feature vector ($\vec{R} = [f_c \ f_e]$) in the uncompressed domain, (b) the same feature vector in the compressed domain, and (c) inside intra density and pruned intra density feature vector ($\vec{R} = [f_i \ f_p]$) in the compressed domain.

Feature vector	T_g	% of eliminated database items	% of erroneously eliminated relevant items
$\vec{R} = [f_c \; f_e]$ in the uncompressed domain	0.1	96	18
	0.2	88	3
	≥0.3	82	0
$\vec{R} = [f_c \; f_e]$ in the compressed domain	0.1	92	26
	0.2	78	16
	0.3	70	6
	0.4	64	1
	0.5	58	0.5
	≥0.6	51	0
$\vec{R} = [f_i \; f_p]$ in the compressed domain	0.1	99	85
	0.2	98	68
	0.3	94	42
	0.4	87	28
	0.5	80	21
	0.6	72	17
	0.7	63	8
	0.8	52	5
	0.9	42	4
	1	32	4

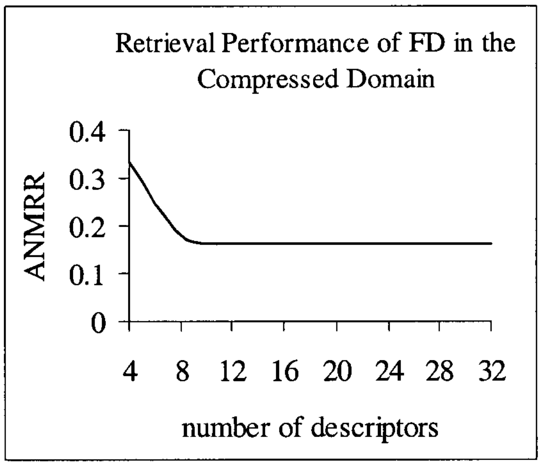
Table 4-2: The percentage of the eliminated irrelevant database items and the erroneously eliminated relevant items for different T_g values for a number of feature vectors.

4.2.6.2 Retrieval with the Fourier descriptors

Recall that we re-sample the video object boundary into 64 equally distanced points prior to Fourier Transformation and employ the scaled and re-ordered magnitudes of the first M Fourier coefficients as our Fourier descriptors. Selecting a large M would result in representing the finer details of the shape information, therefore better shape retrieval. Nevertheless, a larger number of Fourier descriptors require more storage bits. Moreover, representing the finer shape details accurately may make the retrieval algorithm more sensitive to the compressed domain shape approximation errors.



(a)



(b)

Figure 4.8: The retrieval performance using various numbers of the Fourier descriptors in the (a) uncompressed and (b) compressed domains.

The change in ANMRR with the number of Fourier descriptors employed for retrieval is presented in Figure 4.8.a and Figure 4.8.b for both the uncompressed and compressed domains, respectively. Recall that the lower values of ANMRR represent a better retrieval rate. As can be seen from the figures, selecting 8 Fourier descriptors, where NMRR=0.034 in the uncompressed domain and NMRR=0.17 in the compressed domain, offers a good tradeoff between the descriptor size and the retrieval performance.

Next, we present some example query results employing the first 8 Fourier descriptors. Our first query is the 72th VOP of the Singing Girl video object. There are twelve ground truth VOPs that are marked as similar to the query item in a database of more than 2000 VOPs. The first 12 retrieved VOPs are presented in Figure 4.9 and Figure 4.10, for employing the uncompressed and compressed domain Fourier descriptors, respectively. The NMRR measures are 0.12 for the uncompressed and 0.29 for the compressed domain features. As can be seen from the figure, even the VOPs that are considered as mismatch, e.g., the VOP from the Hall Monitor video object (ranked 12th in Figure 4.9), the VOP from the Stephan video object (ranked 10th in Figure 4.10), have in fact very similar shapes to the query. Also, the query VOP from the Stephan video object that is ranked 11th in Figure 4.10 is similar to the query, when rotated by 180^o (i.e. upside down). This also demonstrates the rotation invariance of this shape feature. Another example that demonstrates the rotation invariancy is given in Figure 4.11 where the retrieval results for the 224th VOP of the Coastguard video object is presented. Even though they are marked as mismatch, the VOPs that are ranked 63rd and 89th are very similar in shape to the query VOP when they are rotated by 90^o.

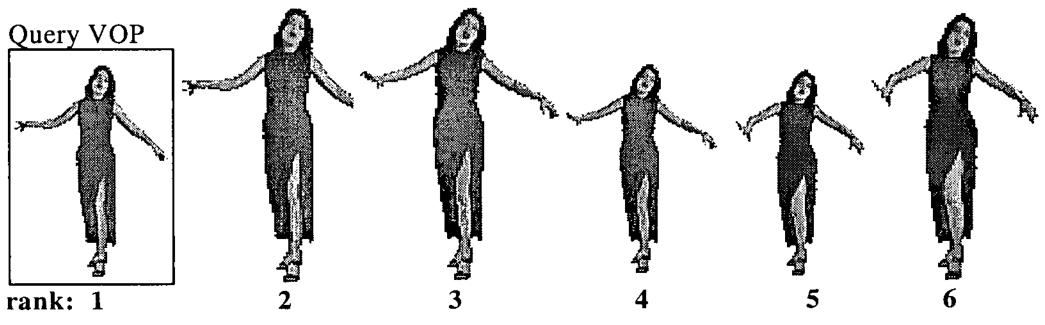


Figure 4.9: The shape retrieval results for the VOP query, the 72th VOP of the Singing Girl video object, employing the Fourier descriptors in the uncompressed domains.

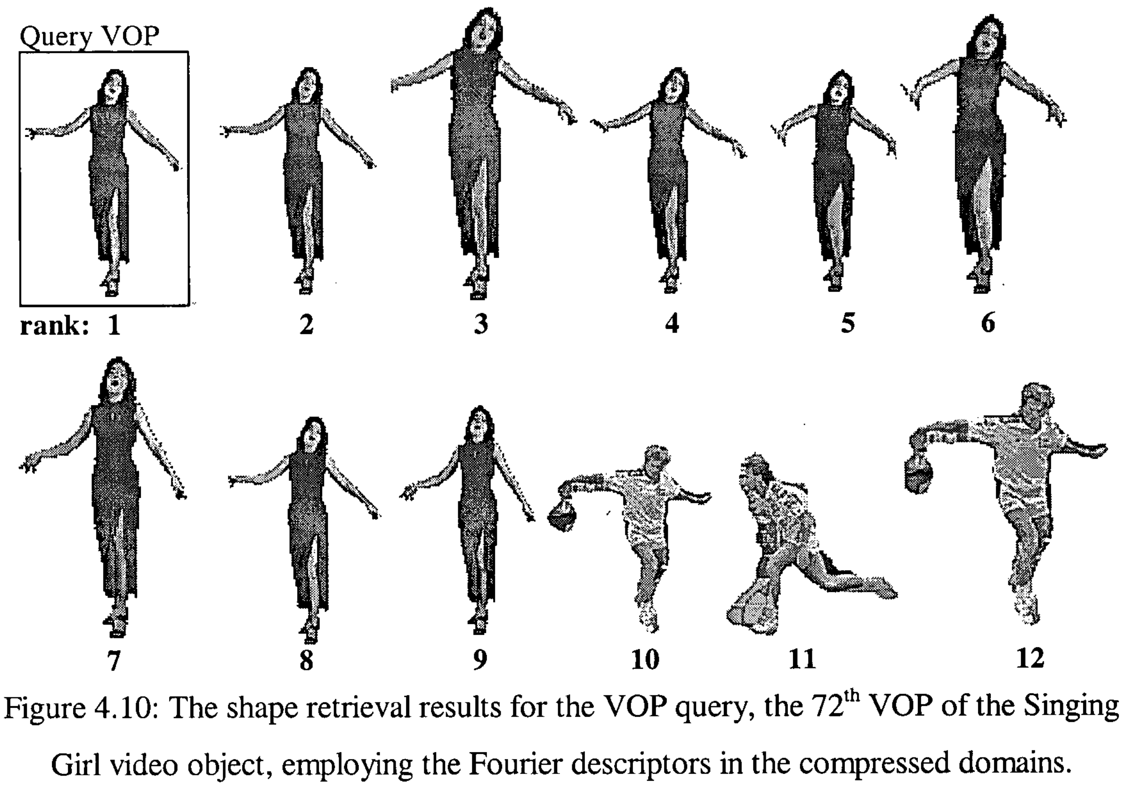


Figure 4.10: The shape retrieval results for the VOP query, the 72th VOP of the Singing Girl video object, employing the Fourier descriptors in the compressed domains.

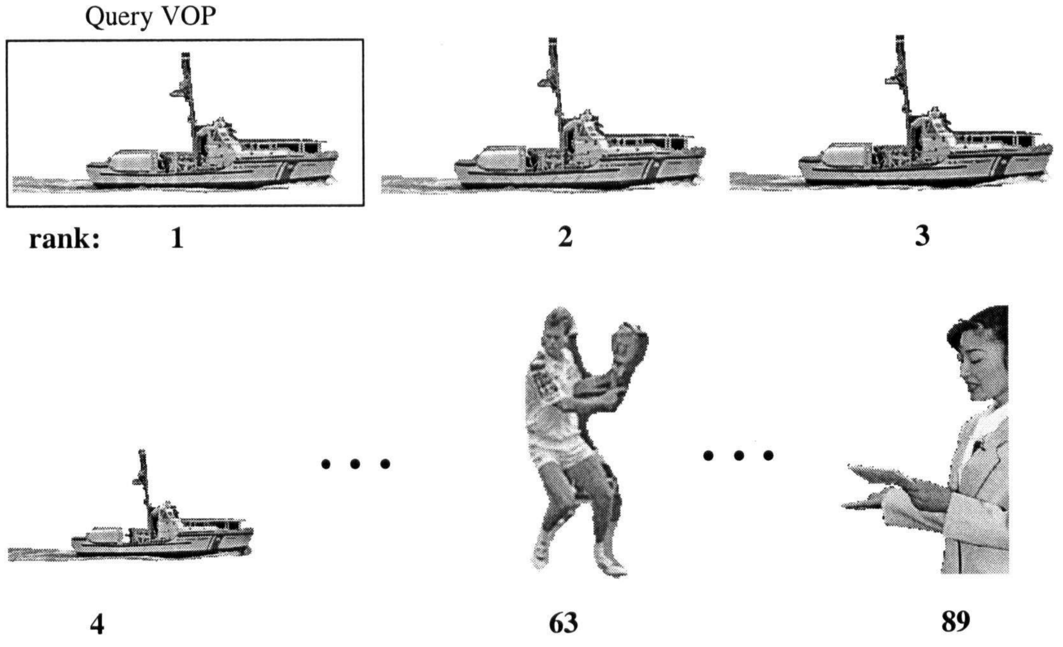
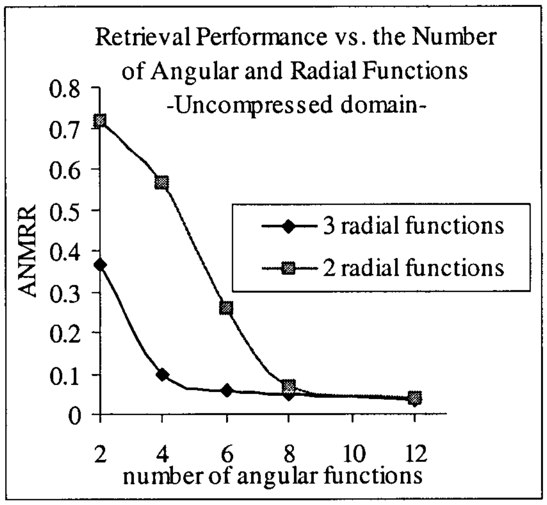


Figure 4.11: The shape retrieval results for the VOP query, the 224th VOP of the Coastguard video object, employing the Fourier descriptors in the compressed domain.

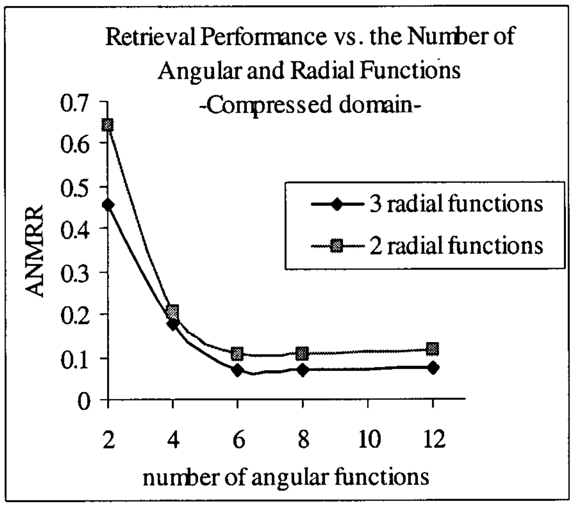
As can be observed from the retrieval performance results presented in Figure 4.8, Figure 4.9, and Figure 4.10, retrieval with the descriptors computed in the uncompressed domain has better accuracy. Our experiments show that the compressed domain descriptors may fail to correctly classify shapes if the resolution of the video object is very small. This problem could be solved by computing the uncompressed domain descriptors if the video object is smaller than a certain size. The main advantage of using the compressed domain descriptor is the substantial reduction in number of computations. The computation time required computing and matching one VOP shape in the compressed domain is approximately 13 msec., which is more than an order of magnitude smaller than the 392 msec. required for performing the same operation in the uncompressed domain.

4.2.6.3 Retrieval with the ART descriptors

The number of angular and radial functions of the ART descriptor determines how accurately the shape is represented. Using a smaller number results in less accurate description but also in a more compact descriptor and faster feature extraction and shape matching. The retrieval performance achieved by using different numbers of angular and radial functions is presented in Figure 4.12 for the uncompressed and compressed domain ART descriptors. As can be observed from the figure, for the uncompressed domain implementation, employing 8 angular and 2 radial functions (ANMRR=0.067), 4 angular and 3 radial functions (ANMRR=0.099), and 6 angular and 3 radial functions (ANMRR=0.057) offer the best tradeoff for the descriptor size and the retrieval performance. Here, we favor the retrieval accuracy and use 6 angular and 3 radial functions for the ART descriptors. In the compressed domain, as shown in Figure 4.12.b, clearly employing 6 angular and 2 radial functions (ANMRR=0.109), and 6 angular and 3 radial functions (ANMRR=0.069) offer the best tradeoffs. Here, we adopt 6 angular and 3 radial functions for the retrieval in the compressed domain as well.



(a)



(b)

Figure 4.12: The change in retrieval rate based on the number of angular and radial functions of the ART descriptors in the (a) uncompressed and (b) compressed domains.

Next, we present the shape retrieval results obtained by querying the 110th VOP of the Hall Monitor video object. There are 8 ground truth VOPs marked as similar to the query VOP item in more than 2000 VOPs. Figure 4.13 and Figure 4.14 show the first 8 retrieved VOPs using the ART descriptors (with 6 angular and 3 radial functions) in the uncompressed and compressed domains, respectively. The NMRR value for the uncompressed domain implementation is 0.078 and that of the compressed domain implementation is 0.033. Extracting and matching the compressed domain descriptors require 39 msec. per VOP versus the 210 msec. required by uncompressed domain descriptors. The compressed domain features offer excellent computational savings with only a small decrease in the retrieval accuracy.

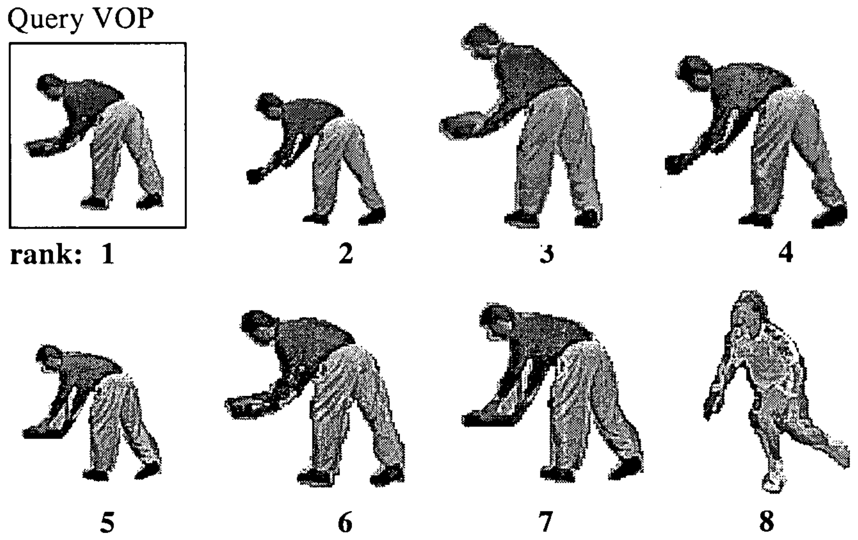


Figure 4.13: The shape retrieval results for the VOP query, the 110th VOP of the Hall Monitor video object, employing the ART descriptors in the uncompressed domain.

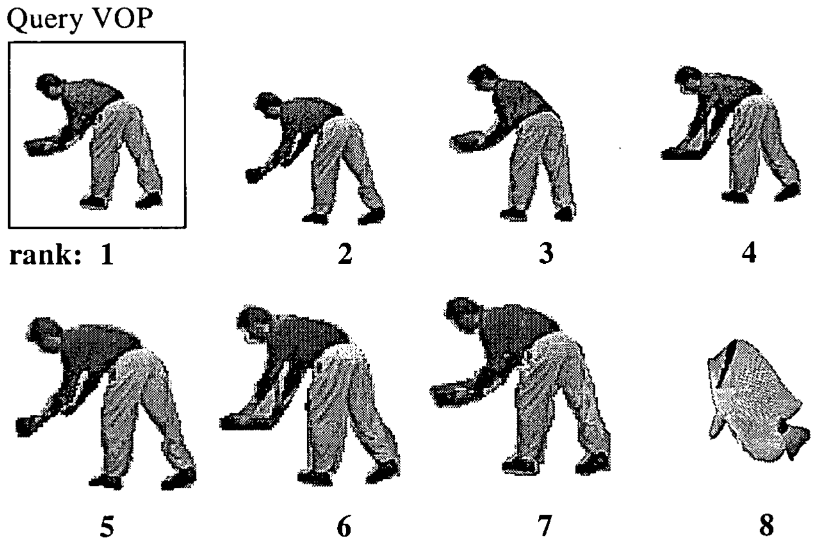


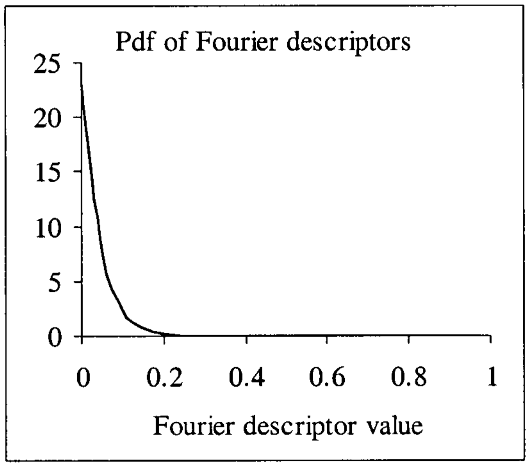
Figure 4.14: The shape retrieval results for the VOP query, the 110th VOP of the Hall Monitor video object, employing the ART descriptors in the compressed domain.

