(P)

M0017108SR.

Highly Scalable 2D Model-based Video Coding

Mingyou HU

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

UniS

Centre for Communication Systems Research (CCSR)
School of Electronics and Physics Science
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

August 2005

To my parents, my wife YANG Chunhua, and my baby HU Hongli

Abstract

With rapid mergers of computer, communications, and entertainment industries, we can expect a trend of growing heterogeneity (in channel bandwidth, receiver capacity, etc.) for future digital video coding applications. Furthermore, some new functions appear, such as object manipulation, which should be supported by the video coding techniques. The traditional video coding approach is very constrained and inefficient to the heterogeneity issue and user interaction. Scalable coding, allowing partial decoding at a variety of resolution, temporal, quality, and object levels from a single compressed codestream, is widely considered as a promising technology for efficient signal representation and transmission in a heterogeneous environment. However, although several scalable algorithms have been proposed in the literature and the international standards over the last decade, further research is necessary to improve the compression performance of scalable video coding.

This thesis investigates scalable 2D model-based video coding method with efficient video compression as well as excellent scalability performance, in order to satisfy the newly appeared requirements. It first examines main model-based video coding techniques and scalable video coding methods. Also, the parametric video models that describe the real world and image generation process are briefly described.

Next, video segmentation algorithms are investigated to semantically represent the video frame into video objects. At the first frame, the texture information and the motion from first several frames are used to extract the semantic foreground objects. For some sequences, user interaction is required to get semantic objects. In later frames, the proposed complexity-scalable contour-tracking algorithm is used to segment each frame. After that, each object is progressively approximated using three-layer 2D mesh model. In order to represent the motion of human face more precisely, face detection and modelling are also investigated. This technique, in which human face is modelled separately, is shown to produce improvements of object motion representation.

Scalable model compression is also outlined in this thesis. Object model is represented into two parts: object shape and interior object model, which are compressed separately. A scalable contour approximation algorithm is proposed. Both intra- and predictive scalable shape-coding algorithms are investigated and proposed to code the object shape progressively.

The encoded coarser layers are used to improve the coding efficiency of the current layer. The effectiveness of these algorithms is demonstrated through the results of extensive experiments.

We also investigate the scalable texture coding of video objects. An improved shape-adaptive SPECK algorithm is employed in intra-texture coding and is also used for residual texture coding after motion compensated temporal filtering. During motion compensated temporal filtering, scalable mesh object model is used, and scalable motion vector coding is achieved using CABAC codec. A hierarchically structured bitstream is created, which is optimised for rate-distortion, to facilitate efficient bit truncation and bit allocation among video frames and video objects. The coding system can encode/decode the video object independently and generate a separate bit stream for each object. As is exhibited in our experiments, such a high coding scalability in the proposed coding system is achieved without a significant cost in compression performance commonly experienced in most scalable coding systems.

Key words: Model-based video coding, scalable video coding, video processing and segmentation, vertex-based shape coding, motion compensated temporal filtering (MCTF).

Acknowledgements

I would like to thank my main PhD supervisor *Dr. Abdul H. Sadka* and my second supervisor *Dr. Stewart Worrall*, whose support, guidance and suggestions were invaluable.

I consider myself very fortunate to have studied at the Centre for Communication Systems Research (CCSR). My graduate research was financially supported by UK ORS Award and funding from CCSR. I would also like to thank *Prof. Ahmet Kondoz* and other colleagues in the multimedia group for their support and especially for providing me a friendly working environment in the multimedia Lab. I would like to acknowledge the support given by the computing staff at CCSR.

Most importantly, I deeply appreciate my family – my parents, my wife Chunhua YANG, and my son Hongli HU – for their love, support and encouragement throughout my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Preamble

Currently, multimedia technology provides content creators and consumers with a myriad of coding, access, and distribution possibilities. At the same time, communication infrastructure is being put into place to enable access to information and multimedia services from almost anywhere at anytime. These make the Universal Multimedia Access possible. In order to achieve an efficient Universal Multimedia Access (UMA), video coding plays an important role, which should support several new functionalities, such as efficient scalable coding, and object manipulation.

Much research has been conducted for video coding. Some recently standardised video coders, such as MPEG-4 [MPEG4-2001] and H.264 [H264-2003], can compress the video efficiently. For example, H.264 can save about 50% of total bits when compared with MPEG-2. However, these coders utilise waveform-based video coding approaches, which are based on the combination of block-based motion compensation with Discrete Cosine Transform (DCT). One of the main problems of these now so-called *first generation coding techniques* is that they did not question the image representation structure imposed by the canonical representation of the image. These techniques use pixel or block of pixels as the basic entities that are coded. In addition, they also share in common the absence of consideration for the human visual system (HVS). These standardised video codecs, such as MPEG-4 and H.264, present severe limitations for very low bit-rate video coding applications.

In order to improve the visual performance of video coding, some totally different video coding methods are introduced, which are known as *second generation video coding techniques* [TORRES-1996]. Based on the employed source models, these methods are divided into region-based video coding [ERYU-1995] [CZEREPINSKI-1997], object-based video coding [GERKIN-1994], knowledge-based video coding [KAMP-1997b] and semantic video coding [CHOI-1994].

In fact, all of these methods can be uniformly considered as the model-based video coding because the encoders compress the video object-by-object, instead of pixel-by-pixel. The block diagram of these methods can be simply depicted in Figure 1.1

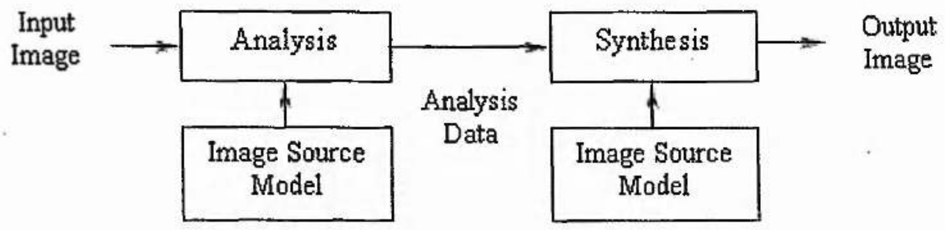


Figure 1.1 – General description of a model-based coding system

Model-based video coding has been an active area of research for a number of years [AIZA-1995] [PEAR-1995]. It makes use of a variety of source models taking into account the structural features of the image. Semantic knowledge of the scene can be also exploited to achieve high efficiency when encoding video sequences for certain scenarios. In model-based video coding, both 2-D and 3-D models are employed at various levels of complexity and accuracy to provide the optimal match between the video scene and the employed model. Therefore, the video frames are described by specifying 2D and/or 3D motion and deformation of the objects in the scene. Since only a few parameters are necessary to qualify the temporal changes, extremely low bit-rate can be achieved. Good compression performance has been achieved at very low bit rates with a model-aided coder [EISERT-2000]. Compared to DCT-based video coders, model-based video coding does not show any blocking artefacts at low data rates.

However, for model-based video coding, 3D model-based codecs have a major disadvantage in that they can only be used for sequences in which the foreground object closely matches the pre-defined reference model. 3D model-based video coding techniques are too rigidly object-specific because the extraction of 3D structure from single objects in an unrestricted environment and the efficient modelling of their surfaces are extremely difficult tasks. However, modeling objects is a very important issue in model-based video coding as the complexity of analysis and synthesis depends on the adopted model. Therefore, at present, 3D model-based video coding is just applied for coding head-shoulder sequences [LI-1993][PEAR-1995]. 3D model-based video coding has another disadvantage that it is very sensitive to channel errors [WORRALL-2002]. As the transmitted parameters will be employed by the decoder to synthesise the original frames, the error of these parameters will affect the final synthesis performance.

Much research has also focused on 2D model-based video coding [ALTUNB-1997] [TEKALP-1997]. Compared with 3D model-based video coding, 2D model has several advantages:

- 2D model-based video coding is rather universal and not limited to head-shoulder sequences. 2D mesh models (unlike 3D wireframe models) can be easily designed for arbitrary scenes.

- 2D parametric motion estimation is a better-posed problem than 3D motion and structure estimation. Therefore, the analysis process of 2D model-based video coding is much easier than that of 3D model-based video coding.

Although *a priori* knowledge of objects can be used to improve the efficiency of 3D model-based video coding, it can also be used efficiently to improve 2D model design and object coding. Research results show that 2D model-based coding with affine / perspective transformation and triangular mesh models can simulate almost all capabilities of 3D model-based approaches using wireframe models at a fraction of the computational cost [ALTUNB-1997] [TEKALP-1997].

1.2 Objective and overall project description

The objective of this thesis is to develop an efficient scalable 2D model-based video coding scheme, which tries to achieve scalable video coding with high compression efficiency and improve the scalability of model-based video coding. The application of 2D object model is to overcome the reduced generality of 3D object model. Moreover, research results show that 2D model-based coding can simulate almost all capabilities of 3D model-based approaches at a fraction of the computational cost. The investigation of scalable coding is to increase the robustness of transmission of the generated bitstreams, which also facilitate the achievement of new functionalities, such as Universal Multimedia Access (UMA) [BORMANS-2003].

To achieve this objective, the overall project will include the following parts:

- Video segmentation and scalable modelling

  Video segmentation and modeling are one of the important steps to achieve model-based video coding. In order to model the video frames using video objects, accurate and semantic video segmentation is necessary and important. Currently, much research has been conducted for video segmentation. Most segmentation algorithms use the low-level features, such as motion or colour information, to distinguish the video object from a video frame. The main problem of these algorithms is the accuracy of segmentation results. Further research is required to achieve accurate (semi-) automatic and semantic video segmentation. In order to improve the coding performance of 2D model-based video coding, *a priori* knowledge of special objects, such as human face, can be employed. To achieve this, face detection and facial feature extraction is also investigated in this project. After getting the video objects, scalable object modelling is also an important step for 2D model-based video coding. The designed object model should represent the object model motion precisely. The accuracy of modelling will affect its compression performance.

- Scalable compression of object model

  Scalable object model compression is one of the major issues in scalable 2D model-based video coding scheme. In this project, the compression of object model includes both interior model compression and model contour compression (or shape coding). In shape coding, content-adaptive arithmetic encoding (CAE) and chain coding techniques are widely used. The CAE scheme is well integrated into the current MPEG-4 scheme. Commonly, it costs several kbits to encode the model of each object. For scalable shape coding, the CAE technique shows a visually annoying staircase effect [BRADY-1997]. Further research is needed to improve the encoding performance and scalability of object shape.

- Scalable texture coding of video objects

  Scalable texture coding is another major issue in scalable 2D model-based video scheme. In past decades, there have been continuous efforts in developing techniques for arbitrarily shaped video objects. In MPEG-4, shape-adaptive DCT (SA-DCT) scheme is used to encode the texture of video objects. However, there are several disadvantages of the SA-DCT technique. First, it inherits the blocking effect from the DCT. Second, in the implementation of SA-DCT, the alignment of the coefficients destroys the spatial correlation to some extent. Therefore, the coding efficiency is degraded. Experimental results show that SA-DCT performance is inferior to that of shape-adaptive wavelet-based scheme [SHIPENG-2000]. Recently, many wavelet-based texture-coding algorithms have been extended to shape-adaptive object texture coding [EGGER-1996] [KIMJ-1998] [SHIPENG-2000]. Further research is necessary to improve its coding efficiency as well as its scalability.

1.3 Source Material and Performance Evaluation

The colour video sequences used in the performance evaluation of the simulated techniques and algorithms are the conventional ITU test sequences. In order to test the efficiency of the developed scalable 2D model-based video coding algorithms, various test sequences with different properties have been utilised, such as Foreman, Carphone, Claire, Akiyo, News, Motr_dhtr, Coastguard sequences. For Foreman, Carphone, Claire and Akiyo sequence, human face is considered as a separate object and modelled separately, which is different from News, Motr_dhtr and Coastguard sequences.

On the other hand, to evaluate the performance of the proposed video coding algorithms, both subjective and objective methods have been adopted. The performance of the considered video algorithm can be evaluated by simply comparing the original and the reconstructed video sequences. However, the subjective evaluation is more desirable even though it requires a number of users to spend much time to view and compare a number of different decoded video sequences.

The most common objective method for comparing video quality is to use the Peak-to-peak Signal to Noise Ratio (PSNR) equation. This equation for PSNR is shown below:

$$PSNR = 10\log_{10}\left(\frac{255^2}{\left(\frac{1}{M \times N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(x(i,j)-\hat{x}(i,j))^2\right)}\right) \tag{1.1}$$

where M and N stand for the dimension of the video sequence. For QCIF (Quarter Common Intermediate Format) sequence, these variables are always 176 and 144.

For a fair performance evaluation of a video-coding algorithm, the bitrate must also be included. The output bitrate of video coders is expressed in bits per second (bits/s). Since the bitrate is directly proportional to the number of frames per second, the frame rate should also be mentioned during the evaluation process.

1.4 Original Achievements

A number of publications and patents have been produced as a result of the research that is described here. These papers and patents are listed in Appendix A. In this thesis, work that is believed to be original can be summarised as:

- Video segmentation: A video segmentation scheme is proposed in our research. We formulate the video segmentation as two sub-problems: semi-automatic video object extraction from the first video frame, and automatic video object extraction from the video sequence based on the available object model. A complexity-scalable contour-tracking algorithm is proposed, which makes the segmentation robust to large motion pattern and partial occlusion.

- Face detection and modeling: Face detection and facial feature extraction (including eye, mouth and chin) are extensively investigated. An automatic face detection algorithm is proposed to robustly extract the facial features automatically. After that, a heuristic scalable 2D face model scheme is developed to construct the model using the facial features and facial muscle distribution.

- Scalable shape coding: Both intra- and inter- shape-coding schemes are extensively investigated. A scalable approximation scheme is proposed to present the object shape, in which curvature scale space (CSS) image is used to get the contour salient features. A novel scalable intra-shape coding scheme is developed in which the information from the coarser encoded layers is employed to improve the coding efficiency of the current layer. A predictive scalable shape-coding scheme is also proposed to improve the coding efficiency further due to the use of temporal information. In the predictive shape-coding scheme, contour motion is estimated through CSS image matching. Experimental results demonstrate that the proposed shape coding algorithms can achieve better R-D performance.

- Wavelet-based object texture intra-coding: An improved shape-adaptive SPECK algorithm is proposed to improve the coding efficiency of the original shape-adaptive SPECK algorithm [LU-2001], in which context-adaptive binary arithmetic codec (CABAC) is incorporated. Experimental results show the efficiency of the improved algorithm. It is about 0.1-0.4 dB improvements when compared with the original shape-adaptive SPECK algorithm. The proposed algorithm is used to code "I-frame" and residual frames after motion compensation.

- A highly scalable 2D model-based texture-coding scheme is proposed. In this scheme, lifting-based temporal filtering is conducted for video objects. During temporal filtering, warping motion compensation is used, instead of blocking-based motion compensation, to reduce the blocking artefact. After temporal filtering, the object texture and residual frames are encoded using the improved shape-adaptive SPECK algorithm. A scalable MV encoding scheme is proposed and rate-distortion optimised bit truncation scheme is employed to achieve bit allocation among the frames within group-of-picture (GOP). The proposed scheme can achieve high coding efficiency and exact bit rate control. It can also achieve temporal, spatial, quality and object scalability simultaneously.

1.5 Structure of Thesis

This first chapter aims to introduce the background and reasons behind the work. It also describes some of the methodology used to evaluate the compression performance and outlines some of the original achievements of the work featured in this thesis. The final chapter summarises the research work that has been performed. It also examines the potential for future research in this area. A list of publications and patents associated with the author is given in the Appendix A. The other chapters are summarised below.

1.5.1 Chapter 2

Chapter 2 acts as background for 2D scalable model-based video coding. Main scalable and model-based video coding techniques are reviewed. First, main video modelling techniques are discussed, which includes camera, illustration, video object and video scene. Then, both 2D and 3D model-based video coding techniques are reviewed and discussed. After that, main scalable video coding techniques are reviewed. They are quality scalability, spatial and temporal scalability, fine-granularity scalability, object-based scalability and wavelet-based scalable coding.

1.5.2 Chapter 3

The third chapter introduces the video segmentation techniques, which is one of the most important steps for scalable 2D model-based video coding scheme. After reviewing the video segmentation techniques, a novel and original video segmentation technique is proposed. The proposed algorithm includes: semi-automatic video object extraction from the first video frame; and automatic video object extraction from the video sequence based on contour-tracking algorithm. A complexity-scalable contour-tracking algorithm is proposed, which makes the segmentation robust to large motion pattern and partial occlusion. The proposed scheme is evaluated through extensive experiments.

1.5.3 Chapter 4

Chapter 4 mainly discusses the face detection techniques and scalable face modelling. The objective of face detection and modelling is to represent the face motion precisely and reduce the rendering error by using *a prior* knowledge in human face. After intensive review of state-of-the-art face detection techniques, an automatic face detection scheme is proposed. The face detection scheme can localise face, face feature and human chin contour automatically and precisely. After detecting face features, a heuristic 2-D scalable face model is designed based on the detected face features and face muscular distribution. Experimental results show that the introduction of scalable face model can improve the accuracy for model-based motion estimation and compensation.

1.5.4 Chapter 5

In chapter 5, scalable object modelling and model compression techniques are reviewed and discussed. The scalable object model is divided into two parts: scalable shape contour and scalable model for object interior. In this chapter, a scalable shape representation algorithm is proposed in which curvature scale space (CSS) image is used to extract the salient feature of contour. Both intra and inter shape coding techniques are investigated and a number of innovative shape coding algorithms are proposed to improve the shape coding performance.

1.5.5 Chapter 6

Chapter 6 mainly presents the algorithms for wavelet-based texture intra-coding of video objects. After introducing the principle and structure for subband/wavelet analysis and shape-adaptive wavelet transform, the author reviewed the main wavelet-based texture coding algorithms, such as SPIHT, SPECK and EBCOT algorithms. The extensions of these algorithms to object-based coding are then discussed. An improved object-based SPECK algorithm is proposed by incorporating Content-adaptive binary arithmetic codec (CABAC) to improve the texture coding performance of video objects. This algorithm is employed in Chapter 7 to encode the texture of "I-frame" and residual frames of video objects.

1.5.6 Chapter 7

In Chapter 7, scalable 2D model-based texture coding scheme is proposed and discussed. The detailed description of this scheme is discussed after reviewing the state-of-the-art highly scalable coding schemes. In this proposed scheme, temporal filtering is first conducted by using motion compensation (MC) and wavelet lifting scheme. Then, warping motion compensation scheme is discussed and employed during temporal filtering, and a scalable motion vector compression scheme is proposed. Rate-distortion optimised bitstream truncation is discussed to achieve arbitrary bitrate coding. Extensive experiments and results are also presented in this chapter.

1.5.7 Chapter 8

The eighth chapter contains the overall conclusions for the thesis, and goes on to make recommendations for future work.

Chapter 2

Model-based and Scalable Video Coding: Overview

2.1 Video modelling

Before digital video processing and coding, we should understand the content of video sequence and describe them in terms of object motion and other effects, such as illumination changes and camera motion. In order to relate changes in the real world to changes in the video sequence, we need parametric models that describe the real world and image generation process, and the parameters can be estimated from the video sequences. The most important models to describe the real world are scene, object, camera and illumination models. Depending on the selected models, the real world can be described with more or less detail and precision. Using image analysis tools, the parameters of the parametric models can be estimated from the video sequence. The real world can be reconstructed and approximated using the parametric models and their estimated parameters.

In the following sections, the camera model, illumination model, object model and scene model are discussed. As the selected parametric object model will decide the coding method and affect the final coding performance and complexity, it will be discussed in detail.

2.1.1 Camera model

The camera model describes the projection of real objects in the real scene onto the image plane of the real camera. Two kinds of camera model are widely used. They are the pinhole camera model and the *CAHV* model (it is defined by C, A, H_0, V_0 vectors as shown in Figure 2.2) [YAKI-1978].

The pinhole camera model has been widely used to approximate the projection of real objects on a real camera target, which is illustrated in Figure 2.1. In this Figure, ***F*** represents the focal length of the camera, and *C* the focal centre. The projected position *x* of a 3-D point *X* is the intersection of the line connecting *X* and *C* with the image plane. Assume that the origin of the 3-D coordinate system is located at the focal centre and its *XY*-plane is parallel to the imaging plane. From the similar triangles illustrated in Figure 2.1 (a), it can be easily concluded that

$$\frac{x}{F}=\frac{X}{Z},\quad \frac{y}{F}=\frac{Y}{Z} \tag{2.1}$$

$$\text{or } x=F\times\frac{X}{Z},\quad y=F\times\frac{Y}{Z} \tag{2.2}$$

This relation is known as perspective projection. A notable character of perspective projection is that the image of an object is smaller if it is further away from the camera. Mathematically, it is described by the inverse between the projected *x*, *y* values and the depth value Z.

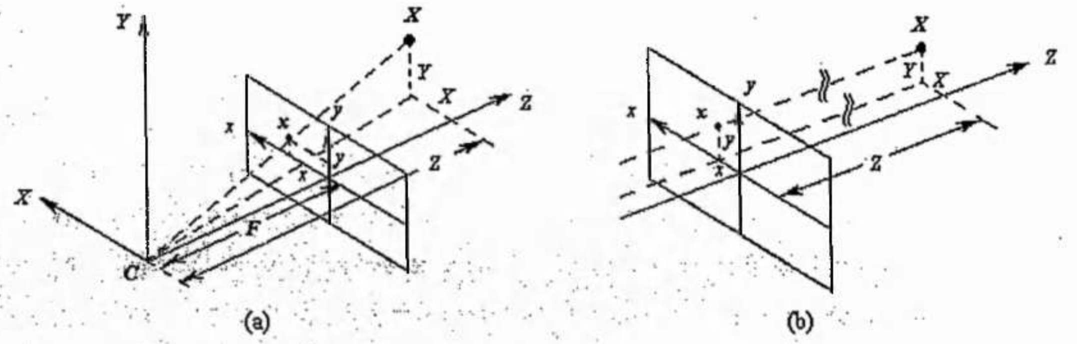


Figure 2.1 – Camera model: (a) using perspective projection in a pinhole camera; (b) using parallel projection as an approximation of a pinhole camera

If the image object is very far from the camera plane, perspective projection can be approximated by orthographic projection, which is also known as parallel projection (Figure 2.1 (b)):

$$x=X,\quad y=Y \tag{2.3}$$

Commonly, as long as the relative depth variation of the object surface is negligible compared to the distance of the object from the camera, this approximation can be used reliably.

The pinhole camera model with its perspective projection is only an approximation of most real cameras. It does not consider the misalignment of the camera axis and the image centre, the low-pass filter effect of the finite size aperture of a real lens, the finite exposure time, and other distortions of the lens.

The *CAHV* camera model can describe a camera such that the camera motion can be accommodated and the camera can be calibrated to compensate for geometrical differences between the pinhole camera model and the real camera. The *CAHV* model describes perspective projection for a pinhole camera model using four vectors:

C: Vector to the camera centre;
A: Unit vector in the direction of the optical axis;
$\boldsymbol{H_0}$: Unit vector in the direction of the horizontal axis of the imaging plane;
$\boldsymbol{V_0}$: Unit vector in the direction of the vertical axis of the imaging plane;

This geometry is depicted in Figure 2.2.

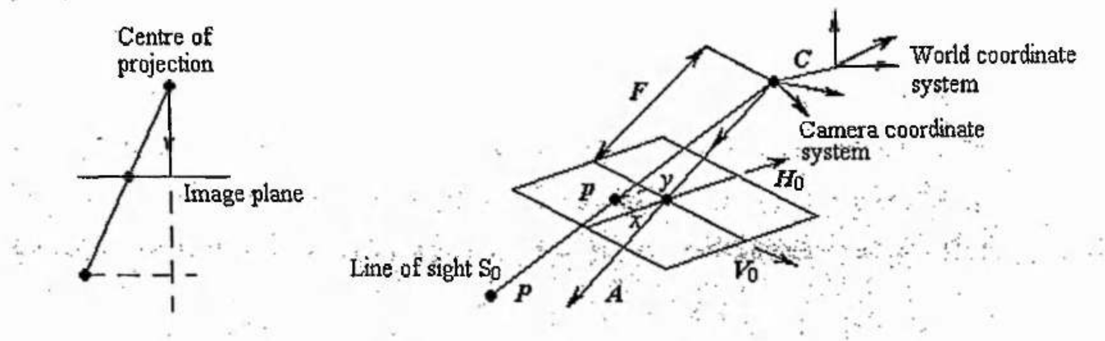


Figure 2.2 – Perspective projection of a point ***P*** in space onto a point ***p*** in the imaging plane using the CAHV camera model [FORSYTH-2003].

Adapting the perspective projection of Equation (2.2) by projecting the vector (***P*** - ***C***) onto the camera axis ***A*** and the imaging plane axis $\boldsymbol{H_0}$ and $\boldsymbol{V_0}$, a point ***P*** is projected onto the image point ***p*** according to [FORSYTH-2003]:

$$\boldsymbol{P} = \begin{pmatrix} x \\ y \end{pmatrix} = \frac{F}{A^T \cdot (P-C)} \begin{pmatrix} H_0^T \cdot (P-C) \\ V_0^T \cdot (P-C) \end{pmatrix} \tag{2.4}$$

The CAVT camera model can characterise a practical camera system by its extrinsic parameters ***C*** and ***A*** and its intrinsic parameters $\boldsymbol{H_0}$, $\boldsymbol{V_0}$, and ***F***. These parameters enable us to describe an imaging plane that is off the camera axis, as well as distortion introduced by the optical system.

Precise knowledge of camera parameters is useful when estimating 3-D shapes and motion from a video sequence. These parameters can be estimated by using geometric camera calibration techniques [FORSYTH-2003].

2.1.2 Illumination model

In order to see an object, we need to illuminate the observed scene. Describing illumination and the reflection of light on object surfaces usually requires complexity models. An illumination model describes how the light incident on an object influences the reflected light distribution.

In video processing, the illumination model is mainly used to describe the temporal changes in the video sequence caused by the changing illumination of the real world. The illumination of a background may change because of an object that moves together with its cast shadow. Since the object surface reflects light, this reflecting source changes the overall illumination of the scene.

When discussing the interaction of a light source with an object surface, there are three types of energy involved. First, incident flux refers to the rate at which energy is emitted from the light source. Second, incident irradiance is the incident flux per unit surface area on the object. Finally, reflected radiance measures the light energy reflected from an object surface. The distribution of the reflected radiance C depends on the distribution of incident irradiance E and the object surface reflectance function r at this point. The relation can be described by [WANG-2002]:

$$C(L,V,N,X,t,\lambda)=r(L,V,N,X,t,\lambda)\cdot E(L,N,X,t,\lambda), \tag{2.5}$$

where X is the location on the object surface, N is the surface normal vector at the location X, L is the illumination direction, V is the viewing direction connecting X to the focal point of the camera, and λ is the wavelength of light [STAU-1993]. The reflectance function r depends on the wavelet length of the incident light, the surface geometry and material properties.

For an ambient source, it radiates the same amount of energy in every direction at any point. Hence, it illuminates objects without casting shadows. When the incident light is such an ambient source and the object surface is diffuse reflecting, the reflected radiance intensity distribution is:

$$C(X,t,\lambda)=r(X,t,\lambda)\cdot E_a(t,\lambda), \tag{2.6}$$

where $E_a(t,\lambda)$ represents the intensity of the ambient light at time t [WANG-2002].

For a point light source, the reflected radiance intensity at X simplifies to [WANG-2002]:

$$C(X,t,\lambda)=r(X,t,\lambda)\cdot E_p(t,\lambda)\max(0,L^T\cdot N) \tag{2.7}$$

Assuming that the scene is illuminated by one stationary, distant point light source and an ambient light, both invariant in time and space, the description of incident irradiance can be represented as:

$$E(N,\lambda)=E_a(\lambda)+E_p(\lambda)\cdot\max(0,L^T\cdot N). \tag{2.8}$$

This is the shading model of Phong used in early computer graphics. The model given by Equation (2.8) has been implemented in an object-based analysis-synthesis coder by Stauder [STAU-1995]. In the 2-D image plane, he assume that the luminance ψ at pixel x and time t is proportional to the reflected radiance at the 3-D point X corresponding to x; that is,

$$\psi(x_{k+1},t+d_t)=k\cdot C(X)=k\cdot r(X)\cdot E(N) \tag{2.9}$$

where k is a constant, and N denotes the normal direction corresponding to X. Thus, the luminance intensity of a point moving from x_k to x_{k+1} (with corresponding surface normals N_k and N_{k+1}) from time t to $t+d_t$, changes according to:

$$\psi(x_{k+1},t+d_t)=\psi(x_k,t)\cdot\frac{E(N_{k+1})}{E(N_k)} \tag{2.10}$$

The simplest and yet most widely used illumination model simply assumes that $E(N)$ is a constant. In this case, the luminance of a moving point does not change, and

$$\psi(x_{k+1},t+d_t)=\psi(x_k,t) \tag{2.11}$$

This is referred to as the constant intensity assumption. This model is widely used for video processing applications, including the video coding standards, such as MPEG-2 and MPEG-4.

2.1.3 Object model

The object model describes assumption about real objects. For video coding, the object can be described by shape, motion and texture models [MUSM-1989]. The texture model describes the surface properties of an object. Commonly, the texture of an object is described by the colour parameters, which contain the luminance and chrominance reflectance. Since we usually assume constant intensity according to Equation (2.11), the colour parameter corresponds to the image signal $\psi(x_k)$. Commonly, different object models, such as 2D object model and 3D object model, use the same texture model, while employing different motion and shape models. In the following sections, different shape models and motion models will be introduced.

2.1.3.1 Shape model

The shape of a 3D object is described by the 3D space that it occupies. Object shapes can be convex, or concave. They can also have holes. It is commonly assumed that an object is topologically connected; that is, that a path can be drawn from any point in the object to any other point without leaving the object. Commonly, the shape of object is described by its surface. Often, a mesh of polygons, referred to as a wireframe, is used.

For a mesh of triangles, it is put up by vertices referred to as control points. In addition to these control points, a list is needed to define which control points define the triangles. The control points of a wireframe are located on the object surface. The number and location of control points are determined by the object shape, as well as the accuracy with which the wireframe model is to describe the object shape. If control points cannot be moved independently, the object is rigid and cannot change its shape. Otherwise, the object is flexible and can change its shape.

When a real object is projected onto the image plane, the shape of a 3D object is projected as a 2D object contour. Similar to 3D object shapes, the number and location of control points along the object contour determine the accuracy of the projected shape. However, a 2D shape cannot be used to decide whether the 3D object is flexible or rigid. When a rigid object is moving, its 2D shape may change.

2.1.3.2 3-D Motion model

The motion of a rigid object can be described in terms of a translation vector $T = (T_x, T_y, T_z)^T$ and a rotation matrix $[R]$. The translation vector T describe a displacement of a point from X to X' by T_x, T_y, T_z in the directions of the coordinate axes X, Y, Z, respectively:

$$X' = X + T. \tag{2.12}$$

If an object rotates around the origin of the 3-D space, we describe the motion of its points with the rotation matrix $[R]$:

$$[R]=[R_z]\cdot[R_y]\cdot[R_x] \tag{2.13}$$

where $[R_x]$, $[R_y]$, and $[R_z]$ are the rotation matrices around the axes X, Y, Z.

These individual rotation matrices are:

$$[R_x]=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\theta_x & -\sin\theta_x\\ 0 & \sin\theta_x & \cos\theta_x\end{bmatrix}, \tag{2.14}$$

$$[R_y]=\begin{bmatrix}\cos\theta_y & 0 & -\sin\theta_y\\ 0 & 1 & 0\\ \sin\theta_y & 0 & \cos\theta_y\end{bmatrix}, \tag{2.15}$$

and

$$[R_z]=\begin{bmatrix}\cos\theta_z & -\sin\theta_z & 0\\ \sin\theta_z & \cos\theta_z & 0\\ 0 & 0 & 1\end{bmatrix} \tag{2.16}$$

If we consider both translation and rotation, the motion of a point on the object surface from X to X' can be expressed as:

$$X'=[R]\cdot X+T \tag{2.17}$$

Equation (2.17) rotates the point X on the object surface around the centre of the world coordinate system. In the case that the object rotates around its own centre only, the object motion can be represented as:

$$X'=[R]\cdot(X-C)+T+C \tag{2.18}$$

where $C=(C_x, C_y, C_z)$ is the coordinate of object centre.

From the above analysis, we can find that the object motion can be described by the parameters $A=(T_x, T_y, T_z, \theta_x, \theta_y, \theta_z)$. For a rigid object, the shape does not change when it is moved with parameter set A.