4.2.7 Efficient quantization and representation of the descriptor values

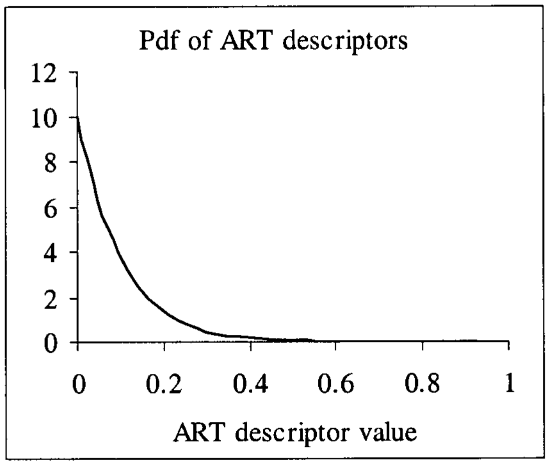
The shape features that are described in the previous section can be extracted from the bit stream at the time of the query or they can be computed prior to the query and attached to the MPEG-4 bit stream as user data. Computing the features offline and storing them for future queries is a commonly used method in database applications. It was shown in the literature that for coding of an average resolution video object, the number of bits required to represent a video object plane in MPEG-4 is approximately 10 Kbits [41]. Representing the Fourier and ART descriptors (which are float numbers) with 4 bytes each, and storing 8 Fourier and 18 ART descriptors in the bit stream would require 832 bits of side information. This corresponds to almost 10% of the coded video information. If these descriptors are quantized and only the quantization values are stored in the memory, or the bit stream, then great savings in memory would be possible. For example, if the descriptors are quantized into 16 levels, then the Fourier and ART descriptors can be stored into 4 (instead of 32) and 9 (instead of 72) bytes respectively, requiring only 104 bits of side information.

One solution here is to employ uniform quantization, without considering the probability distribution of these values. The ART and Fourier descriptor values show an exponential distribution with a steep curve as presented in Figure 4.4 and more than half of the descriptor values lie within 10% of the dynamic range. Therefore, if uniform quantization is employed, only a small range of the bits will be actually used to represent a large range of descriptor values. This would result in a degradation in the retrieval performance. A better solution is to employ non-uniform quantization based on the probability distributions of Fourier and ART descriptor values.

In order to design a non-uniform quantizer for the Fourier and ART descriptors, we first need to find the continuous probability density functions (pdf) associated with the each descriptor. The pdf of these descriptors can be modeled in the form of exponential distribution, i.e. $\alpha e^{-\alpha x}$, where E(X)=1/α and σ^2(X)= 1/α^2. We find the values for the parameters α_F, which defines the pdf of the Fourier descriptors and α_A, which defines the pdf of the ART descriptors, by using non linear regression function of MATLAB, which is based on minimizing the sum of squares of the residuals and employing the Gauss-Newton algorithm with the Levenberg-Marquardt modifications for global convergence [86]. The resulting values for α_F and α_A are 23 and 10, respectively. The corresponding pdf functions are presented in Figure 4.15.



(a)



(b)

Figure 4.15: The pdfs of the (a) Fourier and (b) ART descriptor values.

The most efficient quantizer is the one that assigns equal probability to each of the quantization bins. Given that the Fourier and ART descriptor probability density functions are modeled by $\alpha e^{-\alpha x}$ and the number of desired quantization levels is *L*, the optimum quantization ranges can be found by

$$\int_{q_i}^{q_{i+1}} \alpha e^{-\alpha x}\, dx = \frac{1}{L} \qquad 0 \le i \le L, \quad q_0 = 0, \tag{4.14}$$

where q_i and q_{i+1} are the lower and upper boundaries of the quantization ranges, respectively. The optimum reconstruction levels of the quantizer should be selected such that $P(q_i<x<=X_R) = P(X_R <x<=q_{i+1})$, i.e. the probability of a descriptor value being between the lower quantization bound and the reconstruction level (X_R) is equal to it is being between the reconstruction level and the upper quantization bound. Given this constraint, the reconstruction levels, X_R, for each quantization range are found by

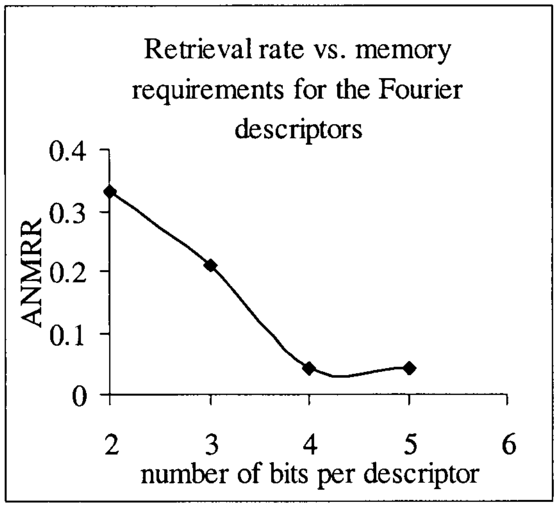
$$\int_{q_i}^{X_R} \alpha e^{-\alpha x}\, dx = \int_{X_R}^{q_{i+1}} \alpha e^{-\alpha x}\, dx \qquad => \tag{4.15}$$

$$-e^{-\alpha x X_R} + e^{-\alpha x q_i} = -e^{-\alpha x q_{i+1}} + e^{-\alpha x X_R} \qquad => \tag{4.16}$$

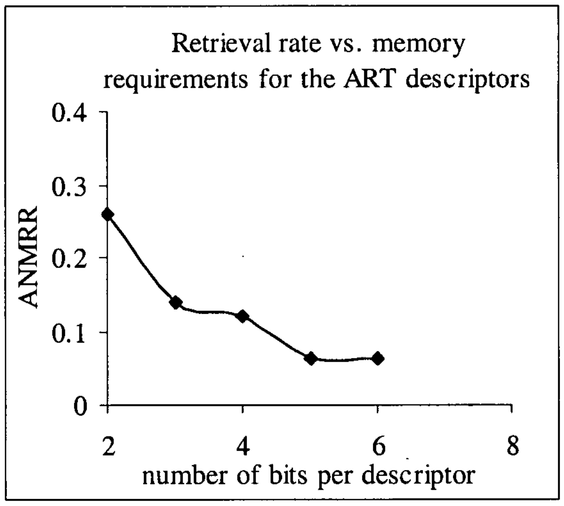
$$X_R = \frac{\ln 2 - \ln(e^{-\alpha x q_i} + e^{-\alpha x q_{i+1}})}{\alpha}. \tag{4.17}$$

The number of quantization bins, L, is optimized based on the tradeoff of retrieval performance and descriptor size. L should be selected such that the number of bits to represent each level should be minimal without causing much degradation in the retrieval rate. The change in the retrieval rate for different quantization step sizes employing the

Fourier descriptors is illustrated in Figure 4.16.a. Here, using 4 bits to represent one descriptor corresponding to L=16 offers a compact representation of the descriptor with minimal quality degradation. The retrieval rate and the quantization step size tradeoff for the ART descriptors is presented in Figure 4.16.b. As can be seen from the figure, selecting the number of descriptor bits to be 3 (i.e. L=8) or 5 (i.e. L=32) appears to offer a good tradeoff between the storage requirements and the retrieval rate. Alternatively, we can use 4 bits per descriptor, where it would be possible to save two descriptors in a byte and the storage of and access to the data would be potentially easier. The quantization ranges and the reconstruction levels of the Fourier and ART descriptors for L=16 are presented in Table 4-3.



(a)



(b)

Figure 4.16: The change in the retrieval rate when different bit rates are employed to represent the (a) Fourier (b) ART descriptors.

Fourier Descriptors			ART Descriptors		
quantizer lower bound	quantizer upper bound	reconstruction level	quantizer lower bound	quantizer upper bound	reconstruction level
0	0.0028	0.0014	0	0.0065	0.0032
0.0028	0.0058	0.0043	0.0065	0.0134	0.0098
0.0058	0.009	0.0074	0.0134	0.0208	0.017
0.009	0.0125	0.0107	0.0208	0.0288	0.0247
0.0125	0.0163	0.0144	0.0288	0.0375	0.033
0.0163	0.0204	0.0183	0.0375	0.047	0.0421
0.0204	0.025	0.0227	0.047	0.0575	0.0521
0.025	0.0301	0.0275	0.0575	0.0693	0.0633
0.0301	0.0359	0.0329	0.0693	0.0827	0.0758
0.0359	0.0426	0.0392	0.0827	0.0981	0.0901
0.0426	0.0506	0.0464	0.0981	0.1163	0.1068
0.0506	0.0603	0.0552	0.1163	0.1386	0.1269
0.0603	0.0728	0.0661	0.1386	0.1674	0.152
0.0728	0.0904	0.0807	0.1674	0.2079	0.1856
0.0904	0.1205	0.1029	0.2079	0.2773	0.2367
0.1205	inf	0.1507	0.2773	inf	0.3466

Table 4-3: The quantization ranges and the reconstruction levels of the Fourier and ART descriptors for L=16.

4.3 Video Object Retrieval by Shape Features

Similar to a video sequence being a collection of two dimensional still images (frames), an arbitrarily shaped video object is a collection of two dimensional video object planes. Therefore, still shape matching techniques can be used to retrieve video objects. Here, we propose to find the shape similarity of two VOs via comparing the shape feature vectors of their VOPs using a new similarity measure. The proposed distance measure requires that for every VOP of VO_A, we find the smallest distance to any key VOP in VO_B. Then the summation of these distances is divided by the number of key VOPs in VO_A to obtain the distance between two video objects. This distance measure is asymmetric, i.e. the distance from VO_A to VO_B is not equal to the distance between VO_A to VO_B. Therefore, we define the final distance as the maximum of two distances as follows,

$$d(VO_A, VO_B) = \left(\max \frac{1}{N} \sum_{i}^{N} \min_{VOP_a \in VO_A} \{ d_{vop}(VOP_{b_i}, VOP_a) \}, \right. \\ \left. \frac{1}{M} \sum_{k}^{M} \min_{VOP_b \in VO_b} \{ d_{vop}(VOP_{a_k}, VOP_b) \} \right), \quad (4.18)$$

where M and N are the number VOPs of video object A (VO_A) and video object B (VO_B), respectively. $d_{vop}(VOP_a, VOP_b)$ is the shape distance between VOP_a and VOP_b, and computed as $d_{vop}(VOP_a, VOP_b) = \left\| \vec{R}_a - \vec{R}_b \right\|$, where $\vec{R}_a$ and $\vec{R}_b$ are the shape feature vectors of the VOP_a and VOP_b, respectively.

4.3.1 Reducing the video object content redundancies

Considering that there may be hundreds of VOPs even in several seconds long video objects, the computation of the distance between two video object shapes could be very computationally intensive. Here, we propose to compute the above distance measure on a subset of VOPs. A straightforward way to obtain such a subset would be temporally sampling the IVOPs in a video object. However, using this method, some important changes of the video object shape content could be missed. A much efficient way of obtaining a subset of VOPs would be employing one of the key VOP extraction algorithms described in Chapter 3 as they select the VOPs that represent the salient shape content of the video object.

The number of key VOPs obtained by these algorithms depends on how much change there is in the shape of a video object during its existence [87]. However, in some cases it might be desired to have an upper limit to the number of VOPs to be compared in order to limit the number of computations. We can achieve this by employing the K-means clustering algorithm [75] and using the already computed shape feature vectors for classifying the key VOPs as presented in Figure 4.17. The computational overhead introduced by summarization of the video content could be omitted considering that it needs to be performed only once when adding a video object to the database.

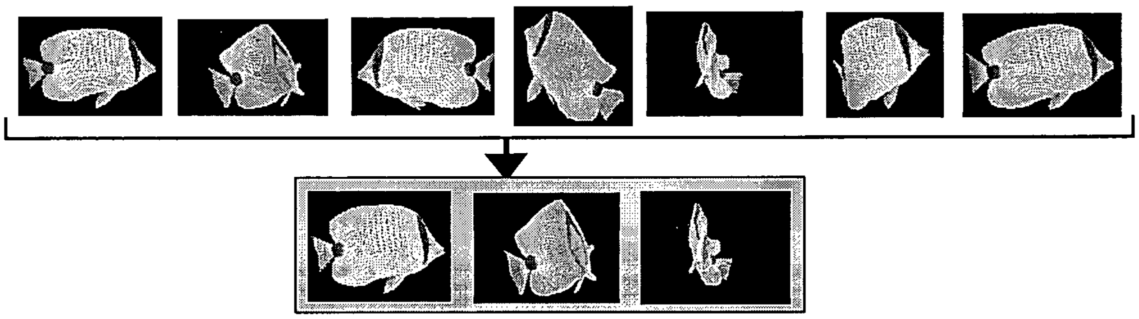


Figure 4.17: Further summarization of the Bream video object into 3 VOPs with K-means clustering.

4.3.2 Retrieval results

In this section, we demonstrate the effectiveness of our proposed video object shape matching method. We perform our experiments on a database of more than 50 video object bit streams. We compare only the key VOPs of the query and database video objects. These key VOPs are found by using the Hausdorff distance based algorithm that is proposed in Chapter 3. Then these key VOPs are clustered such that not more than 5 VOPs represent a video object. The ANMRR values presented here are found by averaging the NMRR values obtained by querying 8 video objects, including the ones shown in Figure 4.18.

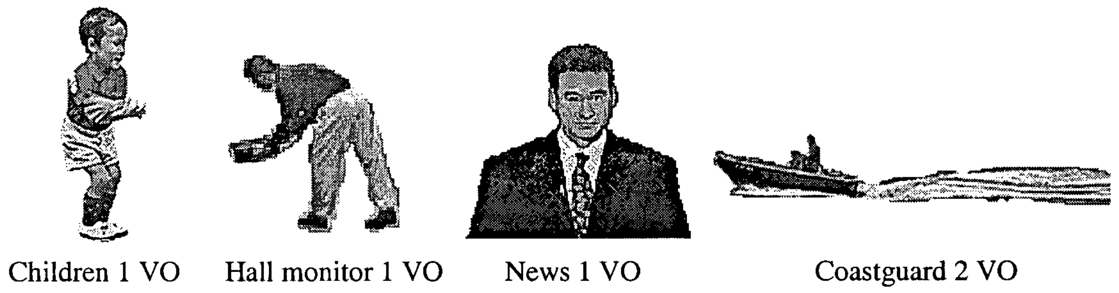


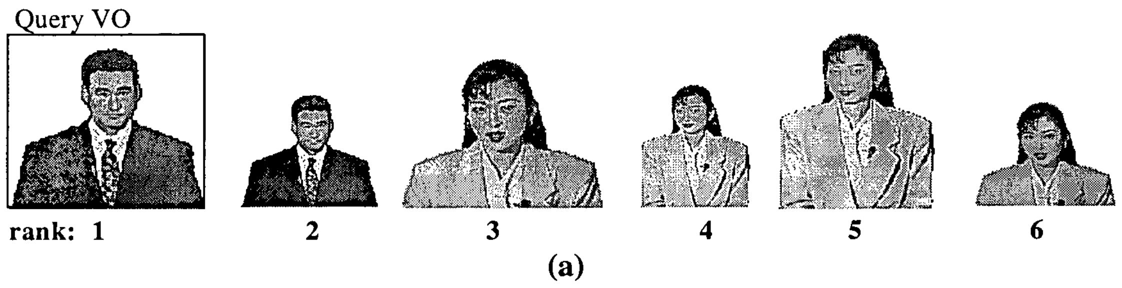
Figure 4.18: Example of query video objects.

The proposed Fourier and ART descriptors can be used individually or together for efficient retrieval. Table 4-4 presents the retrieval performance for each case. As can be expected, combining the Fourier and ART descriptors generally provides better retrieval performance than employing them individually. Also in Table 4-4, we compare the results obtained using our proposed VO shape distance measure (see Section 4.2.1) with the distance measure commonly used in the current systems, i.e. averaging of the feature vectors. As can be seen from the table, our proposed method consistently results in better retrieval performance.

VO shape distance measure	Domain	Fourier descriptors ANMRR	ART descriptors ANMRR	Combined descriptors
Proposed Key VOP comparison	Compressed	0.194	0.139	0.134
	Uncompressed	0.123	0.130	0.103
Averaging	Compressed	0.213	0.266	0.190
	Uncompressed	0.172	0.181	0.168

Table 4-4: The retrieval performance results for Fourier, ART, and a combination of the descriptors.

Figure 4.19 and Figure 4.20 present some example query results using the Fourier and ART descriptors together in both uncompressed and compressed domains. Note that each of the query and database video objects contains approximately 300 VOPs and only the representative VOPs are shown in the figures. Also note that the key VOP selection is performed separately on the each different resolution of the video object that is present in the database, resulting in a different set of key VOPs for the each resolution of the video object. Extracting and matching the Fourier and ART descriptors require 52 msec. in the compressed domain and 602 msec. in the uncompressed domain for each VOP. Considering that these features only need to be computed and matched for the key VOPs and there are a maximum of 5 key VOPs per video object, it would take approximately 260 msec. to process one video object in the compressed domain and approximately 3 seconds to process it in the uncompressed domain. The compressed domain processing clearly provides a very good computational advantage.



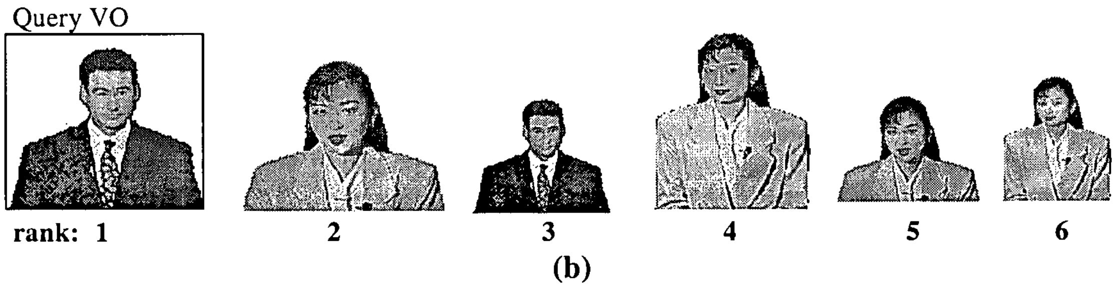


Figure 4.19: The shape retrieval results for the News 1 video object query in the (a) uncompressed (b) compressed domains.