Not all real objects are rigid. One way to describe a flexible object is by decomposing it into two or more rigid components. Each object has its own set of motion parameters A according to Equation (2.18).

Alternatively, flexible objects can be described by superposing small local motion onto the rigid motion parameters. For example, if we have a sailing ship with a flag, the ship motion can be described using a set of rigid parameters and local motion model is used to describe the waving of its flag in the wind. Since flexible objects change their shape due to local motion, it is not obvious whether this change should be described by shape or motion parameters.

2.1.3.3 2-D motion model

Object or camera motion in 3-D leads to 2-D motion. The 2-D motion model depends not only on the 3-D motion model, but also on the illumination and camera models. The most important 2-D motion model is projective mapping, which is often approximated by affine or bilinear mapping.

As we know, the 3-D positions of any object point before and after a rigid motion are related by

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ Tz \end{bmatrix} \tag{2.19}$$

Substituting Equation (2.2) into Equation (2.12), the relation between coordinates before and after the motion can be described as:

$$\begin{cases} x' = F \cdot \dfrac{(r_1x + r_2y + r_3F) \cdot Z + T_xF}{(r_7x + r_8y + r_9F) \cdot Z + T_zF} \\ y' = F \cdot \dfrac{(r_4x + r_5y + r_6F) \cdot Z + T_yF}{(r_7x + r_8y + r_9F) \cdot Z + T_zF} \end{cases} \tag{2.20}$$

When there is no translational motion in the Z direction, or when the imaged object has a planar surface, it is easy to show that Equation (2.20) can be simplified into the following eight-parameter projective mapping:

$$\begin{cases} x' = \dfrac{a_0 + a_1 x + a_2 y}{1 + c_1 x + c_2 y} \\ y' = \dfrac{b_0 + b_1 x + b_2 y}{1 + c_1 x + c_2 y} \end{cases} \tag{2.21}$$

The projective mapping is an important relation in the study of motion between two video frames, or equivalently the registration of two video frames. It models exactly the relation between images of the same object after camera or object motion, if either of the following is true:

(1) The object has a planar surface;

(2) Neither the object nor the camera undergoes translation along the Z-axis.

In practice, in order to circumvent the problem caused by its rational form, the projective mapping is approximated by polynomial mapping, such as affine mapping, bilinear mapping, biquadratic mapping and pseudo-perspective mapping. They are listed in the Equation (2.22) – (2.25), respectively.

$$\begin{bmatrix} x' - x \\ y' - y \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x + a_2 y \\ b_0 + b_1 x + b_2 y \end{bmatrix} \tag{2.22}$$

$$\begin{bmatrix} x' - x \\ y' - y \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x + a_2 y + a_3 xy \\ b_0 + b_1 x + b_2 y + b_3 xy \end{bmatrix} \tag{2.23}$$

$$\begin{bmatrix} x' - x \\ y' - y \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 y^2 + a_5 xy \\ b_0 + b_1 x + b_2 y + b_3 x^2 + b_4 y^2 + b_5 xy \end{bmatrix} \tag{2.24}$$

$$\begin{bmatrix} x' - x \\ y' - y \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x + a_2 y + a_3 xy + b_3 x^2 \\ b_0 + b_1 x + b_2 y + b_3 xy + a_3 y^2 \end{bmatrix} \tag{2.25}$$

2.1.4 Scene model

The scene model describes how the moving object and the camera of a 3-D scene are positioned with respect to each other. There are three kinds of scene model, which are named as 2-D scene model, layered 2-D scene model (or named as 2.5-D scene model), and 3-D scene model, with different level of complexity [WANG-2002].

The 3-D scene model allows description of object motion and occlusion, as well as cast shadows. It can realistically describe the real world. If the objects are in motion, we distinguish between four image areas: static background, moving object, uncovered background region, and newly covered background (or occluded regions).

Figure 2.3 shows two image frames with a moving object. Comparing frames k and $k+1$, we can distinguish the changed and unchanged regions. The unchanged regions show the static background in both images. The moving object is part of the changed regions in image k and $k+1$. In frame k, the change region is defined as the area of the moving object, and the background to be covered in frame $k+1$ due to the object motion. In frame $k+1$, the changed region is defined as the area of the moving object and the uncovered background that was not visible in frame k.

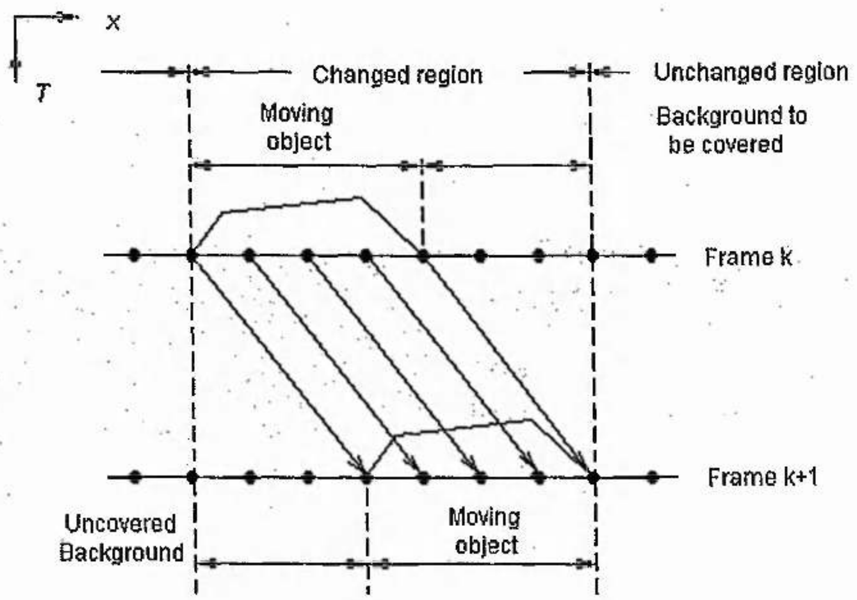


Figure 2.3 – The separation of changed areas into moving objects, uncovered background, and background to be covered [HOTT-1988].

3-D scene model can be simplified as a scene with layered 2-D objects, which is called as 2.5-D scene model. In this model, the camera model uses orthographic projection instead of perspective projection. Therefore, depth has no effect on the image of this scene. This model does not allow the description of effects due to 3-D motion. MPEG-4 standard supports layering of video objects and the layered 2-D scene model.

The simplest 2-D scene model is used in the video coding standards, such as H. 263, MPEG-2, and H. 264. It assumes that all objects are flat and lay on the same image plane and 2-D objects are limited to motion in a 2-D plane. This scene model is useful for video processing.

2.2 Model-based video coding

In past decades, much research has been conducted into the use of object models in video coding and many model-based video coding schemes have been proposed. Different from various conventional waveform coding methods, in which 2-D waveforms of image signals are efficiently compressed, model-based video coding represents image signals using structural object models which in some sense take into account the 3-D properties of the scene. Based on the applied object model in the published schemes, three different scenarios for model-based video coding can be classified: 3D model-based methods [DIEH-1991] [KOCH-1993] [AIZA-1989], 2D model-based methods [NAKAYA-1994] [GERKIN-1994] [WANG-1994], and hybrid 2D/3D model-assisted methods [EISERT-2000]. Various published papers, such as [MUSM-1989], [AIZA-1995], and [PEAR-95], give a detailed review of model-based video coding approaches.

2.2.1 3D model-based video coding

3D model-based coding is a rather specific approach to model-based coding which utilizes a 3-D model of a person's face. In these approaches, 3-D structural models of scenes are adopted. There are two kinds of major approaches to 3D model-based schemes: Approach I makes use of surfaces of the object modelled by general geometric models such as planes or smooth surface [DIEH-1991] [HOTT-1989] [MUSM-1989]; Approach II utilises a parameterised model of the object, such as parameterised facial models [AIZA-1989] [LI-1993].

In Approach I, information such as surface structure and motion information are estimated from image sequences and utilised in video coding. These approaches have been applied, together with motion compensation and interpolation, to improve the performance of the first-generation video coding method, which includes predictive coding, transform coding, vector quantization, etc.

In Approach II, parameterised models are usually given in advance. In these schemes, video sequences are analyzed to estimate the parameters of these models. These parameters are compressed and are sent to the decoder, which uses these parameters to render the video sequence. One of the problems of this approach is that automatic modeling and analysis is not an easy task. Up to now, most of the contributions to 3-D model-based coding have focused on head-shoulder sequences and the parameterized face model. In these methods, automatic facial motion analysis has been done under restricted conditions (e.g. the initial position of the face is known). The extracted facial animation parameters (FAP's) are compressed. Choi et al [CHOI-1994] and

Li et al [LI-1993] reported a direct estimation of head and facial movements, which do not require feature point correspondence. G. Bozdagi, et al [BOZD-1994] described a novel formulation where 3-D global and local motion estimation and the adaptation of a generic wireframe model to a particular speaker are considered simultaneously within an optical flow based framework including the photometric effects of the motion. For all of these methods, heavy computational complexity is required.

Compared with 2D model-based coding approaches, 3D model based video coding schemes have several disadvantages:

- Obtaining detailed object models from a general scene is extremely difficult;
- The adaptation of face models to a particular human face in the sequence is very complicated and time-consuming;
- The analysis process in the video encoder is too complex to get the data required to synthesize the image in the decoder. This is not suitable for handheld mobile devices;
- 3-D object model is very sensitive to channel errors during the transmission.

2.2.2 2D model-based video coding

Currently, 3-D model-based video coding is too rigidly object-specific because the extraction of 3-D structure from single objects in an unrestricted environment and the efficient modelling of their surfaces is an extremely difficult task. Modelling objects is the most important issue in model-based video coding as the complexity of analysis and synthesis depends on the adopted model. Up to now, most of the contributions to 3-D model-based video coding have focused on human facial images, and the parameterised facial models are utilised in advance.

In order to cope with the generality problem of 3D model-based video coding approaches, many 2D model-based video coding methods have been introduced [HOTT-1990] [NAKAYA-1994] [SALE-1995] [ERYU-1995] [CZEREPINSKI-1997]. The 2-D model based coding schemes are rather general approaches. These coding methods exploit visibly important 2-D features, such as edges, contours and regions. In the case of video sequences, moving regions that are detected as changing areas between two successive frames are modelled and coded as arbitrarily shaped 2-D objects [HOTT-1990].

In 2D model-based video coding, both rigid and flexible regions are used for modelling 2-D moving areas. The motion models are not limited to the simple translation model. They exploit affine transform and bilinear transform in order to better approximate the fields of a 3-D moving rigid object and linear transformation such as rotation and zooming. Deformable triangular segmentation of the image and affine transform based motion model are also employed [NAKAYA-1994]. The 2-D deformable triangle-based method divides images into triangle blocks and makes use of an affine motion model. The affine motion parameters of each triangle are uniquely determined by the displacement of its grid points so that no additional information needs to be transmitted except displacement vector of triangle nodes. Motion compensation is performed as follows:

1. Covering the current frame by triangular patches;

   The advantages of the triangle-based motion compensation are: (1) it can deal with linear deformation such as zooming and rotation; (2) it well approximates the motion field of a 3-D moving object; (3) it effectively avoids block artifacts. However, the triangle-based warping motion compensation techniques may suffer from strong inhomogeneous motion, e.g. very fast moving objects, leading to "warping artefact" as reported in [OHM-1996]. Overlapped block motion compensation (OBMC) can be applied to deal with such errors [HEIS-2001].

2. Estimating the motion of the grid points;

3. Synthesizing the prediction image by mapping (warping) the texture of the previous frame onto the corresponding patches of the current frame.

A. M. Tekalp, et.al [TEKALP-1997] compared 2-D and 3-D model-based video coding methods in terms of their capabilities and performance (peak signal-to-noise ratio and visual image quality) for very low bit-rate video coding. The results show that 2-D model-based video coding with affine/perspective transformation and triangular mesh models can simulate almost all capabilities of 3-D model-based approaches using wireframe models at a fraction of the computational cost.

2.2.3 Hybrid model-assisted video coding

The combination of traditional hybrid video coding methods with model-based coding has been proposed by Chowdhury et al. in 1994 [CHOWD-1994], which is called as *hybrid model-assisted* *video coding*. A *switched model-based coder* is introduced that decides between encoded output frames from an H.261 coder and a 3D model-based coder. The coding mode decision is done for a complete frame and therefore the information from the 3D model cannot be exploited if only parts of the frame cannot be described by the model-based coder. An extension to the switched model-based coder is the *layered coder* proposed by Musmann in 1995 [MUSM-1995]. The layered coder chooses the output from up to five different coders, and the coding mode decision is done framewise or objectwise.

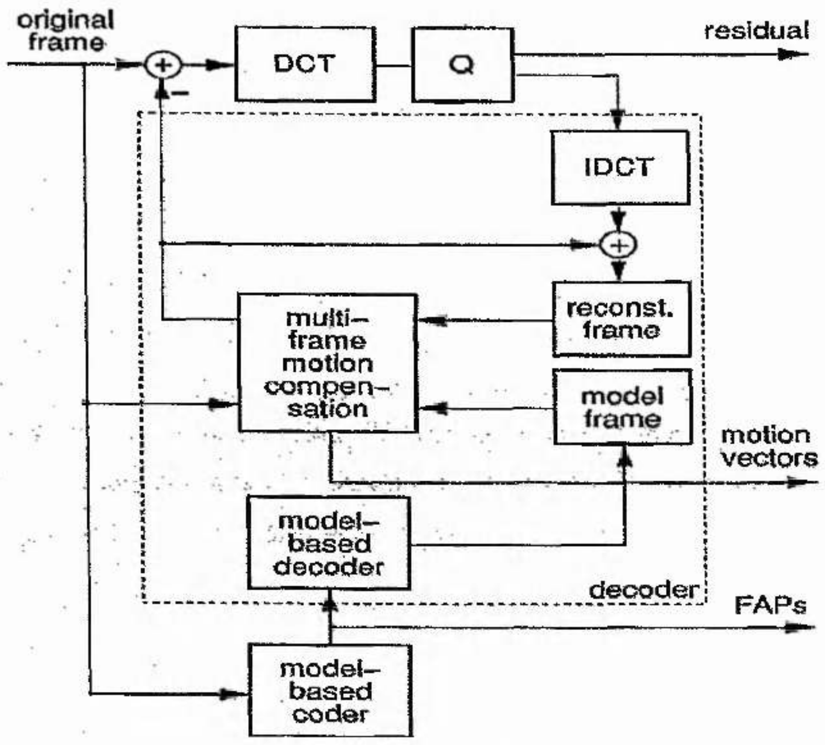


Figure 2.4 – Structure of the hybrid Model-aided video coder [EISERT-2000].

In 2000, Eisert P, et al. [EISERT-2000] proposed a *hybrid model-aided video coding*, which is an extension of an H.263 video codec that incorporation information from a model-based coder in a novel way. Instead of exclusively predicting the current frame of the video sequence from the previous decoded frame, motion compensated prediction using the synthesized output frame of the model-based coder is also considered. In this approach, both traditional waveform coding and 3D model-based coding are combined such that the generality of waveform coding and the efficiency of 3D model-based coding are available where needed. The model-based coder uses a parameterised 3D head model, which consists of shape, texture and the description of facial expressions. Motion and deformation of the 3D head model constitute facial expressions which are represented by facial animation parameters (FAP's) based on the MPEG-4 standard [MPEG-

1999]. For synthesis of facial expression, the transmitted FAP's are used to deform the 3-D head model. Since only a few parameters are encoded and transmitted at very low bit rates, typically less than 1 kbits/s, are obtained if the 3-D models can describe the current video frame. Experimental results proved that the model-assisted video coding scheme could achieve bit rates as low as 5 kbps with good video coding performance.

2.3 Scalable video coding

The aim of model-based video coding is to optimise the coding efficiencies for a fixed bit rate. This presents a difficulty when multiple users try to access the same video through different communication links. If the video stream is scalable, the user with high-bandwidth connections can download the entire bit stream to view the full-quality video, while the users with low-bandwidth will download part of the bitstream and see the low-quality presentation. *Scalability refers to the capability of recovering physically meaningful image or video information by decoding only partially compressed bit streams.*

A scalable stream can also offer adaptivity to varying channel error characteristics, and computing power at the receiving terminal. For wireless communications, scalability allows the adjustment of the source rate and the application of unequal error protection in response to channel error conditions. For internet transmission, scalability enables variable-bit-rate transmission, selective bit discarding, and the adjustment of the source rate to correspond to different modem rates, and diverse device capability. As we move to the convergence of wireless, Internet, and multimedia, scalability becomes increasingly important for rich media access from anywhere, by anyone, at any time, with any device and in any form. Due to its importance, scalable video coding is being intensively investigated currently [MPEG-2003]. However, the coding efficiency of scalable video coding is still not superior to the nonscalable video coding techniques. Therefore, the design goal in scalable video coding is to minimise the reduction in coding efficiency while realising the requirement for scalability.

The scalabilities include quality or SNR scalability, spatial scalability, temporal scalability, Fine-Granularity scalability and object-based scalability, which will be discussed in the following subsections.

2.3.1 Quality scalability

Quality scalability is defined as the representation of a video sequence with varying accuracies in the colour patterns. This is typically accomplished by quantising the pixel values with increasingly fine quantisation stepsize. This type of scalability is also referred to as SNR scalability.

During encoding for quality scalability, the first layer is obtained by applying a coarse quantiser to the original image (or in transformed domain). The second layer contains the quantized difference between the original image and that reconstructed from the first layer, using a quantiser that is finer than that used to produce the first layer. Similarly, each of the subsequent layers contains the quantised difference between the original image and the one reconstructed from the previous layer, using an increasingly finer quantiser.

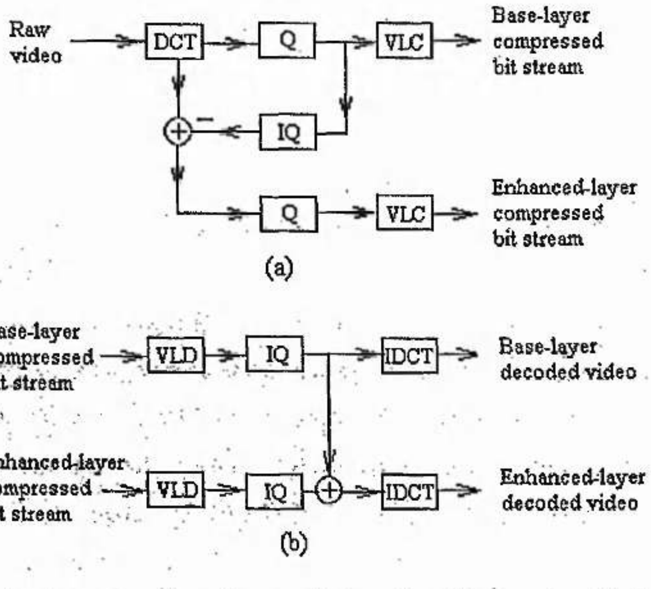

Raw video
DCT
Q
VLC
Base-layer compressed bit stream
IQ
Q
VLC
Enhanced-layer compressed bit stream
(a)
Base-layer compressed bit stream
VLD
IQ
IDCT
Base-layer decoded video
Enhanced-layer compressed bit stream
VLD
IQ
IDCT
Enhanced-layer decoded video
(b)

Figure 2.5 – A two-level quality-scalable codec: (a) encoder, (b) decoder

An encoder and decoder with two-level quality scalability are depicted in Figure 2.5. In the enhanced level, a smaller quantisation parameter is used to achieve better quality than the base level.

2.3.2 Spatial and temporal scalability

Spatial scalability is defined as the representation of the same video in varying spatial resolution. To produce such a layered bit stream with spatial scalability, a multiresolution decomposition of the original image is first obtained. The lowest-resolution image is coded directly to produce the first layer. To produce the second layer, the decoded image from the first layer is first interpolated to the second-lowest resolution and the difference between the original and the interpolated image at that resolution is coded. The bit stream for each of the following resolutions is produced in the same way: first an estimated image at that resolution is formed, based on the previous layers, then the difference between the estimated and the original image at that resolution is coded.

Figure 2.6 shows a block diagram of a two-layer spatially scalable codec. Since the enhanced layer use a smaller quantization parameter, it achieves higher quality than the base layer.

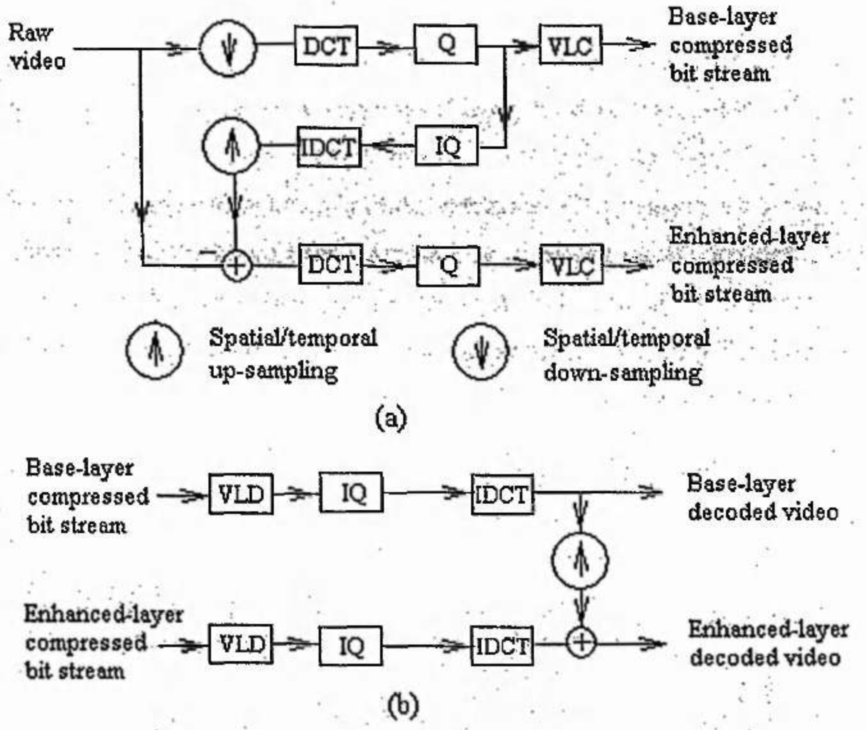

Raw video
DCT
Q
VLC
Base-layer compressed bit stream
IDCT
IQ
DCT
Q
VLC
Enhanced-layer compressed bit stream
Spatial/temporal up-sampling
Spatial/temporal down-sampling
(a)
Base-layer compressed bit stream
VLD
IQ
IDCT
Base-layer decoded video
Enhanced-layer compressed bit stream
VLD
IQ
IDCT
Enhanced-layer decoded video
(b)

Figure 2.6 – A two-level spatial/temporal scalable codec (a) encoder, (b) decoder

Temporal scalability is defined as the representation of the same video in varying temporal resolution or frame rates. Temporal scalability enables different frame rates for different layers of the contents. Typically, temporally scalable video is encoded in a way that makes use of temporally upsampled pictures from a lower layer as a prediction in a higher layer. The codec to achieve temporal scalability has a similar structure as that to achieve spatial scalability, as shown in Figure 2.6. The only difference is that the spatially scalable codec uses spatial down-sampling and spatial up-sampling, whereas the temporally scalable codec use temporal down-sampling and temporal up-sampling.

2.3.3 Fine-Granularity scalability

Fine-granularity scalability (FGS) refers to a coding method by which the rates as well as the quality increase with much smaller steps. In the limiting case, in which a bit stream can provide continuously improved video quality with every additional bit, the underlying coding method is called embedded coding. It is obvious that FGS and embedded coding can adapt to bandwidth variations in real networks more effectively than other scalable methods.

In practice, the requirement that the bit stream has fine granularity is often relaxed. A base layer may be first produced to provide a low but guaranteed level of quality, and then an enhancement layer may be generated to provide improvements in fine granularity. This is the method employed in the FGS model of MPEG-4 [LI-2001] [SCHA-2001]. In this case, the conventional block-based hybrid coding method is employed to produce a base-layer stream at a given frame rate, using a relatively large Quantisation Parameter (QP). Then, for every coded frame, the differences between the original DCT coefficient and the quantised coefficients in the base layer are coded in to a fine-granularity stream. This is accomplished by quantising the refinement coefficients using a very small QP and then representing the quantised indices through successive bit plane encoding. Figure 2.7 illustrates the transmission of portions of the enhancement-layer stream for the support of joint quality-temporal scalability [SCHA-2000a].

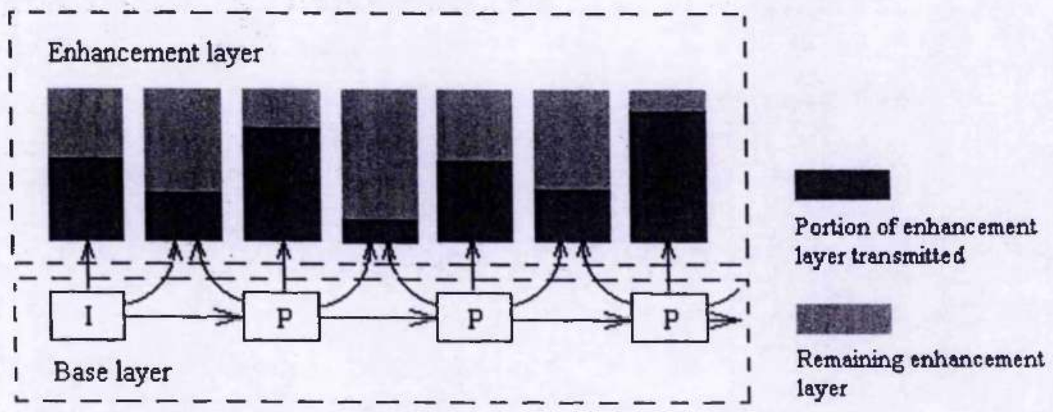

Enhancement layer
I
P
P
P
Base layer
Portion of enhancement layer transmitted
Remaining enhancement layer

Figure 2.7 – Illustration of the transmission of the enhancement-layer stream [SCHA-2000a]

The limitation with FGS scheme is that the base layer must be delivered completely and without error. This may or may not be possible in practical networks. Another problem is that, when the base-layer bit rate is kept low (so as to increase the scalable range of the bit stream), the coding efficiency of the FGS method will be significantly reduced compared to a nonscalable coder. One approach to improving the coding efficiency of the FGS method is to periodically use an intermediate enhancement layer (reconstructed from some but not all bit plane) as a reference for motion-compensated prediction [WU-2001]. Furthermore, it is difficult to use block-based transforms to realize fully embedded coding. Much research has been conducted for wavelet-based coding to achieve this (see section 2.3.5).

2.3.4 Object-based scalability

Object-based scalability is also important for the video format to facilitate content manipulation. In object-based temporal scalability (OTS), the frame rate of a selected object is enhanced such that it has a smoother motion than the remaining area.

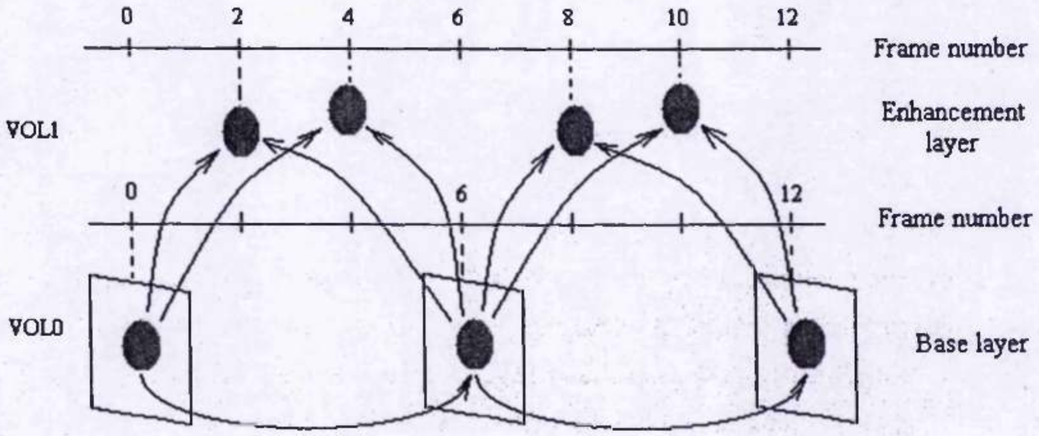


Figure 2.8 – OTS enhancement structure [MPEG4-2001]

Figure 2.8 shows an example of OTS, which uses bidirectional prediction to form bidirectional VOPs (B-VOPs) in the enhancement layer. In this figure, VOL0 (VideoObjectLayer 0) is an entire frame with both an object and a background, whereas VOL1 represents the particular object in VOL0. VOL0 is encoded with a low frame rate, and VOL1 is coded to achieve a higher frame rate than VOL0. In this example, frames 2 and 4 in VOL1 are predicted from frames 0 and 6 in VOL0. Two additional pieces of shape data, a forward shape and a backward shape, are encoded to perform the background composition.

2.3.5 Scalability with wavelet-based coding

The discrete wavelet transform (DWT) has emerged as a powerful tool in image and video compression, due to its flexibility in representing non-stationary image signals and its ability to adapt to human vision characteristics. Wavelet-based image coding techniques have been included in some image and video coding standards, such as JPEG2000 [TAUB-2000] and MPEG-4 [MPEG4-2001].

Like the DCT-based approach, wavelet-transform-based coding for images consists of three steps: (1) wavelet transform; (2) quantisation and (3) entropy coding. Wavelet coefficients after wavelet transform are typically organised into a hierarchical data structure, so that the bit allocation and data compaction can be employed more efficiently. Quantisation allows the data rate to be reduced at the expense of some distortions. Entropy coding encodes the quantised coefficient into a set of compact binary bit streams. When applying wavelets to video coding, mechanisms to reduce temporal redundancy are needed. Motion compensation in both spatial and wavelet domains, 3-D wavelets, and motion compensated 3-D wavelet video coding algorithms have been proposed [WIEN-2004] [WU-2004]. Especially, motion compensated embedded zero block coding (MC-EZBC), which was recently proposed by Chen et al [CHEN-2002], has become prominent because of the excellent performance. In MC-EZBC, each pair of frames is first motion estimated with hierarchical block structure, and then decomposed into a high-band frame and a low-band frame by the motion-aligned lift-based Haar filter. MC-EZBC efficiently solves the problems in the fractional-pel motion aligned temporal transform due to the use of lift-based wavelet transform [LUO-2001]. Promising experimental results have demonstrated that the 3D wavelet-coding scheme can be competitive with the state-of-the-art H.264 video standard on coding efficiency, at the expense of large delay. As the temporal filtering is conducted with one GOP, this restricts its applications, such as real-time video streaming.

However, there are still many problems to be solved before this coding technique can be described as mature. First, the compression efficiency of the 3D wavelet coder is still not satisfactory for video sequences with complex motion patterns. Next, 3D wavelet coder can cause large time delay due to the use of temporal filtering across one Group of Pictures (GOP). This restricts some real-time applications, such as real-time streaming and video conferencing. Furthermore, as more motion vectors are generated during motion compensation (commonly double), they use up a large portion of bits for low-bit rate application. This requires more research to achieve scalable MV coding. Therefore, further research is still needed to achieve all kinds of scalabilities in one codec.

2.4 Conclusions

This chapter gives an overview of video modelling and main techniques of model-based video coding. It also reviews the main scalable video coding techniques.

This chapter begins with a description of video model techniques, such as camera model, illumination model, object model and scene model. This is considered as the theoretical background of the proposed video coding scheme. All of the video coding schemes should be based on a special video model. For example, for video coding techniques adopted in MPEG-2 and MPEG-4, the object models assume object with 2-D motion and the illumination models assume ambient lighting and diffuse reflecting surfaces. These coding standards use 2-D scene model that assumes 2-D objects moving parallel to the imaging plane of the camera. Therefore, the selected video modelling techniques decide the properties of the video coding schemes.

Next, model-based video coding approaches are reviewed, which includes 3D model-based video coding, 2D model-based video coding, and hybrid model-assisted video coding techniques. Both 2D and 3D model-based coding have their advantages and disadvantages. For 3D model-based video coding, current contributions mainly focus on some special video objects only, such as the human face. In hybrid model-assisted video coding scheme, both traditional waveform coding and 3D model-based coding are combined such that the generality of waveform coding and the efficiency of 3D model-based coding are available where needed.

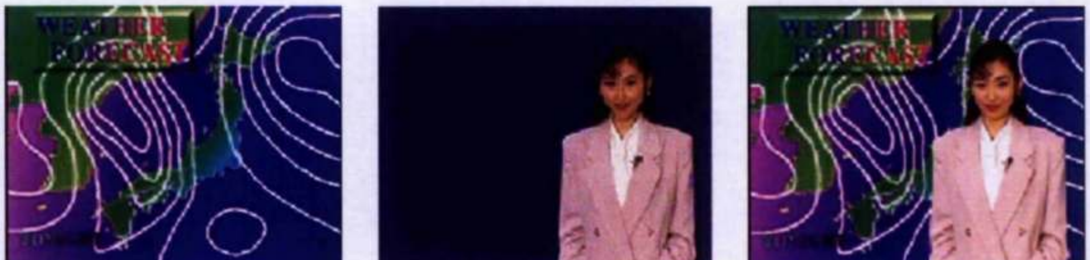
Furthermore, this chapter also gives an overview of scalable video coding. Scalable video coding techniques are very important for rich media access for different users. Four kinds of scalabilities are discussed in detail. Wavelet-based coding techniques can easily be made to produce an embedded stream because the wavelet transform provides a multiscale representation of the signal. However, more research is needed to reach the coding performance of state-of-the-art non-scalable video coding technique at all bit rates.

Chapter 3

Video Analysis and Segmentation

3.1 Introduction

Interactivity with multimedia content is becoming a reality: the user is no longer satisfied being a passive spectator, and wants more of an active role. With the publication of the MPEG-4 [MPEG4-2001] and MPEG-7 standard [MPEG7-2002], the MPEG committee has made a significant contribution towards the development of a new generation of interactive multimedia services. MPEG-4 standard allows the independently encoding of audiovisual objects and MPEG-7 gives the content-based description of audiovisual materials. Furthermore, advanced types of interaction are often based on the understanding of a video scene as a composition of video objects, to which it is possible to associate specific information as well as interactive 'hooks' to deploy the desired application behaviour. To enable such type of interactive services, a representation of the scene semantics that is no longer limited to the frame concept is required. The ability to manipulate such entities in video is a shift in the paradigm from pixel-based to content-based management of visual information.



(a) (b) (c)

Figure 3.1 – Example of the composition of a scene: (a) synthetic object; (b) real object; and (c) composed video scene from (a) and (b)

In the old paradigm, a video sequence is characterised by a set of frames. In the new paradigm, the video sequence is composed of a set of meaningful entities. A wide variety of applications benefit from this shift, which range from video coding to video surveillance, and from virtual reality to video editing – see example in Figure 3.1.

Before introducing the extraction of meaningful entities, the concepts of object and region needed to be introduced. These two concepts provide the additional organisation units to allow semantically structured representations – see example in Figure 3.2.

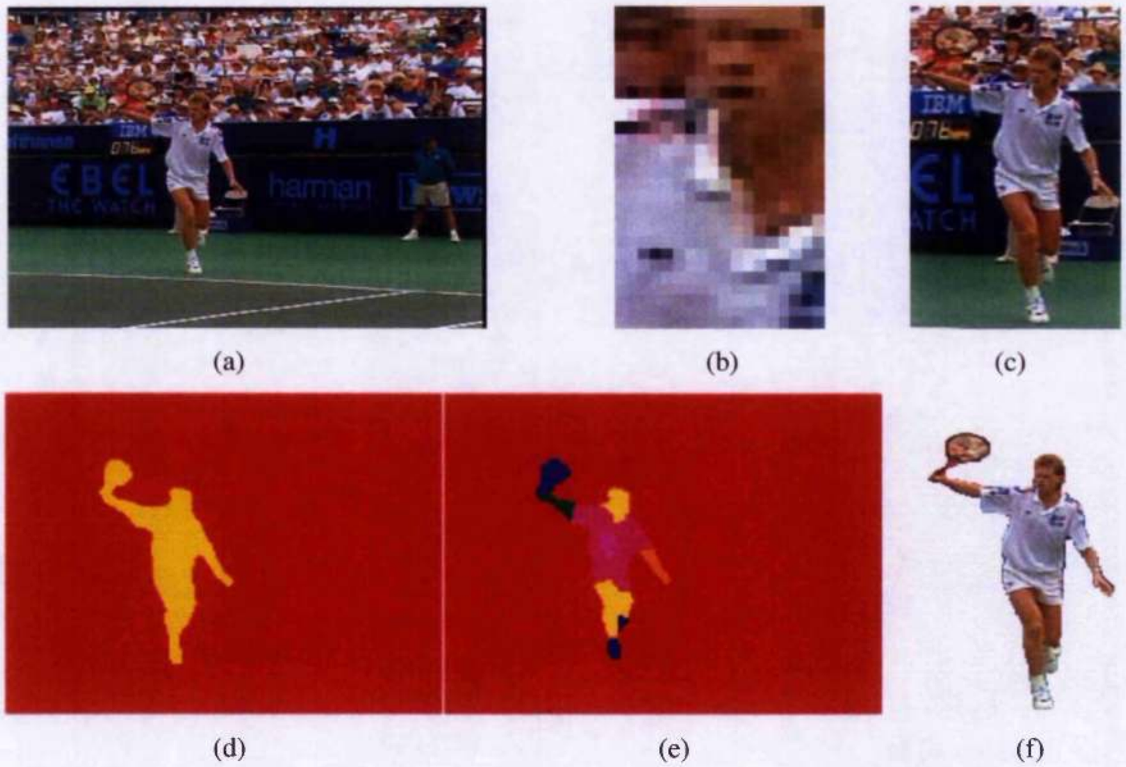


Figure 3.2 – Example of semantically structured representations of a video frame. (a) Original frame; (b) A rectangular set of pixels extracted from (a); (c) A set of pixels showing the relevant video object; (d) A segmentation partition of video object (indicated by yellow colour); (e) The regions of a video object; and (f) The selected video object using the obtained object mask.

In the context of this thesis, a **region** is defined as a set of neighbouring pixels that, at a given time instant, are similar to each other according to some homogeneity criteria that can be objectively measured, and which can be tracked through time during its temporal life span. An **object** is associated with a higher abstract level carrying some semantic value in a given application context, and corresponds to a region or a set of regions. The set of objects and/or regions that completely cover an image at a given time instant, without overlapping, is the temporal instantiation of a **segmentation partition**. Taking a segmentation partition and focusing on a single object by replacing all other labels by a common value allows the **segmentation mask** of that object to be obtained. Usually, **segmentation** denotes the operation aimed at partitioning an image or a video sequence into regions extracted according to a given criterion. In the case of video sequences, this partitioning should also achieve the temporal coherence of the resulting sequence of object masks representing the video object.

The extraction of the meaningful entities is one of the important steps for the success of scalable 2D model-based video coding scheme. This chapter is mainly devoted to the technique of extracting meaningful entities. To that purpose, the main video segmentation techniques are first reviewed and classified in section 3.2. In particular, spatial, temporal and combined spatio-temporal segmentation techniques are discussed. The role of user interaction for segmentation is also examined. After reviewing the published video segmentation algorithms, a complexity-scalable object contour-tracking algorithm is proposed in section 3.3. In the proposed algorithm, semantic objects are first extracted, and object contour is used as the description of a 2D video object. Once the object contour in the first frame is achieved through user interaction and/or auto-segmentation scheme, the contour-tracking algorithm can be applied to segment the whole video sequence. The efficiency of the proposed algorithm is evaluated using many experiments, as shown in section 3.4. Section 3.5 gives the conclusions of this Chapter.

3.2 Video segmentation techniques: A review

Segmentation is one of the most important objectives of a video analysis system targeting object-based coding and description. Unfortunately, a complete theory of video segmentation is not available. Video segmentation techniques are *ad hoc* in their genesis and differ in the way they trade-off one desired property against another.

According to Haralick and Shapiro, image segmentation can be defined as "*a process which typically partitions the spatial domain of an image into mutually exclusive subsets, called regions, each of which is uniform and homogeneous with respect to some property such as tone, hue, contrast or texture and whose property value differs in some significant way from the property value of each neighbouring region*" [HARA-1994].

The extension of this definition to object-based video analysis requires taking into account the temporal dimension. The temporal coherence of the segmentation should be guaranteed. Temporal analysis may be performed by estimating the motion between consecutive frames, thus providing valuable information to merge regions that are not spatially homogeneous but do belong to the same object. Also the tracking of partitions is enabled by the temporal information, ensuring the coherent evolution of the segmented objects with time.

More powerful and complex segmentation criteria may be introduced by using some priori knowledge or by accepting user guidance, which depends on the semantics of the application. This type of information is of major importance in identifying objects semantically relevant to the application. Other generic criteria, such as size, position, or depth order, for instance, may also provide useful information for segmentation purposes.

The remaining part of this section presents an overview of available video segmentation techniques. These techniques can be classified into two groups: automatic video segmentation and semi-automatic video segmentation. The discussion of this section is mostly focused on automatic techniques, as these are the most commonly presented in the literature. Moreover, since the interaction of the user should always be limited to the minimum, automatic segmentation techniques are the aims of all segmentation solutions, even those including user guidance.

Automatic video segmentation techniques are typically grouped into three major categories, depending on the properties looked for to build the image partitions:

- Spatial segmentation – the target regions are homogeneous in terms of their spatial features. Using the luminance and chrominance information, measures such as average and contrast, are computed to find homogeneous regions. These techniques are commonly unable to deal with the temporal homogeneity aspects of the content.

- Temporal segmentation – the target regions are homogeneous in the temporal (motion) dimension. These techniques usually operate on estimated motion vector fields, and are able to produce temporal coherent partitions, but they cannot identify static objects (or parts of objects).

- Combined spatio-temporal segmentation – the target regions are homogeneous both in the spatial and temporal motion dimensions. These techniques allow overcoming many limitations of the spatial and temporal segmentation techniques.

Depending on the specific techniques and related principles used for the segmentation, a further level of classification for the automatic video segmentation techniques is proposed:

- Spatial segmentation – various types of spatial segmentation techniques can be considered, depending on the application addressed. The various techniques may also have different degrees of complexity. The main classes of spatial segmentation techniques are:

- Thresholding – these are simple segmentation techniques that identify regions mainly based on the analysis of image histograms [HARA-1992].
- Texture-based – these techniques are based on the detection of regions with homogeneous textural characteristics. The type of techniques employed is usually effective in detecting highly textured regions [REDN-1984] [OCON-1997] [HILL-2003] [CALLAGHAN-2005].
- Edge-based – these techniques first detect the edges present in the image, and then process the regions to build an image partition [BALL-1982] [JAIN-1989] [PRAT-1991].
- Regions-based – these techniques can be seen as the dual of the edge-based techniques, as they directly detect homogeneous regions in the image, which are separated by a set of edges. The tools employed differ from those used in texture-based techniques, where the separation between regions is not mainly imposed by the presence of edges. Often region merging and splitting techniques are used [HORO-1976] [MEYE-1990] [HARA-1992] [CORT-1995].

- Temporal segmentation – mainly two types of segmentation techniques can be considered, depending on the application targets:
  - Change detection – these techniques target the identification of the areas that change (or not) between successive images, and are not able to identify objects with different motion characteristics [HOTT-1988] [MUSM-1989] [AACH-1993] [MECH-1998].
  - Motion segmentation – these techniques are based on motion homogeneity criteria, such as the velocity field values; in this case, multiple moving objects can be identified even if they have similar texture [BOUT-1993] [WU-1993] [WANG-1994b] [WEIS-1997].
- Combined spatio-temporal segmentation – depending on the way the spatial and temporal information is processed, various types of segmentation techniques are identified:

- Temporal after spatial – a spatial segmentation can be improved by considering also temporal information. For instance, several regions can be merged into the same object if they share common motion characteristics [CHOI-1997].

- Spatial after temporal – these techniques refine the temporal segmentation results using spatial information. For instance, region contours can be corrected, or temporally uniform regions can be split according to their spatial characteristics [MECH-1998] [KIM-1999].

- Temporal and spatial together – these techniques perform the segmentation by simultaneously considering temporal and spatial information [SALE-1994] [MOSC-1998].

The various video segmentation algorithms identified above are discussed in the following subsections.

3.2.1 Spatial segmentation

Spatial segmentation techniques consider each image by itself even if it belongs to a video sequence. A segmentation partition is produced based only on the spatial features of each image and, in particular, motion information is not taken into account. No information from previous frames and subsequent frames, that is, no motion information is used during segmentation. As a consequence, spatial segmentation cannot generate time-coherent partitions.

3.2.1.1 Threshold segmentation

Threshold segmentation techniques are mainly based on the selection of an adequate (set of) separation level(s), i.e., threshold(s), in order to identify areas with different properties in the histogram of some image component, and split the image accordingly. Pixels are allocated to regions depending on the range of values in which they lie, considering the values for the luminance component. This is a very simple approach to the segmentation problem, which can generate accurate results for simple image.

Given an input image I, a simple two-level segmentation algorithm consists in generating an output image O by comparing each luminance pixel with a pre-defined threshold T:

$$O(i,j)=\begin{cases}1, & \text{if} \quad I(i,j)\geq T \\ 0, & \text{if} \quad I(i,j)<T\end{cases} \tag{3.1}$$

This technique, when applied to simple images with a carefully selected threshold, can result in the effective separation of the target objects from a background with a different brightness level.

The selection of the threshold values is not an easy task in order to obtain good segmentation results. It can be done manually or automatically based on global and/or local characteristics of the image. A number of methods for choosing image segmentation thresholds can be found in the literature [HARA-92]. The most popular method is by analyzing the image histogram peaks and valleys.

Segmentation by clustering can also be classified into the Thresholding category, although the threshold doesn't need to be calculated. An image is represented in terms of clusters of pixels that belong together. The specific criterion to be used depends on the application. Pixels may cluster together because they have the same colour; they are nearby; and so on. Simple clustering methods, K-means clustering methods and Graph-theoretic clustering [SHI-2000] [BOYKOV-2001] can be used in this segmentation technique. Based on the properties of the clustering algorithm, the segmentation by clustering can be considered as a general form of threshold segmentation algorithm.

The main advantage of threshold segmentation techniques are their low computational cost, and their effectiveness in segmenting objects that are clearly distinct from each other in some component dimension. The main drawbacks of thresholding techniques are that a large number of (small) regions are generated for textured images, and in some cases, important local spatial relationships are ignored within the image.

3.2.1.2 Texture-based segmentation

Texture-based segmentation techniques have the objective of building a partition in which each region is differently but uniformly textured. Texture is not only related to the way surfaces reflect light, expressed by the luminance and chrominance values, but also to characteristics such as the spatial distribution of tones (or colour) along the image – examples of textures are shown in Figure 3.3.

Texture-based segmentation techniques generally use pattern recognition tools for textural feature extraction and classification. These segmentation techniques can be based on optimization or probabilistic models, such as Markov random fields and Bayesian estimation. Other techniques consider statistical models for regions, built by means of some initialization process and rely on methods like the expectation-maximization algorithm to cluster pixels according to the region models [REDN-1984] [OCON-1997].

The main advantage of texture-based segmentation techniques is their ability to detect homogeneous although highly textured regions. This is not possible to be achieved using other segmentation techniques that do not understand the notion of texture and tend to create over-segmented partitions. However, these techniques usually have a high computational cost and only use spatial information.

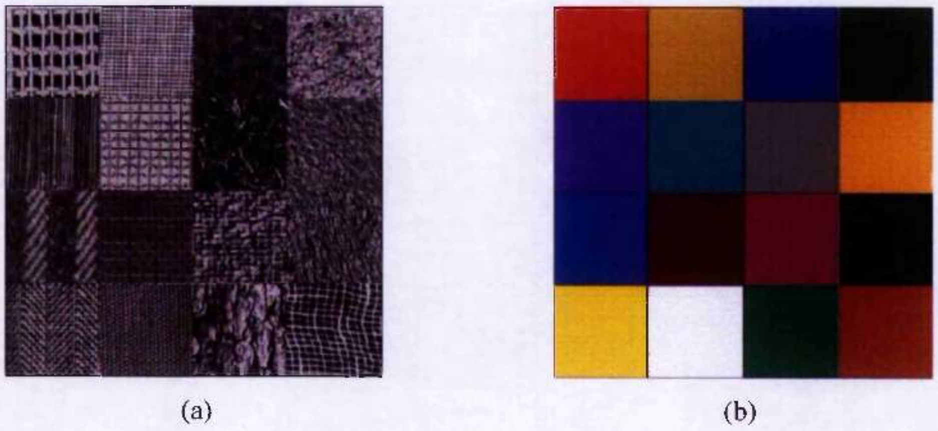


Figure 3.3 – (a) Example image composed by 16 different texture samples, and (b) the corresponding segmentation partition.

3.2.1.3 Edge-based segmentation

Edge-based segmentation relies on boundary or edge detection techniques. These techniques try to find discontinuities in some properties of the pixels, such as grey level, colour, or some local measure, to identify the boundaries between regions. Edge-based segmentation techniques usually perform several major steps:

- Edge detection – this step is typically performed by means of edge detection operators. The outcome of these operators indicates the likelihood of each pixel belonging to an edge, and does not directly correspond to a partition of the image, since multiple edge candidatures may be found close to each other and the resulting edges are usually not connected.

- Edge selection – this step takes the output of the edge detection step and selects the most relevant edge segments. The detected edges are processed and the corresponding pixels are classified as edge or non-edge. A cleaner and sharper edge image results from this procedure. Edge selection can be achieved by applying Thresholding and/or edge relaxation techniques [BALL-1982] [JAIN-1989] [SONK-1993].

- Region boundary identification – this step takes the selected edges and combines them into closed chains, which define the boundaries of the regions in the image. After this step, pixels not separated by an edge are considered as belonging to the same regions. Several techniques have been proposed to achieve region boundary identification [BALL-1982] [JAIN-1989] [SONK-1993].

3.2.1.4 Region-based segmentation

The region-based segmentation techniques partition the image into regions according to some relevant spatial homogeneity criteria. In particular, the regions detected are separated by edge and in this sense, these techniques can be considered as a dual of the edge-based techniques.

Several techniques can be used to perform region-based segmentation. Most of these techniques can be classified into one of the following categories:

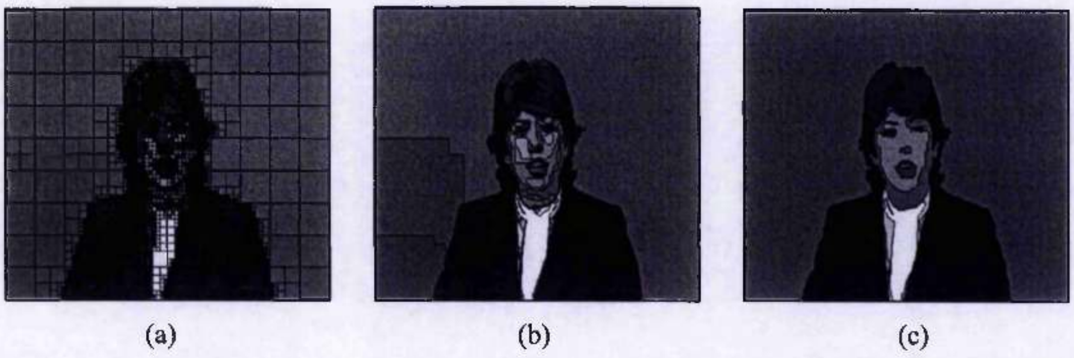


Figure 3.4 – Example of split and merge segmentation: (a) result of splitting step; (b) result after the merging step; and (c) result after eliminating the small-size region to control the number of regions.

- Region Split and merge – this category consists of both splitting and merging of regions, based on their spatial homogeneity. It usually considers a pyramidal image representation, where regions are square shaped and correspond to one of the pyramidal levels. The segmentation starts with an initial division of the image into regions according to the possible pyramid levels. Regions that are not homogeneous are usually split into four subregions according to a quadtree division. The result is a quadtree structure where each leaf node represents a homogeneous region. Due to the quadtree representation usually employed, it is desirable to introduce further processing steps to allow the merging of adjacent homogeneous regions not belonging to the same branch of the segmentation tree. Otherwise, some artificial block boundaries may result. An example of the application of this technique is shown in Figure 3.4.

- Region growing – the algorithms in this category are similar to region merging algorithm, in the sense that neighbouring regions with similar properties are grouped together. The main difference is that the region-growing algorithm does not start with a complete image partition, as in the region merging case, but only with a set of seeds.

  A popular technique for performing region growing is known as the **watershed transforms** [MEYE-1990]. This technique is based on the detection of some image minima, and the 'catchment basins' around them, followed by the execution of a 'flooding' procedure to divide the image into regions. Watershed segmentation is often implemented using mathematical morphological tools. Due to its good segmentation property, watershed algorithm, together with other feature extraction methods, has been widely used image and video segmentation [CALLAGHAN-2005]. Detailed description and examples of watershed segmentation can be found in [MEYE-1990] [VINC-1991] [BEUC-1993].

Often, the region-based segmentation algorithms are preceded by a pre-processing step, where an image simplification is performed using morphological filters. The goal of simplification is to reduce the amount of information to process, while maintaining the relevant boundary information in the image. It also helps to minimize the problem of over-segmentation that often results from region-based techniques.

The main advantages of region-based segmentation are the effectiveness in identifying regions that are homogeneous according to the selected spatial features and accuracy in boundary location. The major drawbacks of these techniques are oversegmentation. One object patch is segmented into several small patches due to the noise or texture. Therefore, image filtering or region merging is required for these techniques.

3.2.2 Temporal segmentation

Temporal segmentation algorithms compute a segmentation partition by evaluating homogeneity in the temporal dimension. To achieve such segmentation, the first step is usually to estimate a motion vector field, from which a partition of the image into coherently moving regions can be estimated.

For motion estimation, a number of different techniques can be used. Most of them are based on the estimation of the apparent 2-D velocity field: the optical flow. However, the estimated optical flow does not always correspond to the true motion field due to:

- The aperture problem; the motion can be estimated uniquely only if the aperture contains at least two different gradient directions. As illustrated in Figure 3.5, to estimate the motion at x_1 using aperture 1, it is impossible to determine whether the motion is upward or perpendicular to the edge, because there is only one spatial gradient direction in this aperture. One the other hand, the motion at x_2 can be determined accurately, because the image has gradients in two different directions in aperture 2.

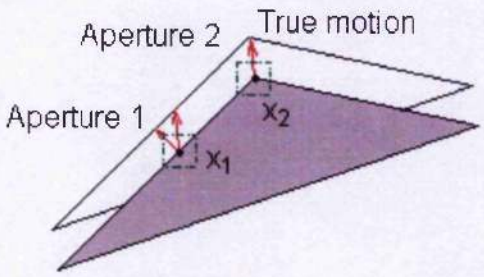


Figure 3.5 – The aperture problem in motion estimation

- The corresponding problem; It is very hard to measure optical flow reliably at featureless pixels because they could hardly correspond to pretty matching everything.

- Image noise and the occlusions between the moving objects;

The two main classes of temporal segmentation techniques are change detection and motion segmentation, which are discussed in the following sub-sections.

3.2.2.1 Change detection segmentation

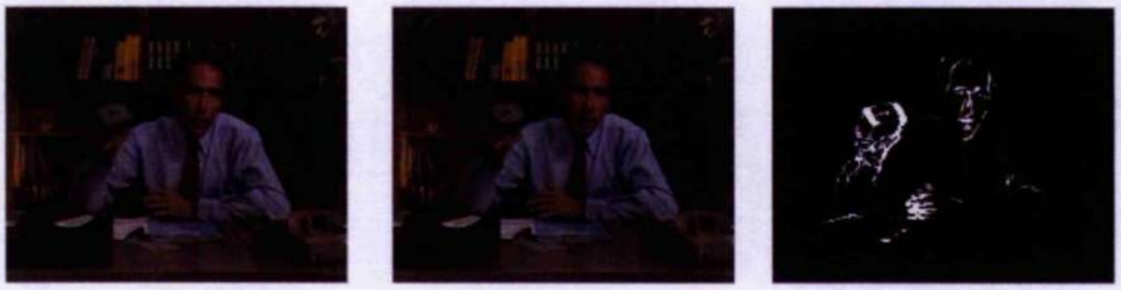
Change detection segmentation is the simplest video segmentation method using temporal information. It can separate the regions that are changing position between successive time instants from those that remain statistic, by comparing the previous with the current image – see example in Figure 3.6.

The elementary steps typically followed to achieve change detection segmentation are:

- Computing the difference between consecutive video frames – the frame difference should be calculated after global motion compensation, to prevent camera motion from influencing the results.

- Thresholding the image difference – in order to obtain a binary image, the image difference must be thresholded. The threshold used can be predetermined. It can also be dynamically computed based on the camera noise variance.

- Post-processing the result – change detection results can be improved by post-processing the output of the thresholding step. For instance, if changes between consecutive images are expected to be small, then parts of the moving objects may not be detected as having changes, resulting in the appearance of undesired holes in the segmentation mask. In this case, when a part of a previously moving object remains static for a small period of time, it can still be considered as part of the detected object by using a segmentation memory [MECH-1998]. Another post-processing is to distinguish between the changed areas that correspond to the moving objects from those corresponding to uncovered static areas. The uncovered static areas are assigned to the detected changed area to improve the moving object segmentation [HOTT-1988] [MUSM-1989] [THOMA-1989].



(a) (b) (c)

Figure 3.6 – Example of initial step of change detection segmentation. (a) and (b): Two video frames; (c): The difference of the luminance part between (a) and (b).

Typically, change detection segmentation techniques use the segmentation result for a given time instant as an initialization for the next time instant segmentation, thus performing a temporal tracking of the detected objects. The limitations of change detection algorithms include:

- First, the motion of uniform objects is very difficult to detect. These areas would become part of the unchanged area, creating holes in the foreground object.

- Next, it cannot be used to segment the still images or objects.
- Furthermore, the global luminance change of background can cause incorrect change detection. Adaptive background models are being investigated to reduce the effects of global luminance change [STAUFFER-1999].

It is worth noting that the MPEG-4 standard includes in its Visual part an informative annex describing a video segmentation algorithm whose temporal analysis is performed with the change detection algorithms presented in [MECH-1998] – see Annex F in part 2 of MPEG-4 standard [MPEG4-2001].

3.2.2.2 Motion-based segmentation

Motion segmentation goes further than simply detecting changing areas between consecutive images, allowing the distinction between differently moving objects. For the example in Figure 3.7, change detection segmentation cannot separate the two video objects. Motion-based segmentation relies on methods that estimate the 2-D velocity field and try to identify regions with homogeneous motion characteristics.

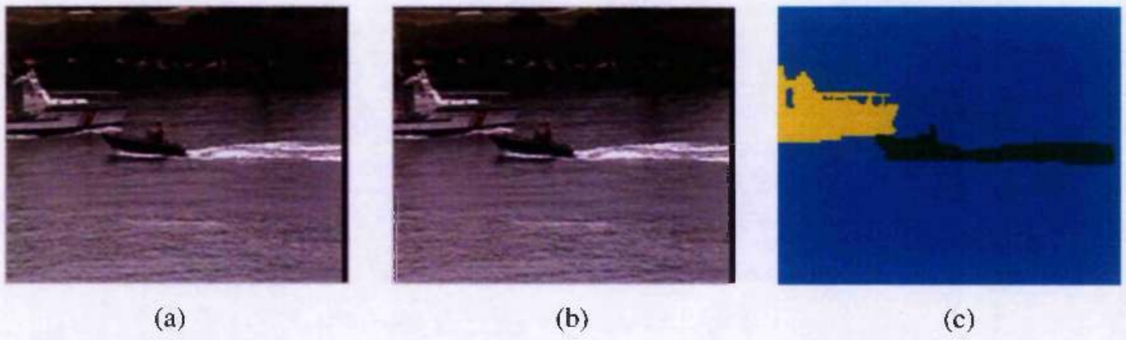


Figure 3.7 – Example of motion-based segmentation partition (c) corresponding to the two frames (a) and (b).

Several techniques have been proposed to perform motion segmentation. The possible classification of these techniques is proposed below:

- Clustering techniques – these techniques try to find clusters of pixels whose estimated motion vectors have similar properties. Clustering can be based on parametric motion models [WANG-1994b], Hough transform [KRUS-1996], Expectation Maximization framework (EM) [BRADY-1996], Maximization of *a posteriori* probability (MAP) [PARAS-2001], and Gaussian mixture models (GMM) [CHAL-1995].

- Hierarchical techniques – these techniques achieve motion segmentation through the successive application of a dominant motion detection algorithm. In each iteration, the largest moving object is identified and it is then removed from the image to be processed in the next iteration, and then the cycle continues until a partition of the image is achieved [WU-1993].

- Markov Random Field (MRF) techniques – these techniques formulate the motion segmentation problem as the probabilistic estimation of a label field, which is modeled by a Markov random field. The global energy function resulting from the MRF modeling can then be minimized using Bayesian techniques [MURR-1987] [BOUT-1993] [ODOB-1996], [GELGON-2000].

Motion-based segmentation techniques identify the presence of a set of moving objects by analyzing the estimated motion field. However, motion estimation in uniform image areas is very difficult due to the lack of texture information to match between consecutive images. Additionally, since motion estimation is sometimes performed using approximate methods, such as block-matching techniques, it is very hard to find the exact position of object contours.

Motion-based segmentation also leads to over-segmented partitions by separating several parts of the same object that exhibit different motion characteristics. Additionally, different objects with similar motion parameters may be merged with each other. To overcome these limitations, the integration with other analysis tools, such as using spatial information or user assistance, should be considered. Furthermore, motion-based segmentation techniques have high computational cost.

3.2.3 Combined spatio-temporal segmentation

For most applications, the usage of both spatial and temporal segmentation techniques can lead to the most reliable results, overcoming the limitations of each of the individual approaches. This combination can be achieved in several ways: temporal after spatial processing, spatial after temporal processing, and simultaneous spatial and temporal processing.

For the temporal after spatial processing techniques, a spatial segmentation, as described in section 3.2.1, is first performed, and afterwards extra segmentation information is added to the spatial-based partition by considering the temporal information. The results of spatial segmentation usually contain too many regions, and temporal segmentation information can be used to group these regions that belong to the same moving object. Additionally, temporal information can be used to maintain the temporal coherence of objects when segmenting the video sequence [CHOI-1997].

For the spatial after temporal processing techniques, temporal segmentation is first performed. Its result is then improved by a spatial segmentation step. The temporal partition improvements include boundary location correction and inclusion of further regions that are not detected by the temporal processing [MECH-1998] [KIM-1999] [WANG-1998].

Simultaneous spatial and temporal processing algorithms are the most powerful approaches. Several algorithms perform video segmentation using this type of technique [SALE-1994] [CHAL-1996] [CHOI-1997] [MOSC-1998]. For example, video segmentation based on contour tracking belongs to simultaneous spatial and temporal processing algorithms [YILMAZ-2004], in which both temporal and spatial information is used to detect and refine the object contour of the current frame.

One of the main advantages of segmentation based on spatio-temporal techniques is its efficiency in identifying regions that are homogeneous in either, or both, spatial and temporal features. Good tracking of objects throughout the sequence and accurate boundary location are also possible. The main drawback is that combining different techniques may result in a high computational cost.

3.2.4 Summary of automatic video segmentation techniques

A summary of the main advantages and disadvantages of the various automatic segmentation techniques is presented in Table 3.1. This table presents the characteristics of each of the main classes of segmentation techniques, and the additional particularities of each of the specific techniques considered. From Table 3.1, we can find that automatic segmentation techniques are most suitable for usage with a given application.

3.2.5 Video segmentation with user interaction

Video segmentation with user interaction, or interactive video segmentation, is nowadays largely recognized as 'an important extra help' to solve the segmentation problem, if the application allows for it. A large number of references about interactive video segmentation techniques have recently appeared in the literature. Some examples are [CHAL-1996] [SMEU-1997] [OCON-1997] [GU-1998] [KWAK-1998] [MARC-1999a] [MARC-1999b] [MARQ-2000].

For video segmentation techniques, two major types of user interaction are considered useful:

- Initial user interaction – used to constrain the analysis process at its start. It can specify the number of relevant objects to be identified by the analysis processing. It also allows the user to select a set of pixels belonging to each of the objects of interest, so that they can constitute the seeds used to constrain the automatic segmentation algorithm [CHAL-1996]. Furthermore, initial user interaction can ask the user to define the position of the contours for the interesting objects by drawing over the original image [GU-1998].

- User refinement – used to refine and correct the automatic analysis results as they are being produced. For example, it can correct a segmentation partition in terms of the number of objects or by refining their boundaries. It can correct the image where a certain object is said to appear for the first time.