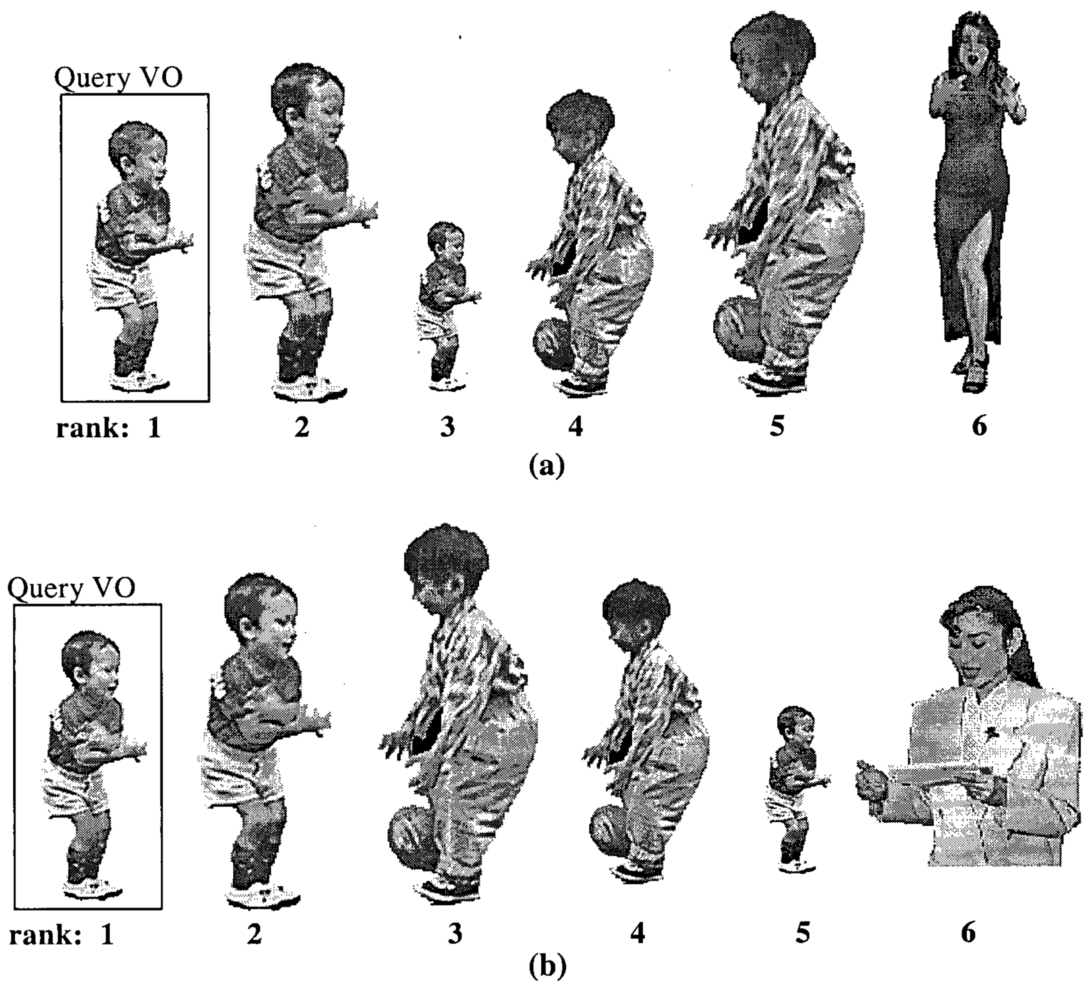


Figure 4.20: The shape retrieval results for the Children 1 video object query in the (a) uncompressed (b) compressed domains.

4.4 Conclusions

Matching of video objects based on their shape information is a very important component of any video object retrieval system. In this chapter, we proposed a method to compute some effective shape features in the compressed domain. Besides addressing the issues related to the retrieval and computational performance tradeoffs, we also addressed the efficient quantization and the compact representation of the proposed shape features. Moreover, we proposed a technique to extend the use of our still shape retrieval methods to arbitrarily shaped video object retrieval through the use of key VOPs. Our experiments show that compressed domain ART descriptors offer better retrieval rates than those of the Fourier descriptors. The ART descriptors can be used individually or together with Fourier descriptors for efficient shape retrieval based on the computation, storage, and accuracy requirements of the target applications. In either case, these compressed domain features offer excellent computational savings, i.e. approximately 99% in the case of Fourier and 90% in the case of ART descriptors, with only a small degradation in the retrieval performance compared to their uncompressed domain counterparts.

Chapter 5

Local Motion Descriptors

"[The body is] a marvelous machine... a chemical laboratory, a power-house. Every movement, voluntary or involuntary, full of secrets and marvels!"

Theodor Herzl (1860-1904)

Unlike still images, video has a temporal dimension that we associate with some motion information. We use this information as one of the key components to describe video sequences, for example "in this video my daughter was waving" or "part of this video contains my son playing basketball". Consequently, motion features play an important role in content-based video retrieval. We can classify the types of motion features into three groups as follows:

- Global motion of video or camera motion (e.g. camera zoom, pan, tilt).
- Global motion of the video objects within a frame (e.g. an object is moving from left to the right in the scene).

- Local motion of the video object (e.g., a person is raising his/her arms).

The camera operation analysis is generally performed by analyzing the directions of the motion vectors that are present in the compressed video bit stream [88][89] or the motion vectors obtained by optical flow techniques in the spatial domain [90]. For example, panning camera motion is likely to be present if most of the motion vectors inside a frame are in the same direction. Similarly, zooming camera motion can be identified by detecting if the motion vectors at the top/left of the frame have opposite directions than the motion vectors at the bottom/right of the frame [91][93].

Global motion of video objects is typically represented with their motion trajectories, which are formed by tracking the location of video objects (generally the object's mass center or some selected points on the object) over a sequence of frames. Forming motion trajectories generally requires segmentation of video objects in a video scene. In MPEG-4, the location information of the video object bounding box (the upper-left corner) is already available in the bit stream making the formation of the trajectory a simple task [94]. The classification and matching of object motion trajectories is a challenging issue as the trajectories contain both the path and the velocity information of the objects. In [95], Little et al. proposed to extract separate curves for the object path and speed, and match these two components separately. Furthermore, Rangarajan et al. demonstrated two-dimensional motion trajectory matching through scale-space [96] and Chang et al. proposed to match the motion trajectories via a wavelet decomposition [97].

Most available content-based video retrieval systems in the literature employ camera motion features and/or global object motion for retrieval by motion. For example, the Jacob [19] system supports queries using common camera motion changes, such as pan, zoom, and tilt. Another retrieval system, VideoQ, employs a spatio-temporal segmentation algorithm in order to retrieve individual objects with their global motion inside a scene [22]. It allows the user to specify an arbitrary polygonal trajectory for the query object and retrieves the video sequences that contain video objects with similar trajectories. Similar to VideoQ, NeTra-V supports spatio-temporal queries and utilizes motion histograms for global camera and video object motion retrieval [21]. Moreover, the content-based description standard MPEG-7 [27]-[33] supports motion descriptors, in particular camera motion, which characterizes the 3-D camera operations, motion trajectory, which captures 2-D transitional motion of objects, parametric motion, which describes the global deformations, and motion activity, which specifies the intensity of action.

On the other hand, processing database queries such as "find soccer video sequences where the player is scoring" and "find a video sequence where people are salsa dancing" would be possible only by enabling the retrieval of video objects with their local motion. The current research in detecting the local motion of video objects has been restricted mostly to specific domains. Stalidis *et al.* employed a wavelet-based model using boundary points of MRI images to describe the cardiac motion in [101]. Miyamori *et al.* proposed to classify of the actions of tennis players by using 2-D appearance-based matching [102]. Hoey *et al.* suggested a method for classification of motion, which is based on representation of flow fields with Zernike polynomials in [103]. Their method is applied to the classification of facial expressions. In [104], Fujiyoshi *et al.* presented a process to analyze human motion by first obtaining the skeleton of the objects and then determined the body posture and motion of skeleton segments to determine human activities. Human motion classification was also studied by other researchers, including Little *et al.* in [105], where they proposed to recognize individuals by periodic variation in the shape of their motion, and by Heisele *et al* in [106], where they suggested discriminating pedestrians by characterizing the motion of the legs. Moreover, Cutler *et al.* propose to characterize the local motion by detecting periodicity of the motion by Fourier analysis on the gray scale video [107]. Most of the work in this area focuses on "recognizing" the motion of specific objects and they assume prior knowledge about the video content.

In this chapter, we propose three generic (content independent) descriptors that describe the local motion of video objects for video retrieval. Motivated by the fact that any significant motion of video objects within their bounding box would very likely result in changes in their shape, our motion descriptors are based on the shape deformations of video objects. Recall that the Fourier and Angular Radial Transform (ART) coefficients efficiently describe shape information as presented in Chapter 4. Here, we first propose to employ the variances of Fourier and ART coefficients to identify the changes in the shape of video objects. Our Fourier coefficient variance based descriptor captures the deformations at the object's boundary whereas the ART coefficient variance based descriptor captures the changes in the object's region. An important point to note here is that the location information is distributed among the transform coefficients after performing either the Fourier or ART transforms. Therefore, the exact location of the object motion would not be captured in these descriptors. To address this issue, we propose a third descriptor, the Angular Circular Local Motion (ACLM) descriptor, which is extracted by dividing the video object area into a number of angular and circular segments and computing the variance of each segment over a period of time. All the proposed descriptors here can be derived directly from the MPEG-4 compressed domain or computed using the binary shape masks of the video objects in the spatial domain.

The rest of this chapter is as follows. In the next three sections, we describe the three proposed local motion descriptors, as well as their extraction and matching in the uncompressed and compressed domains. Section 5.4 presents the experimental results that illustrate the retrieval performance of our methods and the tradeoffs associated with extracting the proposed features in different domains. A comparison of the proposed descriptors and conclusions are given in Section 5.5.

5.1 Fourier Transform Based Local Motion Descriptor

Fourier coefficients efficiently represent the boundary of arbitrary shaped objects as we presented in Chapter 4. As the shape of a video object changes with time, the magnitude of these coefficients would also vary. Considering that the amount of variation in different

Fourier coefficients corresponding to different frequencies would be a good classifier of different types of motion, we propose a motion descriptor based on the variances of the Fourier coefficients. The computation of our proposed descriptor is as follows. We first compute the complex Fourier coefficients of individual VOPs as described in Chapter 4 in either the compressed or uncompressed domains. Next, a Fourier vector, $\vec{F} = [F_1\, F_2\, F_3\, F_4\, \ldots\, F_M]$, is formed for each VOP, where F_m is the mth Fourier coefficient. Then the proposed local motion descriptor is computed by

$$\vec{F}_{\sigma^2} = \frac{1}{K} \sum_{k=0}^{K} \left(\vec{F}_k - \vec{F}_\mu\right)^2, \tag{5.1}$$

$$\vec{F}_\mu = \frac{1}{K} \sum_{k=0}^{K} \vec{F}_k, \tag{5.2}$$

where K is the number of VOPs in a video object, $\vec{F}_\mu$ is the mean Fourier vector and $\vec{F}_k$ is the Fourier vector of the kth VOP. The mean Fourier vector, F_μ,, is the average of the feature vectors of all VOPs in a video object.

Recall that the complex Fourier coefficients of individual VOPs are rotation and scale variant and their scaled magnitude is employed in still shape retrieval in order to provide rotation and scale invariance. Here, the above variances are computed prior to any scaling on both real and complex components of Fourier coefficients. This way, any size changes and rotation of the individual VOPs would be captured by our descriptor. Nevertheless, it is still desired that the final motion descriptor is invariant of the video object size and rotation. We provide scale invariance by dividing each element of $\vec{F}_{\sigma^2}$

with $\max_{VOP_k \in VO} \left\{ F_1(VOP_k)^2 \right\}$, where $F_1(VOP_k)$ is the first non zero Fourier coefficient of the kth VOP. Note that $F_1(VOP_k)$ gives the radius of the circle with an area equal to that of the kth VOP. Rotation invariance is ensured by employing the magnitude of the resulting complex coefficients as shown below:

$$F_k = \frac{1}{N} \sum_{n=0}^{N-1} z_n \, e^{-\frac{2j\pi n k}{N}}, \tag{5.3}$$

$$F_k^{\alpha} = \frac{1}{N} \sum_{n=0}^{N-1} \left(z_n \, e^{\alpha} \right) e^{-\frac{2j\pi n k}{N}} \Rightarrow F_k^{\alpha} = e^{\alpha} F_k, \tag{5.4}$$

$$\vec{F}^{\alpha}_{\sigma^2} = \frac{1}{K} \sum_{k=0}^{K} \left(\vec{F}_k^{\alpha} - \vec{F}_{\mu}^{\alpha} \right)^2, \tag{5.5}$$

$$\vec{F}^{\alpha}_{\sigma^2} = \frac{1}{K} \sum_{k=0}^{K} \left(e^{\alpha} \vec{F}_k - e^{\alpha} \vec{F}_{\mu} \right)^2 = \frac{1}{K} \sum_{k=0}^{K} e^{2\alpha} \left(\vec{F}_k - \vec{F}_{\mu} \right)^2, \tag{5.6}$$

$$= e^{2\alpha} \vec{F}_{\sigma^2} \Rightarrow \left| \vec{F}^{\alpha}_{\sigma^2} \right| = \left| \vec{F}_{\sigma^2} \right|. \tag{5.7}$$

The number of Fourier coefficients, M, employed to compute our proposed descriptor determines the type of motion we capture. Employing a lower number would allow capturing of only very low frequency motion, i.e. the type of motion that would affect the shape of the video object globally. Selecting a high number of descriptors would make it possible to capture the motion more accurately, but would potentially make the feature more sensitive to segmentation noise.

5.2 Angular Radial Transform Based Local Motion Descriptor

We already demonstrated the effectiveness of the ART coefficients in shape retrieval in Chapter 4. Here, we propose to use the variances of the ART coefficients, computed using each VOP of a video object, to describe the object's local motion.

Using the ART coefficients, the proposed local motion descriptor matrix, ***R***, is computed as follows,

$$\boldsymbol{R} = \begin{bmatrix} a_{0,0} & \dots a_{0,m} \dots & a_{0,M-1} \\ \vdots & \vdots & \vdots \\ a_{n,0} & \dots a_{n,m} \dots & a_{n,M-1} \\ \vdots & \vdots & \vdots \\ a_{N-1,0} & \dots a_{N-1,m} \dots & a_{N-1,M-1} \end{bmatrix}, \tag{5.8}$$

$$a_{nm} = \frac{1}{K}\sum_{k=0}^{K-1}(A_{nm}(VOP_k) - \mu_{nm})^2 , \tag{5.9}$$

$$\mu_{nm} = \frac{1}{K}\sum_{k=0}^{K-1} A_{nm}(VOP_k) , \tag{5.10}$$

where K is the number of VOPs in a video object, N and M are the number of angular and radial ART functions, respectively, and $A_{nm}(VOP_k)$ is the ART coefficient of order n and m of the kth VOP.

Here, similar to the Fourier descriptors, the rotation variant complex ART coefficients are employed for the variance computation in order to capture the rotation of video objects. Finally, the magnitudes of these variances are used as rotation invariant descriptors.

5.3 Angular Circular Local Motion (ACLM) Descriptor

The variance of the object area offers valuable information about the amount of shape deformation or local motion present in a video object. Motivated by this fact, we propose to divide the shape mask of a video object into M angular and N circular segments and use the variances of the pixels that fall into each segment to describe the local motion. Unlike the Fourier and ART based descriptors, where the location information is distributed among the transform coefficients, this descriptor captures the location as well as the type of the object's motion.

After the video object is divided into M angular and N circular segments, as illustrated in Figure 5.1, we first compute the variances for each angular circular segment in the temporal direction using the VOPs of the video objects. Then the local motion feature matrix is formed for each video object as follows:

$$\boldsymbol{R} = \begin{bmatrix} \sigma_{0,0}^2 & \ldots\sigma_{0,m}^2\ldots & \sigma_{0,M-1}^2 \\ \vdots & \vdots & \vdots \\ \sigma_{n,0}^2 & \ldots\sigma_{n,m}^2\ldots & \sigma_{n,M-1}^2 \\ \vdots & \vdots & \vdots \\ \sigma_{N-1,0}^2 & \ldots\sigma_{N-1,m}^2\ldots & \sigma_{N-1,M-1}^2 \end{bmatrix}, \tag{5.11}$$

where M and N are the number of angular and circular sections, respectively. $\sigma_{n,m}^2$ is the normalized variance of the pixels that fall into the segment (n,m) and computed as follows,

$$\sigma_{n,m}^2 = \frac{1}{S(n,m)^2 K} \sum_{k=0}^{K-1} \left(\mathrm{P}_{n,m} - \mu_{n,m} \right)^2 , \tag{5.12}$$

$$\mu_{n,m} = \frac{1}{K} \sum_{k=0}^{K-1} \mathrm{P}_{n,m} , \tag{5.13}$$

$$\mathrm{P}_{n,m} = \sum_{\theta=\theta m}^{\theta_{m+1}} \sum_{\rho=\rho_n}^{\rho_{n+1}} VOP_k(\rho,\theta), \tag{5.14}$$

where K is the number of the VOPs of the video object, VOP_k is the binary shape map of the video object at kth instant, $VOP_k(\rho,\theta)$ is the value of the binary shape mask in VOP_k at the position (θ, ρ) in the polar coordinate system centered at the mass center of VOP_k. $S(n,m)$ is the area, θ_m is the start angle, ρ_n is the start radius of the angular circular segment (n,m) and they are defined as

$$S(n,m) = \frac{\pi(\rho_{n+1}^2 - \rho_n^2)}{M}, \; \theta_m = m \times \frac{2\pi}{M}, \; \rho_n = n \times \frac{\rho_{\max}}{N}, \tag{5.15}$$

where M and N are the number of angular and circular sections, respectively. $\rho_{\max}$ is found by

$$\rho_{\max} = \max_{VOP_k \in VO} \{\rho_{VOP_k}\}, \tag{5.16}$$

where VOP_k is the k^th^ video object plane of the video object, and ρ_{VOP_k} is the radius of the tightest circle around VOP_k that is centered at the mass center of VOP_k.

In the uncompressed domain, the value of the binary shape map is "1" for pixels that fall inside the video object and "0" for pixels that fall outside the video object. In the compressed domain, it is "1" for the opaque coded shape blocks, "0.5" for the intra coded shape blocks, and "0" for the transparent coded shape blocks. This way, the intra coded shape blocks are counted as half of the opaque coded blocks in the variance computation.

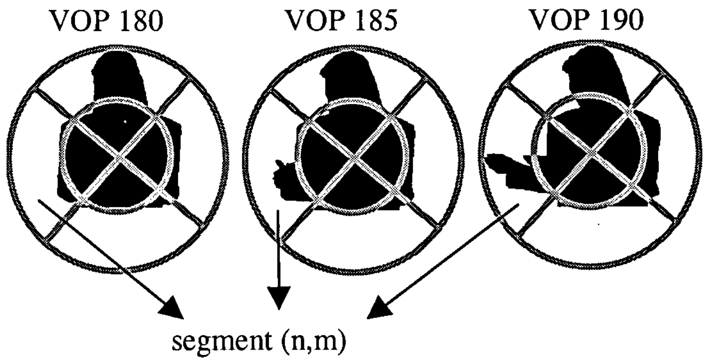


Figure 5.1: The binary shape maps of the 180th, 185th, and 190th VOPs of a video object divided into 4 angular and 2 circular segments.

The proposed descriptor is scale invariant since the number of angular and circular segments is the same for all video objects and the size of each segment is scaled with ρ_{max}. Rotation invariance is obtained as follows. We first re-order the feature matrix $\boldsymbol{R}$ so that the angular segment with the largest variance is in the first column of $\boldsymbol{R}$. This is achieved by first summing the columns of $\boldsymbol{R}$ to obtain the 1×M projection vector $\vec{A}$ and then finding the maximum element of $\vec{A}$, which corresponds to the angular segment m that has the largest variance. Finally, we circularly shift to the left the columns of $\boldsymbol{R}$ by m to obtain a rotation invariant feature vector.

5.4 Experimental Results

Here, we demonstrate the performance of each of our proposed local motion descriptors. Our database contains over 50 MPEG-4 video object streams and the NMRR measure (see Appendix A) is employed to evaluate the performance of our descriptors. The average NMRR values presented in this section are obtained by averaging the retrieval results of 12 query video objects that have a large variety of local motions. The ground truth objects are decided by having human subjects rank the video objects for their local motion similarity to the query video objects. As in the previous chapter, computation times presented here are obtained by using a Pentium 233 MHz. computer with 128 MBs of memory.

The results presented in the following sections are obtained by computing these descriptors using the IVOPs of video objects. The similarity distance between two shapes is measured by computing the Euclidean distance between their local motion descriptors. The distance is computed on the square roots of the variance based descriptors.