Table 3.1 – Summary of advantages and disadvantages for automatic segmentation techniques

Segmentation technique	**Advantages**	**Disadvantages**
SPATIAL-BASED	• Accurate boundaries	• Cannot ensure temporal coherency • Often result in oversegmentation • Unable to merge regions with similar motion characteristics
Thresholding	• Low computational cost • Effective for simple scenes	• Only use global image information ignoring spatial relationships
Texture-based	• Effective for highly textures images	• High computational cost due to the texture feature extraction
Edge-based	• Effective in detecting spatial amplitude variations in images	• Sensitive to noise • Requires complex edge link process to get closed boundaries leading to a partition
Region-based	• Effective in detecting regions with the selected spatial features • More robust to noise than edge-based techniques	• Reasonable to high computational cost
TEMPORAL BASED	• Allows temporal tracking of objects	• Unable to detect static objects or parts of objects • Low textured objects are hard to detect • Boundaries may not be very accurately located
Change detection	• Low computational cost	• Allows moving (and all static) objects are merged together
Motion-based	• Effective in detecting regions homogeneous in motion	• Objects with similar motion may appear merged together • Differently moving parts of an object are identify as different objects • Reasonably high computational cost
SPATIO-TEMPORAL BASED	• Effective in detecting regions homogeneous in temporal and/or spatial characteristics • Good tracking of objects • Precise boundary location	• Reasonably high to high computational cost

3.3 Proposal of contour tracking for video segmentation

This section discusses in detail the proposed video segmentation method, which is based on a complexity-scalable contour tracking approach. This method can be classified into the combined spatial-temporal segmentation category. In past decades, many methods have been proposed that use temporal tracking [WANG-1998] to achieve video segmentation. After being detected and segmented from one video frame, the video objects can be segmented by using a video-tracking algorithm in the subsequent frame, with semantically meaningful object shape. The difference between the proposed methods with other temporal tracking methods is that a complexity-scalable contour-tracking algorithm is proposed and employed in segmentation method, which can achieve more accurate object boundary detection. Furthermore, as the contour-tracking algorithm is complexity-scalable, it can satisfy more application requirements. Before discussing the proposed method, a brief review about video tracking techniques is given as follows.

Video tracking has become an important technique for image and video-based applications, such as video segmentation (e.g. [MPEG4-2001] [MEIER-1998] [WANG-1998]), video surveillance, motion capture (e.g. [MOES-2001]), and gestural human-machine interfaces (e.g. [CROWLEY-2000]). Many video tracking methods have been proposed, which can be roughly divided into region-based tracking [MEIER-1998] [SCLAR-1998] [GOKCE-2000], contour-based tracking [PARAG-2000] [GU-1998], and feature-based tracking [SHI-1994].

The tracking technique in [MEIER-1998] is based on a Hausdorff distance. A binary model for the video object is first obtained from the edge image. The method then matches the model to the objects in subsequent frames. The object model is updated at every frame to follow the change of object shape. This method has the limitation that it cannot deal with complex scenes, and cluttered backgrounds. In [SCLAR-1998], active blobs employ a new region-based approach to nonrigid motion tracking. Shape is defined in terms of a deformable triangular mesh that captures object shape plus a colour texture map that captures object appearance. Nonrigid shape registration and motion tracking are achieved by posing the problem as an energy-based, robust minimization procedure. However, this method cannot cope with the object occlusion, complex motion and deformation.

Instead of tracking pixels of the whole object, contour-based methods track only the contour of the object. First, the object contour of the previous frame is projected onto the current frame using motion information. Then, the predicted shape is adapted to the object in the current frame. The tracking method in [GU-1998] first estimates the parameters of a perspective motion model and then predicts the position of the contour in the next frame based on these parameters. To deal with non-rigid body motion, the method adjusts the approximated boundary by means of a morphological watershed. This method can track an object contour with pixel-wise accuracy. But it cannot handle large non-rigid movements. Active contours (snakes) are efficient methods for tracking both rigid and nonrigid objects [PARAG-2000]. One of the important features of the active contour technique is that it can fuse both edge and texture information to improve the tracking accuracy.

Recently, particle filters have become popular tools in solving the tracking problem [ISARD-1998] [PEREZ-2004], which, for the visual tracking context, are pioneered by Isard and Black [ISARD-1998]. One important advantage of particle filtering is that it allows the information from different measurement sources to be fused in a principle manner. However, within the visual tracking context, efficiently fusing different cues has not been fully exploited to increase the reliability of object tracking algorithms.

During our research, a complexity-scalable object contour tracking method is proposed, which is based on multiple cues (including motion, texture, edge, etc). It can achieve robust object tracking under different conditions. In particular, it can achieve complexity-scalable contour tracking so it can adapt to different applications and complexity requirements. No prior training is required, and a non-parameterized contour model is used. The block diagram for contour tracking algorithm is illustrated in Figure 3.8.

This method can be considered as a hybrid scheme of feature-based, region-based and contour-based techniques. The proposed scalable contour-tracking algorithm consists of three steps, each of which can be exploited for different applications. At first, an object contour is predicted using feature-based and mesh-based object tracking schemes. Then, it is refined using texture information along the contour region, in which a local maximal likelihood detection scheme is conducted. Finally, the active contour model is applied to track object contour with pixel-wise accuracy and alleviates the possible error detection in step 2.

The proposed method features the following novelties:

1. A robust piecewise contour prediction scheme is employed, in which reliability evaluation is first conducted for the estimated motion vectors of mesh vertices and a local motion model is estimated and used to predict the object contour.

2. A local maximal likelihood detection scheme is used to correct the predicted contour, which is efficient for nonrigid object movement.

3. Both local photometric (foreground/background region statistical properties) and geometric (such as edge, contour and region smoothness constraints) information is incorporated into the active contour model to further refine the contour.

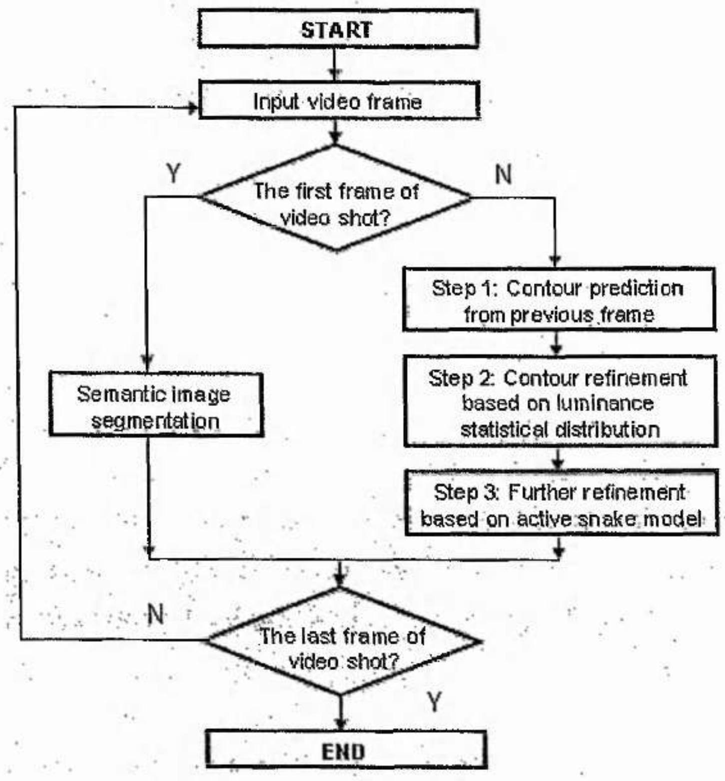

START
Input video frame
The first frame of video shot?
Y
N
Step 1: Contour prediction from previous frame
Step 2: Contour refinement based on luminance statistical distribution
Step 3: Further refinement based on active snake model
Semantic image segmentation
The last frame of video shot?
N
Y
END

Figure 3.8 – Block diagram for contour tracking algorithm within one video shot

3.3.1 Semantic image segmentation

Before carrying out contour tracking to achieve segmentation, we should semantically segment and specify the object to be tracked. Using automatic segmentation techniques, as discussed in Section 3.2, or using user interaction can achieve this. Some advanced interactive image cut algorithms, for example the method in [ROTHER-2004], can also be used to minimize the human interaction and achieve semantic object segmentation.

In order to achieve semantic image segmentation of the first frame, combined spatial-temporal segmentation technique is employed. The proposed algorithm consists of spatial segmentation, and segmentation refinement.

3.3.1.1 Spatial segmentation

First, morphological open/closing by reconstruction filters are used to simplify the first video frame. As we know, morphological open/closing by reconstruction filters have good performance for keeping the edge of the structure while removing small texture pieces whose size is smaller than that of the structure element of morphological filters. In our experiments, the size of structure element n is decided by the testing sequence such as the size of frame and complexity of texture.

After getting the simplified image, its gradient is estimated by the Sobel edge detector. The gradients of two chrominance components are also estimated and used to improve the performance, which is based on the following formula:

$$Grad = \max\{\alpha G_Y, \beta G_U, \gamma G_V\} \tag{3.2}$$

where α, β, and γ are the weighting factors applied.

The watershed transformation algorithm [VINC-1991] is used on the gradient image to get the segmentation. Before watershed transformation, the gradient image is thresholded by a value. That is, small gradient values, which are less than the threshold value, are set to zero, otherwise they remain the same. The threshold value varies from sequence to sequence, which is chosen from 8 to 15 in our experiments. After the watershed transformation, the region-merging algorithm is used to merge the small-sized patches based on intensity homogeneity.

3.3.1.2 Segmentation refinement

In the proposed scheme, motion, colour, and user input are integrated to merge the adjacent 'similar' patches. Block-matching or change detection algorithm is used to derive the motion information. Change detection is more useful for static background/ moving foreground and moving background/static foreground sequences, such as Akiyo sequence. However, block-matching algorithm is more useful for moving background/moving foreground sequences, such as Carphone sequence.

After getting the spatial segmentation results and temporal information, they are fused to get the foreground and background objects. The same fusion algorithm in MPEG-4 is used in our proposed algorithm to get the final segmentation results – see Annex F of part 2 in MPEG-4 standard [MPEG4-2001]. For example, if the change detection algorithm is used during the motion estimation and when most of the spatially segmented region belongs to the changing region in the change detection mask, the whole area of the spatially segmented region is declared as a foreground, otherwise it belongs to background. The parts exceeding region boundaries in the spatially segmented region can be excluded from the foreground region when the portion of the exceeding pixels compared to the neighbouring regions is small. While uncovered parts of the spatially segmented region by the object mask can be all foreground when their portion of areas are small compared to the part of the region covered by the foreground part of the object mask. Sometimes, both foreground and background have similar motion pattern. In this case, user interaction is required in order to achieve semantic segmentation.

Figure 3.9 shows the segmentation results for the Carphone sequence using the proposed method. During the image simplification process, the size of structure element M_n of morphological filters increases from 3x3 to 7x7. The applied weighting factors α, β, and γ are chosen as 1, 0.5, 0.5 respectively during the gradient fusion. These selected parameters can achieve good segmentation performance for all video sequences by fusing the luminance and the chrominance components. The gradient value is thresholded by 10 before watershed transformation.

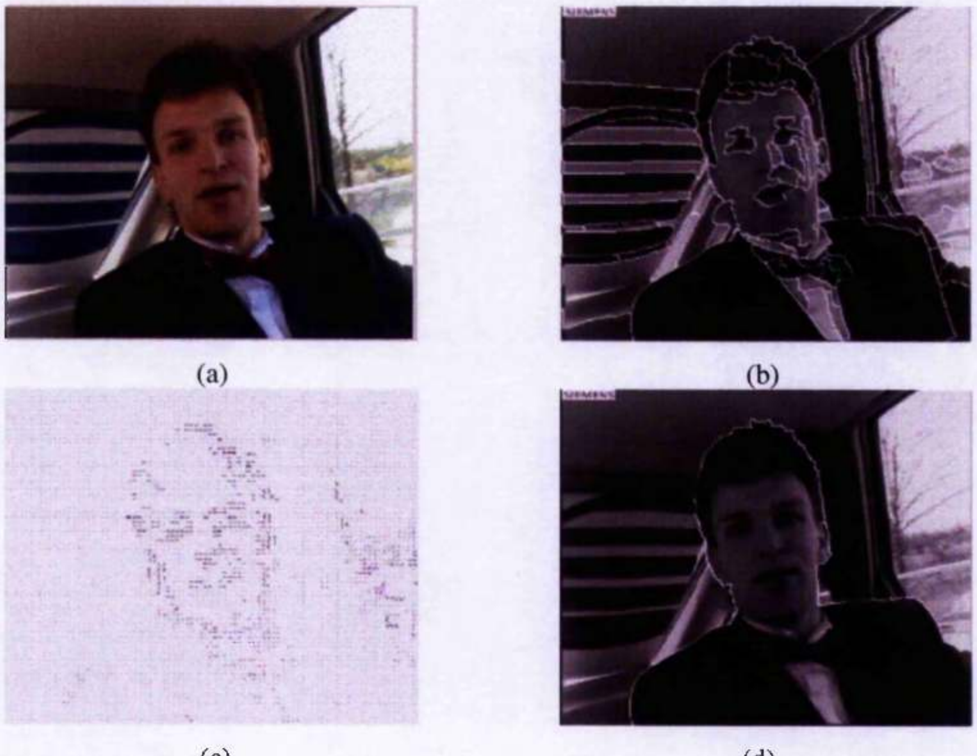


Figure 3.9 – Semantic segmentation of Carphone sequence. (a) Original frame 0. (b) Its spatial segmentation results. (c) Motion vectors estimated by hierarchical block matching algorithm, and (d) Final segmentation results.

Figure 3.10 shows the segmentation results of Akiyo sequence. For this sequence, as the foreground object is static and background objects are moving, the change detection algorithm is used to extract the motion information.

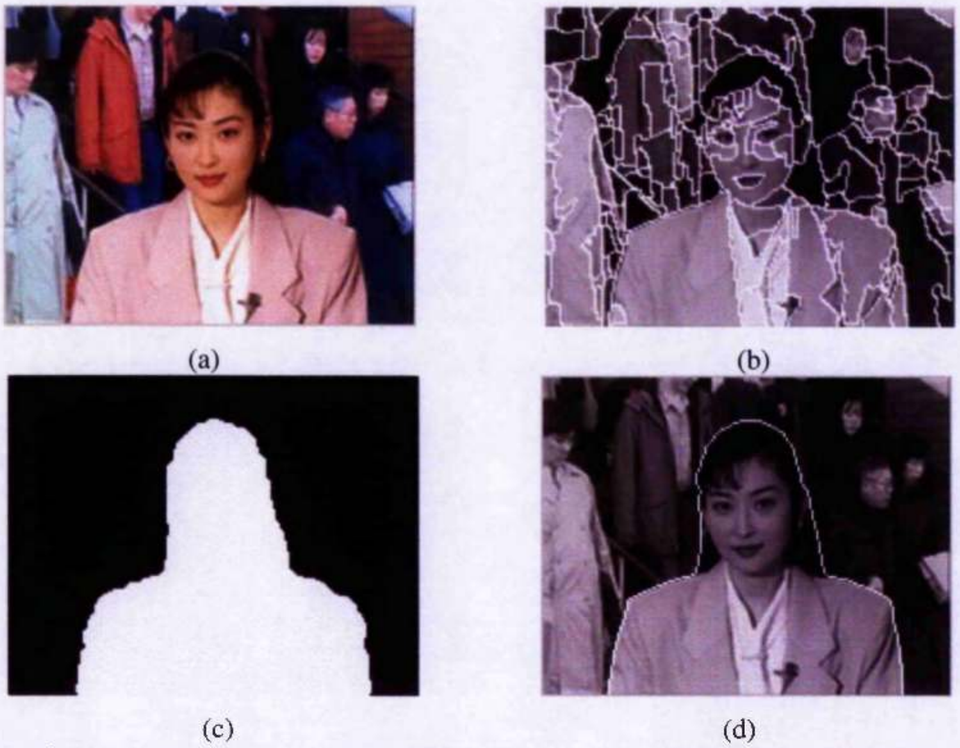

(a) (b) (c) (d)

Figure 3.10 – Semantic segmentation of Akiyo sequence. (a). Original frame 0; (b). Spatial segmentation results; (c) Motion information estimated by Change Detection algorithm and (d) Final segmentation results

Figure 3.11 shows the process for estimating the motion information of the Akiyo sequence in frame 0. (a) and (b) are two video frames; (c) is the difference between these two frames; (d) is the change decision based on statistical significant test; (e) is the foreground mask after Bayesian estimation and relaxation; (f) is the final foreground mask after post-processing, trying to remove the small-size patches.

3.3.2 Complexity-scalable contour tracking for video segmentation

After getting the object with semantic meaning, object contour is used as the descriptor of video object. Then, the subsequent video frames are segmented through contour tracking. In our search, a complexity-scalable contour tracking scheme is proposed, which includes contour prediction using motion, contour refinement using colour information and further refinement through active snake model. The detail descriptions are presented in the following subsections.

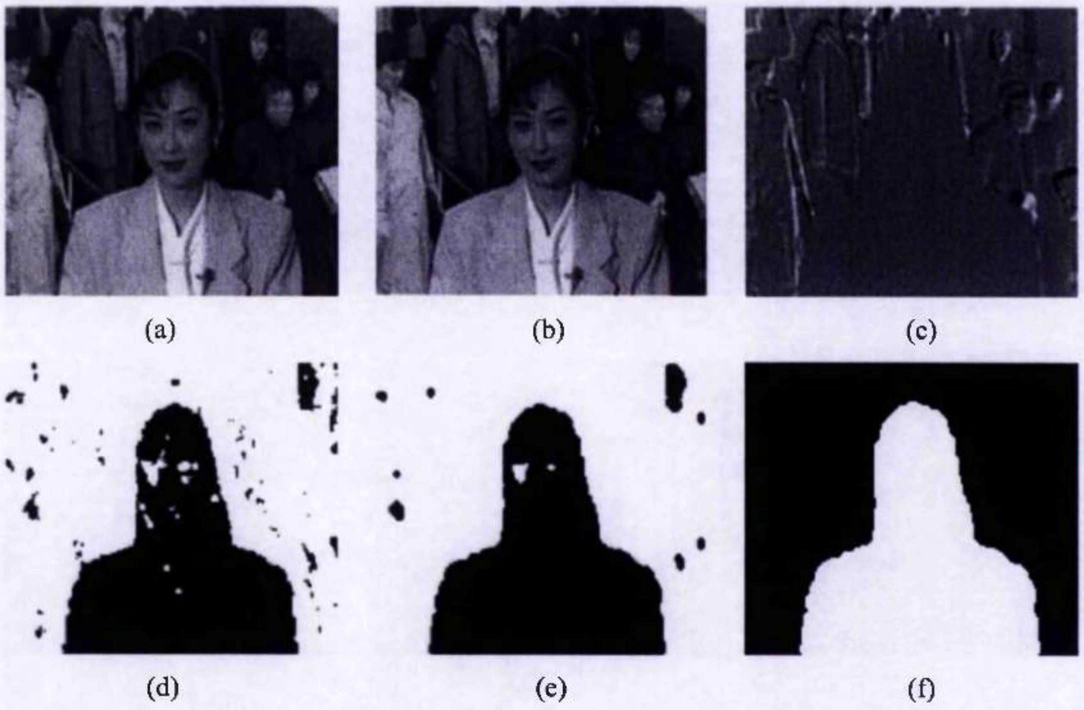


Figure 3.11 – Motion information extraction of Akiyo sequence by using change detection algorithm.

3.3.2.1 Contour prediction

For the object in the previous frame, robust feature points, based on the criteria of [SHI-1994], are selected on the object and the object contour is approximated through a series of contour points. A mesh model is then constructed using the allocated feature points.

After object mesh construction, both forward and backward motion vectors of control points between frame $I(\bar{x},t-1)$ and frame $I(\bar{x},t)$ are estimated using the Shi-Tomasi feature-tracking algorithm [SHI-1994]. That is, the forward motion vector of the i th node location V_i in frame $t-1$, moves to location $V_i^{'}$ in frame t. Then the backward motion vector at the location $V_i^{'}$ in frame t maps back to $V_i^{''}$ in frame $t-1$.

After both forward and backward motion estimation, their motion "reliability" is estimated based on both forward and background motion vectors. The "reliability" is evaluated by the following formula:

$$\mathrm{Re} = \exp\left(-\frac{\left\|V_i - V_i^{"}\right\|^2}{2\sigma_m^2}\right) \tag{3.3}$$

where σ_m is the free parameter.

From Equation (3.3), it shows that the smaller the difference between V_i and $V_i^{"}$, the more reliable the motion vector of i th node. For the nodes whose reliability is smaller than a threshold (0.3 is chosen in our experiments), they are not considered during contour prediction.

Then, a mesh-based motion estimation scheme [GOKCE-2000] is applied to refine the motion vectors of the control points. The positions of the nodes with higher "reliability" are initialised by the feature-based motion estimation results. The initial positions of the nodes with smaller "reliability" are estimated from their neighbours. The mesh-based motion estimation scheme can keep the mesh structure during MV's refinement.

After estimating the motion vectors of interior points, the points along the contour can be predicted from their m nearest motion vectors. The weighted least squares estimation algorithm (WLS) is used to determine the affine motion that best describes the motion of the contour segment. Each motion vector is weighed according to its "reliability".

For a rigid moving object, the object contour is predicted with high accuracy. However, for nonrigid moving object, for example, human head profile contour appearance, further refinement is necessary.

3.3.2.2 Contour refinement

In this step, it is assumed that the statistical characteristic of pixels of frame t is similar to its adjacent neighbors in frame $t-1$. This step is efficient in tracking objects with large nonrigid motion such as the in-plane rotation of human head.

During contour refinement, local maximal likelihood detection is used to refine the predicted contour in frame t. Only the pixels in the band around the object contour are employed to estimate the statistical properties of foreground and background, in which non-parametric kernel density estimation is employed [FORSYTH-2003]. While estimating the local statistical properties of contour points in the previous frame, the selected pixels are constrained by rectangle and the defined band, as shown in Figure 3.12 (a) for point P.

The estimated local statistical properties of foreground and background around point P are used to refine the corresponding contour segments around the predicted contour point P' in frame t, using maximal likelihood (ML) detection and Bayesian relaxation. First, a band is generated around the predicted contour in current frame t. It means that the object contour can only be searched in the band. Then, based on the local statistical probability of both foreground and background of P, the pixels around P', which should also be located in the circle as shown in Figure 3.12 (b), are classified as foreground and background. The radius of circle is decided by the distance between P' and its neighbour contour points.

As the criterion of this step is that the statistical characteristic of the pixels of frame t is similar to its adjacent neighbours in frame $t-1$, it is efficient for tracking the object with large nonrigid motion such as human head in-plane rotation. However, as the contour smoothness constraint is not employed during refinement, some contour parts are zigzag-patterned and visually uncomfortable, especially for the position where statistical probabilities of foreground and background are similar. Therefore, further refinement is necessary to get smooth object contour.

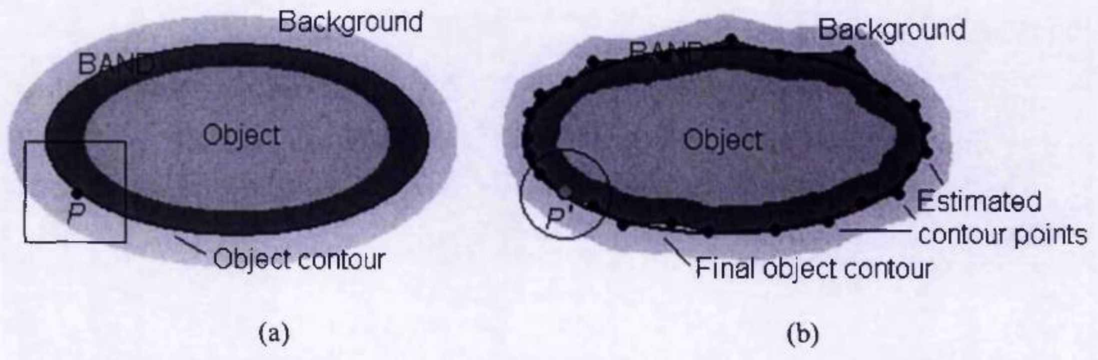


Figure 3.12 – Illustration for contour refinement process; (a) estimating local statistics of foreground and background around P in previous frame; (b) refining contour segment around the estimated P' in current frame.

3.3.2.3 Further refinement using active snake

After contour prediction and refinement, a precise object contour can be achieved if there is high contrast between the tracked object and background along the contour. If the contrast is low, the generated contour is a zigzag and some contour shape errors may occur.

In the third step, the active contour model is used to refine the contour further. Suppose that N

discrete points are selected, which have similar foreground and background surroundings, along the given contour C. The discrete snake energy E_{snake} can be expressed as follows:

$$E_{snake} = \sum_{i=0}^{N-1} E_i = \sum_{i=0}^{N-1} \left(E_{\text{int},i} + E_{ext,i}\right) = \sum_{i=0}^{N-1} \left(E_{\text{int},i} + E_{edge,i} + E_{region,i}\right) \tag{3.4}$$

where each E_i depends on the contour segment between up to three points v_{i-1}, v_i and v_{i+1}. In our research, the dynamic programming algorithm in [AMINI-1990] has been used to search for the maximum of Equation (3.4).

During algorithm implementation, for node i, the search locations are restricted along the bisector lines of the angle $\angle \overline{v_i v_{i-1}}, \overline{v_i v_{i+1}}$. As the points are placed at regular intervals along the contour, the internal energy in node i is selected as:

$$E_{\text{int},i} = 2 - 2\cos\angle \overline{v_i v_{i-1}}, \overline{v_i v_{i+1}} \tag{3.5}$$

An edge map is selected as one part of external energy E_{ext} of Equation (3.4). The Canny edge detector has been used to obtain the edge map. Low threshold has been used during edge detection to detect the weak edge. Short edges are removed in order to reduce the effect of noise. A binary edge map *BinaryEdgeMap* is generated, which is then smoothed by Gaussian filter with variance σ_e ($\sigma_e = 2$ is selected in the experiments). The average gradient along the contour segment is used as external energy E_{edge}.

$$E_{edge,i} = \frac{BinaryEdgMap_{\sigma_e}\left(v_i v_{i-1}\right) + BinaryEdgeMap_{\sigma_e}\left(v_i v_{i+1}\right)}{l\left(v_i v_{i-1}\right) + l\left(v_i v_{i+1}\right)} \tag{3.6}$$

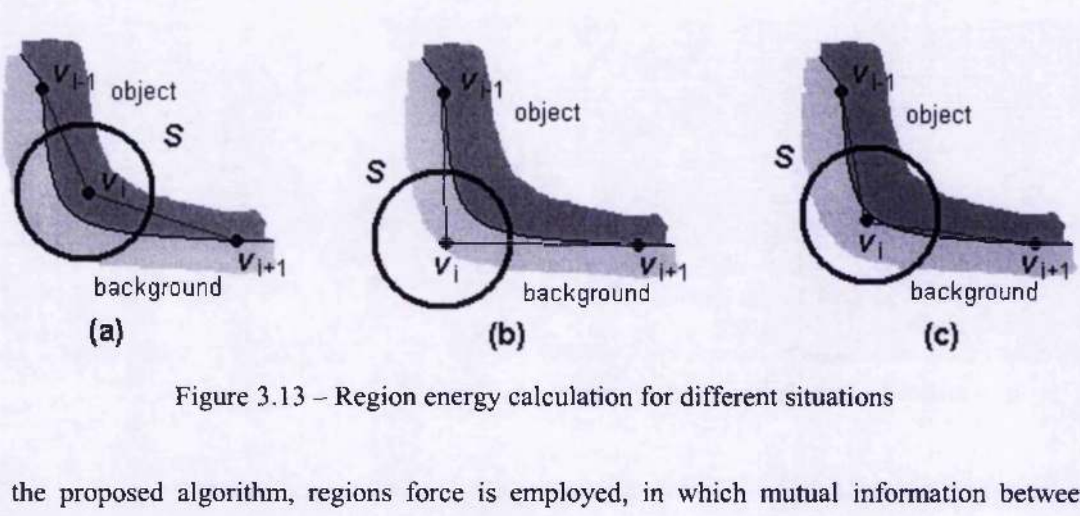


Figure 3.13 – Region energy calculation for different situations

In the proposed algorithm, regions force is employed, in which mutual information between image intensity and its label is maximized [KIM-2002].

$$\begin{aligned} E_{region,i} &= I(I(x); L(x)) \\ &= H(I(x)) - \Pr(L(x)=F)H(I(x)|L(x)=F) - \Pr(L(x)=B)H(I(x)|L(x)=B) \end{aligned} \tag{3.7}$$

where B and F represent the background and foreground respectively within the support S, as shown in Figure 3.13. $E_{region,i}$ will be maximized if and only if the labelling $L(\cdot)$ gives the correct segmentation [KIM-2002], as in Figure 3.13 (c).

$$H(I(x)|L(x)=F) = -\int_S \Pr_z(x)\log \Pr_z(x) dZ \tag{3.8}$$

$\Pr(x|L(x)=F)$ and $\Pr(x|L(x)=B)$ are the density of foreground and background which are estimated using the fast gauss transform algorithm in [CHAN-2001].

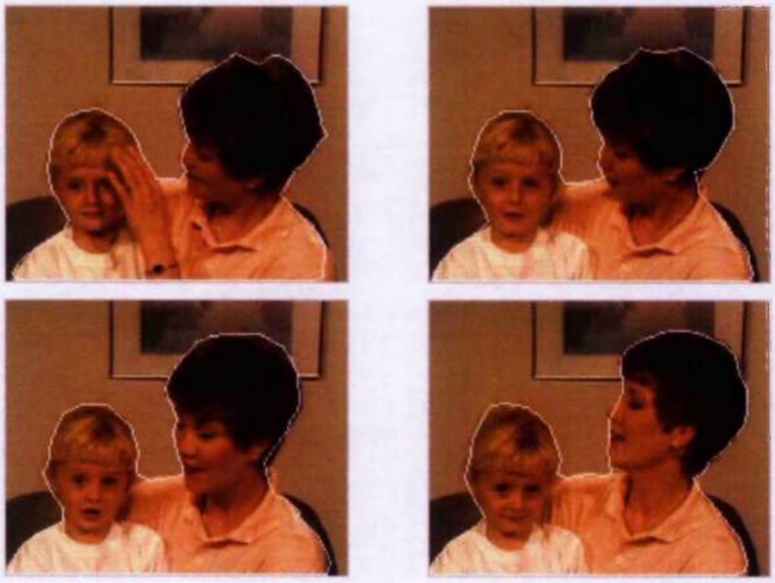
After achieving the object contour of frame t, object mesh structure and contour approximation are updated in order to track object contour along the whole video sequence.

3.4 Experimental results and analysis

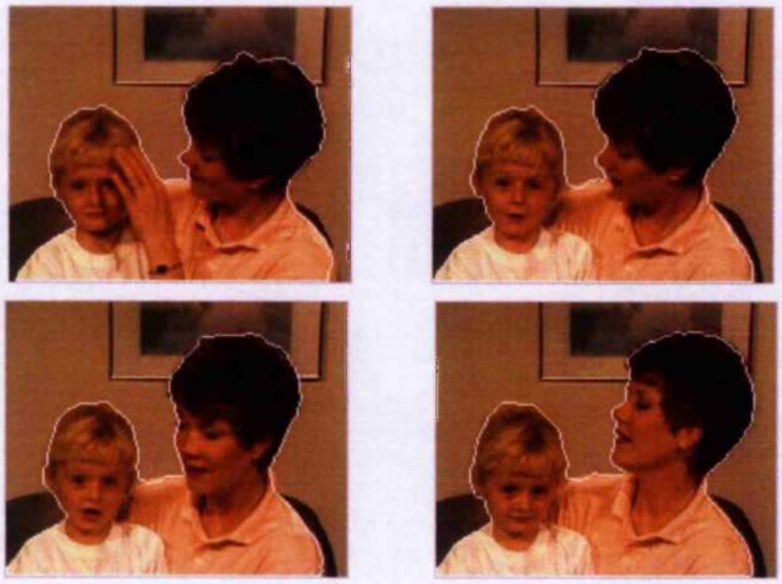
Several sequences have been used to test the performance of our proposed methods. For Motr_dhtr and Claire Sequences, the experimental results are shown in Figure 3.14 and Figure 3.15, respectively.

For the Motr_dhtr sequence, 700 frames are segmented without human interaction except for the first frame. Figure 3.14 (a) shows the predicted contours of four frames using motion information. For most sequences, the performance is acceptable. Large prediction errors may occur if in-plane rotation happens, such as the bottom-left image in Figure 3.14 (a). Figure 3.14 (b) illustrates the refined object contours. The object contour can be detected correctly.

Figure 3.15 shows the segmentation results of Claire. The proposed algorithm can segment 500 frames without human interaction after the first frame. The experimental results show that this contour tracker is robust for tracking nonrigid motion, even with partial occlusion, as shown in Figure 3.14, and Figure 3.15.



(a)



(b)

Figure 3.14 – Video segmentation results (four frames) for Motr_dhtr sequence. (a) Results after contour prediction; and (b) Results after further refinement using active snake

One reason for the success of this algorithm is the use of band constraint during the refinement process. There are several advantages of using the band around the contour compared to using the complete region during probability estimation and pixel classification:

- First, it allows object tracking by adapting to the local changes around the object contour;
- Next, the contour search space is reduced, which can save the computational time;

- Furthermore, the effects of noise and artefact from both foreground and background are reduced.

Most importantly, the proposed contour tracking algorithm can achieve complexity-scalable object tracking. Based on the requirements of the tracking accuracy, not all of the steps of the proposed scheme need to be implemented. This can release much computation burden.

In our proposed algorithm, contour prediction and contour refinement steps achieve the effective prior result of current frame. It is a prediction taken from the posterior result of the previous frame. Further refinement step using active contour can be considered as the measure process, in which the observation information of current frame is used to refine the contour.

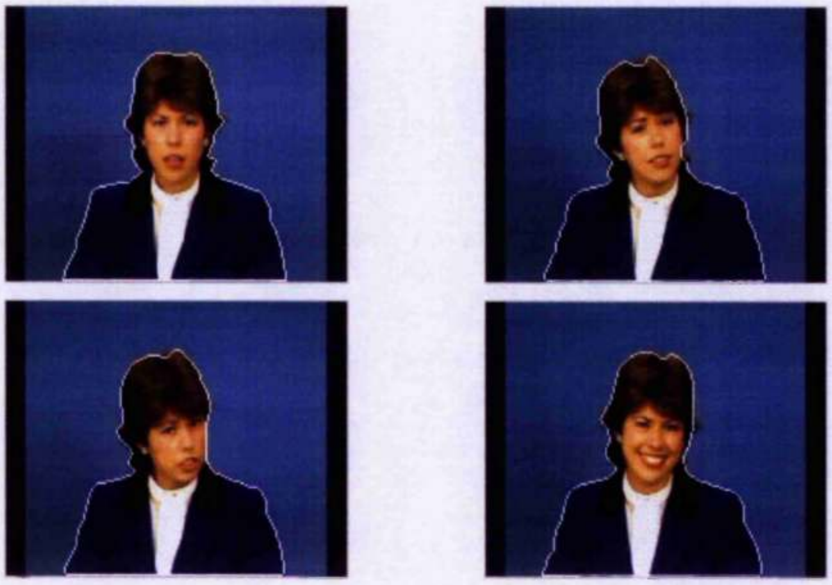


Figure 3.15 – Video segmentation results (four frames) for Claire sequence after further refinement using active snake

3.5 Conclusions

In this Chapter, the importance of video segmentation for object-based or model-based video coding has been explained. Moreover, the main objectives of video segmentation have been presented. For video segmentation, the main conclusion was that there are many techniques available, but individual techniques often consider specific constraints, and thus provide useful results only for the targeted applications, showing intrinsic limitations for dealing with generic audiovisual sequences content. For instance, a spatial segmentation technique does not take the temporal information into account, and a temporal segmentation algorithm is not able to identify static object. Since video segmentation is recognised as a complex problem, the solution often consists of the combination of several analysis techniques in order to exploit their advantages and overcome their shortcomings. Whenever possible, user interaction should be allowed to guide the process, and correct or improve the automatic segmentation results.

After reviewing the main segmentation algorithms, a contour-tracking algorithm has been proposed and employed for video segmentation, which can achieve complexity-scalable object tracking with different tracking accuracies. In this algorithm, a combined spatial and temporal segmentation technique has been used to detect and segment the video object to be tracked. Its contour is used to as the initialisation of the contour-tracking algorithm. Sometimes, user interaction is required to achieve semantic video object segmentation for complex video sequences.

After initialising the object contour, a three-step contour-tracking algorithm is proposed to detect the object contour of the subsequent frames. The tracking results of each step can be used for some special applications with different accuracy requirements. The proposed video segmentation algorithm can be considered as a hybrid feature-based, texture-based and contour-based tracking algorithm.

The experimental results show that this contour tracker is robust for tracking the object contour with nonrigid and large motion, even with partial occlusion. Further research is being conducted within the EC FP6 funded VISNET project (http://www.visnet-noe.org) to improve the performance of contour tracking algorithm and to achieve the multiple object contour tracking with a view to comparing the proposed method with the particle filtering method.

Chapter 4

Face Detection and Its Scalable Modelling

4.1 Introduction

Due to the application of the pre-defined 3D wireframe model in both encoder and decoder, 3D model-based video coding can achieve very low-bit rate coding. Only the analysis parameters need to be transmitted to the decoder. In 3D model-based coding, the encoder tries to recognise the objects (such as faces) in a video scene. As soon as the coder recognises an object, it uses human knowledge about this object to improve the coding performance. For head and shoulder sequences, the algorithms for facial feature recognition and face model adaptation are proposed in [KAMP-1997b]. These algorithms include:

- Detection of face features, such as eyes and mouth, including their corners and contours;

- Adaptation of the predefined face model to the detected human face by scaling the face model horizontally to match the distance between the eyes and vertically to match the distance between the mouth and eyes;

Experimental results show that the efficiency of 3D model-based video coding exploiting the priori knowledge of objects is higher than that of object-based video coding without priori knowledge of objects [KAMP-1997b]. For the proposed scalable 2D model-based video coding scheme, *a priori* knowledge on detected objects can also be employed during the coding of head-shoulder sequences, although the proposed scheme is rather universal and not limited to coding head-shoulder sequences.

This chapter addresses the algorithms for face detection, scalable face model design and its performance evaluation. After reviewing the main face detection techniques, a robust and adaptive face detection method is proposed, which is based on piecewise skin colour distributions.

Next, reliable algorithms are proposed for detecting eyes, mouth and chin that are used to verify the face candidatures. Then, based on the detected facial features and facial muscular distributions, a heuristic scalable face model is designed to represent the rigid and nonrigid motion of head and facial features. An efficient motion estimation method is proposed to evaluate the efficiency of the designed model. The proposed method features three major novelties:

- A robust and simple face detection scheme is proposed. Illumination-piecewise statistical skin colour model and Bayesian detection/relaxation schemes can achieve robust detection to different lighting conditions and skin colour.
- A reliable and simple facial feature detection scheme is proposed, which is very important for its application to scalable 2-D model-based video coding.
- Facial muscular distribution is introduced to build the scalable face model, which can describe face motion more precisely, hence reducing the warping error during scalable model-based video coding.

4.2 Face detection techniques: A review

In recent years, facial feature detection has received considerable attention due to its wide range of applications, such as face recognition, human computer interface, and model-based video coding. Many approaches have been proposed [ROWLEY-1998] [SUNG-1998] [MAIO-2000] [SOBOTTKA-1996] [HSU-2001] [KUO-2002] [WONG-2003]. These approaches apply different techniques, such as neural networks (NN), support vector machine (SVM), geometrical modelling, motion extraction, and colour analysis. Ming-Hsuan Yang gives a more detailed review on face detection algorithms [YANG-2002]. Existing face detection techniques can be classified into four categories, as listed in Table 4.1. These categories are:

1. Rule-based methods.

   These methods are designed mainly for face localization [YANG-1994] [KOTROP-1997]. They encode human knowledge of what constitutes a typical face. Several rules are proposed and employed during face detection. Face features in an input image are extracted first, and face candidatures are identified based on the coded rules. A verification process is usually applied to reduce false detections. One problem with these methods is the difficulty in translating human knowledge into well-defined rules. If the rules are too detailed or strict, they will fail to detect faces that do not pass all the rules.

Moreover, it is difficult to extend these approaches to detect faces in different poses since it is challenging to enumerate all possible cases.

2. Feature invariant approaches.

   The underlying assumption of these methods is based on the observation that humans can effortlessly detect face and objects in different poses and lighting conditions and, so, there must have features which are invariant over these variabilities. These algorithms aim to find structural features that exist even when the pose, viewpoint, or lighting conditions vary, and then use these to locate faces. These methods are designed mainly for face localization [GRAF-1995] [LEUNG-1995] [YOW-1997] [DAI-1996] [MCKEN-1998]. One problem with these methods is that the image features can be severely corrupted due to illumination change, noise, and occlusion. So they are not robust enough.

3. Template matching methods.

   In template matching, a standard face pattern is manually predefined or parameterised by a function. Given an input image, the correlations between an input image and the stored patterns are computed for detection. The existence of a face is determined based on the correlation value. These methods have been used for both face localization and detection [CRAW-1992] [LANITIS-1995] [DENG-1997]. The advantage of these methods is simple implementation. However, it has proved to be inadequate for face detection since it cannot effectively deal with variation in scale, pose, and shape.

4. Appearance-based methods.

   In contrast to template matching, the models in appearance-based methods are learned from a set of training images that should capture the representative variability of facial appearance. In general, appearance-based methods rely on techniques from statistical analysis and machine learning to find the relevant characteristics of face and non-face images. These methods are designed mainly for face detection [TURK-1991] [SUNG-1998] [ROWLEY-1998] [OSUNA-1997]. [RAJAGO-1998] [COLMEN-1997]. The disadvantage of these methods is the computational complexity for learning the face and non-face models.

However, some techniques can be classified into more than one category [GOVIND-1996] [WONG-2003]. For example, template-matching methods usually use a face model and sub-templates to extract facial features, and then use these features to locate or detect faces.

Table 4.1 – Summarisation of the algorithms for face detection within four categories

Approaches	**Representative algorithms**
Rule-based approaches	
	Multiresolution rule-based method [KOTROP-1997].
Feature-invariant methods	
– Facial feature	Grouping of edges [YOW-1997].
– Texture	Space Gray-level dependence matrix of face pattern [DAI-1996].
– Skin colour	Mixture of Gaussian [HSU-2001] [MCKEN-1998] [CHAI-1999].
– Multiple features	Integration of skin colour, size and shape [KJELD-1996] [SOBOTTKA-1998] [WONG-2003].
Template matching algorithms	
– Pre-defined face model	Shape template [CRAW-1992].
– Deformable templates	Active shape model [LANITIS-1995].
Appearance-based schemes	
– Eigenface	Eigenvector decomposition and clustering [TURK-1991].
– Neural network and Naïve Bayes Classifier	Gaussian distribution and multilayer perception [SUNG-1998] [ROWLEY-1998].
– Support Vector Machine (SVM)	SVM with polynomial kernel [OSUNA-1997].
– Hidden Markov Model (HMM)	Higher order statistics with HMM [RAJAGO-1998].
– Information-theory approach	Kullback relative information [COLMEN-1997].

4.3 Proposal for automatic face detection

In this section, an automatic face detection technique is investigated. Although many face detection techniques have been developed in the past, one of the disadvantages of these methods, such as [SUNG-1998], [SOBOTTKA-1996] [SOBOTTKA-1998] [HSU-2001] [CHAI-1999] [KUO-2002], is heavy computational complexity (some of them include training), which makes them unsuitable for the proposed scalable 2D model-based video coding. Another disadvantage is that they are not robust enough to cluttered backgrounds and different lighting conditions. In our research, an automatic face detection technique has been proposed, which includes the algorithms for face detection, eye-mouth extraction and chin detection. These proposed algorithms are discussed in the following subsections.

4.3.1 Face detection and eye-mouth extraction

The proposed face detection algorithm consists of face localisation, eyes detection and mouth extraction. During the detection process, it is assumed that the faces in video sequences are in frontal or near-frontal views. This assumption is reasonable to the application of scalable 2D model-based coding. It can also make facial feature detection and scalable face modelling easier.

4.3.1.1 Face localisation

A robust and adaptive face segmentation method is proposed to locate and regularise face candidatures. The method is based on luminance-piecewise skin colour distributions. It consists of three steps:

1. Detect face candidatures based on a luminance-piecewise statistical skin colour model and Bayesian decision/relaxation;
2. Regularise the face candidatures using spatial segmentation results;
3. Evaluate face candidatures using both shape and size.

There are many methods to locate face candidatures based on skin colour model [CHAI-1999] [HSU-2001]. However, it is found that none of these methods can detect the face robustly under poor and strong lighting conditions. The detected face is full of holes, or in a zigzag shape. In fact, the skin colour model, that is, the distributions of chrominance components C_r and C_b, is related to the illumination value Y. In our research, non-parametric kernel density estimation is used to build the piecewise statistical skin colour distributions. 43 million skin pixels from 900 images in [PHUNG-2002] are used to train the skin models as shown in Figure 4.1. In order to increase its robustness to different lighting conditions, the skin models are separated into 6 parts based on luminance value Y, as shown in Figure 4.1 from (a) to (f). It is shown that Figure 4.1 (a) and (f) have totally different statistical properties from other models in (b) – (e). These two cases are very important for the face detection for dark skin colour and under different lighting conditions.

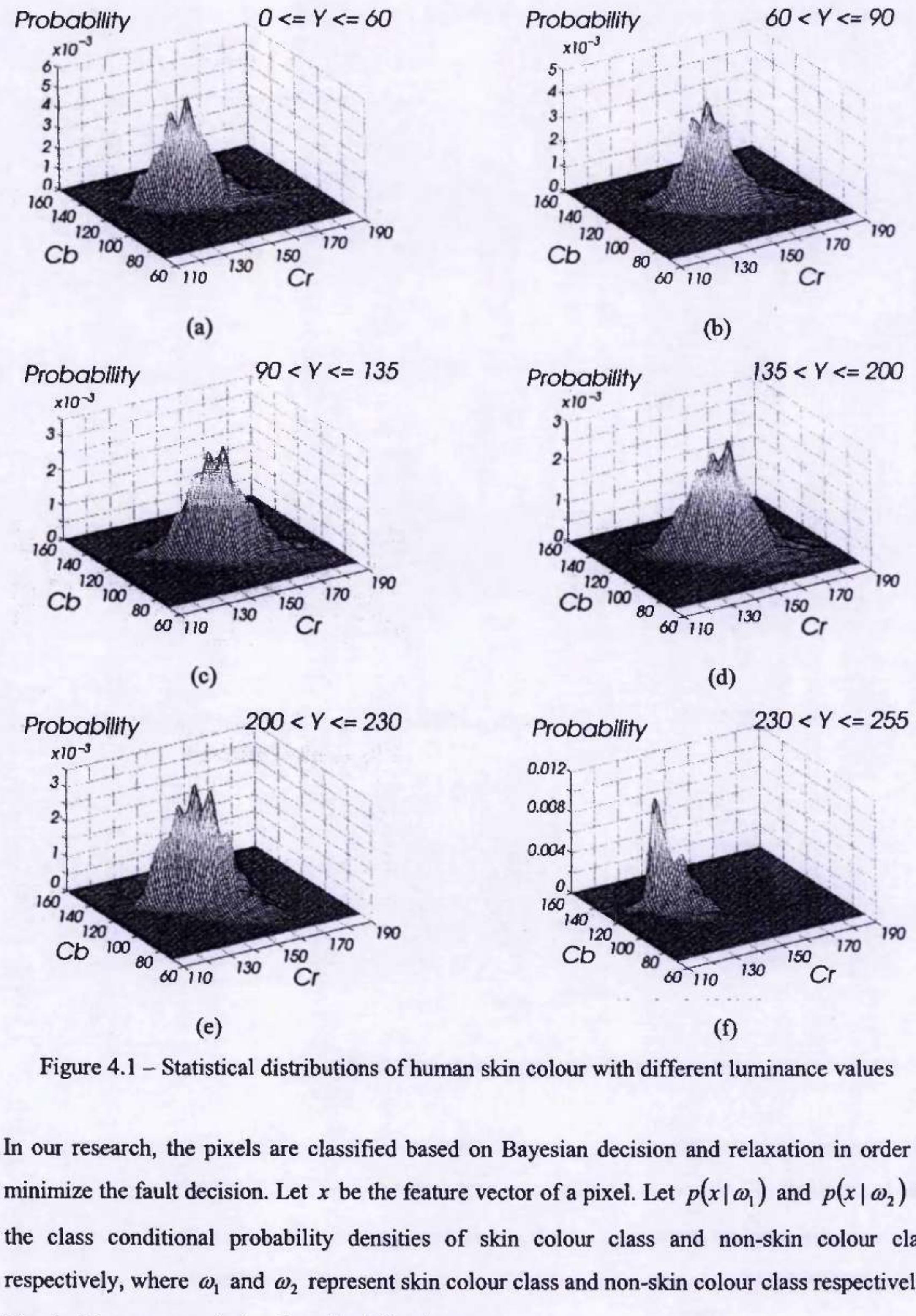


Figure 4.1 – Statistical distributions of human skin colour with different luminance values

In our research, the pixels are classified based on Bayesian decision and relaxation in order to minimize the fault decision. Let x be the feature vector of a pixel. Let $p(x \mid \omega_1)$ and $p(x \mid \omega_2)$ be the class conditional probability densities of skin colour class and non-skin colour class respectively, where ω_1 and ω_2 represent skin colour class and non-skin colour class respectively. The decision commonly involves the following process:

$$\begin{aligned} &\mathit{If}\ \ L(x) = \frac{p(x \mid \omega_1)}{p(x \mid \omega_2)} \geq TH, \mathit{then}\ \ x \in \omega_1 \\ &\mathit{else}\ \ x \in \omega_2 \end{aligned} \qquad (4.1)$$

By applying the Bayesian formula and minimizing the misclassification cost, the following relation exits:

$$\begin{aligned}&\text{If } L(x)=\frac{p(x\mid\omega_1)}{p(x\mid\omega_2)}\geq\frac{C_{2,1}}{C_{1,2}}\frac{P(\omega_2)}{P(\omega_1)},\\ &\text{then } x\in\omega_1\\ &\text{else } x\in\omega_2\end{aligned} \tag{4.2}$$

where loss parameter $C_{i,j}$ means the loss incurred when a pixel of type i is classified as having type j. Since loss associated with correct classification should not affect the design of the classifier, $C_{i,i}=C_{j,j}=0$. During the experiments, we select $C_{1,2}=2*C_{2,1}$ so as to detect all of the possible face skin pixels. Furthermore, it is very hard to get the non-skin class conditional probability density of all kinds of backgrounds pixels, and prior probabilities $p(\omega_1)$ and $p(\omega_2)$. Hence, it is assumed that the non-skin class conditional probability density conforms to uniform distribution, and the probabilities $p(\omega_1)$ and $p(\omega_2)$ are equal. Therefore, the threshold TH in Equation (4.1) is calculated to be 0.5.

The above decision process does not take the relationship among adjacent pixels into consideration, that is, the neighbours of a skin colour pixel are more likely to be skin colour pixels. Therefore, after Bayesian classification, the Bayesian relaxation algorithm proposed in [AACH-1993a] [AACH-1993b] is exploited. The decision is based on the following formula:

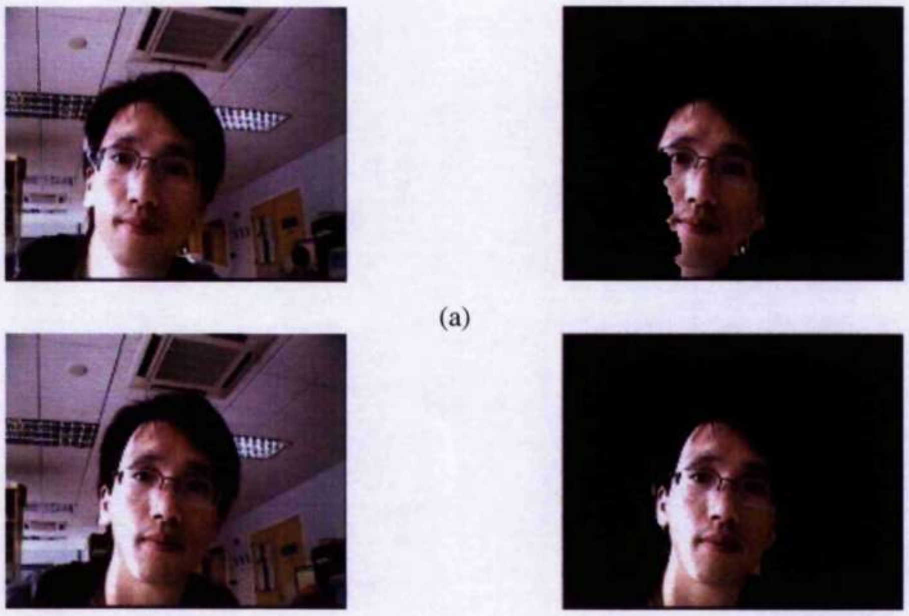
$$\begin{aligned}&\text{If } \frac{p(x\mid\omega_1)}{p(x\mid\omega_2)}\geq\left(TH+8*(B+C)-4*(v_B(s)*B+v_C(s)*C)\right)\quad\text{then } x\in\omega_1\\ &\text{else } x\in\omega_2\end{aligned} \tag{4.3}$$

where $v_B(s)$ is the number of skin pixels which border pixel x horizontally or vertically, and $v_C(s)$ is the number of skin pixels that are diagonal neighbours of pixel x. The cost parameters B and C in relaxation algorithm are so-called potentials, which, when positive, incur an energy increase for each border pixel pair present in a change mask [AACH-1993b]. In our experiments, B and C are chosen to 0.25 and 0.125 respectively, which can achieve stable and reliable results.

After Bayesian decision and relaxation, spatial segmentation is used to regularise the face candidatures. Watershed transform is used to achieve spatial segmentation. Face candidatures are then superimposed on top of the spatial segmentation mask to regularise the shape of the face candidature. In our experiments, if 80% of the spatially segmented patch belongs to a face candidature, the whole segmented patch is considered as part of the face candidature. If 20% of the spatially segmented patch belongs to a face candidature, the whole segmented patch is not part of the face candidature. However, if the ratio lies between 20% and 80%, no change occurs.

For every candidate, the size and shape are evaluated, assuming that human faces in the video are not too small and their shape is characterized by elliptical or oval shape [SOBOTTKA-1998]. Some face candidatures that do not meet the above conditions are considered as non-face patches.

Figure 4.2 demonstrates the results of face localization algorithms by using (a) the skin colour model in [CHAI-1999] and (b) the proposed skin colour model. The images have the same bright lighting conditions. The left hand images in (a) and (b) are the original images containing the face candidature. The right hand images are the detected face candidatures. The results show that our proposed skin colour model can locate the face more precisely.

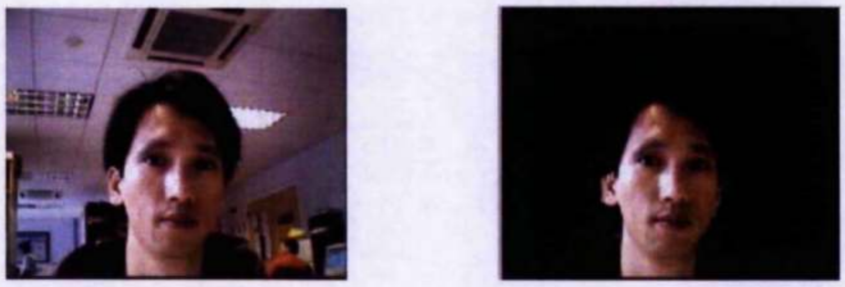


(a)

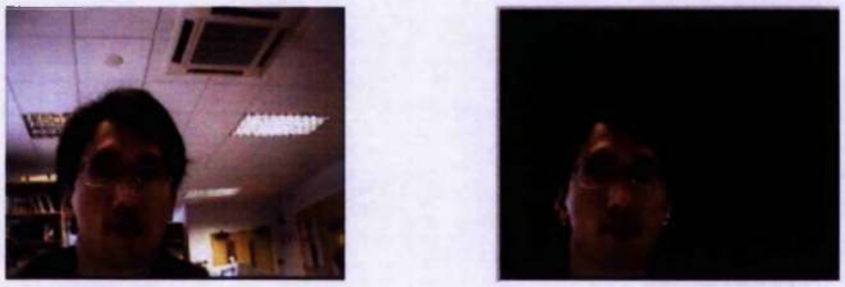
(b)

Figure 4.2 – Face localization using different skin colour models. (a): Original image and the detected face candidature using literature method in [CHAI-1999]; (b): Original image and the detected face candidature using our proposed method. The left images of (a) and (b) are the original images and the right ones are the localised face candidature.

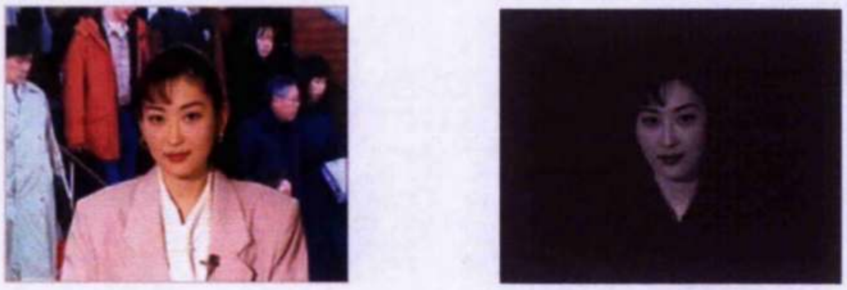
Figure 4.3 shows the face location results for several sequences under different lighting conditions. Parts (a) and (b) are the sequences captured in the Labs with controlled lighting conditions. We can find that the proposed skin colour model can locate faces correctly under both bright and dark lighting conditions, which can not be achieved by using the skin colour model in [CHAI-1999]. Several standard video sequences are also used to test the performance of our proposed scheme, such as Miss_am, Claire, Akiyo, Carphone, etc. The detection results of Akiyo and Carphone sequences are shown in parts (c) and (d) of Figure 4.3. More face localisation results are presented in section 4.5.



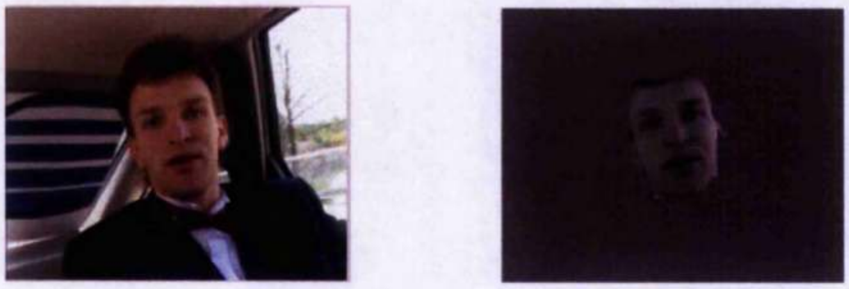
(a)



(b)



(c)



(d)

Figure 4.3 – Results of face localization algorithm. The images in left column are the original images; and the images in right column are the localized face candidatures

4.3.1.2 Locating eyes and mouth

After locating face candidatures, the eyes and mouth should be detected to verify the face candidatures. As it is assumed that the detected face is frontal or near-frontal view, it is not difficult to detect the eyes and mouth.

In our research, colour and luminance are used to locate eye and mouth position, which is based on the observation that high C_b and low C_r values are found around the eyes, and eyes contain both dark and bright pixels in the luminance part. The mouth region contains red lips and some pixels with small luminance value are located between the upper and lower lips. The search procedures for eye and mouth candidatures are illustrated in Figure 4.4. They can be described as follows:

- Enhance C_b and C_r by using histogram equalisation; Calculate colour map $MapC = C_b + (255 - C_r)$, and then enhance it using histogram equalization;

- Emphasize the dark pixels in the Y component using the morphological dilation operation, and calculate the map: $MapY = (dilation(Y)/(errosion(Y) + 0.0001))$. Then, enhance it by using histogram equalisation;

- Calculate eye/mouth decision map: $EyeMouthMap = MapY + MapC$, and normalize it to brighten both eyes and mouth, and to suppress other noises;

- The eye and mouth candidatures are initially estimated by iterative thresholding of $EyeMouthMap$. The iterative thresholding method in [PEREZA-1987] is used.

The search region is restricted to the located face candidatures. Figure 4.5 shows the results of the above search procedures for the Carphone sequence, and the final location of the eyes and mouth. In Figure 4.5, (a) is the luminance component; (b) and (c) are the enhanced C_b and C_r. (d) and (e) show the calculated maps $MapY$ and $MapC$. (f) is the calculated $EyeMouthMap$ and is used to locate the position of eye-mouth candidatures in (g). Experimental tests show that the proposed method has two advantages when compared with that in [HSU-2001]:

- First, it requires less computation for eye-mouth localisation, which can save about 30% computational complexity for different face sizes.

- Next, it is more robust for faces with different kinds of lighting conditions because some lip colour is faint or is similar to its surrounding skin.

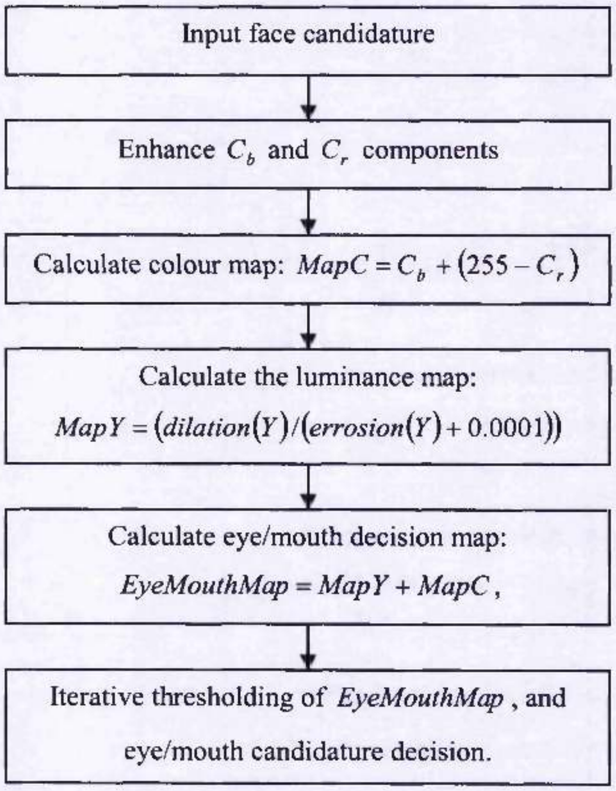

Input face candidature
Enhance C_b and C_r components
Calculate colour map: $MapC = C_b + (255 - C_r)$
Calculate the luminance map:
$MapY = (dilation(Y)/(errosion(Y) + 0.0001))$
Calculate eye/mouth decision map:
$EyeMouthMap = MapY + MapC$,
Iterative thresholding of $EyeMouthMap$, and
eye/mouth candidature decision.

Figure 4.4 – Flowchart of search algorithm for eye and mouth candidatures

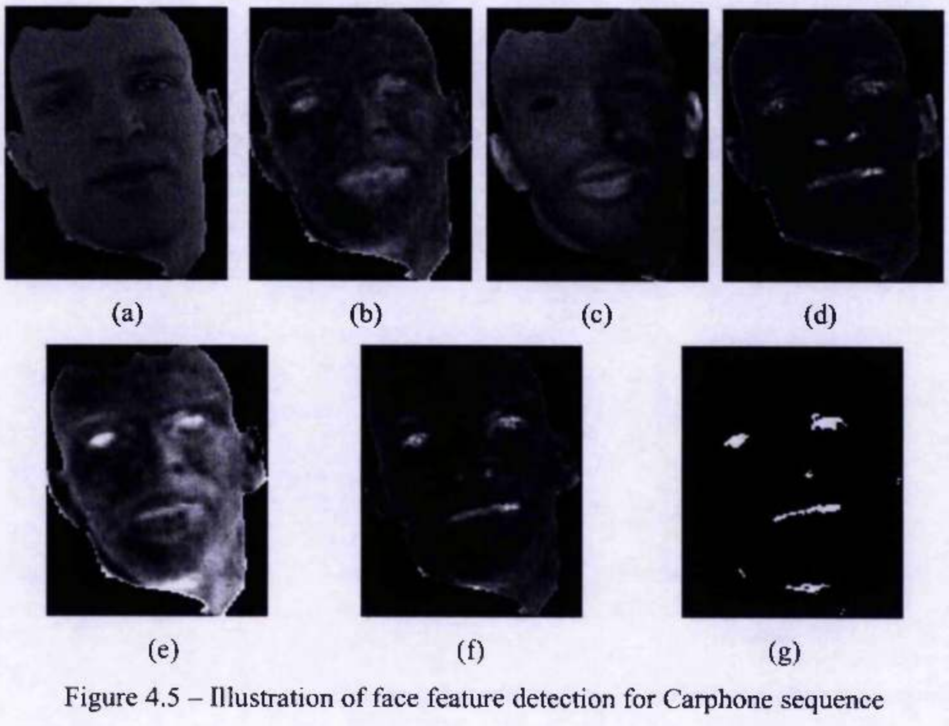


Figure 4.5 – Illustration of face feature detection for Carphone sequence

4.3.1.3 Verifying eyes and mouth pairs

For the detected eyes and mouth candidatures in Figure 4.5, there are 6 eye or mouth candidatures and $C_6^3 = 20$ kinds of eye-mouth combinations, theoretically. However, the geometry and orientation information can be used to reduce this number. In our research, the symmetry of eyes and mouth localization are proposed to verify the eyes and mouth pairs. Figure 4.6 illustrates the geometry and orientation relations among face, eyes and mouth. The following criteria are used during verification, which are satisfied for frontal or near-frontal view of faces:

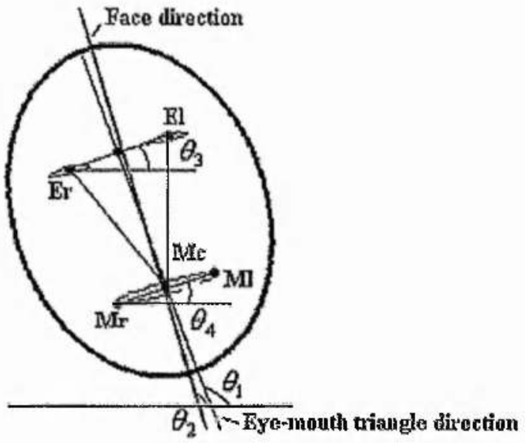


Figure 4.6 – Face and facial feature geometry and orientation

1. The face is upright and eye pair should be located in the upper half face (above the minor axis of the fitted ellipse. This can reduce the number of eye-mouth pair from 20 to 9 for the face candidature in Figure 4.5.