5.4.1 Fourier Transform based local motion descriptor

Table 5-1 shows the retrieval performance of the proposed Fourier coefficient based local motion descriptor with various numbers of Fourier coefficients. Note that the smaller

ANMRR values represent a better retrieval performance. Employing a larger number of coefficients improves the retrieval performance but also increases storage requirements. As can be observed from Table 5-1, using the first 10 coefficient variances offers a good tradeoff between retrieval performance and descriptor size. In addition, we observe that computing this descriptor in the compressed domain results in a small decrease in the retrieval performance. Nevertheless, processing in the compressed domain offers significant computational advantages considering that the computation time required to compute the descriptor for one video object is 0.72 seconds in the compressed domain compared to 26.36 seconds in the uncompressed domain.

Number of Fourier coefficients	ANMRR – uncompressed domain	ANMRR – MPEG-4 domain
5	0.170	0.271
10	0.139	0.237
20	0.119	0.232
32	0.122	0.227

Table 5-1: Local motion retrieval results using the Fourier Transform based descriptor with various lengths.

5.4.2 ART based local motion descriptor

The retrieval performance achieved by using different number of angular and radial functions when computing the ART descriptors is presented in Figure 5.2. As can be observed from the figure, employing 4 angular and 2 radial basis functions offers a good tradeoff between the retrieval performance (ANMRR=0.18) and the compactness of the descriptor. The ANMRR is equal to 0.20 for using the same number of basis functions in the MPEG-4 compressed domain, which is only slightly higher (indicating slightly lower retrieval performance) than that of the uncompressed domain. The computation time of this descriptor per video object is 15.34 seconds in the uncompressed domain and 2.5 seconds in the compressed domain.

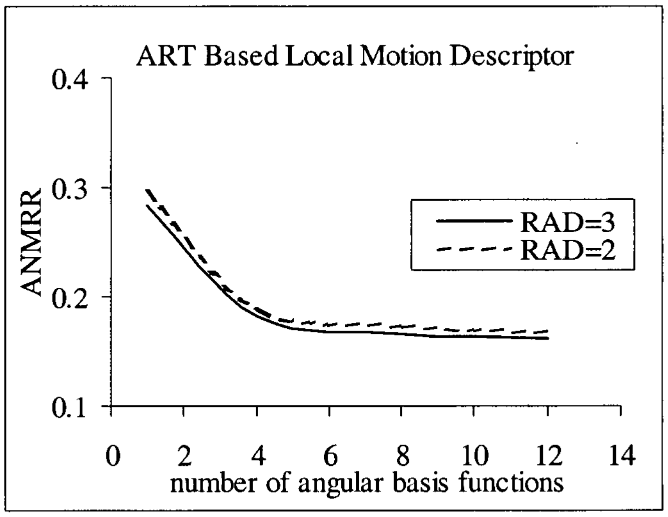


Figure 5.2: Retrieval results of the ART based local motion descriptor obtained by employing different numbers of angular and radial (RAD) basis functions.

5.4.3 ACLM descriptor

Retrieval performance results using the Angular Circular Local Motion (ACLM) descriptor with various number of angular and circular segments are presented in Figure 5.3. Employing a large number of angular and circular bins results in a better retrieval performance but with the cost of more bits required for representing the descriptor. The highest retrieval rates here (i.e. lowest ANMRR) are obtained by using 6 angular and 3 circular segments (ANMRR=0.090) and 8 angular and 2 circular segments (ANMRR=0.089). The ANMRR values obtained by extracting this descriptor directly from the MPEG-4 compressed bit stream are 0.26 for 6 angular and 3 circular segments and 0.27 for 8 angular and 2 circular segments. The time required to compute the compressed domain descriptor for one video object is 0.29 sec., and that of the uncompressed domain descriptor is 20.54 sec.

Even though the computational requirements of the compressed domain descriptor is 2 orders of magnitude lower than that of the uncompressed domain, the decrease in the retrieval rate (increase in ANMRR from 0.09 to 0.26) may not be acceptable in some applications. One way to improve the retrieval performance in the compressed domain is to upsample the shape approximation before dividing it into angular circular segments. This way, more pixels would fall into each segment, making the variance computation more accurate and more robust to shape approximation errors. On the other hand, computing this descriptor on an upscaled version of the shape approximation would take more computation time. This tradeoff is depicted in Figure 5.4, where we plot the required computation time and the retrieval rate for employing different numbers of upscaling factors. Employing an upscaling factor of 4 results in a good retrieval rate (ANMRR=0.16) with computation time less than 0.5 seconds.

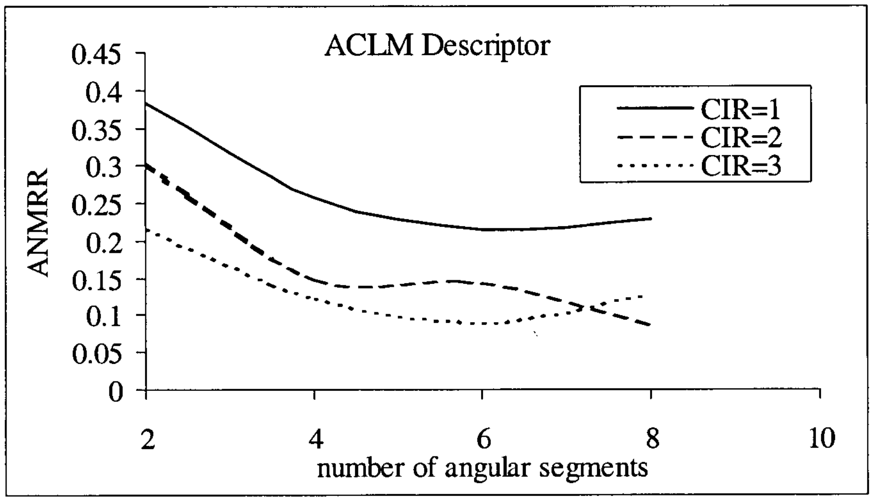


Figure 5.3: Retrieval results of the ACLM descriptor obtained by using a various number of angular and circular (CIR) segments.

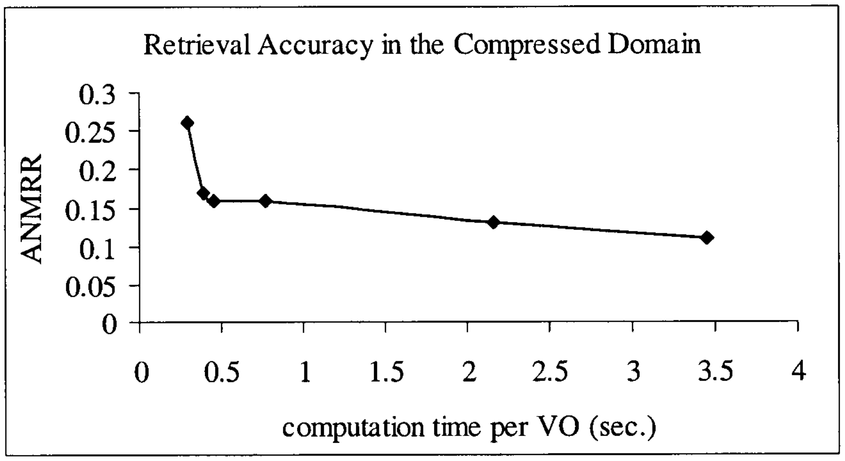


Figure 5.4: The change in retrieval accuracy versus computation time requirements when different upscaling factors are employed in the compressed domain.

Some query examples using 6 angular and 3 circular segments are presented in Figure 5.5, Figure 5.6, and Figure 5.7. Note that the dimensions given in the parentheses are not the dimensions of the video objects, but dimensions of the video sequences that they are extracted from. The dimensions of the video objects are different for each VOP of the video object. The first query, shown in Figure 5.5, is a very low motion anchorperson video object, News_1, which is coded in two different resolutions in our database. As presented in Table 5-2, using the ACLM descriptor, the two different resolutions of the News_1 video object are retrieved as the first two items. The other highly ranked two anchorperson video objects, illustrated in Figure 5.5, are also characterized by low motion. The Coastguard video object, ranked in 7th, 8th, and 9th, is also an object without any articulated parts (a boat object and its waves) and with moderate local motion. The second query is the video object obtained from a video sequence showing two children playing with a ball as shown in Figure 5.6. Here, we query the local motion of the child on the left. As can be seen from the results presented in Table 5-3, the three different resolutions of the two children video objects are ranked the highest among the retrieved video objects. Our last query, Hall Monitor 1, is the video object of a walking man captured by a surveillance camera as shown in Figure 5.7. The query results for this object are presented in Table 5-4. The three different resolutions of the video object are ranked the highest and another walking man video object from the same sequence, Hall Monitor 2, is ranked immediately after. The fish object, which has large moving fins and a tail as depicted in Figure 5.7, is ranked 7th. The different resolutions of a video object that contain a person playing tennis are ranked 8th and 9th. As can be seen from these query examples, the ACLM descriptor successfully classifies the local motion of the video objects.

Rank	Video object	Query distance
1	News 1 (360x240)	0.00
2	News 1 (180x120)	6.68
3	Akiyo (360x240)	11.07
4	News 2 (360x240)	12.55
5	Akiyo (180x120)	14.06
6	News 2 (180x120)	19.52
7	Coastguard 2 (352x288)	27.12
8	Coastguard 2 (176x144)	27.63
9	Coastguard 2 (528x432)	27.68

Table 5-2: Local motion retrieval results for the News 1 video object query.

Rank	Video object	Query distance
1	Children 1 (352x288)	0.00
2	Children 1 (176x144)	4.00
3	Children 1 (528x432)	8.57
4	Children 2 (176x144)	55.04
5	Children 2 (352x288)	55.23
6	Children 2 (528x432)	55.41

Table 5-3: Local motion retrieval results for the Children 1 video object query.

Rank	Video object	Query distance
1	Hall Monitor 1 (360x240)	0.00
2	Hall Monitor 1 (540x360)	2.89
3	Hall Monitor 1 (180x120)	10.23
4	Hall Monitor 2 (180x120)	46.85
5	Hall Monitor 2 (360x240)	50.25
6	Hall Monitor 2 (540x360)	50.31
7	Fish 1 (352x240)	84.59
8	Stefan (176x144)	90.31
9	Stefan (352x244)	90.80

Table 5-4: Local motion retrieval results for the Hall Monitor 1 video object query.

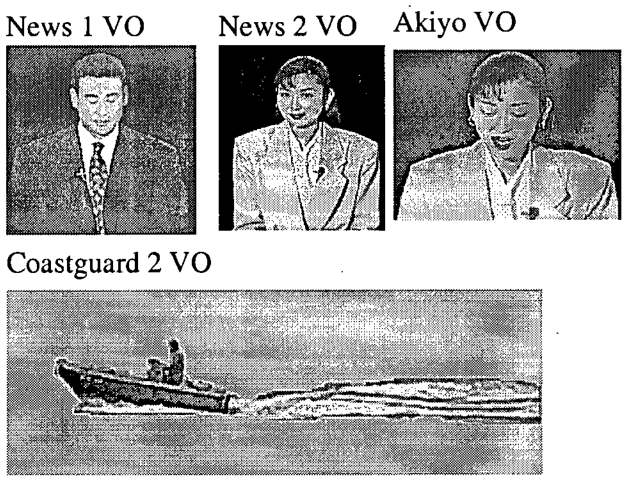


Figure 5.5: The video objects classified as being similar in terms of their local motion to the query video object News 1.

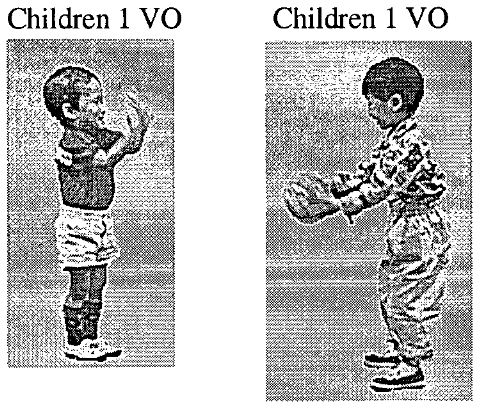


Figure 5.6: The video objects classified as being similar in terms of their local motion to the query video object Children 1.

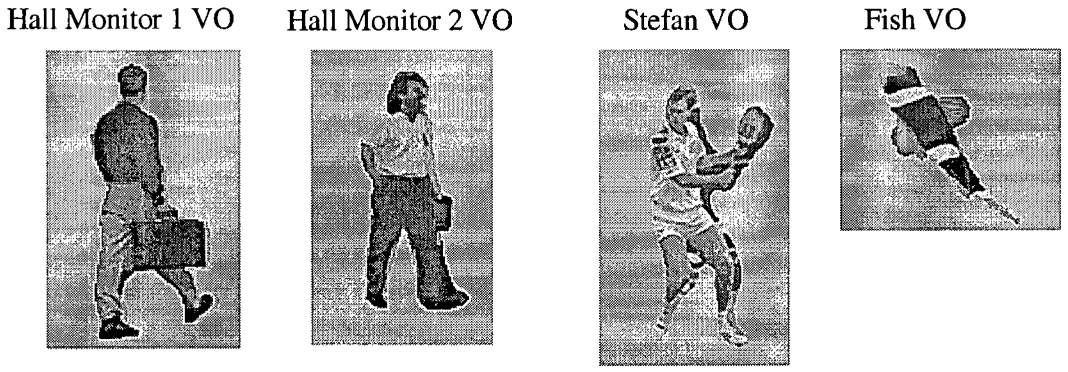


Figure 5.7: The video objects classified as being similar in terms of their local motion to the query video object Hall Monitor 1.

5.5 Conclusions

In this chapter, we proposed three local motion descriptors for efficient retrieval of video objects. Our descriptors are based on the shape deformations of the objects as the changes in an object's shape offer valuable information about its local motion. These descriptors can be extracted using the uncompressed domain shape masks or compressed domain shape approximations. The corresponding computation times, descriptor sizes, and the resulting retrieval performances for all three descriptors are summarized in Table 5-5. As presented in the previous section, the retrieval ranking obtained by employing our descriptors closely matches with the human ranking. According to the ANMRR scores obtained, the ACLM descriptor offers the best retrieval rate, in both compressed and uncompressed domains. That is mainly because this descriptor preserves the location of the motion (as the object is divided into angular/circular segments), where this information is distributed among the coefficients in the ART and Fourier based descriptors. Nevertheless, matching of the ACLM descriptor is more complex as presented in Section 5.3 and it requires 18 bytes (assuming that 1 byte is used to represent each descriptor value) to represent, where Fourier and ART based descriptors require 10 and 8 bytes, respectively. The computation time of the ACLM descriptor is the shortest compared to the other two in the compressed domain. Overall, the ACLM descriptor offers better tradeoffs than the other two descriptors. Nevertheless, if the ART and Fourier coefficients of the VOP shape data is already computed and attached to the video objects as metadata for shape retrieval, then the extra computations required to compute the local motion descriptors based on the ART and Fourier coefficients would be minimal. In that case, employing the Fourier and ART based descriptors could be more advantageous.

	Fourier based descriptor		ART based descriptor		ACLM descriptor	
Domain	comp.	uncomp.	comp.	uncomp.	comp.	uncomp.
ANMRR	0.24	0.13	0.20	0.18	0.16	0.09
computation time (sec)	0.72	26.4	2.5	15.3	0.45	20.5
descriptor size	10	10	8	8	18	18

Table 5-5: Retrieval performance, computation time, and descriptor size comparisons of the proposed local motion descriptors in the compressed and uncompressed domains.

Chapter 6

Similarity Matching of Video Objects using Color Feature

"Has anybody seen my Mopser?
A comely dog is he
With hair the colour of a Charles the Fifth
And teeth like ships at sea."
Walter de la Mare (1873-1956)

Color features are commonly used for visual data retrieval, as they are relatively simple to extract, not much sensitive to noise, and invariant to image scaling, translation, and orientation [108]. They are also relatively easy to associate with some semantic information. For example, if one would like to retrieve nature images from a visual database, he/she can search the database for images with dominant green or brown colors. Alternatively, if one is looking for a brick house in a real estate database, he/she can query the database with a picture of another brick house, where the color of the query image would be automatically extracted by the system and houses with similar colors can be retrieved.

The most commonly used color features in current systems include dominant color, average color, and color histograms. The dominant color feature captures one or several most common colors in an image. Although it provides a compact representation of the color content, capturing only a few dominant colors may not be sufficient in some applications, say for instance if one is searching for a small object in an image (for example a yellow flower). The average color descriptor is typically found by computing the mean value of the colors of all the pixels in an image. Again, this is a compact representation, but the accuracy would not be satisfactory in many applications. Also, averaging all the colors may result in an average color that does not represent the image accurately, for example if the image contains blue and yellow colors the average color would be green.

Color histograms, on the other hand, indicate the number of occurrences of a particular color intensity value in an image, and they work very well in quantifying global color content in visual data. Therefore, they have been employed by many image/video retrieval systems [12][21][22][109], which differ primarily in their approaches to color space selection, quantization, and histogram distance computation problems. For example, VideoQ [22] employs a color histogram that is based on HSV color space and uniform quantization of color values. In NeTraV [21], the RGB color space is employed and the colors are vector quantized into 256 bins using the Generalized Lloyd Algorithm.

In QBIC [12], the image is first represented with an RGB color histogram consisting of 4096 bins. Each color channel is then transformed to the Munsell color system using the MTM transform. Color histogram representation is also standardized by MPEG-7 [27], where the histogram is formed in the HSV color space and a scalable number of color quantization bins are employed.

As digital images and video are becoming available mostly in the compressed formats, several researchers have suggested methods for obtaining color histograms from the coded bit stream without requiring full decompression and subsequent reconstruction of the visual data [110][111]. Their methods are based primarily on the idea that, in block based coding, the mean value of each 8×8 pixel block can be computed by a simple scaling of the first DCT coefficient (DC). These average color values can then be used to form a color histogram. Because the DC coefficient can be obtained from the bit stream without full decompression [112][113], this method makes it is possible to extract color histograms from the compressed bit streams very efficiently.

In this chapter, we discuss the problems arising from the adaptation of DC based color histogram representation to arbitrarily shaped MPEG-4 video objects, which can be summarized as follows:

- Extraction of the DC coefficient based histogram from the MPEG-4 video bit stream is more complex than extraction from the MPEG-1/2 and H.263 bit streams, mainly due to the MPEG-4 bit stream structure and the predictive coding of DC coefficients.

- Unlike frame-based video sequences, video objects generally have low resolution, small variations in color, and their color content usually remains consistent. Therefore, a color histogram representation that is optimal for the frame-based video is not necessarily optimal for the object-based video.
- When the chroma-keying technique[2] is used to obtain video objects, the chroma key value can penetrate into the object. This would happen particularly if the video object shape is not accurately extracted prior to the MPEG-4 encoding and/or the MPEG-4 encoder does not employ the LPE padding technique described in the MPEG-4 verification model [53]. As a result, the chroma color of the background could contribute to some color artifacts that would eventually affect the color histogram of the video object.

In this chapter, we first present a description of the extraction of DC coefficients from the MPEG-4 bit stream without full decompression. Then, a discussion of the selection of color spaces and quantization accuracy are given in Section 6.2. In Section 6.3, we present a color histogram computation method for MPEG-4 video objects. In Section 6.4, we propose a method for detecting and compensating for the chroma keying artifacts that may occur at the boundaries of the video objects. Experimental results and conclusions are given in Section 6.5 and Section 6.6, respectively.