2. For every face candidature, the direction θ_2 of the major axis of the fitted ellipse should be almost the same as the direction of the vector from the midpoint of the two eyes to the mouth θ_1. If the difference between θ_1 and θ_2 is less than a threshold (10 degree is used in our experiments), it is a face. Otherwise, it is not a face.

3. The vector, which is perpendicular to the interocular segment ($\overline{E_r E_l}$ in Figure 4.6) and passing the midpoint of two eyes, should pass the mouth candidatures.

4. The line passing two mouth corners should be almost parallel to the line passing two eyes. This means that the θ_3 and θ_4 in Figure 4.6 should be the same (5 degree difference is tolerated).

Experimental results show that the eye-mouth pair can be detected and verified correctly based on the above four criteria. 50 face images are tested and the detection correction rate of eye-mouth pairs is 100%. Experimental results show that the computational complexity of the proposed method is about 30% - 50% of that of the method in [HSU-2001].

4.3.1.4 Detecting the corners of eyes and mouth

After locating the position of the eyes and mouth, their four corners are detected to build the scalable face model. For eye corner detection, two methods have been proposed for different face sizes.

- If face size is small (smaller than 32 x 64), the method is based on the Morphological Open by Reconstruction Filter (MORF) and thresholding.
- Otherwise, a deformable template matching algorithm is used to detect the eye corners with high accuracy, but also with high computational complexity.

For detecting eyes with a small size face, the procedure consists of:

- MORF is used for the eye patch, followed by thresholding in order to obtain a binary map.
- For every column of this patch (from left to right), the first and last columns with zero elements are chosen as the column on which the eye corners are located. The centre of the eye can be estimated based on the eye corners. This scan step is illustrated in Figure 4.7.

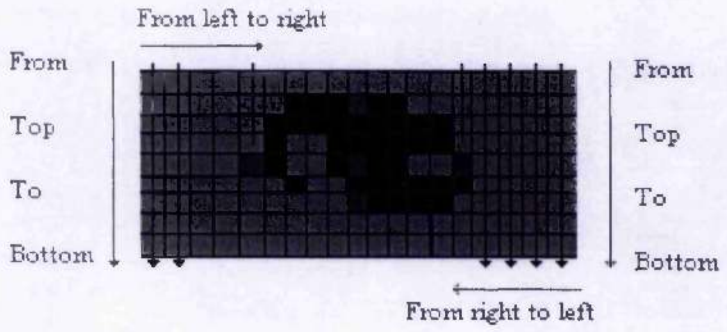


Figure 4.7 – Scan procedure for eye corner detection

- The upper and lower eyelids (two points) can be estimated, as the line joining these two points is perpendicular to the line joining the two eye corners.

For detecting eyes of larger size, it is not easy to estimate the eye corners precisely by using the above method. Deformable template matching algorithm is used. The edge and valley energies are used to adjust the template, and are defined as those in [YULLE-1992]. Interested readers are referred to [YULLE-1992] for the detailed descriptions.

For mouth corner detection, deformable template matching algorithm is used to detect the four mouth corners. However, several modifications are made to improve its speed:

- The SUSAN corner detector [SMITH-1997] is used to detect the right and left mouth corner candidatures. This can reduce the search region for deformable template matching.

- Lip colour distribution is used to further reduce the search position of the deformable template. Colour distribution inside the mouth is modelled as a Gaussian mixture with three components: a dark aperture, pink lips and bright reflection of light from the teeth or lips. The parabolas of the upper and lower lips should try to include more pink lips in the template.

Figure 4.8 shows the eyes, and mouth detection results of Akiyo and Carphone sequences. It shows that eyes and mouth can be detected precisely by using the proposed methods.

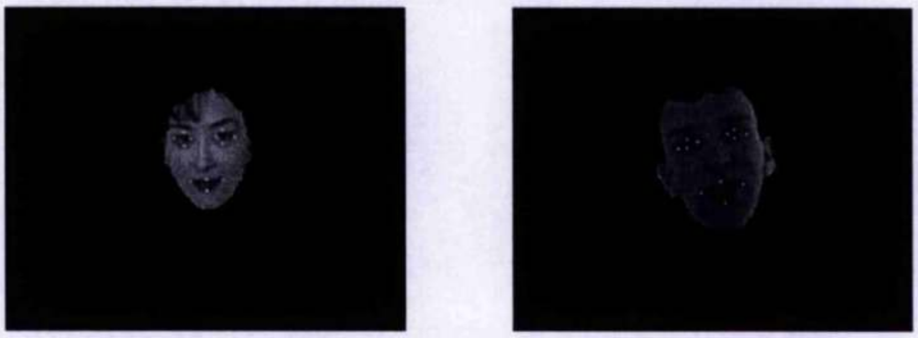

(a) (b)

Figure 4.8 – Eyes and mouth detection for Akiyo and Carphone sequence

4.3.2 Chin Detection

For video coding of videophone sequences, the human face undergoes both rigid and non-rigid motion. In order to represent the face motion precisely and reduce the warping error, efficient chin detection algorithm is necessary.

Several methods have been proposed to estimate the chin contour [RUDIA-1996] [KAMP-1997a] [GOTO-2002] [AHLBERG-2002]. Although the authors claim that their methods can achieve the optimal chin position, which is commonly decided subjectively, their methods have been proved to have some shortcomings. In [RUDIA-1996], the active contour model (Active Snake) has been used to estimate the chin contour. The active snake model is an energy-minimizing spline influenced by external forces and image features. However, when the chin contour appears loosely marked due to weak contrast of the chin in relation to the neck below it, the reliability of the Active Snake model is low. Furthermore, the initialisation of the Snake and the chosen external force affect its performance seriously. In [KAMP-1997a], the concept of deformable templates is used to estimate the chin contour. Two parabolas are used to represent the chin. A cost function is minimized to find the best fit of the template to the chin. However, experimental results have shown that a deformable template can only be used for chins with parabola-like shapes, for example, the chin of Akiyo and Claire. It cannot detect the chin of Carphone correctly. This is due to the variety of chin shape.

In [GOTO-2002], the described chin detection method consists of three steps. First, three points are found on both sides and the bottom tip of the chin. Next, a curve is found to connect these three points. Finally, the position of this curve is modified to fit the chin better. However, it is not easy to find these three points. If the camera is not placed exactly in the front view, the curve fitting result cannot be accurate. In paper [AHLBERG-2002], Ahlberg presented a way to regard the facial feature detection problems as an optimisation problem. Deformable graphs are used to represent the relative position of features. However, this method is very complex and time-consuming. The results given in [AHLBERG-2002] are not satisfactory.

In our research, an efficient chin detection method is proposed, which combines deformable template matching with the active contour method. The prior chin shape is trained and exploited in the active contour model to improve detection accuracy. The proposed method features three major novelties.

1. A deformable template-matching algorithm is applied to initialise the chin contour. Therefore, no user interaction is needed to initialise the snake model. It can also avoid the disadvantage of the method in [GOTO-2002].

2. In order to reduce the effect of weak chin edges and strong background edges, an edge normalization step is introduced. Gradient vector flow (GVF) [XU-1997] is used to get the smoother external force, which is derived from the normalized edge distribution.

3. Chin shape priors are trained and are included in the active snake model to improve its robustness to weak contrast of the chin in relation to the neck below it, and partial occlusion.

In section 4.3.2.1, the deformable template matching method is discussed for initial chin detection. Section 4.3.2.2 describes the refinement of chin contour by using the active snake method, which includes the scheme description, edge detection and the GVF calculation, and prior shape training.

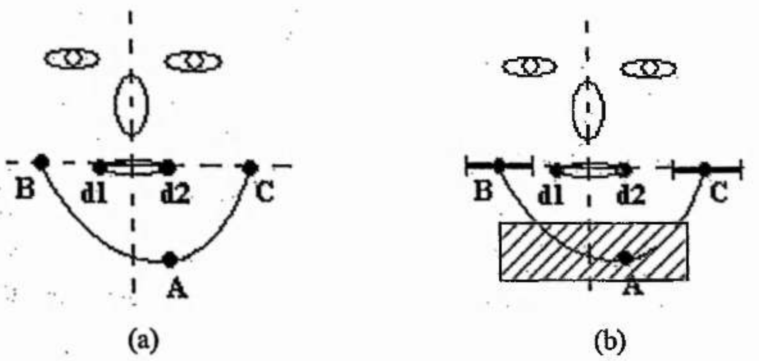


Figure 4.9 – (a) Deformable template model for the chin contour; (b) Search region for points A, B and C.

4.3.2.1 Initial chin estimation

It is known that the initialization of the active snake model will affect its final detection performance. Sometimes, user interaction is required for the initialization, such as in [RUDIA-1996]. In order to initialize the active contour model automatically, a deformable template matching method is employed to initially estimate the chin contour. The deformable template for the chin contour is shown in Figure 4.9 (a), which can be described by the positions of A, B, and C. The points $d1$ and $d2$ are the detected mouth corners. Points B and C belong to the chin parabolas and are located on the line which goes through the mouth corners. Based on the positions (x_A, y_A), (x_B, y_B) and (x_c, y_c) of A, B, and C, respectively, the chin contour can be described. During our experiments, it is found that the accuracy and complexity of the deformable template matching method are affected by the search ranges of point A, B, and C. The search ranges of A, B, and C during the detection process are different from those used in [KAMP-1997a] to make the search more robust, which are shown in Figure 4.9 (b).

In order to reduce the computational complexity further, the position of A, B, and C can be estimated sequentially. The detailed procedures are described as follows:

1. Points B and C are estimated by using the gradient value of chrominance difference $C_r - C_b$. They are searched along the line that goes through the mouth corners. They are also in the detected face region.
2. Point A is moving in the search region to maximize cost function. The luminance valley and the gradient are combined as the cost function. A larger search range is used for point A in order to make it robust for all circumstances, such as face rotations.

The detected chin contour is decided by maximizing the total energy along the chin contour, which is defined as:

$$E_{total} = E_{valleys} + E_{Gradient} \tag{4.4}$$

$$E_{valleys} = Normalize(255 - Y) \tag{4.5}$$

$$E_{gradient} = Normalize\left(sqrt\left(Y_x^2 + Y_y^2\right)\right) \tag{4.6}$$

where function $Normalize(x)$ normalizes the value x to the range $[0,1]$. The detected chin contour is used as the initial position of the active snake.

4.3.2.2 Refinement of chin contour

4.3.2.2.1 General description

Not all chin contours have a parabola-like shape. For example, the deformable template matching method does not perform very well for the Carphone sequence. Therefore, a refinement step is needed to get correct chin detection for all of the test sequences.

In our research, the snake (active contour) model, incorporating prior chin shape, is used to refine the initial chin contour. The snake is an energy-minimizing spline, defined within an image domain that can move under the influence of internal forces within the curve itself and external forces derived from the image data. The internal and external forces are defined so that the snake will conform to an object boundary or other desired features within an image. However, this method may be sensitive to the starting position and may leak through the object if the edge feature is not salient enough in a certain region of image. In our research, a prior chin shape model is trained and incorporated in order to improve the robustness of the active snake.

There are many methods incorporating prior shape during boundary finding [STAIB-1992] [COOTES-1995] [YONG-1998]. The method in [YONG-1998] is selected in our investigation due to its less computation, which is ten times faster than the method in [COOTES-1995]. The objective is to maximize the *a posterior* density of the final shape given the input edge image and shape parameters, which can be expressed as (similar to that in [YONG-1998]):

$$J = \arg\max_{Q} \left(\beta_1 * \left(\sum_{i=i}^{t+1} \left[-\frac{(q_i - m_i)^2}{2\sigma_i^2} \right] \right) + \beta_2 * E_{snake} \right) \tag{4.7}$$

where vector $Q = (q_1, q_2, \cdots q_{t+1})$ is the pose and shape parameters, which will be defined and discussed in section 4.3.2.2.3. m_i is the mean value of shape parameters, which is defined to be zero relative to the mean shape. σ_i^2 is the eigenvector's corresponding eigenvalues calculated from the train sets. E_{snake} is the snake gradient-curvature energy along current snake contour. β_1 and β_2 are used to balance the influence of the shape model and snake model. The trade-off between shape and image depends on how much faith one has in the shape model and the imagery for a given applications. In our experiments, we set these parameters empirically as $\beta_1 = 2$ and $\beta_2 = 1$.

Above maximization can be achieved using the following processes:

1. The active snake model guides the contour evolution, which maximizes the cost function E_{snake}.

2. Update the pose and shape parameter vector Q to best fit the newly found points, which can also maximize Equation (4.7).

3. Repeat until convergence

The detail description of above processes are presented in section 4.3.2.2.2 and section 4.3.2.2.3

4.3.2.2.2 Snake-guided contour evolution

The objective of the first step in the above maximization algorithm is to evolve the contour to position with large gradient. Instead of using the active snake model in [RUDIA-1996] directly, the following improvements have been conducted in order to achieve robust chin detection:

- First, Canny edge detector [CANNY-1986] is used to detect the weak edge by selecting its proper parameters and generate the binary edge map. Commonly, there exists weak contrast of the chin in relation to the neck below it.
- Next, in order to get a smoother external edge and wider convergence range, GVF is calculated and used as the external force, instead of using binary edge map directly.

Suppose that N discrete points are selected along the given active contour C. The discrete snake energy E_{snake} can be expressed as follows:

$$E_{snake} = \sum_{i=0}^{N-1} E_i = \sum_{i=0}^{N-1}\left(E_{\text{int},i} + E_{ext,i}\right) \tag{4.8}$$

where each E_i depends on the contour segment between up to three points v_{i-1}, v_i and v_{i+1}.

The dynamic programming algorithm in [AMINI-1990] has been used to search for the maximum of (4.8). In the implementation, for node i, the search locations are restricted along the bisector lines of the angle $\angle\overline{v_i v_{i-1}}, \overline{v_i v_{i+1}}$. As the points are placed at regular intervals along the contour, the internal energy in node i is selected as:

$$E_{\text{int},i} = 2 - 2\cos\angle\overline{v_i v_{i-1}}, \overline{v_i v_{i+1}} \tag{4.9}$$

In our research, an edge map is selected to induce the external energy E_{ext}. The Canny edge detector has been used to obtain the edge map [CANNY-1986]. The low threshold value that is used in the hysteresis step of the Canny edge detector is set to 0 in order to detect the weak chin edge. Short edges are removed in order to reduce the effect of noise in the face area, and a binary edge map ' *BinaryEdgeMap* ' is generated.

If the binary edge map is used directly, the convergence of the active contour algorithm is still affected by the chin shape. In order to improve its convergence, gradient vector flow (GVF) [XU-

1997] is used as the external force, instead of using gradient directly. The main advantage of the GVF is that it can capture a snake from a long range and extract it into concave regions.

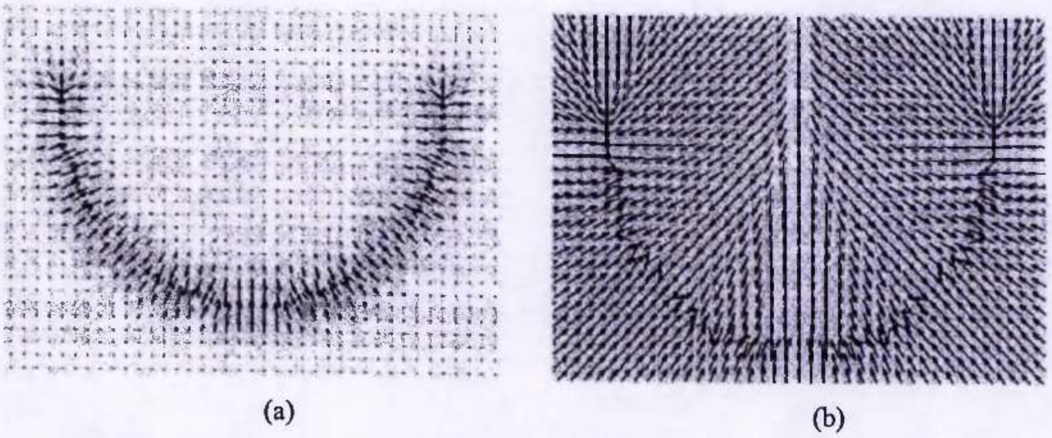
We define the edge map $f(x,y) = BinaryEdgeMap$, which is derived from the detected face patch. Gradient vector field is the vector field $V(x,y) = (u(x,y), v(x,y))$ that can be derived by minimizing the following function:

$$\varepsilon = \iint \mu\left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right) + |\nabla f|^2 |V - \nabla f|^2 dxdy \tag{4.10}$$

where u and v are the horizontal and vertical coordinates of vector field V respectively; μ is a regularization parameter governing the trade-off between the first term and second term. This vector field is smoother than the gradient when there is no intensity change. Particularly, when $|\nabla f|$ is small, the energy is dominated by partial derivatives of the vector field. On the other hand, when $|\nabla f|$ is large, the second term dominates the integrand and is minimized by setting $V = |\nabla f|$.

Using variational calculus, the GVF can be found by solving the following Euler equations:

$$\begin{aligned} \mu\nabla^2 u - (u - f_x)(f_x^2 + f_y^2) &= 0 \\ \mu\nabla^2 v - (v - f_y)(f_x^2 + f_y^2) &= 0 \end{aligned} \tag{4.11}$$



(a) (b)

Figure 4.10 – (a): Edge map after Gaussian kernel smoothing; (b): GVF image

From Equation (4.10), we note that in homogeneous regions, the second term of both equations is zero. Therefore, within these regions, u and v are each determined by Laplace's equation. Figure

4.10 shows the calculated edge map gradient of a simulated half-circle edge, and the calculated edge GVF field. It is found that the edge map GVF is smoother and has a wide convergence range. The external force E_{ext} in Equation (4.8) is replaced by $V(x,y)$.

4.3.2.2.3 Prior shape calculation

Before incorporating the prior shape into the contour detection scheme, a set of training chin contours are exploited to deduce the model distribution, which is based on the mean positions of the points on the aligned shapes and the main variation of the points from the mean [YONG-1998].

Suppose each shape can then be represented by a $2N$-element vector $Z = (x_1, y_1, \cdots, x_N, y_N)^T$. When all chin contours of the training image are aligned into a common coordinate frame, a cloud in the $2N$ dimensional space is formulated. In our research, Principal Component Analysis (PCA) is applied to model the data, which computes the main axes of this cloud. Based on this model, we can generate new examples, similar to those in the original training set, and we can examine new shapes to decide whether they are plausible examples.

Based on S aligned training samples, we can calculate the mean shape and the covariance about the mean as:

$$\overline{Z} = \frac{1}{S}\sum_{i=1}^{s} Z_i \tag{4.12}$$

$$C = \frac{1}{s-1}\sum_{i=1}^{s}(Z_i - \overline{Z})(Z_i - \overline{Z})^T . \tag{4.13}$$

It can be shown that, by principal component analysis, the eigenvectors of the covariance matrix C, corresponding to the largest eigenvalues describe the most significant shape model of variation. After computing the eigenvectors p_i, and their corresponding eigenvalues λ_i of C, the first t largest eigenvalues are chosen such that:

$$\sum_{i=1}^{t}\lambda_i \geq 0.98\sum_i \lambda_i \tag{4.14}$$

Any shape in the training set can be approximated using the mean shape and a weighted sum of deviations obtained from the first t shape models:

$$Z_i \approx \overline{Z} + Pq \tag{4.15}$$

where $P = (p_1 \mid p_2 \mid \cdots \mid p_t)$ contains t eigenvectors of the covariance matrix and $q = (q_1, q_2, \cdots q_t)$ is the vector of weights, which is also the set of t shape parameters to be optimised in Equation (4.7). Equation (4.15) allows us to generate new examples of shapes by varying the parameters $q = (q_1, q_2, \cdots q_t)$.

During our research, 350 training chin contours are selected from face databases [PHUNG-2002] [FACE-2004a] [FACE-2004b], which contain different face shape, pose, and expressions. Before aligning all chin contours, they are normalized by the distance d_1 horizontally and by distance d_2 vertically, and rotated by α degrees to make the line between the two eye centres horizontal, as shown in Figure 4.11 (a). After being normalized and rotated, each contour is annotated with 46 points, which are sampled from -90^0 to 90^0 with 4^0 between each two consecutive points. The sampled contours are listed in Figure 4.11 (b), where the mouth centre is the origin of the coordinates. The iterative approach in [COOTES-1995] is applied to align the sampled contours, as shown in Figure 4.11 (c).

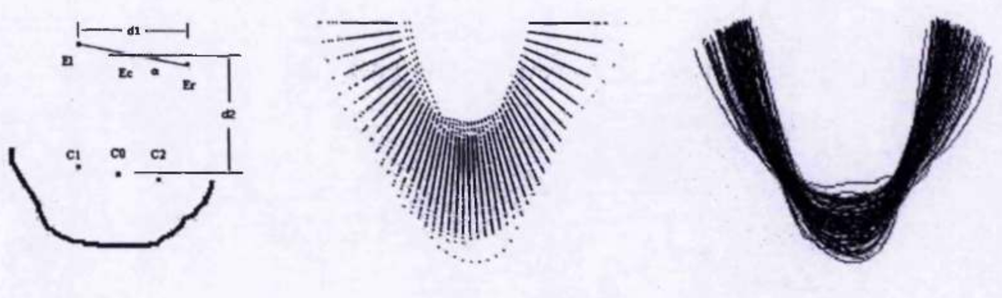


Figure 4.11 – Trained chin contour. (a): Normalized chin contour; (b): Sampled chin contours after normalisation; (c): Aligned chin contours

After aligning the training set, the PCA algorithm is used to calculate the model distribution. We found that the shape model is mainly decided by the first 4 shape parameters, which can explain 98% of the variance of point position in the training set. Therefore, $t = 4$ is chosen and applied throughout the experiments.

Figure 4.12 shows the effect of varying the four shape parameters in turn between ± 2.5 standard deviations, leaving all other parameters at zero. It is found that these models can explain global variation due to different face shape, pose, and expression. Less significant modes cause smaller and more local changes, such as in Figure 4.12 (c) and (d).

Before updating the pose and shape parameter vector Q to best fit the newly found points, which are detected by the proposed active snake algorithm, the detected points should be rotated by an angle α, which is constant during evolution. Therefore, the pose and shape parameter vector Q only consists of one scale parameter s and four shape parameters (q_1, q_2, q_3, q_4). We can adjust $Q = (s, q_1, q_2, q_3, q_4)$ to achieve the objective in formula (4.7). In (4.7), the mean m_i for each shape parameter is 0 and its variance σ_i is the eigenvector's corresponding eigenvalues. The mean for scale is 1 and variance is set to 0.5.

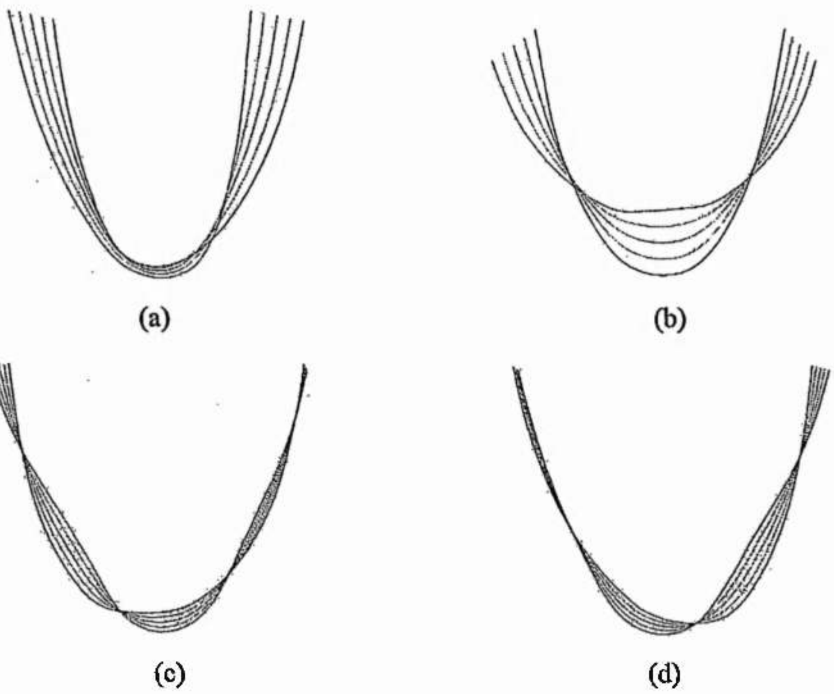


Figure 4.12 – Effects of varying each of first four chin model shape parameters in turn between ±2.5 standard deviations, leaving all other parameters at zero. (a) Model 1; (b) Model 2; (c) Model 3; (d) Model 4

4.4 2-D Scalable face model design

In the proposed scalable 2D model-based video coding scheme, the human face is considered as a special object and is modelled separately from other video objects since the human face undergoes both rigid and non-rigid motion. Its motion description is complex. Furthermore, the human face is more important and small warping error during video coding is annoying. Much research has been conducted on facial feature motion analysis and description [WATERS-1987]. This *a priori* information can be used for our scalable model design.

In our research, a heuristic scalable face model is constructed, based on the muscle distribution of human faces in [WATERS-1987]. During scalable model design, more important nodes in the lower level are allocated to the face features and intersection points between different muscles in order to represent the facial motion more precisely and reduce the estimation and warping error during video coding.

In our research, a three-level scalable face model is designed which is shown in Figure 4.13. The design process is described as follows:

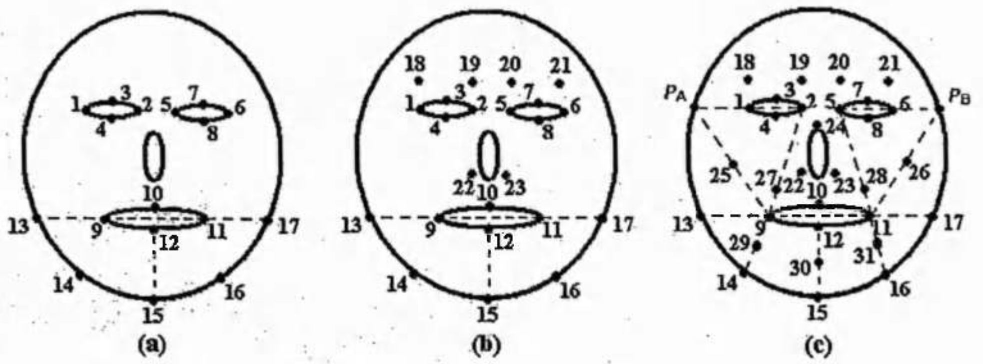


Figure 4.13 – Scalable face model design. (a) layer 1; (b) layer 2; and (c) layer 3

First, eight nodes and four nodes are allocated on the eyes and mouth respectively to represent their movement. In addition, five nodes are used to approximate the contour of chin. The points 13 and 17 are found by extending the interconnecting lines between the mouth right and left corners towards the borders of the face segment. Point 15 is the intersection point between the line formed by interconnecting the mouth upper and lower corners (point 10 and point 12), and the detected chin. These points are very important to represent the movement of the face and are included in the lowest level 1 (from 1 to 17 in Figure 4.13 (a)).

For level 2, shown in Figure 4.13 (b), six additional points are introduced to represent the movement of eyebrows and nose (from 18 to 23). They are useful for head motion estimation. For points 22 and 23, if the face size is small, they are merged at one point located in the middle of their positions.

For level 3, 8 additional points are allocated mainly based on the face muscle distribution [WATERS-1987]. The scalable model of level 3 is shown in Figure 4.13 (c). The points *PA* and *PB* are only auxiliary points to deduce other points' position, which are not included in the scalable face model. They are found by extending the interconnecting lines between the predetermined points of eye corners towards the borders of the face segment. Points 25 and 26 are located at the middle of line segments joining mouth corners to points *PA* and *PB*. The locations of other points are shown in Figure 4.13 (c). This heuristic model is based on the face muscle distributions used to represent the face motion. A more complex model can be designed if more complex face expression needs to be encoded.

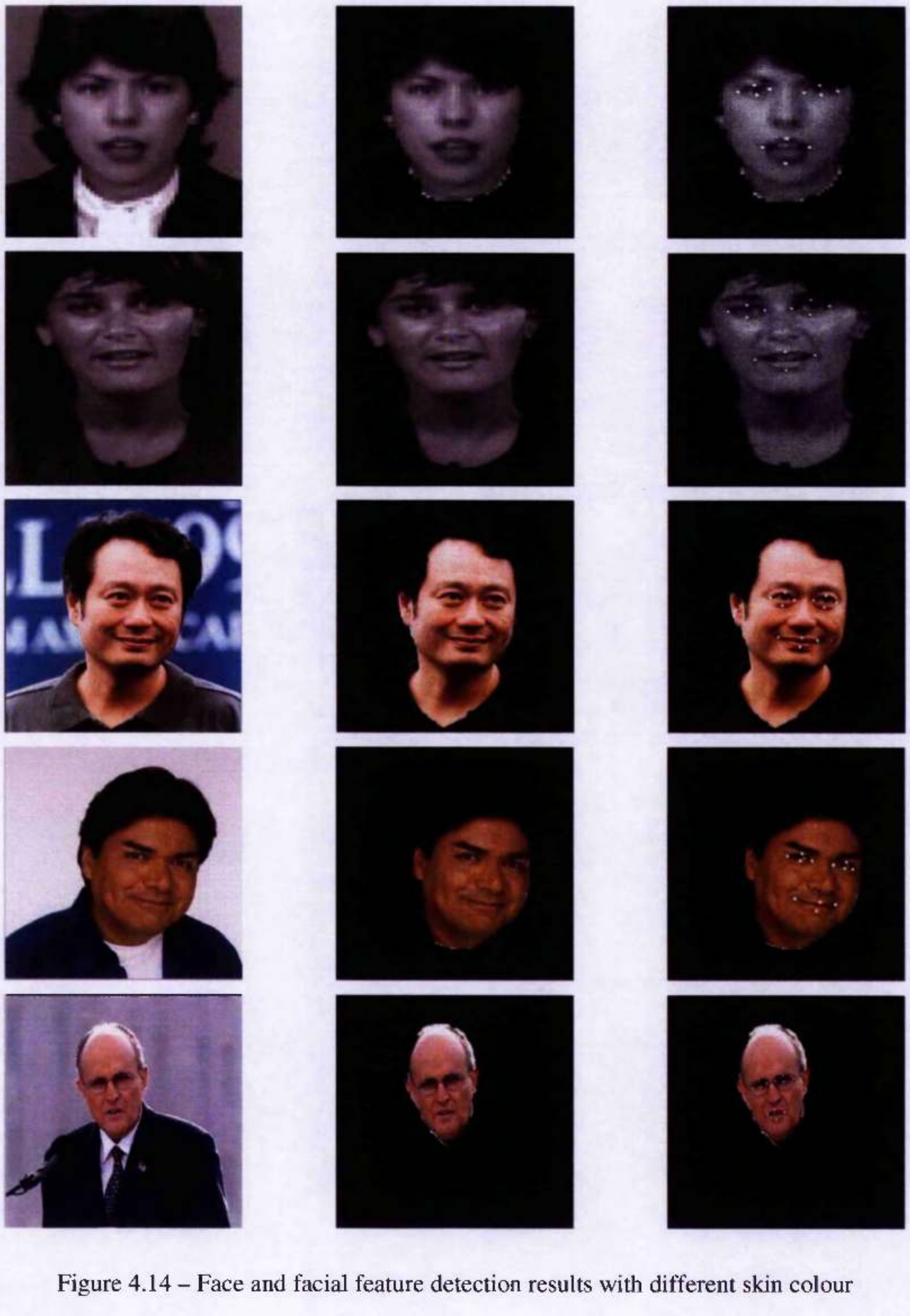


Figure 4.14 – Face and facial feature detection results with different skin colour

4.5 Experimental results

4.5.1 Results for eye-mouth detection

The objective of our proposed methods in section 4.3.1 is to detect the frontal or near-frontal view of the face under varying lighting conditions so that the scalable face model can be designed automatically. Therefore, profile views of the face are not considered in the experiments. In section 4.3.1, some results are presented that demonstrate the performance of the proposed method. More images and image sequences are used in this section to test the performance of the proposed methods for face and facial feature detection. Almost 300 human faces are used. These faces cover several racial groups and varying lighting conditions. All of the faces are localised correctly (accurate rate is 100%) and eye-mouth features are extracted properly.