[2] Chroma keying is one of the most popular methods to obtain arbitrarily shaped video objects, where the video object is separated from the background by placing it in front of a screen with a unique color.

6.1 DC Coefficient Extraction in the MPEG-4 Bit Stream

In most of the current image and video coding standards, such as JPEG [2], MPEG-1/2 [4][5], and H.263 [6], each frame is divided into 8×8 blocks, followed by DCT, quantization, zigzag scan, and run length coding. In Intra coded frames (I-frames), the mean value of each of the 8×8 luminance and chrominance blocks can be obtained by a simple scaling of the first DCT coefficient (DC) of the corresponding block as described next.

The definition of the 2-D DCT transform for an 8×8 block is given by [76]:

$$F(k,l) = \frac{c(k)\,c(l)}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} f(i,j) \cos\left(\frac{(2i+1)k\pi}{16}\right) \cos\left(\frac{(2j+1)l\pi}{16}\right), \tag{6.1}$$

where $c(x) = \begin{cases} \frac{1}{\sqrt{2}} & if\ x=0 \\ 1 & otherwise \end{cases}$ and *f(i,j)* is the value of the pixel at the location *(i,j)*.

Therefore, the DC coefficient is computed as follows:

$$DC = F(0,0) = \frac{c(0)\,c(0)}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} f(i,j) \cos\left(\frac{(2i+1)0\pi}{16}\right) \cos\left(\frac{(2j+1)0\pi}{16}\right) \tag{6.2}$$

$$= \frac{1}{4\sqrt{2}\,\sqrt{2}} \sum_{i=0}^{7} \sum_{j=0}^{7} f(i,j) = \frac{1}{8} \sum_{i=0}^{7} \sum_{j=0}^{7} f(i,j) \tag{6.3}$$

Considering that the mean value, M, of the 8×8 block is found by $M = \frac{1}{64}\sum_{i=0}^{7}\sum_{j=0}^{7} f(i,j)$, the relationship between the DC value and the mean of the block can be written as

$$DC = \frac{1}{8}\sum_{i=0}^{7}\sum_{j=0}^{7} f(i,j) = 8M \Rightarrow M = \frac{DC}{8}. \tag{6.4}$$

Taking into account that the chrominance values are shifted by 128 prior to coding, the mean of C_b block, M_{cb}, and the mean of C_r block, M_{cr}, are found by

$$M_{Cb} = \frac{DC_{Cb}}{8} - 128, \quad M_{Cr} = \frac{DC_{Cr}}{8} - 128, \tag{6.5}$$

where DC_{Cb} and DC_{Cr} are the DC coefficients of the C_b and C_r blocks, respectively.

In MPEG-1/2 and H.263 I-frames, the DC coefficient can be obtained by simply performing parsing of the headers and run length decoding. However, in MPEG-4, an extra step is required as the DCT coefficients of macroblocks are predictively coded even in intra VOPs. In intra macroblocks, the DC coefficients are predicted from the DC coefficient of either the left or the above block as presented in Chapter 2. Therefore, the DC coefficient needs to be reconstructed first by summing with the prediction value followed by dequantization to obtain DC_Y, DC_{Cb}, and DC_{Cr}. These values are then used in Equations (6.4) and (6.5) to compute the mean values of the 8×8 luminance and chrominance blocks for color histogram computation.

Parsing of the MPEG-4 bit stream in order to extract the DC information is also more complex than parsing of the MPEG-1/2 and H.263 bit streams. Recall that in

MPEG-4, the shape information is placed before the texture information in the bit stream. Therefore, arithmetic decoding of the shape is required before obtaining the DC coefficient. Nevertheless, reconstruction of the shape data is not necessary. Also in MPEG-4, if the data packetization option is utilized, the shape information and the DC coefficients of intra coded VOPs are separated from the AC coefficients with a unique dc marker bit sequence. This provides better error protection for the MPEG-4 DC coefficients, making the color histogram more robust to the potential bit stream errors.

6.2 Color Space Selection and Quantization

In histogram computation, the proper choice of color space and quantization method is very important, as they can have a profound effect on the retrieval performance of the system. Video in MPEG-4 domain is represented in YC_bC_r color space, as in MPEG-1/2. While YC_bC_r representation is good for efficient compression, it is not a desirable representation in visual retrieval as it is not a perceptually uniform color space. Perceptual uniformity means that two colors that are equal in distance in a color space are perceived as equal in distance by viewers. The RGB color space, which is commonly used for display purposes, has a linear relation with the YC_bC_r color space, it is therefore also non uniform. The other two color spaces typically employed to form color histograms are HSV (Hue, Saturation, Value) and MTM (Mathematical Transformation to Munsell). The HSV color space, which is adopted for the MPEG-7 color histogram descriptor, more closely resembles human perception, but it is again a non uniform color space [74].

Munsell is a perceptually uniform color space that very closely represents the human way of perceiving colors [74][114][115]. The C.I.E. L*a*b* method commonly used to obtain a quantitative expression for the Munsell space of color classification. On the other hand, Miyahara et al. showed in [114] that the Mathematical Transformation to Munsell (MTM) offers the best method for Munsell space representation. In the MTM space, the colors are represented by Hue (H), Value (V) and Chroma (C) components that can be calculated from the R, G, B or Y, C_b, C_r values using a non linear transform. The conversions from the YC_bC_r space to HSV and MTM spaces are given in Appendix C.

Another important issue in color histogram computation is the quantization of colors. Using the entire color space (approximately 16 million colors) without quantization clearly increases the storage and computation requirements. In a typical retrieval system, each of the components of a color space is divided into a number of color bins prior to histogram computation. Uniform quantization is commonly employed, especially in the absence of information regarding the color distribution of the target image/video databases. The best results with uniform quantization are obtained when applied to a perceptually uniform color space.

6.3 Histogram Computation

We obtain the color histograms of individual VOPs by using only the color components that correspond to the blocks that are either completely inside (i.e., opaque) or on the boundary (i.e., intra) of the video object planes. This information is extracted directly from the MPEG-4 bit stream. On average, only half of the pixels in a boundary block lie in a video object. Therefore, when computing the color histograms for individual VOPs, we count the color components of the boundary blocks as half of the color components of the opaque blocks.

After constructing the histograms for the individual VOPs, the histogram for video objects can be formed by using one of the following techniques commonly used for frame-based video:

- Average histogram: Obtained by accumulating the histogram values over a range of frames and normalizing that by the number of frames.
- Median histogram: The bin values of this histogram are computed by taking the median values of each corresponding histogram bin of the individual frames.
- Intersection histogram: This histogram contains only the colors that are common to all the frames.

According to the literature, employing the average histogram yields the best results for frame-based video retrieval [116]. Video object color generally remains consistent during its lifespan, and in most cases an average histogram accurately represents the video object color content. The median histogram is most useful when there are some frames in a video sequence that differ significantly from the others – which is usually not the case for video objects. Also, there is an increased computational cost associated with the median operation because of the sorting performed for each bin. The intersection histogram is also not very suitable for video object color representation. This is because when objects are entering to/exiting from a scene or when they are occluded, only a small part of their color range is visible: The intersection histogram would only capture these colors. In conclusion, considering the characteristics of arbitrarily shaped video objects, the average histogram is clearly the most appropriate choice to represent their color histograms.

The average histogram can be computed using the individual histograms of all the IVOPs in an MPEG-4 video object. A better alternative that reduces the computational requirements would be to compute the histogram on a temporally sampled subset of IVOPs or on key VOPs that represent the salient content of the video object.

6.4 Detection and Compensation of the Chroma Keying Artifacts

Although there have been great advances in the area of video segmentation, chroma keying remains one of the most popular methods to obtain semantic video objects without requiring a high degree of supervision. In chroma keying, the foreground object is separated from the background by placing the object in front of a color screen that has a unique chroma key value (typically blue or green) and defining the pixels that belong to the screen as outside the video object. Ideally, the coded video object should not contain any pixels from the background. However, if an MPEG-4 encoder does not approximate the video object shape very accurately and/or it does not perform the Low Pass Extrapolation (LPE) padding technique prior to DCT (which is defined in the MPEG-4 verification model [53] but not part of the MPEG-4 standard [1]), the boundary blocks of the video object could contain some severe chroma keying artifacts. These artifacts could result in chroma DC values (DC_{Cb} and DC_{Cr}) of the boundary blocks that include the chroma key color along with the actual video object color, resulting in an inaccurate computation of the color histogram.

In order to overcome this problem, we propose that the existence of such artifacts be detected and then compensated accordingly. Our experiments show that if a video object plane has any chroma artifacts, it is likely to affect all the blocks on the video object boundary. Therefore, if such effects are detected in one or several boundary blocks, it is reasonable to assume that the most of the boundary blocks of the video object have such artifacts. We propose to detect the chroma artifacts at the decoder, assuming no apriori information about the encoder, by decoding the texture and the shape of the first boundary block of the video object plane and then computing the mean chroma values (C_b and C_r) for the pixels that are inside and outside the video object area using the shape mask for that particular block. If the difference between the chroma values corresponding to the inside and outside of the video object is very small, then it could be concluded that the segmentation was performed properly and that the LPE technique was employed prior to encoding. Then the DC values of the boundary blocks correctly reflect the real video object color and no further processing is required. However, if the inside and outside mean chroma values differ significantly, then we define the chroma key values (K_{Cb}, K_{Cr}) as equal to the mean chroma values of the outside pixels.

After chroma keying artifacts are detected and the chroma key values are determined, the average colors of the boundary blocks are adjusted to reduce the chroma artifacts. Considering that, on average half of the pixels in a boundary belong to the inside of the object and the other half belongs to the outside of the object, the following approximations can be made to find the actual mean value of the pixels inside the video object:

$$\begin{gathered} M_{Cb} \approx \frac{V_{Cb} + K_{Cb}}{2},\ M_{Cr} \approx \frac{V_{Cr} + K_{Cr}}{2} \Rightarrow \\ V_{Cb} \approx 2M_{Cb} - K_{Cb},\ V_{Cr} \approx 2M_{Cr} - K_{Cr}, \end{gathered} \tag{6.6}$$

where M_{Cb} and M_{Cr} are the average values of the 8×8 C_b and C_r blocks extracted from the bit stream, K_{Cb} and K_{Cr} are the approximated chroma key values, and V_{Cb} and V_{Cr} are the approximated mean chrominance values of the video object pixels in the corresponding 8×8 blocks.

It is possible to make this approximation even more accurate by considering the coding modes of the luminance blocks corresponding to the chrominance blocks. In MPEG-4, as in MPEG-1/2, chrominance values are typically downsampled by two in both vertical and horizontal directions and a 16×16 luminance block spatially corresponds to a 8×8 chrominance block. Therefore, by looking at the coding modes (intra coded/not coded) of each of the 8×8 luminance blocks in a macroblock, we can predict with a better accuracy what percentage of the chroma block lies in the video object border. For example, if 3 out of 4 luminance blocks are not coded, then it is likely that more than 75% of the chroma pixels in the corresponding block are outside the video object. Taking the coding modes of the luminance blocks into consideration, the following approximation can be made to find the mean value of the chrominance pixels inside the video object.

$$M_{Cb} \approx \frac{0.5\alpha(V_{Cb} + K_{Cb}) + (4-\alpha)K_{Cb}}{4} \quad \Rightarrow \tag{6.7}$$

$$V_{Cb} \approx \frac{8M_{Cb} - 8K_{Cb} + \alpha\, K_{Cb}}{\alpha},$$

where α is the number of intra coded luminance blocks in a boundary macroblock, M_{Cb} is the average chrominance value corresponding to that macroblock, K_{Cb} is the chroma key value, and V_{Cb} is the mean chrominance values of the pixels that belong to the video object in a boundary block. V_{Cr} is also found by using a similar approximation. Finally, we compute the video object color histogram by using the approximated V_{Cb} and V_{Cr} values of the chrominance components, along with the unmodified luminance component, M_Y.

6.5 Color Retrieval Results

In this section, we present color retrieval results obtained by querying video objects as well as individual VOPs. Retrieval performance is measured by NMRR and ANMRR measures that are described in Appendix A. Our database consists of more than 50 MPEG-4 bit streams containing more than 2000 Intra coded color VOPs. The color histograms of the video objects are found by averaging the color histograms of their key VOPs. The histogram distances between two VOs or two VOPs are computed using the

L1 norm, which demonstrated superior performance for measuring the histogram distances [116][117].

6.5.1 Retrieval by employing various color spaces and number of quantization bins

Here, we present the retrieval performance levels when employing HSV, MTM, and YC_bC_r color spaces and different numbers of quantization bins. Our experiments are performed by querying MPEG-4 video object planes. We uniformly quantize each of the color components to reduce the number of histogram bins. Since hue has most of the color information that humans can recognize [118], we allocate a larger number of bins to the hue (H) color component when quantizing the HSV and MTM color spaces.

Table 6-1 presents the color retrieval performance comparison employing histograms computed in the three different color spaces. Using the YC_bC_r color representation does not require conversion to another color space, however it gives the lowest retrieval performance. Using the MTM representation, which matches most closely human perception, clearly offers a superior retrieval performance. Table 6-2 shows the retrieval results when different numbers of quantization bins are used to represent the color components of the MTM space. As can seen from the table, employing a 128-bin histogram offers the best tradeoff between retrieval performance and memory requirements.

Query video object plane	**HSV** 128 bins: H:8 S:4 V:4	**MTM** 128 bins: H:8 V:4 C:4	**YC_bC_r** 125 bins: Y:5 C_b:5 C_r:5
Bream	0.0007	0.0004	0.0097
Fish	0.0876	0.0249	0.2400
Stefan	0.0116	0.0208	0.1303
Singing girl	0.2686	0.2006	0.2715
Average NMRR	**0.0912**	**0.0617**	**0.1629**

Table 6-1: NMRR values obtained by querying various video object planes and employing color histograms computed in three different color spaces.

Query video object plane	256 bins: H:16 V:4 C:4	128 bins: H:8 V:4 C:4	64 bins: H:4 V:4 C:4	32 bins: H:4 V:2 C:2
Bream	0.0003	0.0004	0.0728	0.0202
Fish	0.0411	0.0249	0.0425	0.0194
Stefan	0.0302	0.0208	0.1841	0.1900
Singing girl	0.0666	0.2006	0.3349	0.3812
Average NMRR	**0.0346**	**0.0617**	**0.1586**	**0.1527**

Table 6-2: The retrieval performance results (in NMRR) when using different numbers of quantization bins for the H, V, and C color components of the MTM color histograms.

6.5.2 Video Object Plane and Video Object retrieval results with and without chroma keying artifacts

Next, we demonstrate the performance of our proposed technique in the presence of chroma keying artifacts. All the results presented here are obtained by utilizing the MTM color space and 128-bin uniform quantization for the color histograms of the VOPs. The video object histograms are formed by histogram averaging on their key VOPs. The key VOPs are found using the Hausdorff distance based algorithm we proposed in Chapter 3.

Table 6-3 demonstrates the retrieval results for several video object plane queries from the video objects Children, Stefan and Hall Monitor. The first column shows the results when there are no chroma keying artifacts. The second column gives the retrieval performance when the query VOP and several hundred VOPs in the database are coded with simulated chroma keying artifacts. Simulation of such artifacts was achieved by simply imposing a blue background against the objects and encoding the video objects with no LPE padding. As seen from Table 6-3, chroma keying artifacts result in poor retrieval performance. After applying the proposed technique to reduce these effects, the retrieval performance improves significantly as presented in the last column of the table. Similar experiments are conducted for the video object sequences as well and the results are presented in Table 6-4. Again, significant performance gains are obtained by employing the proposed method for reducing the chroma keying artifacts.

Query VOP	Without artifacts	With artifacts	With reduced artifacts
Children 1	0. 0022	0. 0777	0. 0087
Stefan	0. 0509	0.1172	0. 0835
Hall monitor 2	0. 0118	0. 5886	0. 2946
Average NMRR	**0. 0216**	**0. 2612**	**0. 1289**

Table 6-3: Video object plane retrieval performance results (in NMRR) without any chroma artifacts, with chroma artifacts, and after compensation for the artifacts with the proposed method.

Query video object	Without artifacts	With artifacts	With reduced artifacts
Children 1	0. 0000	0. 6396	0. 3333
Stefan	0. 0741	0. 0410	0. 0370
Hall monitor 2	0. 0250	0. 4250	0. 1500
Average NMRR	**0. 0330**	**0. 3685**	**0. 1734**

Table 6-4: Video object retrieval results (in NMRR) without any chroma artifacts, with chroma artifacts, and after compensation for the artifacts with the proposed method.

6.6 Conclusions

In this chapter, we discussed the issues arising from the use of the DC based color histogram technique in the MPEG-4 compressed domain. Compared to extraction from the MPEG-1/2 and H.263 bit streams, extracting the color histogram from the MPEG-4 bit stream requires more computations, due mainly to the intra DC prediction. Nevertheless, the MPEG-4 bit stream offers more protection to the DC coefficient, making the histograms more robust if the bit stream is susceptible to errors.

We employ the color histogram feature for color retrieval, as it offers a more complete representation than the other available color features. Because they contain more information, color histograms generally require more storage than the other color descriptors. While this would be a consideration in image database applications, the overhead introduced by employing color histograms in our case is negligible, since the video data typically requires considerably more storage.

Our experimental results show that the color histogram technique works best for video object retrieval when the MTM color space is employed. Using the YC_bC_r space also offers reasonable retrieval performance without requiring color space conversion. In the presence of chroma keying artifacts, the computed color histograms may contain the chroma key value (generally a very distinct color), causing a significant drop in the retrieval performance. As presented in Section 6.5.2, great improvements in retrieval performance are obtained by employing our proposed method to compensate for such artifacts.

Chapter 7

Conclusions and Future Research Directions

Efficient retrieval of video is becoming increasingly important as video content becomes available from a growing number of different sources. Classifying and retrieving arbitrarily shaped video is of particular importance, since it is an enabling technology for many multimedia applications ranging from surveillance to broadcasting, education to medicine. This thesis proposed techniques for automatic, generic, effective, and low complexity content-based retrieval of arbitrarily shaped video, specifically targeting the MPEG-4 object-based compressed domain representation.

In this thesis, we first summarized MPEG-4 object-based video coding, as most of our algorithms are based on this representation. Next, we presented a method for summarization of arbitrarily shaped video content through selection of key video object planes. Our method is based on computing the Hausdorff or Hamming distances on the shape approximations obtained from the compressed bit stream. Experimental results demonstrated that our proposed algorithm selects key video object planes that efficiently represent the salient content of video objects.

Next, we focused on shape retrieval of video objects and video object planes. We gave a brief overview of Fourier and ART descriptors as well as some global shape features, and presented a method to extract these features directly from the compressed bit stream without full decompression. In addition, two new compressed domain shape features were proposed. Processing in the compressed domain and using our proposed shape feature extraction techniques, we demonstrated computational savings as high as two orders of magnitude. Compact representation of these shape features is also important, especially if they are computed prior to querying and are stored in the video bit stream or in a database. We designed efficient quantizers for these features based on their statistical distributions, which resulted in compaction of the feature vectors by a factor of eight. We also proposed a domain independent measure to find the distance between two video objects based on their key video object planes. Using this technique, we achieved significantly better video object retrieval performance compared to the other existent techniques in the literature.