Some face and facial feature detection results are illustrated in Figure 4.14. The first column shows the original images. The second column gives the detected face patches. The third column shows the detected eyes, and mouth components. From the results, the proposed algorithms can detect the facial features correctly, irrespective of whether the face is under strong lighting or uneven lighting conditions.

4.5.2 Experimental results for chin detection

Extensive simulations have been conducted to evaluate the performance of the proposed chin detection algorithm in section 4.3.2 and compare it to other published methods [YULLE-1992] [KAMP-1997a]. Figure 4.15 shows the performance of a number of chin detection algorithms. The first method is the chin detection algorithm using deformable template matching. The second one is the initial version of our proposed method without incorporating the prior chin shape [HU-2003]. The third method is the proposed method including the prior chin shape.

For column (a) in Figure 4.15, deformable template matching algorithm has been applied. For some face shapes, such as Akiyo, the chin contour can be detected correctly. However, for most faces, its performance is rather poor. This is because not all chin shapes can be represented by parabolic curves. Column (b) lists the detection results using our method in [HU-2003] without considering the prior shape. Compared with those in (a), the proposed chin detection method achieves higher detection accuracy than deformable template matching. For example, for the Carphone face in row 2, chin detection is improved greatly. It is shown that the proposed method can detect chins with different face shapes. However, as no global information is used, the detected chin contour is not smoother than (a). Images in column (c) of Figure 4.15 show detection results generated by our proposed method including the prior shape information. As the prior shape of the chin contour has been taken into account, the proposed method can achieve a smoother and more accurate chin contour. Figure 4.16 shows the ability of the proposed algorithm to cope with weak chin edges and cluttered backgrounds. Image (a) is the original image taken in Lab. Image (b) shows the chin detection result by using the active snake model in [RUDIA-1996].

It is found that the algorithm in [RUDIA-1996] is seriously affected by strong edges and cannot converge to an accurate chin position, as shown in Figure 4.16 (b). Figure 4.16 (c) shows the result from using the method in [HU-2003]. As the low threshold value, used in the hysteresis step of the Canny edge detector, is set to 0, the weak chin edge is detected. Then, all the edge gradient values are set to 1 and short edges are removed. This can remove the effect of strong edges and noises around the chin contour. Furthermore, as deformable template matching method is used to detect the initial chin contour, this can improve the detection performance greatly. However, as no prior shape information is incorporated in the snake model, the detected chin contour is not smooth enough. The chin contour in Figure 4.16 (d) is detected using our proposed method incorporating prior shape information in the snake model. It can be seen that this method achieves a smoother chin contour than the method in [HU-2003] without taking the prior shape into consideration.

The performance of the proposed method for coping with occlusion has also been investigated. The main idea of introducing the chin shape prior is to allow the detection method to cope with missing or misleading information. In the case of occlusion, we expect the statistical shape prior to inducing a reconstruction of the shape contour that is not visible. Figure 4.17 (a) shows the image, in which a small part of chin is occluded. We found that the methods in [RUDIA-1996] and [HU-2003] can not detect the chin contour correctly, as illustrated in Figure 4.17 (b) and (c). The detection result of the proposed method is illustrated in Figure 4.17 (d). The chin shape is reconstructed according to the shape prior in the area where the occlusion occurs. However, if the size of occlusion is very large, the quality of the detection result degrades because the information contained in the image may not be sufficient to correctly guide the snake model to the correct position

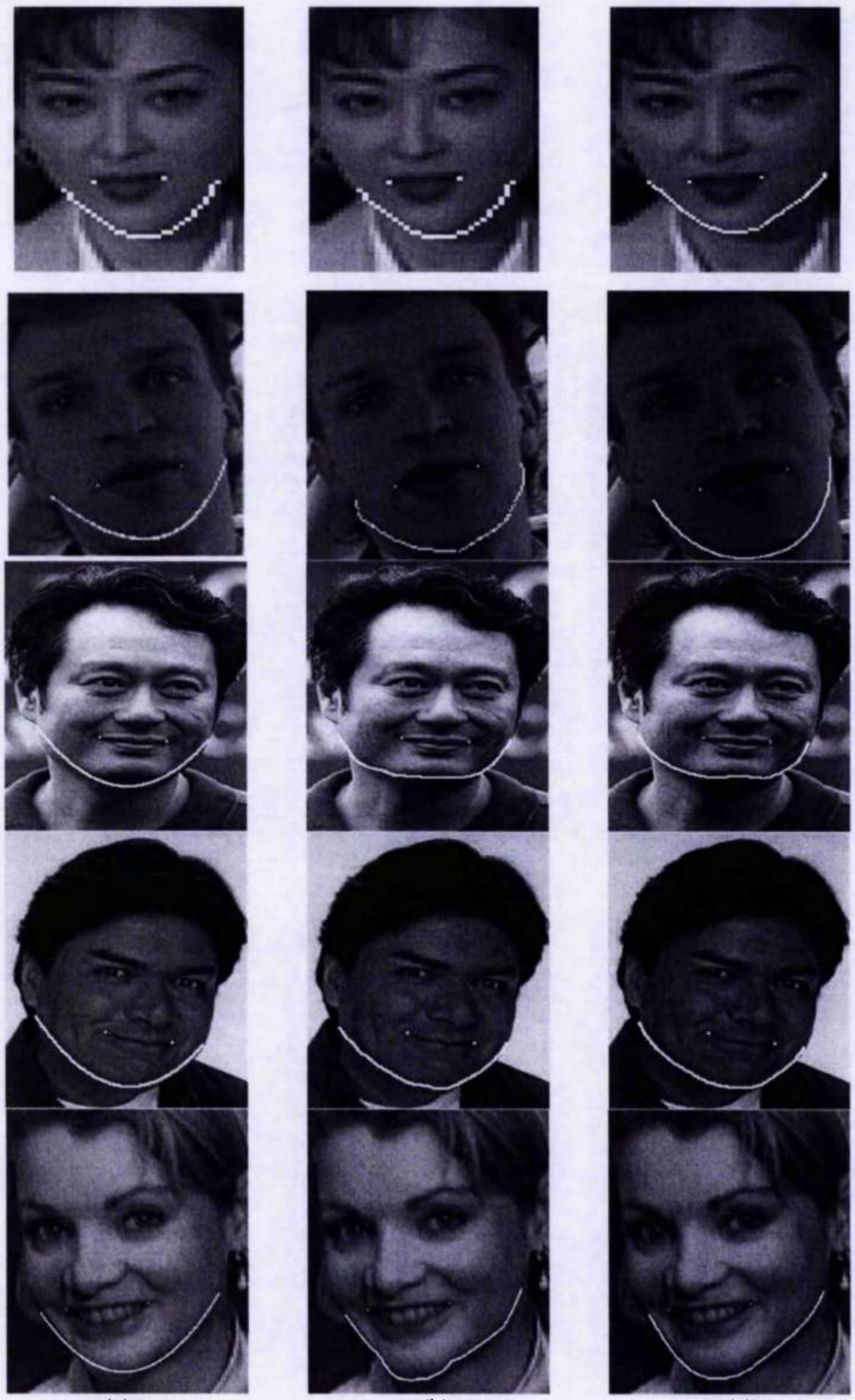


Figure 4.15 – Comparison of chin detection results by using different methods: Column (a): deformable template matching in [YULLE-1992]; Column (b): Our proposed method in [HU-2003] without taking the prior shape into consideration; (c): The proposed chin method in section 4.3.2 incorporating prior chin shape.

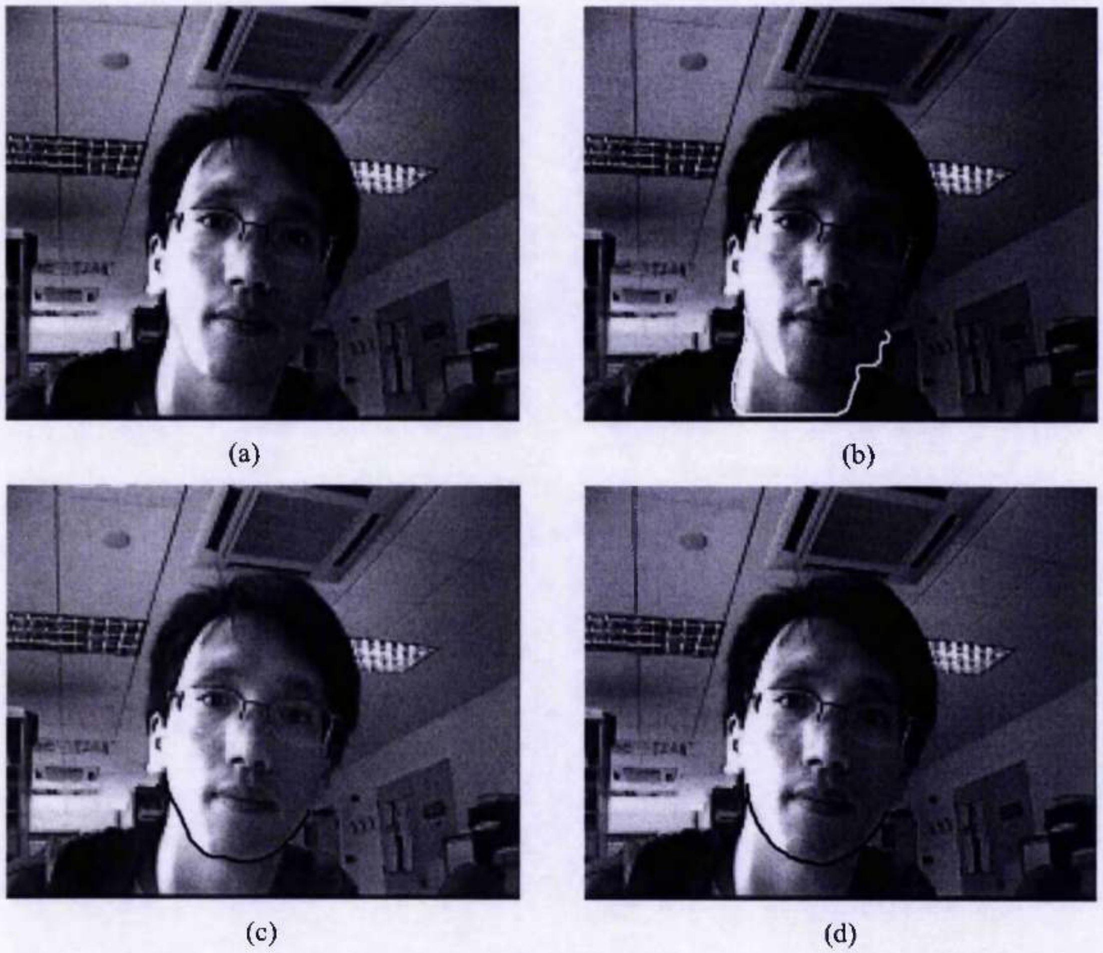
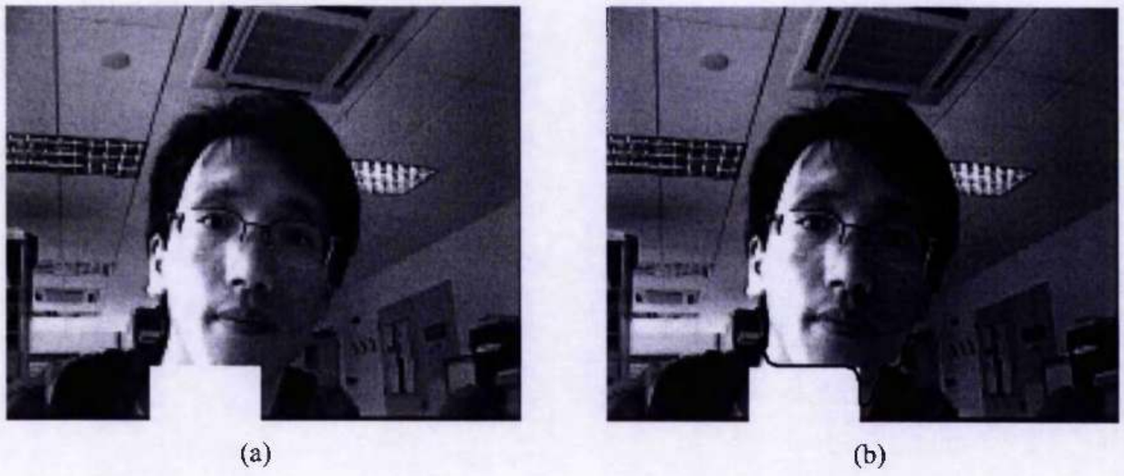


Figure 4.16 – Chin detection results for the image with weak chin edge and cluttered background; (a): Original image; (b): Detection method in [RUDIA-1996]; (c): Our proposed method in [HU-2003] without taking the prior shape into consideration; (d): The proposed chin method in section 4.3.2 incorporating prior chin shape.



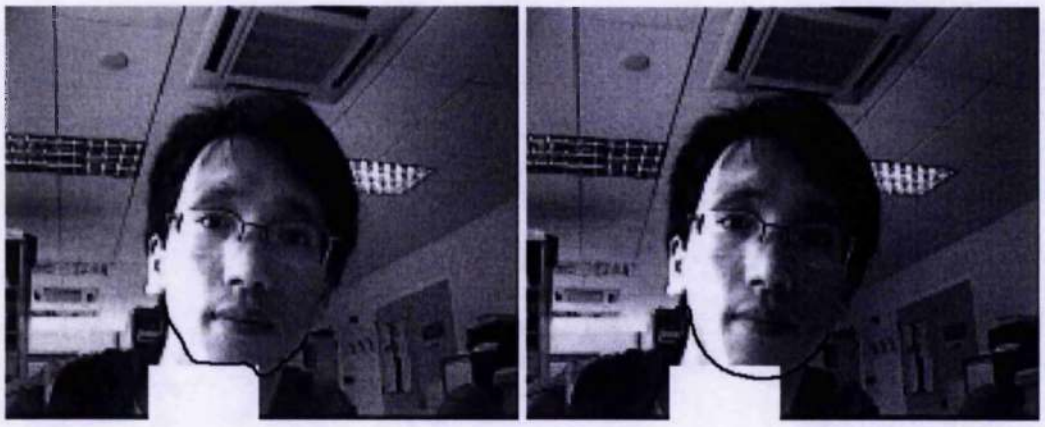


Figure 4.17 – Chin detection results for the image with occlusion; (a): Original image; (b): Detection method in [RUDIA-1996]; (c): Our proposed method in [HU-2003] without taking the prior shape into consideration; (d): The proposed chin method in section 4.3.2 incorporating prior chin shape

4.5.3 Results for scalable face model design and evaluation

Four head-shoulder sequences (QCIF) are used to test the performance of the designed scalable face model for representing face motion through video sequence. They are Carphone, Akiyo, Claire, and Miss_am.

Before designing the scalable model of the foreground head-shoulder objects, we first segment the object into face object and human body object (including hair part). Then, for a human body part, the content-adaptive scalable model is designed, where three levels are chosen. For face objects, the method described in section 4.4 is used to design the scalable face models. They are combined to achieve three-level representation of the foreground head-shoulder object.

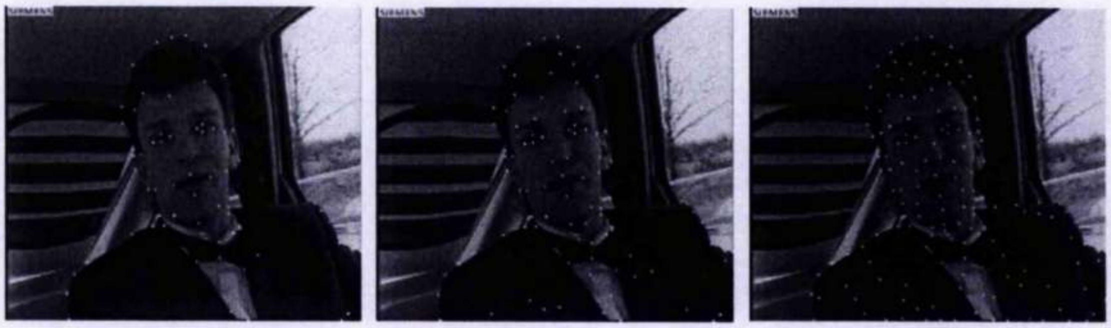


Figure 4.18 – Scalable object models (three levels) for Carphone sequence, the number of control points are 63, 37, 50 for (a) Level 0, (b) Level 1, (c) Level 2, respectively

Figure 4.18 and Figure 4.19 demonstrate the designed scalable models of head-shoulder objects for Carphone sequence and Akiyo sequence. TABLE 4.2 lists the number of control points in different levels for Carphone, Akiyo, Claire, and Miss_am sequences.

Table 4.2 – The number of control points for different levels

	Carphone	Akiyo	Miss_am	Claire
Level 0	63	54	42	40
Level 1	37	35	30	30
Level 2	50	51	48	40

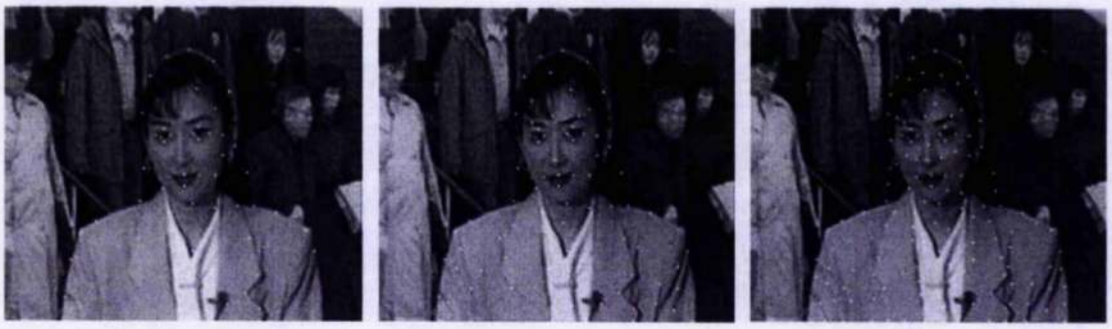

(a) (b) (c)

Figure 4.19 – Scalable object models (three levels) for Akiyo sequence, the number of control points are 54, 35, 51 for (a) Level 0, (b) Level 1, (c) Level 2, respectively

Table 4.3 – Average warping PSNR values (dB) for different levels of representation

	Average PSNR Value (dB)		
	Level 0	Level 1	Level 2
Claire (QCIF)	31.46	34.27	39.02
Miss_am (QCIF)	33.14	36.57	40.23
Carphone (QCIF)	32.82	34.39	35.78
Akiyo (QCIF)	31.25	32.91	35.91
PSNR of Akiyo (QCIF) using method in [BEEK-1999]	28.64	29.27	30.56

In order to test the performance of scalable face model, for every video sequence, four frames (frame 2, 4, 6 and 8) are warped from frame 0 based on the designed scalable model. The motion vectors (MVs) of the control points are estimated by using our proposed method with ¼ - pixel resolution that will be discussed in section 5.3.2 of Chapter 5 in details. Then, average PSNR values of every level are calculated for the warped four frames (frame 2, 4, 6, and 8) with reference to their corresponding original frames. During the PSNR calculation, only the intersection of the warped and original VOP alpha-plane regions is considered. TABLE 4.3 lists the PSNR values of Carphone, Claire, Miss_am and Akiyo sequences. Compared with the results in [BEEK-1999], for Akiyo sequence, the proposed method can achieve about 2-5 dB improvements. It shows that the designed scalable face models can represent the object motion more precisely than the published methods.

4.6 Conclusions

In this chapter, facial feature detection and scalable face model design techniques are investigated for achieving scalable 2D model-based video coding. First, a luminance-adaptive skin colour model is proposed, which is robust to different lighting conditions. A reliable and efficient face localisation and facial feature extraction scheme has also been proposed. These methods can achieve precise and reliable eye, and mouth detection. As the chin contour is one of the most important face features for designing 2D scalable face models, it has been intensively studied during our research. An efficient chin detection scheme has been presented to estimate the chin contour of the human face. After face localization and mouth corner detection, deformable template matching method is used to detect the rough chin position. This is in turn used as the initialization of the active snake model for contour refinement. In order to improve its performance, prior chin shape is trained and incorporated into the active snake model. During external energy calculation for snake model, Canny edge detector is first applied to detect weak chin edges by choosing proper parameters. Then, the GVF of a binary edge map is used as the external force in order to improve the convergence of the active snake model and increase robustness in the case of weak chin edges. For face detection, the proposed algorithm in section 4.3.1 is robust to different skin colour and luminance. For chin detection, some comparisons with other published detection methods have been presented to show the robustness of the proposed method in section 4.3.2 to weak chin edges and partial occlusion.

After extracting the facial features, a heuristic scalable face model is designed based on face muscular distributions and the detected facial features. In order to evaluate the designed scalable face model, a novel motion estimation scheme is proposed which can estimate the model motion precisely although some points are allocated on the textureless region, such as the part of human face. A thorough experimental study has been conducted to show the efficiency of the proposed method in Section 4.4 for scalable face model design. Experimental results show that the designed scalable model can represent the face motion more precisely than previously published techniques and the proposed algorithms can achieve automatic facial model design.

Chapter 5

Scalable Object Modelling and Model Compression

MPEG-4 is an object-based multimedia compression standard which allows encoding different audio-visual objects (AVO's) separately [MPEG4-2001]. These AVO's are decoded and then composited at the user terminal according to a transmitted scene description script. For example, each video object is modelled by its shape (includes 2D and 3D), motion and texture (colour). For shape modelling of video object, 2D mesh modelling can be considered as a projection of a 3D polygon mesh model onto the image plane by perspective projection. Moreover, implicit in 2D object mesh models is a compact representation of the shape of each VOP. This is given by the polygonal boundary of the mesh, which is named as 2D vertex-based shape description. 2D mesh representation of video objects enables the following functions:

1. Video object compression

   Mesh model may improve coding efficiency visually. It provides better motion compensation than translational-block models, and results in fewer blocking artefacts at low bit rates.

2. Video object manipulation

   It includes augmented reality, spatio-temporal interpolation and synthetic-object transfiguration/animation. For example, it can enable the replacement of a natural video object in a video clip by another video object.

3. Content-based video indexing

   It can provide accurate information that can be used to retrieve visual objects with specific motion. It can also provide vertex-based object shape representation, which is more efficient than the bitmap representation used for shape-based object retrieval in MPEG-7 [MPEG7-2002].

In 2-D model-based video coding, the information about model (such as object shape and content-adaptive object mesh model) should be included in the encoded bitstream and sent to the decoder, which is the difference between the model-based / object-based coding scheme and the conventional frame-based coding schemes. Commonly, the cost of model compression is high, especially at low bit rate environment. On the other hand, object model, especially object contour, contains important and sensitive visual information to human eyes. Object shape should be represented precisely, encoded efficiently, and transmitted robustly. A large number of techniques have been proposed in order to transmit the object contour efficiently [CONNELL-1997] [GERKIN-1997] [CHUNG-2000] [JONG-2000].

For 2D mesh modelling of video objects, it is commonly possible to model the shape and motion in a unified framework. However, during our research, object shape and interior model are treated separately. That is, we separate the vertices of the object mesh model into two parts: vertices for shape and vertices for interior object (or vertices of the interior object), and different representation and compression schemes are proposed for them. The reasons for separating the vertices of the object mesh model into two parts are:

- Object shape contains more important and visually sensitive information. More protection is required when sending it over error-prone channels.
- Object shape information has other important functionalities, such as video object index and retrieval. Above separation can facilitate further manipulation;

The objective of this Chapter is to investigate scalable object modelling and model compression. In section 5.1, the well known lossy and lossless shape representation schemes are reviewed, followed by the discussion of main model design and coding techniques that include content-adaptive mesh model design, block-based shape coding, contour-based shape coding, and scalable shape compression. In section 5.2, scalable shape representation and compression (both intra and inter) are investigated. Both intra- and predictive scalable shape-coding algorithms have been proposed during our investigation to improve the coding efficiency of object shape. Section 5.3 gives a detailed description of the proposed schemes for object mesh design (interior object) and compression. For the vertices of the interior object, a coarse-to-fine strategy is used to allocate the control points due to its simple implementation at no loss of accuracy. The most important proposal on this is trying to make the mesh edge conform to the object boundary. In this way, the warping error can be reduced and object motion can be estimated precisely [NAKAYA-1994].

Extensive experiments have been conducted and some results are illustrated in section 5.4, followed by conclusions of this Chapter.

5.1 Overview

The emergence of new multimedia applications, such as searching, indexing and manipulation of visual information at the semantic object level, requires further research on video representation and coding. In past decades, much research has been conducted on content-adaptive mesh design, shape representation and compression, and some are standardized in MPEG-4 [MPEG4-2001].

There are two kinds of mesh applied to represent the object motion: *regular mesh* and *content-adaptive mesh*. As regular meshes can be setup at both the encoder and decoder without geometry overhead, it is sufficient to transmit only node motion vectors. However, regular meshes cannot adapt the mesh structure based on the content of the video object to represent the motion more accurately. Content-adaptive meshes can overcome this drawback with the cost of transmitting initial mesh geometry [NAKAYA-1994] [WANG-1994a] [HUANG-1994] [ECKERT-1997]. Several techniques have been discussed on how to design content-adaptive object meshes [ALTUNB-1997], which are reviewed in section 5.1.1.

In recent years, significant research has been performed on shape coding in the framework of MPEG-4 standardization activities. Shape also plays an important role in image database search and retrieval applications, which are addressed by MPEG-7 standard [MPEG7-2002]. A large number of shape coding algorithms have been proposed [BRADY-1997] [ETOH–1997] [YAMAG-1997] [CONNELL-1997] [GERKIN-1997] [CHUNG-2000]. These shape coding algorithms can be classified into two categories: block-based and contour-based [KATSAGGELOS-1998], which are reviewed in section 5.1.3 and 5.1.4 respectively.

Furthermore, *scalable* shape representation and compression is an important requirement of new applications. For example, one of the key requirements of MPEG-7 applications is to perform very fast shape filtering and browsing through rough shape reconstruction, but also has the ability to perform full resolution shape rendering. Scalable shape representation and compression lend themselves naturally to these requirements. Scalable shape compression also facilitates shape transmission over error-prone channels or the channels with variable bandwidth. Many scalable shape-coding methods have also been proposed [QIAN-1997] [JORDAN-1998] [MELNIKOV-2000a] [MELNIKOV-2000b], which will be reviewed in detail in Section 5.1.4.

5.1.1 Content-adaptive mesh model design and coding

Y. Nakaya et al [NAKAYA-1994] proposed a hexagonal matching procedure for motion compensation based on a uniform mesh. Uniform meshes are suitable for "motion compensation by redesign", that is, a new uniform mesh is overlaid on each frame k and motion vectors of the object model are estimated from frame k to $k-1$ for motion compensation. Unfortunately, uniform meshes are often inadequate for representing the motion near object boundaries, where a patch may contain two or more different motions. This problem may be addressed by splitting these patches with more than one motion into smaller patches [HUANG-1994], resulting in a hierarchical mesh. Information about this splitting must be transmitted as overhead.

A more fundamental approach to overcome the problems of uniform mesh elements is to design a content-adaptive mesh model. Wang et al [WANG-1994a] proposed an optimization framework for motion compensation based on an active mesh that adapts to scene content. However, this content-adaptive mesh is not suitable for motion compensation by redesign because transmission of all node locations at each frame constitutes an excessive amount of overhead. Furthermore, these mesh models enforce connectivity of the structure everywhere, which imposes a global smoothness constraint on the 2-D motion field and is unsuitable for motion compensation across the motion and occlusion boundaries.

Y. Altunbasak et al [ALTUNB-1997] proposed an occlusion-adaptive mesh model to solve this problem. Occlusion regions are classified as Background to Be Covered (BTBC) and Uncovered Background (UB). No node points are allowed in the BTBC regions and the meshes within the failure regions are redefined for subsequent tracking. The success of forward tracking is closely related to how well it can detect occlusion and model failure regions together with the motion estimation near their boundaries. In motion compensation by forward tracking, positions of all nodes need to be transmitted only for selected key frames. For other frames, it is sufficient to transmit the boundaries of the BTBC regions and the locations of the newly added nodes in the mesh model.

In order to reduce the warping errors occurring over object borders, M. Eckert, et al [ECKERT-1997], proposed an object-based motion compensation scheme. In this scheme, all border nodes are assigned to the adjacent regions and every region is triangulated individually. The single object meshes are then connected to a complete mesh over the whole image, which covers all region contours with triangular edges. During motion estimation, the mesh has to be split object by object at the points where motion discontinuities occur. The particular objects are transformed individually. Uncovered parts can be detected during the process of motion compensation. In paper [IZQUIER-1999], E. Izquierdo proposed a 3-D modeling method for arbitrary objects based on geometric surface conformity. The proposed technique uses feature points and relevant edges in the images as the node and edges of an initial 2-D wire grid. Starting from this initial 2-D model, the 3-D wireframe is generated by fitting the 2-D model to a previously recovered depth map of the object. The 3-D wireframe is deformed and updated from frame to frame according to the motion of the chosen nodes.

In recent years, hierarchical representation of 2-D dynamic meshes has attracted attention. It provides rendering at various levels of detail. It not only allows scalable transmission of the object geometry and motion information, but also enables improved tracking performance. Detailed algorithms on hierarchical content-adaptive model design and update are discussed in [BEEK-1999] [CELASUM-2000]. In the design algorithm of paper [BEEK-1999], a finely detailed 2-D mesh is designed for the initial video object plane. Then, a hierarchical representation is constructed by simplifying this mesh from fine to coarse levels. During the finely detailed 2-D mesh design, nodes are placed at salient intensity corner points on the boundary and interior of the video object plane. The initial mesh topology is constructed using constrained Delaunay triangulation of the node points, where the boundary edge segments serve as constraints in order to confine the resulting triangles within the boundary polygon. During the simplification of the fine mesh, an independent set of nodes from the finer level is removed to obtain the coarser level. It is naturally desirable to eliminate less "important" nodes first in the mesh simplification process, such that essential mesh features of the mesh geometry are retained. Some complex image-, shape-, and motion-based criteria are proposed in [BEEK-1999] [CELASUM-2000] for determining the importance of a node adaptively. Dynamic programming is used to optimize the mesh. These methods share the disadvantages of high computational complexity. Furthermore, the edge of triangles does not guarantee to conform to the object boundary and interior edges.

5.1.2 Fourier descriptor of object shape

In the Fourier descriptor of object contour, the object contour is considered as a set of ordered points defined on the complex plane [SALE-1996]. The coordinates of a closed contour can be seen as a periodic complex sequence. Fourier descriptors are first calculated as the Fourier series coefficients of this complex sequence; then the Fourier descriptors are encoded. In order to get an accurate reconstructed contour, the bitrate needed to accurately encode these Fourier descriptors is usually high. Therefore, the coding efficiency of the Fourier descriptors approach is not enough for high quality contour coding.

5.1.3 Block-based shape coding – CAE

In block-based shape coding, object shape is represented by a binary image mask and context-based arithmetic encoding (CAE) [BRADY-1997] is used to coding the mask. CAE is one of the most successful methods for binary image coding and is applied in JBIG standard [JBIG-1993]. It is also applied successfully in binary shape coding, and is a block-based shape coding scheme. In CAE [BRADY-1997] [ETOH–1997] [YAMAG-1997], it is assumed that a high degree of local correlation exists in the shape image. Each pixel is encoded according to a conditional probability distribution that is conditional upon its context – the value of pixels in a local neighbourhood. The neighbourhood's shape and size are represented by a template, and this context is used to access a table containing probability distributions. The table is created by a training procedure prior to coding. It also can be adapted during the coding procedure in the case of the adaptive CAE. The widely used templates for the Intra- and Inter-mode coding are shown in Figure 5.1.