We proposed three new descriptors for retrieval of video objects by their local motion, which are based on the variances of the ART and Fourier coefficients, and the variances of angular circular segments of video objects. When compared to human ranking, these descriptors demonstrated their ability to successfully classify local object motion in both compressed and uncompressed domains. The angular circular local motion descriptor had very low complexity and performed more robustly in both domains. Nevertheless, it was concluded that employing ART and Fourier coefficient variances as local motion descriptors can be more advantageous in terms of complexity in cases where coefficients were already computed , e.g. for shape retrieval, and exist in the video stream as metadata.

We also presented color content matching of arbitrarily shaped MPEG-4 video objects. Specific issues arising from computation of the DC coefficient based color histogram in the MPEG-4 compressed domain were discussed. We presented color retrieval results employing YC_bC_r, HSV, and MTM color spaces and various quantization parameters on an MPEG-4 database. We also proposed a method to detect and compensate for chroma keying artifacts that may exist in the bit stream depending on the segmentation technique used to obtain the video object and the MPEG-4 encoding. Employing our method in the presence of such artifacts resulted in significant retrieval performance gains.

The effectiveness and excellent computational tradeoffs associated with our proposed shape, local motion, and color retrieval techniques make them suitable for arbitrarily shaped video database applications, especially for those containing MPEG-4 bit streams. Our video object summarization algorithm can also be used for fast browsing of video content in databases and storyboard generation. The shape, color, and motion features we employ can be made MPEG-7 standard compliant by coding with the

Description Definition Language (DLL). Once these features are computed and coded in an MPEG-7 bit stream, they can be utilized by other database applications supporting the MPEG-7 standard.

Besides database applications, our techniques are potentially useful in video communications as well. For example, in a wireless video telephony application where bandwidth is limited, rather than the whole video sequence, key video object planes that demonstrate the speakers' key actions can be transmitted. In a digital camera application, only the video object planes that efficiently summarize the video object content need to be stored, resulting in significant memory savings. Moreover, if some video objects in a video sequence are identified as being important –using their shape, color, or motion content– for a particular application, they can be coded with a better accuracy than the rest of the video objects, or transmitted in a more robust channel, providing communication efficiency.

Although we employed the MPEG-4 object-based representation in this thesis, most of our techniques can potentially be applied to other object-based representations as well. Here, the MPEG-4 compressed bit stream is mostly utilized to obtain a low resolution version of the object shape information. Therefore, as long as we can obtain a low resolution version of the object's shape and color information from a compressed bit stream without full decompression, our techniques would be applicable to such domain.

7.1 Summary of Thesis Contributions

The main contributions of this dissertation can be summarized as follows:

- A video object content summarization algorithm in the MPEG-4 compressed domain. Our algorithm has a demonstrated ability to find the object planes that efficiently summarize the salient content of arbitrarily shaped video.
- A method to compute some commonly used shape features in the compressed domain. This method demonstrated computational savings of more than two orders of magnitude with only a small degradation in the retrieval performance compared to the uncompressed domain implementations.
- Three generic (i.e. content independent) features to describe the local motion of video objects. These descriptors demonstrated the ability to successfully classify local object motion in both compressed and uncompressed domains.

Other important contributions of this thesis are as follows.

- A distance measure to compute the shape distance between two video objects based on the still shape distances of their representative video object planes. Employing this measure resulted in a significant retrieval performance improvement.
- A method to detect and compensate for the potential chroma keying artifacts in MPEG-4 video objects that may cause erroneous color histogram computation. This method was shown to result in significant retrieval performance gains in the presence of such artifacts.

7.2 Future Research Directions

Object-based video retrieval is an emerging research area, which has been driven by the increasing availability of arbitrarily shaped video content, due to the recent standardization of the MPEG-4 object-based representation and the advancements in the segmentation technology. Although some of the techniques used in image and video retrieval are applicable to content-based matching of video objects, there are many issues specific to arbitrarily shaped video retrieval that remain to be explored. In this thesis we addressed some of these issues with a special emphasis on compressed domain processing. We envision the possible research directions that would extend the work of this thesis as follows.

- Combining multiple features: Content matching by using all or a subset of the suggested shape, motion, and color features could potentially lead to better retrieval accuracy. However, selection of the most effective set of features and the weighing their importance in distance computation are still issues that need to be addressed.
- Compact representation: Efficient coded representations of the color, shape, and motion descriptors of video objects need to be developed in order to reduce the storage overhead of these descriptors. For example, this can be achieved by exploiting the possible redundancies in the descriptors of temporally close VOPs.

- Efficient indexing: As the database size increases, proper indexing of content-based features becomes very important, as it will have a direct effect on retrieval speed.
- Optimization for specific applications: Although our proposed techniques are generic, such that they do not assume any prior information about the video content, they can be further optimized for specific applications containing a particular type of video content (for example, head and shoulder sequences, sports, and surveillance sequences), possibly resulting in better retrieval performance in those domains.
- Retrieval of video scenes: The techniques we proposed for retrieval of individual video objects can be extended to the retrieval of video scenes, which are typically composed of one or more video objects.

Bibliography

[1] MPEG-4 Video Group, "Coding of audio-visual objects: Video," ISO/IEC 14496-2, 2000.

[2] W. Pennebaker and J. Mitchell, "JPEG still image compression standard", Van Nostrand Reinhold, New York, 1993.

[3] M. Gormish, D. Lee, M. Marcellin, "JPEG 2000: Overview, architecture, and applications", In Proc. IEEE International Conference on Image Processing, 2000.

[4] ISO/IEC, "Information technology - coding of moving pictures and associated audio for digital storage media at up to about 1.5 mbits/s: Video," 11172-2, 1993.

[5] ISO/IEC, "Information technology - generic coding of moving pictures and associated audio information: Video," 13818-2, 1995.

[6] ITU-T, "Video coding for low bit rate communication," Recommendation H.263, 1996.

[7] L. H. Keister, "User types and queries: impact on image access systems", In Challenges in indexing electronic text and images, pp. 7-22, 1994.

[8] WebSeek: A Content-based image and video search and catalog tool for the web, available on the Internet at http://disney.ctr.columbia.edu/WebSEEk/, 1999.

[9] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, J. Malik, "Blobworld: A system for region-based image indexing and retrieval", International Conference on Visual Information and Information systems, pp. 509-516, 1999.

[10] ImageRover: A content-based image browser for the world wide web, available on the Internet at http://www.cs.bu.edu/groups/ivc/ImageRover/approach.html, 1999.

[11] J.Z. Wang, G. Wiederhold, O. Firschein, S. X. Wei, "Wavelet-based image indexing techniques with partial sketch retrieval capability", Proceedings of the Fourth Forum on Research and Technology Advances in Digital Libraries, pp. 13-24, May 1997.

[12] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, D. Lee, D. Petkovic, D. Steele, P. Yanker, "Query by image and video content: the QBIC system", IEEE Computer, pp. 23-32, September 1995.

[13] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. C. Jain, and C. Shu, "Virage image search engine: an open framework for image management", SPIE Proceedings on Storage & Retrieval for Image and Video Databases, pp. 76-87, 1996.

[14] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Tools for content-based manipulation of image databases", In Proceedings of the SPIE Storage and Retrieval Image and Video Databases II, pp. 34-47, 1994.

[15] Y. Rui, T. Huang and S. Mehrotra, "Content-based image retrieval with relevance feedback in MARS", In Proc. IEEE International Conference on Image Processing, pp. 815-818, 1997.

[16] E. Paquet, and M. Rioux, "Nefertiti: A query by content system for three-dimensional model and image databases management", Image and Vision Computing, pp. 157-166, 1999.

[17] Visual Retrieval Ware, available on the Internet at "http://vrw.excalib.com:8015/cst", 1999.

[18] J.R. Smith and S.F. Chang, "VisualSEEk: A fully automated content-based image query system", ACM Multimedia, pp. 87-98, 1996.

[19] E. Ardizzone and M. La Cascia, "Automatic video database indexing and retrieval", Journal of Multimedia Tools and Applications, no.1, pp. 29-56, January 1997.

[20] S. Srinivasan, D. Ponceleon, A. Amir, D. Petkovic, "What is in that video anyway?: In search of better browsing", IEEE International Conference on Multimedia Computing and Systems, pp. 388-393, 1999.

[21] Y. Deng and B. S. Manjunath "NeTra-V: Toward an object-based video representation", IEEE Transactions on Circuits and Systems for Video Technology, vol.8, no.5, pp. 616-627, September 1998.

[22] S. F. Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong, "A fully automated content-based video search engine supporting spatiotemporal queries", IEEE Trans. on Circuits and Systems for Video Technology, vol.8, no.5, pp. 602-615, September 1998.

[23] S. K. Choubey, "Generic and fully automatic content based image retrieval using color, shape, and texture", Dissertation, The University of Southwestern Louisiana, 1997.

[24] T. Gevers and A. W. M. Smeulders, "PicToSeek: Combining color and shape invariant features for image retrieval", IEEE Trans. on Image Processing, vol.9, no. 1, pp. 102-119, 2000.

[25] Y.A. Aslandogan and C. T. Yu, "Techniques and systems for image and video retrieval", IEEE Transactions on Knowledge and Data Engineering, vol. 11, pp. 56-63, 1999.

[26] J. Wang, "Visual content based image retrieval using shape and color features", Dissertation, State University of New York at Buffalo, 1998.

[27] ISO/IEC JTC1/SC29/WG11, "Multimedia Content Description Interface - Part 3 Visual". Publicly available at http://mpeg.telecomitalialab.com/working_documents.htm, March 2001.

[28] M. Abdel-Mottaleb, N. Dimitrova, L. Agnihori, S. Dagtas, S. Jeannin, S. Krishnamachari, T. McGee, and G. Vaithilingam, "MPEG-7: A content description standard beyond compression", 42nd Midwest Symposium on Circuits and Systems, vol. 2, pp. 770-777, 2000.

[29] M. I. Sezan and R.J. Qian, "MPEG-7 standardization activities", International Conference on Image Processing, pp. 517-520, 1998.

[30] R. Yong, T. S. Huang, and S. F. Chang, "Digital image/video library and MPEG-7: Standardization and research issues, IEEE International Conference on Acoustics, Speech and Signal Processing, vol.6, pp. 3785-3788, 1998.

[31] F. Nack and A. T. Lindsay, "Everything you wanted to know about MPEG-7. Part 1", IEEE Multimedia, vol. 6, issue 3, pp. 65-77, July-September 1999.

[32] F. Nack and A. T. Lindsay, "Everything you wanted to know about MPEG-7. Part 2", IEEE Multimedia, vol. 6, issue 4, pp. 64-73, October-December 1999.

[33] V.V. Vinod and A. Lindsay, "MPEG-7: Its impact on research, industry, and the consumer", IEEE International Conference on Multimedia Computing and Systems, pp. 406-407, 1999.

[34] D. Zhong, and S. F. Chang, "AMOS: An Active System for MPEG-4 Video Object Segmentation", International Conference on Image Processing, vol. 2, pp. 647-651, 1998.

[35] 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, September 1998.

[36] H. T., Nguyen, M. Worring, and A. Dev, "Detection of moving objects in video using a robust motion similarity measure", IEEE Transactions on Image Processing, vol.9, no. 1, pp. 88-101, 2000.

[37] 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, pp. 1147-1169, December 1999.

[38] T. Meier and K. N. Ngan, "Video segmentation for content-based coding", IEEE Transactions on Circuits and Systems for Video Technology, vol. 9, pp. 1190-1203, 1999.

[39] D. Zhong and S. Chang, "An integrated approach for content-based video object segmentation and retrieval", IEEE Transactions on Circuits and Systems for Video Technology, vol. 9, no. 8, pp. 1259-1268, Dec 1999.

[40] B. S. Manjunath, J. -R. Ohm, V. V. Vinod, and A. Yamada, "Color and Texture descriptors," IEEE Trans. Circuits and Systems for Video Technology, vol. 11, no. 6, pp. 703-715, 2001.

[41] B. Erol, A. Dumitras, and F. Kossentini, "Emerging MPEG standards: MPEG-4 and MPEG-7", Handbook of Image and Video Processing, pp. 611-626, Academic Publishers, 2000.

[42] S. Battista, F. Casalino, and C. Lande, "MPEG-4: A multimedia standard for the third millennium. Part 1", IEEE Multimedia, vol. 6, issue 4, pp. 74-83, Oct.-Dec. 1999.

[43] S. Battista, F. Casalino, and C. Lande, "MPEG-4: A multimedia standard for the third millennium. Part 2", IEEE Multimedia, vol. 7, issue 1, pp. 76-84, Jan.-March 2000.

[44] R. Schafer, "MPEG-4: A multimedia compression standard for interactive applications and services", Electronics Communication Engineering Journal, pp. 253-262, 1998.

[45] N. Brady, "MPEG-4 standardized methods for the compression of arbitrarily shaped video objects, IEEE Transactions on Circuits and Systems for Video Technology, pp. 1170-1189, 1999.

[46] F. Pereira, "MPEG-4: A new challenge for the representation of audiovisual information", in Picture Coding Symposium, pp. 7-16, Melbourne - Australia, March 1996.

[47] T. Sikora and L. Chiariglione, "MPEG-4 video and its potential for future multimedia services", Proceedings of IEEE International Symposium on Circuits and Systems, vol .2, pp. 1468-1471, June 1997.

[48] S. Aramvith and M.T. Sun, "MPEG-1 and MPEG-2 video standards", Handbook of Image and Video Processing, pp. 597-610, Academic Publishers, 2000.

[49] MPEG-4 Systems Group, "Coding of audio-visual objects: Systems," ISO/IEC 14496-1, 2000.

[50] MPEG-4 Audio Group, "Coding of audio-visual objects: Audio," ISO/IEC14496-2, 2000.

[51] R. Talluri, "Error-resilient video coding in the ISO MPEG-4 standard", IEEE Communications Magazine, pp. 112-119, June 1998.

[52] I. Moccagatta, S. Soudagar, J. Liang, and H. Chen, "Error-resilient coding in JPEG-2000 and MPEG-4", IEEE Journal on Selected Areas in Communications, pp. 899-914, 2000.

[53] ISO/IEC JTC1/SC29/WG11, "MPEG-4 video verification model 18.0", Publicly available at http://mpeg.telecomitalialab.com/working_documents.htm., January 2001.

[54] J. S. Boreczky, and A. R. Lawrence, "Comparison of video shot boundary detection techniques", Proceedings of SPIE, vol. 2670, pp. 170-179, 1996.

[55] G. Lupatini, C. Saraceno, and R. Leonardi, "Scene break detection: a comparison", Proceedings of RIDE'98, pp. 34-41, February 1998.

[56] V. Kobla, D. Doermann, and K. I. D. Lin, "Archiving, indexing, and retrieval of video in the compressed domain", Proceedings of SPIE conference on Multimedia Storage and Archiving Systems, vol. 2916, pp. 78-89, November, 1996.

[57] V. Kobla, and D. Doermann, "Indexing and retrieval of MPEG compressed video", Journal of Electronic Imaging, vol. 7(2), pp. 294-307, April, 1998.

[58] K. Tse, J. Wei, and S. Panchanathan, "A scene change detection algorithm for MPEG compressed video sequences", Canadian Conference on Electrical and Computer Engineering (CCECE '95), vol.2, pp 827-830, 1995.

[59] B. L. Yeo, and B. Liu, “A unified approach to temporal segmentation of motion JPEG and MPEG compressed video”, IEEE Conference on Multimedia Computing and Systems, pp. 81-88, May 1995.

[60] B. Gunsel, A. M. Tekalp, and P. J. L. Van Beek, “Content-based access to video objects: Temporal segmentation, feature extraction and visual summarization”, IEEE Transactions on Signal Processing, vol. 66, no. 2, pp. 261-280, April 1998.

[61] A. M., Ferman, B. Gunsel, and A. M. Tekalp, “Object-based indexing of MPEG-4 compressed video”, Proceedings of IS&T/SPIE Symp. on Electronic Imaging, vol. 3024, pp. 953-963, February 1997.

[62] 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, pp. 850-863, September 1993.

[63] Microsoft, “MPEG-4 video encoder/decoder,” 2000.

[64] I. Biederman, “Recognition-by-components: A theory of human image understanding”, Psychological Review, Vol.94, No.2., pp. 115-147, 1987.

[65] J. Eakins and M. Graham, “Content-based image retrieval”, A report to the JISC Technology Applications Programme Newcastle, UK: Institute for Image Data Research, University of Northumbria at Newcastle, 1999.

[66] M. Safar, C. Shahabi, and X. Sun, “Image retrieval by shape: A comparative study”, IEEE International Conference on Multimedia and Expo, pp. 141-144, 2000.

[67] A. Khotanzad and Y. H. Hong, “Invariant image recognition by Zernike Moments”, IEEE Transactions on Pattern Analysis and Image Recognition, pp. 489-497, 1990.

[68] S. Abbasi, F. Mokhtarian, and J. Kittler, “Curvature scale space image in shape similarity retrieval,” Springer Journal of Multimedia Systems, vol. 7, number 6, pp. 467-476, 1999.

[69] F. Mokhtarian, "Silhouette-based isolated object recognition through curvature scale space", IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 17, no. 5, pp. 539-544, 1995.

[70] E. Persoon and K. Fu, "Shape discrimination using Fourier descriptors", IEEE Trans. on Pattern Analysis and machine Intelligence, vol. PAMI-8, No. 3, pp. 388-397, 1986.

[71] H. Kauppinen, T. Seppanen, and M. Pietikainen, "An experimental comparison of autoregressive and Fourier-based descriptors in 2D shape classification", IEEE Transactions on Pattern Analysis and Machine Intelligence", vol. 17, no. 2, pp. 201-207, 1995.

[72] Y. Rui, A. C. She, and T. S. Huang, "A modified Fourier descriptor for shape matching in MARS", A chapter in Image Databases and Multimedia Search, Series of Software Eng. and Knowledge Eng., World Scientific Publishing, 1998.

[73] Y. Rui, and T. S. Huang, "Image retrieval: Current techniques, promising directions, and open issues", Journal of Visual Communication and Image Representation, vol. 10, no. 4, pp. 39-62, 1999.

[74] A. Del Bimbo, Visual Information Retrieval, Morgan Kaufmann Publishers, California, 1999.

[75] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Addison-Wesley Publishing Company, 1993.

[76] A. K. Jain, Fundamentals of Digital Image Processing, Prentice Hall, 1989.

[77] A. Vailaya, Y. Zhong, A. K. Jain, "A hierarchical system for efficient image retrieval", Proceedings of ICPR, pp. 356-360, 1996.

[78] A. Del Bimbo and P. Pala, "Visual image retrieval by elastic matching of user sketches", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 19, No. 2, 1997.

[79] J. Wang, W. Yang, and R. Acharya, "Efficient access to and retrieval from a shape image database", IEEE Workshop on Content-Based Access to Image & Video Libraries, pp. 63-67, 1998.

[80] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," to appear in Neural Information Processing Systems, 2001.

[81] J.P. Eakins, K. Shields, and J. Boardman, "ARTISAN - a shape retrieval system based on boundary family indexing", SPIE Conference on Storage and Retrieval for Still Image and Video Databases, pp. 17-28, 1996.

[82] Y. S. Kim and W. Y. Kim, "Content-based trademark retrieval system using visually salient feature", Journal of Image and Vision Computing, pp. 931-940, 1998.

[83] B. Gunsel, and M. Tekalp, "Similarity analysis for shape retrieval by example", Proceedings of 13th Ann. Intl. Conference on Pattern Recognition, pp. 330-334, 1996.

[84] K. Chakrabarti, M. Ortega-Binderberger, K. Porkaew, and S. Mehrotra, "Similar shape retrieval in MARS", IEEE International Conference on Multimedia and Expo II, pp. 709-712, 2000.

[85] ISO/IEC JTC1/SC29/WG11, "MPEG-7 Visual part of eXperimentation model version 8.1", Publicly available at http://mpeg.telecomitalialab.com/working_documents.htm, Jan 2001.

[86] D. Bates and D. Watts, "Nonlinear regression analysis and its applications", Wiley, pp. 271-272, 1988.

[87] B. Erol and F. Kossentini, "Automatic key video object plane selection using the shape information in the MPEG-4 compressed domain", IEEE Transactions on Multimedia, pp. 129-138, June 2000.

[88] A. Smolic, M. Hoeynck and J.-R. Ohm, "Low-complexity global motion estimation from P-frame motion vectors for MPEG-7 applications", IEEE International Conference on Image Processing, vol. 2, pp. 271-274, September 2000.

[89] H.J. Zhang, Y.L. Chien, and S.W. Smoliar, "Video parsing and browsing using compressed data," Proc. Multimedia Tools and Applications, Vol. 1, No. 1, pp. 89-111, 1995.

[90] M. Shah, K. Rangarajan, and P.S. Tsai "Motion trajectories", IEEE Trans. on Systems, Man and Cybernetics, pp. 1138-1150, 1993.

[91] R. Brunelli, O. Mich, and C. Modena, "A survey on video indexing", IRST Technical Report 9612-06, 1996.

[92] S. Dagtas, W. Al-Khatib, A. Ghafoor, and R.L. KashYap, "Models for motion-based video indexing and retrieval" IEEE Trans. on Image Processing, vol.9, no. 1, pp. 88-101, 2000.

[93] H. Sawhney ans S. Ayer, "Model-based 2D&3D Dominant Motion Estimation for Mosaicing and Video Representation", In Proceedings of ICCV, 1995.

[94] A. M. Ferman, B. Günsel, and A. M. Tekalp, " Motion and shape signatures for object-based indexing of MPEG-4 compressed video", in IEEE Proc. ICASSP, vol. IV, pp. 2601-2604, Apr. 1997.

[95] J. Little and Z. Gu, "Video retrieval by spatial and temporal structure of trajectories", To appear in proceedings of SPIE, 2001.

[96] K. Rangarajan, W. Allen, and M. Shah "Matching motion trajectories using scale-space", Pattern Recognition, pp. 595-610, 1993.

[97] W. Chen and S.F. Chang, "Motion trajectory matching of video objects", Proc. SPIE Storage and Retrieval for Media Databases, pp. 544-553, 2000.

[98] Z. Aghbari, K. Kaneko, and A. Makinouchi, "Modeling and querying videos by content trajectories", IEEE International Conference on Multimedia and Expo, pp. 463-466, 2000.

[99] Z. Aghbari, K. Kaneko, and A. Makinouchi, "A motion-location based indexing method for retrieving MPEG videos", DEXA Workshop, pp. 102-107, 1998.

[100] S. Jeannin and B. Mory, "Video motion representation for improved content access", International Conference on Consumer Electronics, pp. 284-285, 2000.

[101] G. Stalidis, N. Maglaveras, A. Dimitriadis, and C. Pappas, "Modeling of cardiac motion using wavelets: comparison with Fourier-based models", Computers in Cardiology, pp. 733-736, 1998.

[102] H. Miyamori and S. Iisaku, "Video annotation for content-based retrieval using human behavior analysis and domain knowledge" IEEE International Conference on Automatic Face and Gesture Recognition, pp. 320-325, 2000.

[103] J. Hoey and J. Little, "Representation and recognition of complex human motion", IEEE Conference on Computer Vision and Pattern Recognition, vol 1, pp. 752-759, 2000.

[104] H. Fujiyoshi, and A.J. Lipton, "Real-time human motion analysis by image skeletonization", Proceedings of IEEE Workshop on Applications of Computer Vision, pp. 15 - 21, Oct. 1998.

[105] J. Little and J. E. Boyd, "Recognizing People by Their Gait: the Shape of Motion", Videre: Journal of Computer Vision Research, vol. 1, no. 2, 1998.

[106] B. Heisele and C. WoehlerC, "Motion-based recognition of pedestrians", IEEE International Conference on Pattern Recognition, vol. 2, pp. 1325-1330, 1998.

[107] R. Cutler and L.S. Davis, "Robust real-time periodic motion detection, analysis, and applications", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 781-796, 2000.

[108] S. Panchanathan, Y.C. Park, K. S. Kim, and F. Golshani, "The role of color in content-based Retrieval", IEEE International Conference of Image Processing, pp. 517-520, 2000.

[109] A. M. Ferman, S. Krishnamachari, A. M. Tekalp, M. Abdel-Mottaleb, and R. Mehrotra, "Group-of-frame/picture color histogram descriptors for multimedia applications", Proceedings of. IEEE International Conference on Image Processing, September 2000.

[110] R. Chang, W. Kuo, and H. Tsai, "Image retrieval on uncompressed and compressed domains", IEEE International Conference of Image Processing, vol. 2, pp. 546-549, 2000.

[111] M. Shneierand and M. Abdel-Mottaleb "Exploiting the JPEG compression scheme for image retrieval", IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 18, no. 8, pp. 849-853, 1996.

[112] K. Shen and E. J. Delp, "A fast algorithm for video parsing using MPEG compressed sequences", Proceedings of the IEEE International Conference on Image Processing, pp. 252-255, 1995.

[113] B. L. Yeo and B. Liu, "On the extraction of DC sequence from MPEG compressed video", International Conference on Image Processing, vol.2, pp. 260-263, 1995.

[114] M. Miyahara and Y. Yoshida, "Mathematical transform of (R,G,B) color data to Munsell (H,V,C) color data," in SPIE Visual Communications and Image Processing, vol. 1001, pp. 650-657, 1988.

[115] J. D. Foley, A. Van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*, Addison-Wesley, 1990.

[116] M. Ferman, S. Krishnamachari, M. Tekalp, M. Abdel-Mottaleb, and R. Mehrotra, "Group of Frames Color Histogram Descriptors for Multimedia Applications", IEEE International Conference of Image Processing, 2000.

[117] R. Brunelli and O. Mich, "On the use of histograms for image retrieval", Proceedings of ICMCS, vol. 2, pp. 143-147, 1999.

[118] D. Androutsos, K.N. Plataniotis and A.N. Venetsanopoulos, "Efficient color image indexing and retrieval using a vector-based scheme", Proceedings of the Multimedia Signal Processing Conference, pp. 14-20, December 1998.

Appendix A

Retrieval Performance Measures

Typically, performance of a retrieval system is evaluated by first having human subjects mark the relevant database items (which are also referred to as ground truth items) for a given query and then measure the level of agreement between human evaluators and the system ranking of the relevant database items. Establishing a generic metric that would be applicable to a wide variety of content for measuring the similarity in the human and system ranking is still an open research issue.

In traditional information retrieval, performance of document classification is measured by *precision* and *recall* metrics [74]. Recall measures the ability of the system to retrieve all documents that are relevant as follows,

$$\text{Recall} = \frac{\text{relevant and correctly retrieved}}{\text{all relevant}}. \tag{A.1}$$

Note that, if all database items that are relevant to the query item are retrieved, recall would take its maximum possible value, i.e. "1". On the other hand, *precision* measures the ability of the system to retrieve only the documents that are relevant and it is given by

$$\text{Precision} = \frac{\text{relevant and correctly retrieved}}{\text{all retrieved}}. \tag{A.2}$$

Here, if all the retrieved items are relevant to the query item, *precision* would be equal to "1".

Although precision and recall measures are simple to compute and are useful in assessing the performance of retrieval systems where a large test set is available, they have the following limitations.

- Precision and recall metrics do not consider the ranking of the relevant items. For example, let's assume that for a given query human subjects marked 10 items as relevant (ground truth items) and the retrieval systems A and B output 50 items as relevant, including the 10 ground truth items. Clearly, 40 items outputted by the retrieval systems are not relevant to the query. Lets say system A ranks the ground truth items in the first 10 and system B ranks the relevant items in the last 10, i.e. from 40 to 50. Clearly, system A shows a superior performance. However, precision and recall measures would be identical in both cases, as they do not consider ranking.
- Precision and recall measures require large databases in order to accurately measure the retrieval performance. Performing subjective tests and marking the ground truth items in large databases is very time consuming and costly.

- In retrieval, typically a threshold is used to eliminate the items that are not relevant to the query item. That is, if the distance between the query and database item is larger than a threshold, it is not retrieved. Precision and recall values very much depend on this threshold value. For example, two identical systems could have inconsistent precision and recall rates for the same query, if the threshold values are chosen differently.

There have been some attempts in the literature to propose metrics that would overcome these limitations [78]. However, most of these metrics are customized for specific retrieval applications.

Recall that MPEG-7 is aimed towards standardizing content-based description interface. Similar to the previous MPEG standards, MPEG-7 also had a competition stage where different technologies competed for the best performance. In order to measure the retrieval performance, a new metric, Normalized Modified Retrieval Rank (NMRR), was employed in various core experiments [40]. NMRR not only indicates how much of the correct items are retrieved, but also how highly they are ranked among the retrieved items, making this measure more robust even for small databases. Also, because of the way this measure is defined, the dependency on a threshold value (that is used to eliminate irrelevant items) is removed. These properties make the NMRR metric very suitable to measure the performance of our proposed techniques. Next, we present a detailed description on the computation of NMRR.

A.1 Normalized Modified Retrieval Rank

The Normalized Modified Retrieval Rank (NMRR) is given by

$$\text{NMRR(n)} = \frac{\left(\sum_{k=1}^{NG(n)} \frac{Rank(k)}{NG(n)}\right) - 0.5 - \frac{NG(n)}{2}}{K + 0.5 - 0.5 * NG(n)}, \tag{A.3}$$

where *NG(n)* is the number of ground truth items marked as similar to the query item *n*, *Rank(k)* is the ranking of the ground truth item *k* by the retrieval algorithm. K equals to *min(4*NG(n), 2*GTM)*, where *GTM* is the maximum of *NG(n)* for all the queries. A rank of *(K+1)* is assigned to each of the ground truth images that are not among the first *K* retrieved items.

The NMRR is in the range of [0 1] and the smaller values represent a better retrieval performance. Average NMRR (ANMRR) is defined as the average NMRR over a range of queries.

A.2 NMRR versus Precision/Recall

Considering that the NMRR is a relatively new metric and some readers may be more familiar with precision and recall metrics, we provide some example retrieval results and present the relevant NMRR, precision, and recall values. In our example, we assume that there are 10 items in a database that are relevant to the query (i.e. marked as similar to the query by subjective testing) and that 20 items are retrieved in total by the retrieval system. Table A.1 shows the different NMMR rates and corresponding precision and recall values for various retrieval performance levels. Note that the maximum value of the precision here is 0.5, as twice as many items as the ground truth items are retrieved in all cases. The table also shows the hits, misses, false alarm rates, and the ranking of the relevant items that correspond to each NMRR value. Here, "hits" refers to the number of relevant items that are in the first 20 retrieved items. Considering that, in this example, there are 10 items marked as relevant to the query, when "hits" takes a value of 10, this means that all the relevant items are retrieved. The "Misses" column refers to the number of relevant items that are not ranked in the first 20. The "False alarm rate" column, on the other hand, corresponds to the number of falsely retrieved (irrelevant) items ranked in the top 10.

As can be observed from the corresponding hits, misses, and false alarm rates, when the NMRR values are smaller than 0.1, the retrieval performance is very close to the ideal case, i.e., almost all the relevant items are retrieved correctly. When the NMRR values are between 0.1 and 0.2, the retrieval performance is still very good, as more than 80% of the items are correctly retrieved. The NMRR values above 0.4 correspond to poor retrieval performance.

NMRR	Recall	Precision	hits[*]	misses[**]	false alarm rate[***]	ranking of the relevant items
0	1	0.5	10	0	0	1,2,3,4,5,6,7,8,9,10
0.01	1	0.5	10	0	1	1,2,3,4,5,6,7,8,10,11
0.05	1	0.5	10	0	2	1,2,3,4,5,6,7,8,13,14
0.08	0.9	0.45	9	1	1	1,2,3,4,5,6,7,9,10
0.10	0.9	0.45	9	1	1	1,2,3,4,5,7,8,9,10
0.14	0.9	0.45	9	1	3	1,2,3,4,5,6,7,13,14
0.16	1	0.5	10	0	5	1,2,3,4,5,11,12,13,14,15
0.19	0.8	0.4	8	2	3	1,2,3,4,5,8,9,11
0.21	0.9	0.45	9	1	2	1,2,4,6,7,8,9,10,19
0.25	0.8	0.4	8	2	3	1,2,5,6,7,8,9,13
0.29	1	0.5	10	0	5	1,3,5,7,9,11,13,15,17,19
0.35	0.7	0.35	7	3	5	1,2,3,6,8,12,14
0.42	0.5	0.25	5	5	5	1,2,3,4,5
0.63	0.3	0.15	3	7	7	1,2,3
0.65	1	0.5	10	0	10	11,12,13,14,15,16,17,18,19, 20
0.9	0.5	0.25	5	5	10	16,17,18,19,20
1	0	0	0	10	10	-

Table A-1. NMRR vs. precision/recall values for various retrieval performances.

[*] Correctly retrieved items (i.e. relevant to the query) in top 20.

[**] Relevant items that are not retrieved (i.e. missed) in top 20.

[***] Falsely retrieved items (i.e. not relevant to the query) in top 10.

Appendix B

Video Object Database

Our database contains video objects obtained by employing automatic and semi-automatic post-production segmentation tools, as well as employing chroma keying technique where the foreground object is separated from the background by placing the object in front of a color screen that has a unique chroma key value (typically blue or green). The representative VOPs of the database video objects are presented in Figure B.1 to Figure B.20. As can be seen from the figures, our database covers a good variation of shapes and motion. Nevertheless, because it is desired that similar database items exist in the database for some queries, the database is selected such that some of the video objects in the database show similarity.

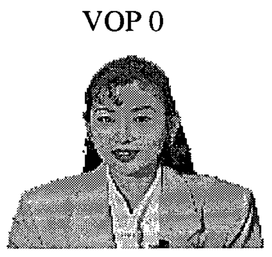


Figure B.1: Akiyo video object.

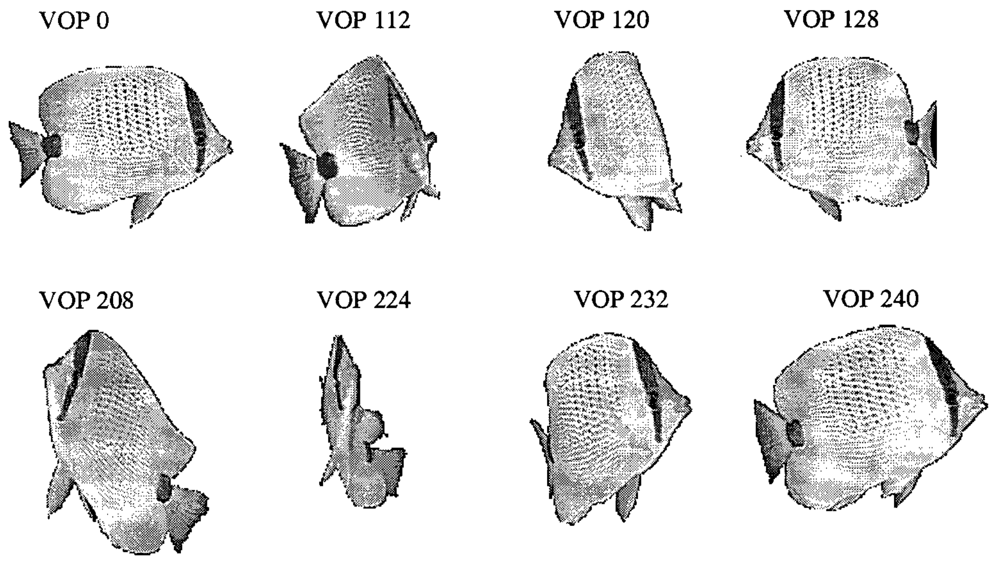


Figure B.2: Bream video object.

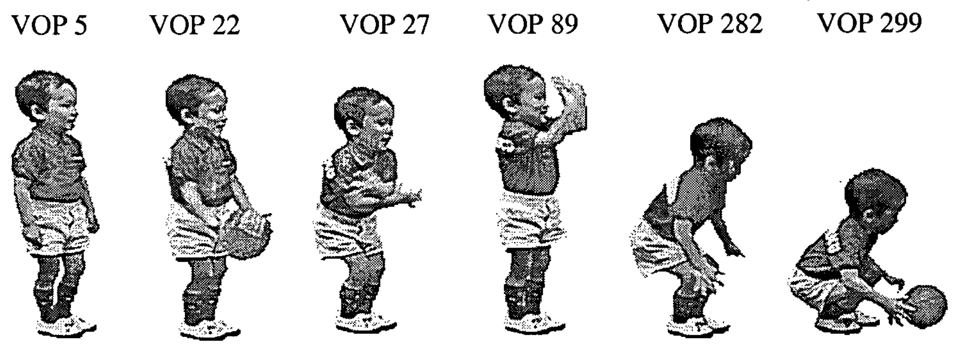


Figure B.3: Children 1 video object.

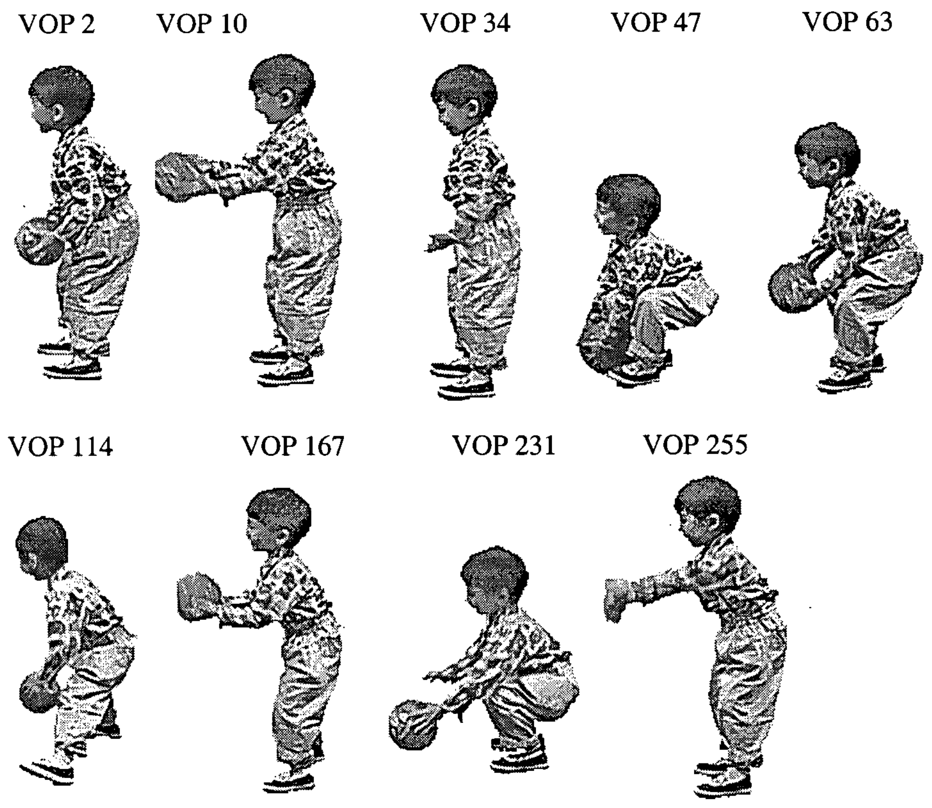


Figure B.4: Children 2 video object.

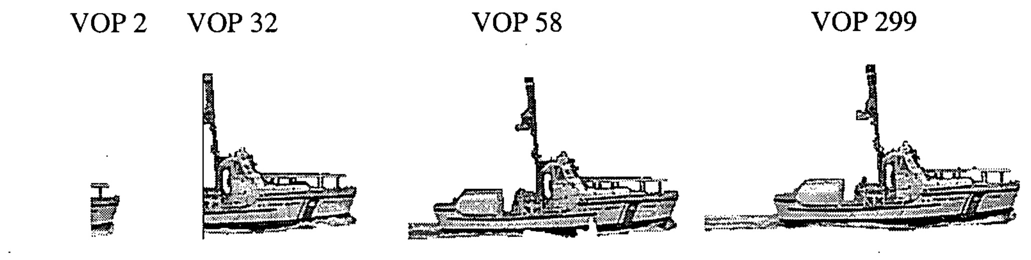


Figure B.5: Coastguard 1 video object.

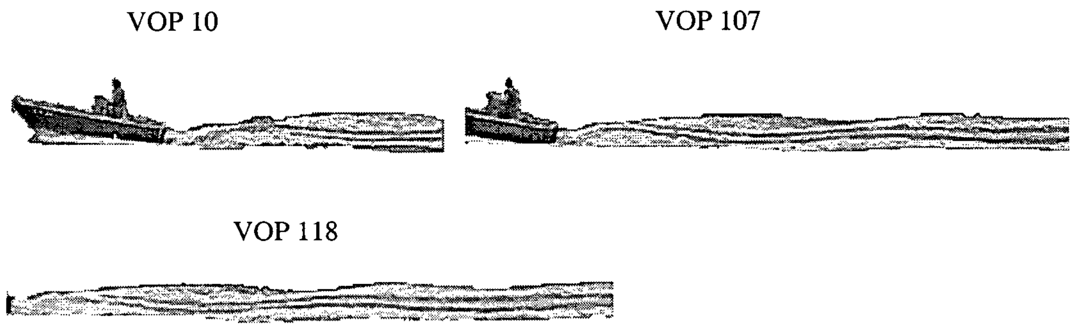


Figure B.6: Coastguard 2 video object.

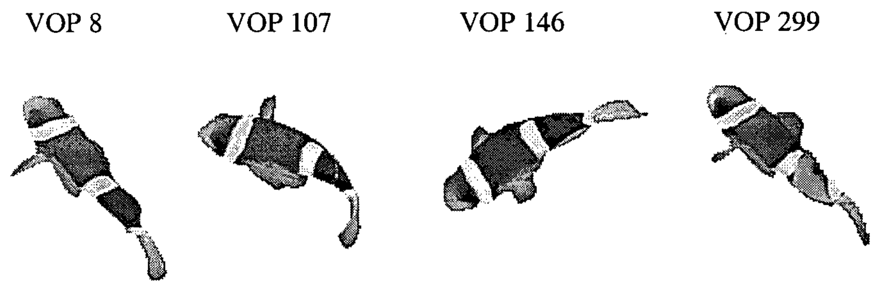


Figure B.7: Fish 1 video object.

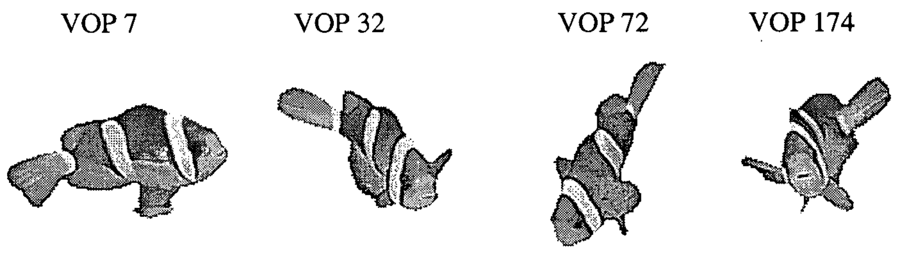


Figure B.8: Fish 2 video object.

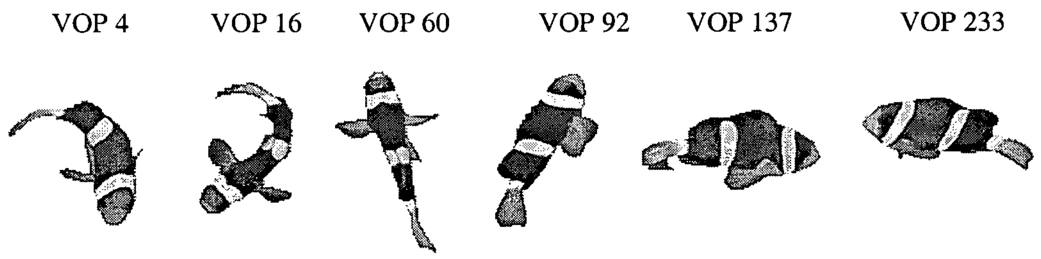


Figure B.9: Fish 3 video object.

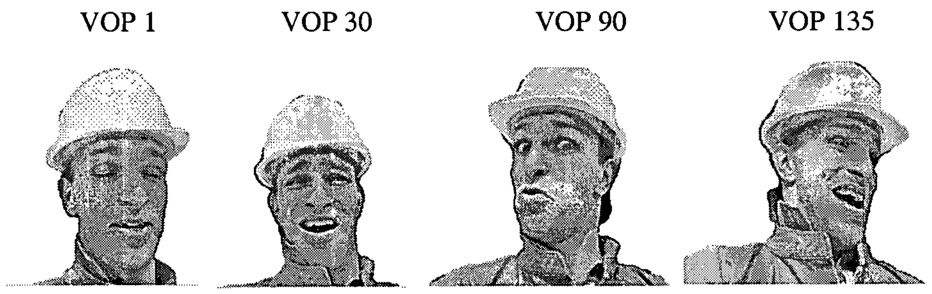


Figure B.10: Foreman video object.

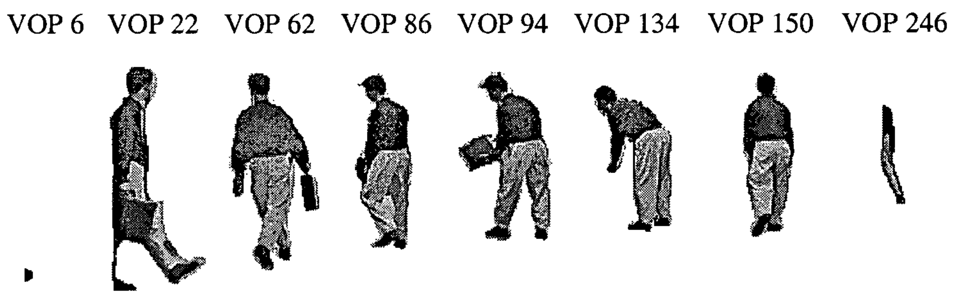


Figure B.11: Hall Monitor 1 video object.

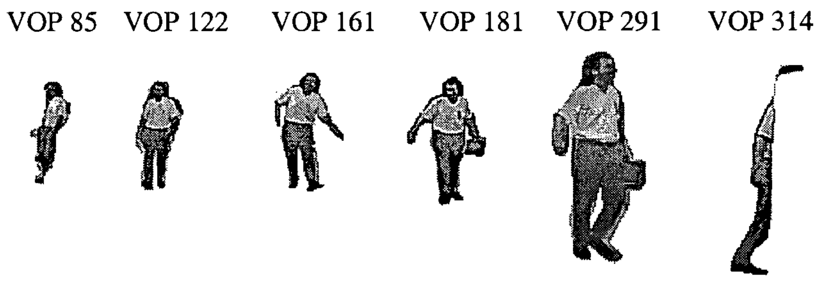


Figure B.12: Hall Monitor 2 video object.

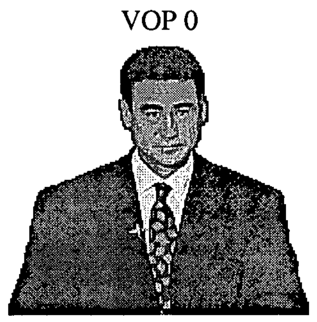


Figure B.13: News 1 video object.

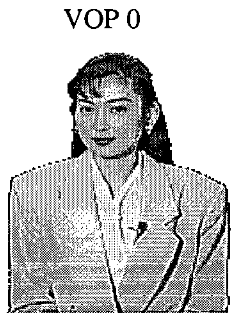


Figure B.14: News 2 video object.

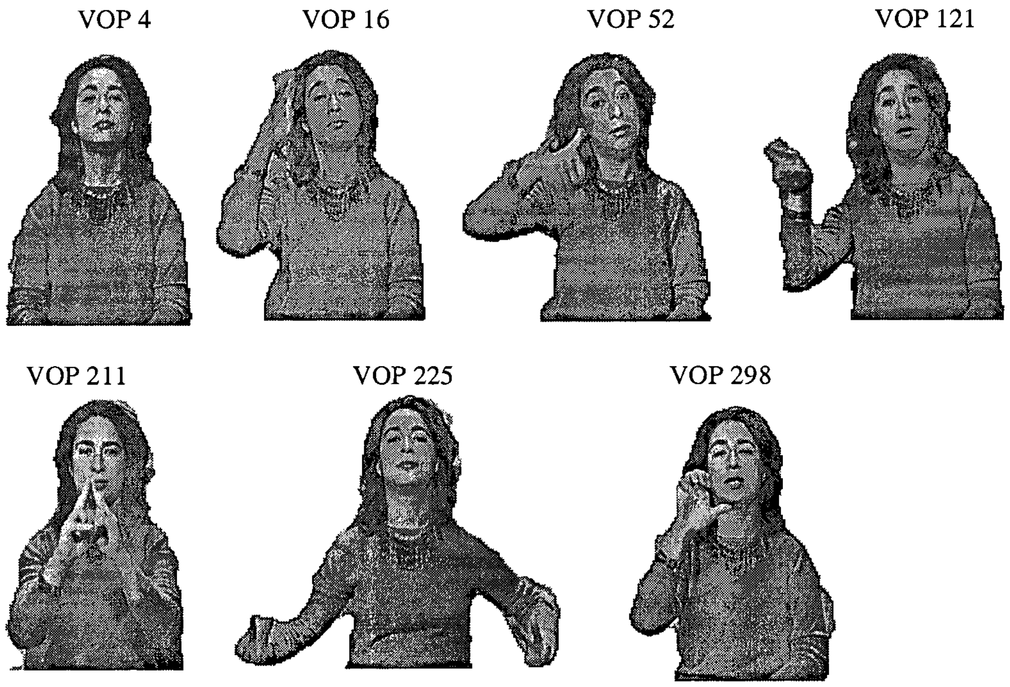


Figure B.15: Silent video object.

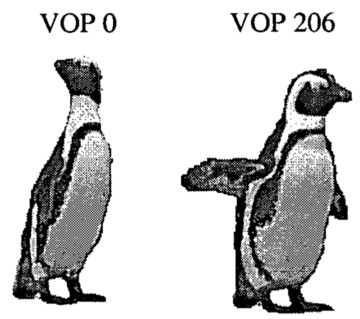


Figure B.16: Penguin video object.

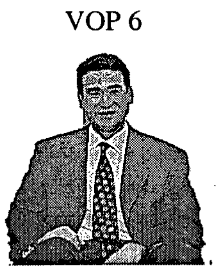


Figure B.17: Sean video object.

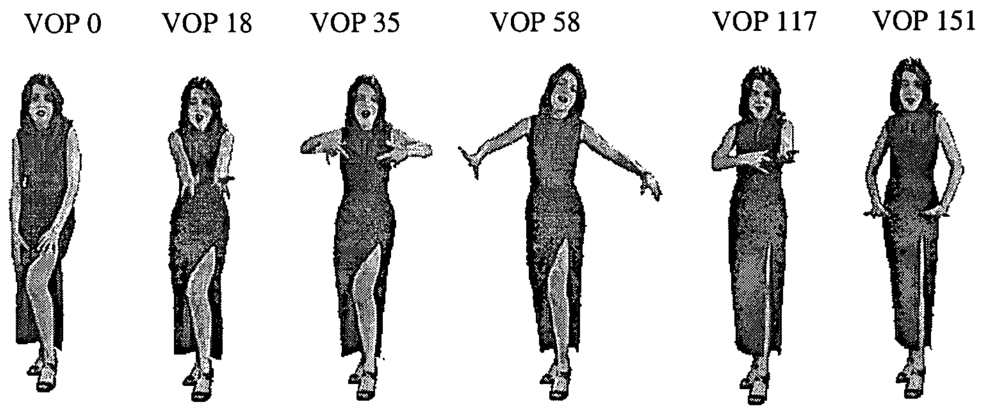


Figure B.18: Singing girl video object.

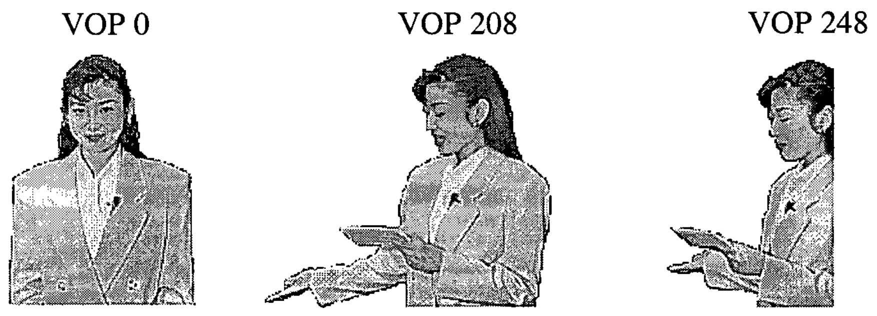


Figure B.19: Weather video object.

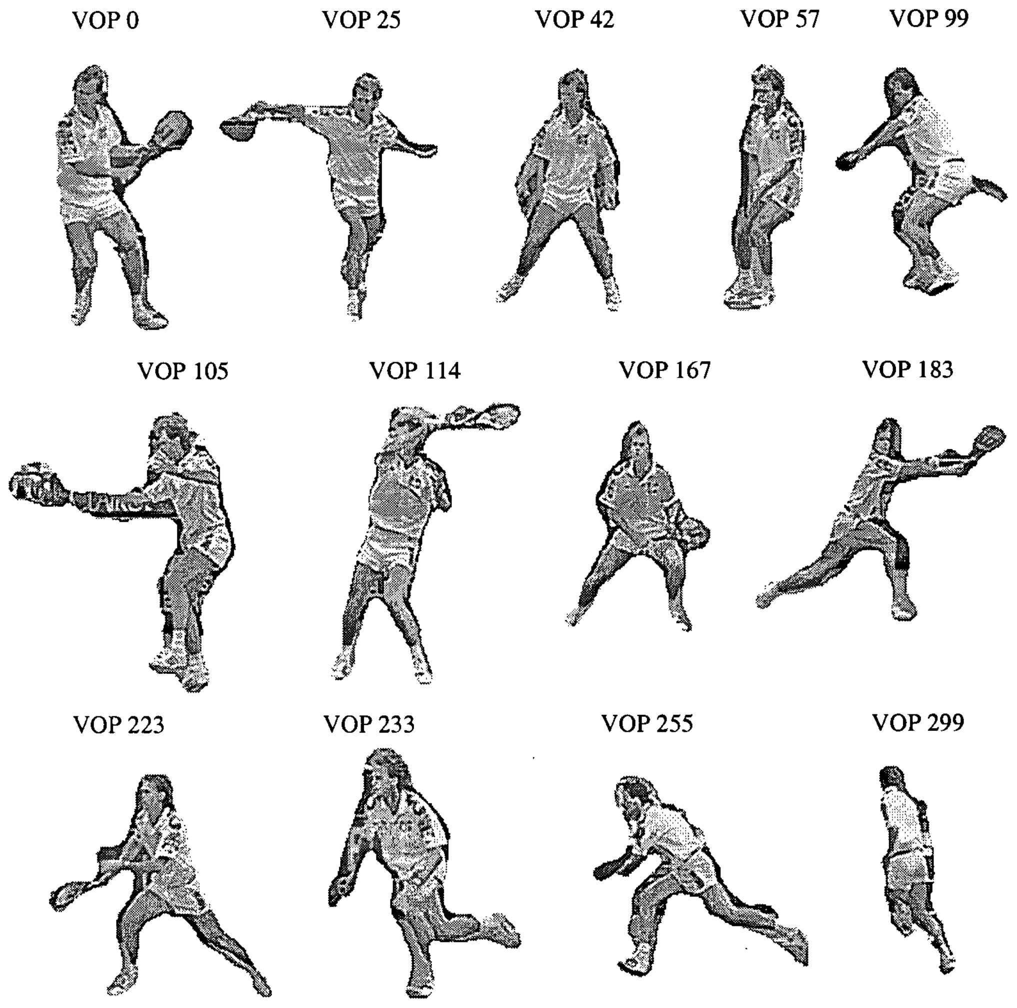


Figure B.20: Stefan video object.

Appendix C

Color Space Conversions

In this section, we present the color space conversions from the YC_bC_r to the HSV and MTM (Mathematical Transformation to Munsell) spaces. More information regarding to these color spaces can be found in [26][74][114][115].

C.1 YC_bC_r to HSV Conversion

The HSV color space consists of Hue (H), Saturation (S), and Value (V) components. H represents the dominant spectral tone of the color, S indicates how dominant the color is and V represents the brightness of the color. Converting the Y (luminance), C_b (chrominance b), and Cr (chrominance r) components to the H, S, and V components requires a non linear transformation as given below.

$$R = Y + 1.403\, C_r, \tag{C.1}$$

$$G = Y - 0.344\, C_b - 0.714\, C_r, \tag{C.2}$$

$$B = Y + 1.773\, C_b, \tag{C.3}$$

$$H = acos\left(\frac{\frac{1}{2}\left((R-G)+(R-B)\right)}{\sqrt{(R-G)^2+(R-B)(G-B)}} \right), \tag{C.4}$$

$$S = 1 - \frac{3}{R+G+B}\min(R,G,B), \tag{C.5}$$

$$V = \frac{R+G+B}{3}. \tag{C.6}$$

C.2 YC_bC_r to MTM Conversion

The MTM color space is formed by Hue (H), Value (V) and Chroma (C) components, which are computed from the Y, C_b, and Cr components in three steps as follows.

Step 1: Linear conversion to X, Y, Z: First convert the Y, C_b, and Cr components to R, G, and B components using the equations B.1, B.2, and B.3. Then perform the following conversion:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.608 & 0.174 & 0.200 \\ 0.299 & 0.587 & 0.144 \\ 0.000 & 0.066 & 1.112 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{C.7}$$

Step 2: Non linear conversion concerned with the human visual system is performed as:

$$H_1 = V(X_c) - V(Y), \tag{C.8}$$

$$H_2 = V(Z_c) - V(Y), \text{ and} \tag{C.9}$$

$$H_3 = V(Y), \tag{C.10}$$

where $V(n)=11.6n^{1/3} - 1.6n$, and n= X_c, Y, and Z_c.

Step 3: Finally, the H, V, and C components are computed as follows:

$$H = atan\left(\frac{H_1}{0.4 H_2}\right), \tag{C.11}$$

$$V = 0.23 H_3, \text{ and} \tag{C.12}$$

$$C = \sqrt{{H_1}^2 + 0.16 {H_2}^2} \, . \tag{C.13}$$