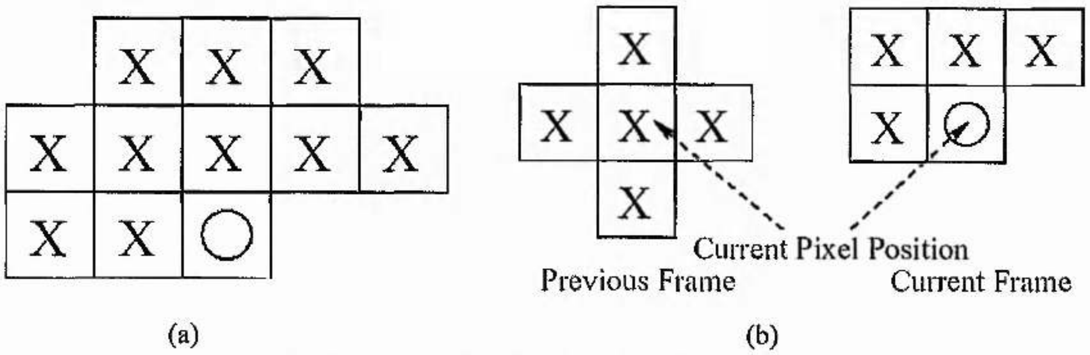


Figure 5.1 – Templates used in context-based arithmetic coding. (a) Template for intra mode; (b) Template for inter mode

The main feature of block-based shape coding is its superior coding efficiency, while bearing a relatively low complexity. It is also well adapted for low delay applications. For video coding applications, it has been extended to achieve block-based coding and temporal prediction, as described in [BRADY-1997]. The CAE has been adopted and well integrated into the current MPEG-4 standard. However, the block size conversion in the MPEG-4 shape-coding scheme, which applies the CAE technique, shows a visually annoying staircase effect [BRADY-1997].

5.1.4 Contour-based shape coding

Object shape can also be represented by its contours. Contour-based shape coding schemes perform compression along the outlines of a segmented object, so that each contour can be endowed with dedicated semantics to describe the object in a contour-by-contour manner. That is of interest in applications where a high level, semantic representation is needed. In the past decades, many contour-based shape-coding methods have been proposed [FREEMAN-1961] [CONNELL-1997] [GERKIN-1997] [CHUNG-2000] [KANEKO-1985] [LU-1991] [LEE-1999] [CHO-1999] [JONG-2000]. These methods can be further classified into lossless coding [FREEMAN-1961] [KANEKO-1985] [LU-1991] and lossy coding schemes [CONNELL-1997] [GERKIN-1997] [CHUNG-2000] [CHO-1999] [JONG-2000].

For lossless contour coding, chain coding is one of the most frequently used methods [FREEMAN-1961] [KANEKO-1985] [LU-1991]. In the chain coding method, the contour information is encoded pixel by pixel. From the starting point, the directional vectors between successive contour pixels are encoded. Since Freeman's chain code was introduced in 1961 [FREEMAN-1961], improvements in chain code representation have yielded several simple lossless contour compression methods [KANEKO-1985] [LU-1991]. In method [LU-1991], Lu and Dunham developed chain coding schemes using higher-order Markov models combined with arithmetic coding, which offer 50% and 25% coding gains over Freeman's chain codes and differential chain coding, respectively. The codes are close to the theoretical upper bound on the compression ratio for lossless differential chain coding. Lossless schemes, however, are not sufficiently flexible to allow the control of shape bit-rate in order to negotiate service quality with available bandwidth. Therefore, lossy shape coding is needed.

In order to achieve lossy shape coding schemes, the object contour should be approximated by an ordered set of vertices properly selected from a given contour. Only the positions of the selected vertices need to be compressed. Therefore, the lossy shape coding methods mainly consist of three steps: vertex selection, vertex encoding, and approximation reconstruction. There are many vertex selection algorithms for computer vision and pattern recognition [ANSARI-1991] [DUNHAM-1986]. The iterated refinement method (IRM) [GERKIN-1997] has been widely used because it can be easily implemented and makes control feasible. Recently, optimized vertex selection methods [KATSAGGELOS-1998] [SCHUSTER-1998] have been proposed. They give optimal vertices in the rate-distortion sense but also result in high computational complexity.

For better coding efficiency, various vertex encoding schemes have been proposed. In [CONNELL-1997], O'Connell presented an object-adaptive vertex-encoding (OAVE) scheme, which adjusts the dynamic range of the relative addresses for each contour and use an octant representation for each vertex address. This idea extends chain coding for lossy shape coding. Experimental results show that OAVE combined with adaptive arithmetic coding for encoding a composite relative address can achieve better compression performance than CAE [CONNELL-

1996]. In 2000, Jae-won Chung introduced a new binary shape coder for high coding efficiency, in which 1D vertex detection, vertex reordering and initial vertex encoding methods are included [CHUNG-2000].

In [LEE-1999], Lee, etc., proposed a baseline-based shape coding method for both lossless and lossy contour coding. In this method, a baseline is first chosen for a given shape image such that the projection of the shape onto the x-axis is the longest. Then, the distance from each contour point to this baseline is extracted as well as the turning points. The distance data are encoded by an entropy coder. In the lossy mode, the distance data are subsampled first and then are entropy encoded. On the decoder size, the reconstructed data are interpolated to get the whole contour.

In [KATSAGGELOS-1998] and [SCHUSTER-1998], a framework for the rate-distortion operationally optimal encoding of shape information in the intra mode is proposed. First order (polygons) and higher order (i.e. splines) approximation techniques are adopted to represent the boundary, and the control points of these curves are encoded to achieve the R-D optimised result. For these techniques, one of the disadvantages is their computational complexity during R-D optimisation for vertex selection.

Wang, etc., recently proposed an efficient rate-distortion optimal shape-coding scheme utilising a skeleton-based decomposition [WANG-2003]. The approach decouples the shape information into independent signal data sets: the skeleton and the boundary distance from the skeleton. The major benefit of this approach is that it allows for a flexible tradeoff between approximation error and bit budget. Experimental results in this paper demonstrate that the proposed algorithms result in a significant improvement in rate-distortion efficiency with respect to other rate-distortion optimised shape encoders. However, this method has the same disadvantage as those in [KATSAGGELOS-1998] [SCHUSTER-1998] due to its high computational complexity during R-D optimisation.

Temporal information has also been exploited in many shape-coding methods, such as [GERKIN-1997] [GU-1995] [CHO-1999] [MELNIKOV-1999] [JONG-2000], to achieve higher coding efficiency. Gerkin demonstrated an improved vertex-based coding scheme using a vertex-list-update predictive coding algorithm [GERKIN-1997]. As the vertex prediction performance is dependent on vertex selection, and the size of the list update information is significant for a large number of vertices, the performance of this method is not very satisfying. Gu [GU-1995] proposed the predictive shape-coding technique, exploiting the temporal correlation. In this technique, only contours yielding relatively large motion-failure (MF) regions are encoded for transmission, in which the lossless chain coding technique is employed for encoding the contour.

However, the MF regions are distributed in a sparse and isolated manner, requiring large overheads to represent each isolated contour using chain codes. Sung Ho Cho, et. al [CHO-1999] proposed a technique based on the segment-based chain-coding scheme. First, a two-stage motion compensation technique is present in order to cope with complex motion. Furthermore, by defining the error band, the method can be applied for lossy encoding, by which the bits required for the contours can be adjusted according to the channel condition. Jong Il Kim, et. al [JONG-2000], introduced a generalized predictive shape coding (GPSC) scheme, which improves the performance of the method in [CHO-1999] by introducing a 1-D reference index-based coding scheme.

Although the methods in [CHO-1999] and [JONG-2000] can improve the contour coding efficiency by incorporating the temporal information during coding, they cannot achieve scalable shape coding that is desirable for multimedia networks and devices with different bandwidths and available decoding powers. Furthermore, during motion estimation, the contour motion is assumed as translational motion. This assumption does not work well when there is zooming and/or rotation. For non-rigid object motion, different motion patterns exist for different contour segments. It is difficult to use one motion model to describe them.

5.1.5 Scalable shape coding

In multimedia networks, devices with different bandwidths and available decoding powers are interconnected. Bitstream scalability is desirable, so that simple decoding of the first bits results in a coarse shape approximation that may be further refined. Scalable representation and coding is also desirable for new functionalities such as indexing and retrieval of the shape information. Many scalable shape-coding methods have been published in the past decades [CONNELL-1996] [QIAN-1997] [JORDAN-1998] [MELNIKOV-2000a] [MELNIKOV-2000b]. These schemes can be considered as extensions of block-based shape coding and contour-based shape coding.

For block-based shape coding techniques, in order to achieve scalability, the binary map is first decomposed into several layers of different resolution. The basic layer, that is the layer with the lowest resolution, is coded using the classical non-scalable technique. The enhancement layers are then encoded in a similar fashion but using a different template [CONNELL-1996] [QIAN-1997].

For contour-based shape coding, the object contour should be represented progressively in order to achieve scalable coding. First, a coarse polygon approximation is built. The resulting vertices corresponding to salient points along the contour are encoded by any existing vertex-based coding method so that the decoder can rapidly access them for fast browsing/retrieval. Complementary lossless representation for final rendering is achieved by successively transmitting the polygonal approximation refinements. The insertion orders of the refinement vertices, as well as their positions, are encoded relative to the coarser polygon edges.

Several methods have been published for progressive representation of object contours [GERKIN-1997] [JORDAN-1998] [MELNIKOV-2000a] [MELNIKOV-2000b]. The most popular method is the iterated refinement method (IRM) in [GERKIN-1997] due to its simplicity. However, this method can only achieve sub-optimal vertex selection. In order to improve its performance, some hybrid schemes have been proposed by using the information of contour geometry [JORDAN-1998], vertex adjustment method [CHUNG-2000], and rate-distortion optimization scheme [MELNIKOV-2000a] [MELNIKOV-2000b].

Although much research has focused on scalable shape coding in recent years, the experimental results show that these methods cannot achieve higher compression efficiency for (near-) lossless shape coding and they lack optimality in both intra and inter modes of the operation [KATSAGGELOS-1998]. Therefore, further research is still necessary for scalable shape coding.

5.2 Proposal for scalable shape representation and coding

In our research, scalable vertex-based shape coding schemes (both intra- and inter-coding) are studied to achieve higher coding efficiency. For scalable vertex-based shape coding, the efforts include:

- First, an optimal vertex selection scheme is proposed, which can achieve less approximation vertex number as shown by the results in section 5.4.2.1. Furthermore, the most important vertices, which correspond to the salient feature of object shape, are included in the coarser layers to facilitate shape manipulations, such as shape retrieval.

- Second, an efficient vertex-based intra-encoding scheme is proposed. The information of the transmitted coarser layers is exploited to improve coding efficiency of current layer.

- Third, a scalable predictive shape-coding scheme is presented. In this scheme, object contours are effectively compressed with the aid of temporal information. The proposed intra and predictive coding schemes offer the content-based shape description together with the quality scalable representation.

5.2.1 Scalable shape representation

In recent years, many vertex selection algorithms have been proposed, such as [GERKIN-1997] [JORDAN-1998] [MELNIKOV-2000b]. The iterative refinement method (IRM) [GERKIN-1997] has been widely used as it can be easily implemented. However, this method cannot find the optimal position of the approximating vertices. In paper [JORDAN-1998], a digital polygon approximation was presented, taking the intrinsic image grid quantisation into consideration. Recently, an R-D optimised vertex selection method [MELNIKOV-2000b] has been proposed. However, this method has high computational complexity.

In our research, a new vertex selection scheme is proposed to approximate an object contour progressively. During approximation, the vertices are classified into several layers according to the selected error bands. During our experiments, 4 layers are used for QCIF sequences. Layer 0, 1, 2, and 3 have the corresponding desired error band $d_{\max}=4$, $d_{\max}=2$, $d_{\max}=1$, $d_{\max}=0$ respectively. These selections are based on the research results in [JORDAN-1998], which show that:

- It does not seem useful to encode more than 4 layers;
- Lossy shape coding should be limited to small distortions for video coding.

The detailed vertex-selection algorithm can be summarised as follows:

- Curvature scale space (CSS) image [MOKHT-1992] has been exploited during vertex selection for layer 0. The algorithm for CSS image calculation will be discussed in Section 5.2.3.1.

  Figure 5.2 shows a video object contour and its traced CSS image. It is found that the traced CSS image carries the most important feature of object contours and can detect these salient features easily. The vertices of layer 0 should include the salient points of the object contour, which can feature contour efficiently. These vertices should be first encoded and decoded, as they are very important for shape retrieving and matching. However, for contours with small curvature, the method based on curvature cannot guarantee that the contour approximation satisfies the predefined error band. Therefore, the IRM method is used after curvature-based vertex selection is employed.

- For other refinement layers, IRM, together with a novel merging scheme, has been exploited. Each approximating polygon edge of the coarser layers is recursively split by introducing a new vertex at the contour point with the largest distance, until the desired accuracy $d < d_{\max}$ is reached. The intrinsic image grid quantisation is taken into account during the approximation.

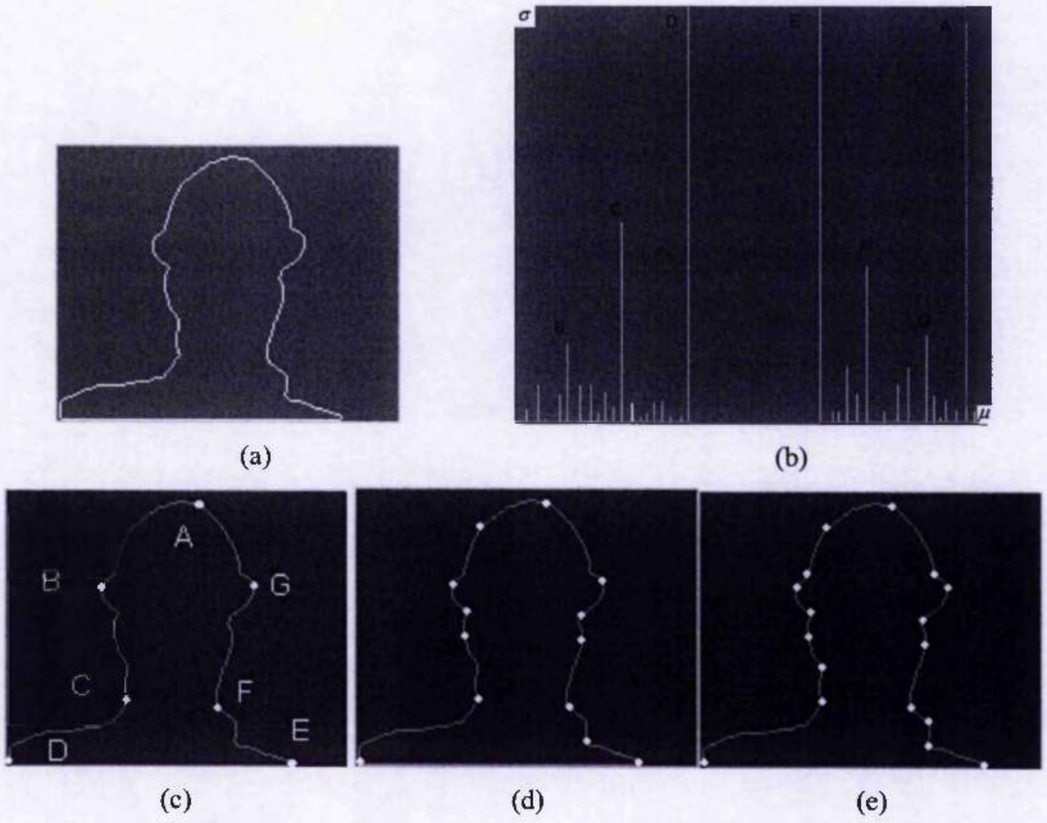


Figure 5.2 – The correspondence between (b) the traced CSS image and (c) the salient features ($\sigma \geq 80$) for original contour (a). (d) and (e) are the salient features for $\sigma \geq 40$ and $\sigma \geq 30$.

In order to reduce the approximation vertices and encoding bit-rate, a new merging algorithm is proposed and summarized as follows:

1 Suppose, at layer $i \in \{1,2,3\}$, the contour has been divided into $\sum_{k=0}^{i-1} N_k$ segments. Along every segment, if the vertex number of layer i is 2 or more, these vertices will be evaluated. If the vertex is removed and the polygon approximation still satisfies the error condition, remove it. Otherwise, keep it.

2 If $d_{\max} = 0$, and, for every segment $\overline{p_1 p_2}$, the vertex number of layer i is $n_j \geq 2$, arrange these vertices along the contour in array $pt[]$, including two terminals. Then:

```
For (k=0; k< $n_j - 1$; k++) {
    For (kk= $pt[k]$; kk< $pt[k+1]$; kk++) {
        If (kk == $pt[k]$)
            If we can find $m_j < n_j - k$ points to approximate segment $\overline{kk, p_2}$,
            recode the points and number;
        Else
            If we can find $m_j < n_j - k - 1$ points to approximate segment
            $\overline{kk, p_2}$, recode the point positions and number;
    }
}
If (no such point sets are found) {No merge is needed;}
Else {Replace the vertices of layer $i$ with the point set with the smallest number.}
```

Figure 5.3 illustrates an example for the lossless approximation, where (a) is achieved by the methods published in [GERKIN-1997] and [JORDAN-1998] (both achieve the same results); and (b) is achieved by the proposed method. For the approximation of the contour segment between $X1$ and $X2$, four points are required by using the method [GERKIN-1997] and [JORDAN-1998]. While using our proposed method, only one point is chosen. More experimental results in Section 5.4 show that our proposed method requires fewer vertices to approximate the object contour, especially for (near-) lossless approximation.

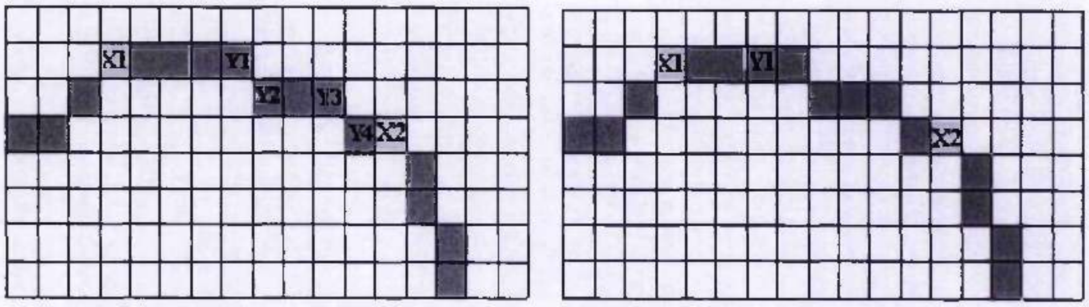


Figure 5.3 – Comparison of different lossless approximation methods. (a) Approximation result using IRM method only [GERKIN-1997]; (b) Approximation result using our proposed method

5.2.2 Scalable intra-shape coding

Figure 5.4 shows the proposed scalable intra-shape coding scheme. For scalable vertex-based shape coding, there are two kinds of information to be encoded: contour configuration and contour location. Contour configuration is represented by an ordered set of vertices that can be used by the decoder to correctly produce the ordered list of vertices. The contour location is represented by the coordinates (x, y) of the vertices. In the proposed intra-shape coding scheme, the scalable vertex encoding consists of:

- The encoding of the vertices of layer 0;
- The encoding of vertex connectivity of the refinement layers;
- The position encoding of the refinement layers;

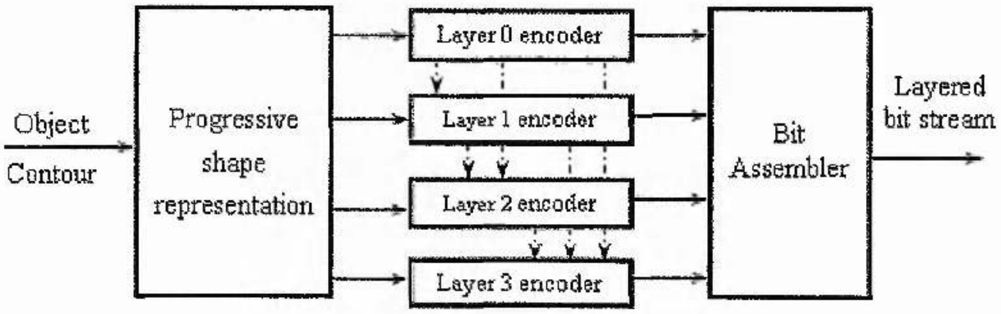


Figure 5.4 – Scheme illustration of scalable intra-shape coding scheme

5.2.2.1 Encoding of layer 0

As there are no coarser approximation layers for layer 0, the corresponding vertices have to be encoded directly. The encoding of layer 0 consists of two parts:

- The encoding of the initial vertex;
- The encoding of other vertices;

In our proposed method, a vertex-reordering step is conducted before the initial vertex encoding. As we know, the relative distance between the initial vertex v_0 and the last vertex v_{N_0-1} does not need to be coded. In the object-adaptive vertex-based shape-encoding (OAVE) scheme described in [CONNELL-1997], the bit assignment during encoding is dynamically determined by the maximum relative distance. Therefore, the reordering of vertices can reduce the number of encoded bits.

After the initial vertex is decided, two prediction models have been proposed to encode its position:

- Relative to the origin of the video object plane (VOP) boundary box;
- Relative to the origins of other contours;

During experiments, we found that, if the number of contours of a VOP is high, (b) is more efficient than (a).

In the proposed scheme, two encoding methods have been used to encode the non-initial vertices:

- OAVE encoding scheme in [CONNELL-1997];

  It provides a compact representation when the vertices are closely spaced. Experimental results in [CHUNG-2000] show that OAVE algorithm, combined with adaptive arithmetic coding for encoding a composite relative address, shows good coding performance. However, the OAVE algorithm may be not efficient when the vertices are widely spaced, which usually occurs for the coarser approximating layers [CONNELL-1997].

- Absolute addressing [CHO-1996];

  It can provide a compact representation when an object's vertices are widely spaced (as in a large object approximated by few vertices). It is inefficient when the vertices are closely spaced.

For above two methods, the method generating shorter bit stream is selected. During the encoding of the vertices of layer 0, the coding model selection information is also included in the encoded bit stream and transmitted to the decoder.

5.2.2.2 Vertex-connectivity encoding of refinement layers

For scalable shape coding, the connectivity and the number of child vertices along the coarser polygon edges should be encoded and transmitted to the decoder. In our research, a 2-D symbol

(A, B) is defined to indicate the positions, where new vertices will be added. A is to define the number of edges (or vertices) before a vertex should be added. B is defined as the number of vertices to be added along this approximation edge. For each approximation layer (except layer 0), 2-D symbols are formed and encoding using variable-length coder (VLC).

The following example illustrates the symbol construction. Assume that we plan to encode the vertices along the object contour in Figure 5.5. First, 10 vertices in layer 0 are encoded and the vertex S is chosen as the starting point. For layer 1 and layer 2, the 2-D reference symbols are formed as follows:

(4,1), (1,1), (2,1), (2,2) (Layer 1)

(1,1), (3,1), (3,4), (2,1), (1,1), (1,2), (2,1), (2,2) (Layer 2)

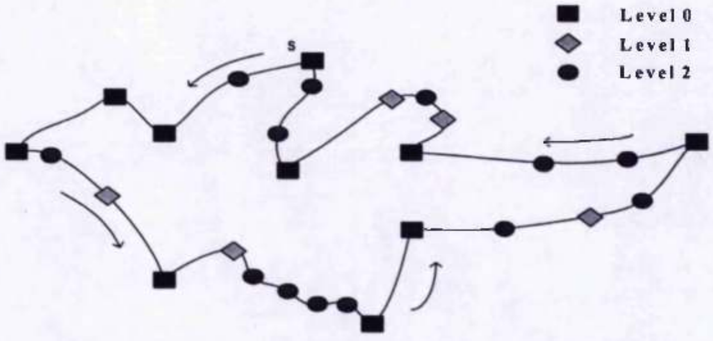
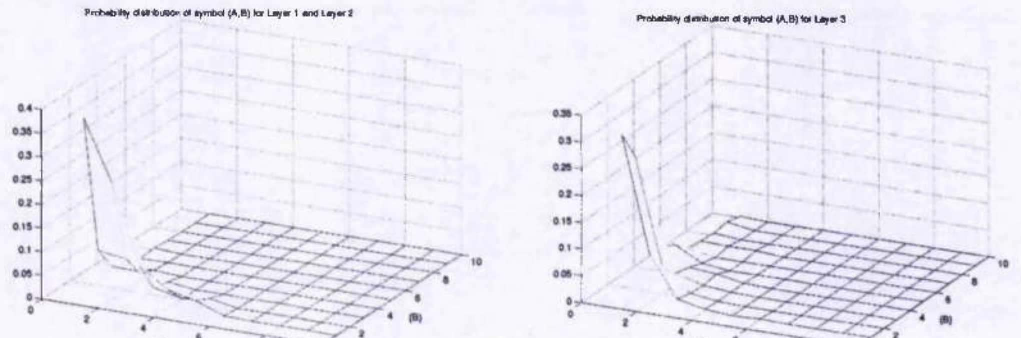


Figure 5.5 – Illustration of vertex-connectivity coding



(a) (b)

Figure 5.6 – Statistical distribution of (A, B) pairs for (a) Layers 1, 2, and (b) Layer 3

During encoding of these symbols, a variable length coder (VLC) has been used. As the number of vertices for every layer has been included in the layer header, the end-of-layer (EOL) information is not needed.

Fifty video objects from QCIF sequences have been used to study the statistics of symbols, which are shown in Figure 5.6. It is found that layers 1 and layer 2 have similar statistics, but are different from these of layer 3. Therefore, two VLC tables are designed. From the data shown in Figure 5.6, it is also found that, for layer 1 and layer 2, $B = 1$ for almost 75% of (A, B) symbols; and for layer 3, $A = 1$ for almost 75% of (A, B) symbols. During vertex position encoding, this has been used to improve the encoding efficiency.

5.2.2.3 Vertex-position encoding of refinement layers

For the vertex position encoding of refinement layers, an improved OAVE encoding algorithm in [CONNELL-1997] has been proposed. In this algorithm, the information from the already encoded coarser layers and pre-defined error band of the current layer is exploited for high encoding efficiency. The encoding process of the current layer includes:

- Determine and encode the dynamic range indicators for x and y components of the current layer;
- Encode the octant numbers of the vertices;
- Encode the major and minor components;

Determine and encode the dynamic range indicators for x and y components

The ability to adapt the vertex representation for all of an object's vertices is provided by indicating the dynamic range of the relative locations [CONNELL-1997]. The dynamic range of the relative locations of the object's vertices of current layer can be determined and indicated in the compressed bitstream by:

- Calculating the relative locations of the vertices $R_i = V_i - V_{i-1}$ for $i = 1, \cdots, N-1$ ($V = \{V_0, V_1, \cdots, V_{N-1}\}$ represents the ordered set of N vertex locations approximating the object contour), after selecting the initial vertex;
- Determining the $x\_\max\_magniture$ and $y\_\max\_magniture$, the maximum absolute values of the x and y components of the relative location R_i. In the proposed method, only the segments containing the newly inserted vertices are used during the determination.

- Selecting and encoding indicators $indicator_x$ and $indicator_y$ based on the Table 1 in [CONNELL-1997]. The selected indicators are then encoded using a 3-bit FLC.

Encode the octant numbers

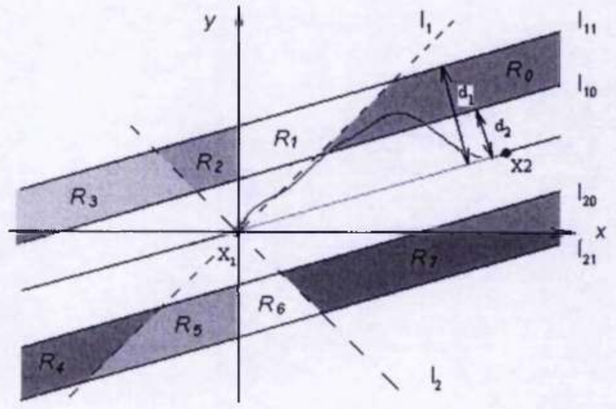
In the proposed method, the determinant of octants of current layer is different from that in [CONNELL-1997]. Figure 5.7 shows the determination of octant number using: (a) the method in [CONNELL-1997]; and (b) the proposed method for *Case 1* that only one vertex is allocated between the segment $\overline{X_1X_2}$. Figure 5.7 (c) shows the octant number determination for *Case 2* that two or more vertices are allocated between the segment $\overline{X_1X_2}$. The main difference of our proposed method with the method in [CONNELL-1997] is the defined area for octant 0, 3, 4, and 7. The proposed method can reduce the dynamic range of x and/or y components, which depends on the octant number. For example, in Figure 5.7 (a), the dynamic range of y component for octant 0 is $y\_\max\_magniture$. In Figure 5.7 (b), its dynamic range of y component is $(d_1 - d_2)*\sqrt{2}$, which is commonly smaller than $y\_\max\_magniture$. The same conclusion can be drawn for the case in Figure 5.7 (c).

According to the vertex number, n_j, of current layer along the approximation edge, two octant determination methods are proposed:

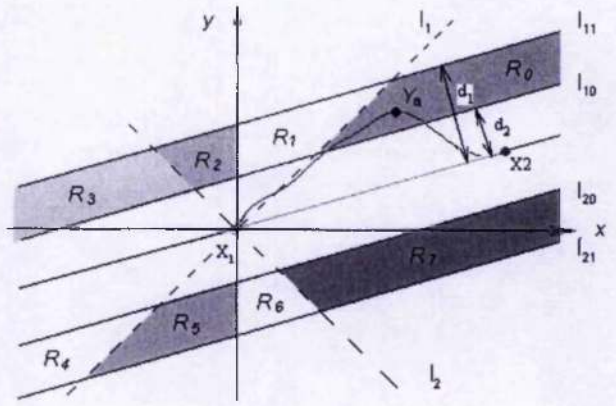
- *Case 1*: If $n_j = 1$, its octant is decided from Figure 5.7 (b). In Figure 5.7 (b), X_1 and X_2 belong to the coarser layers. Now, if one point Y_a belongs to current layer, it must be located in the regions between l_{10} and l_{11} or between l_{20} and l_{21}.

- *Case 2*: If $n_j \geq 2$, their octants are decided by Figure 5.7 (c). These vertices are located in the region between l_{11} and l_{21} in Figure 5.7 (c). For example, in Figure 5.6 (c), the octant of Y_a is based on the relative position between $\overline{X_1Y_a}$ and $\overline{X_1X_2}$. The octant of Y_b is based on the relative position between $\overline{X_1Y_b}$ and $\overline{X_1X_2}$, instead of the relative position between $\overline{Y_aY_b}$ and $\overline{X_1X_2}$.

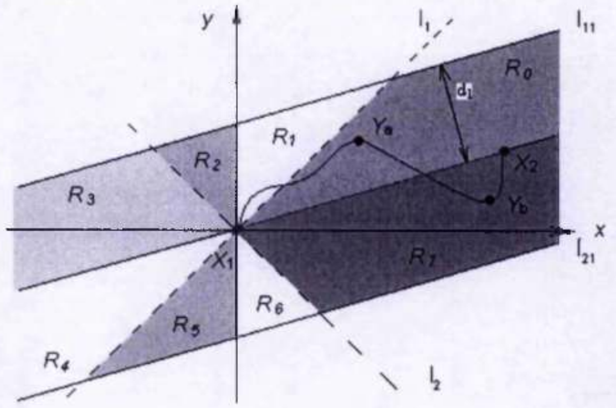
The regions corresponding to octant from 0 to 7 are separated by l_1 and l_2, as indicated in Figure 5.7 (b) and (c) from R_0 to R_7 respectively. After deciding the octants, they are encoded by using conditional differential chain coding (CDCC). The differential octant is obtained by the difference between neighbouring octant values. For example, in Figure 5.7 (b), the differential octant of Y_a is decided by the octant of Y_a and the octant of X_1. In Figure 5.7 (c), the differential octant of Y_b is decided by the octant of Y_a and the octant of Y_b.



(a)



(b)



(c)

Figure 5.7 – Octant number determination: (a) using the method in [CONNELL-1997]; (b) using the proposed method for *Case 1*; (c) using the proposed method for *Case 2*

Encode the major and minor component

The vertex number along the approximation edges also determines the encoding methods for the vertex position.

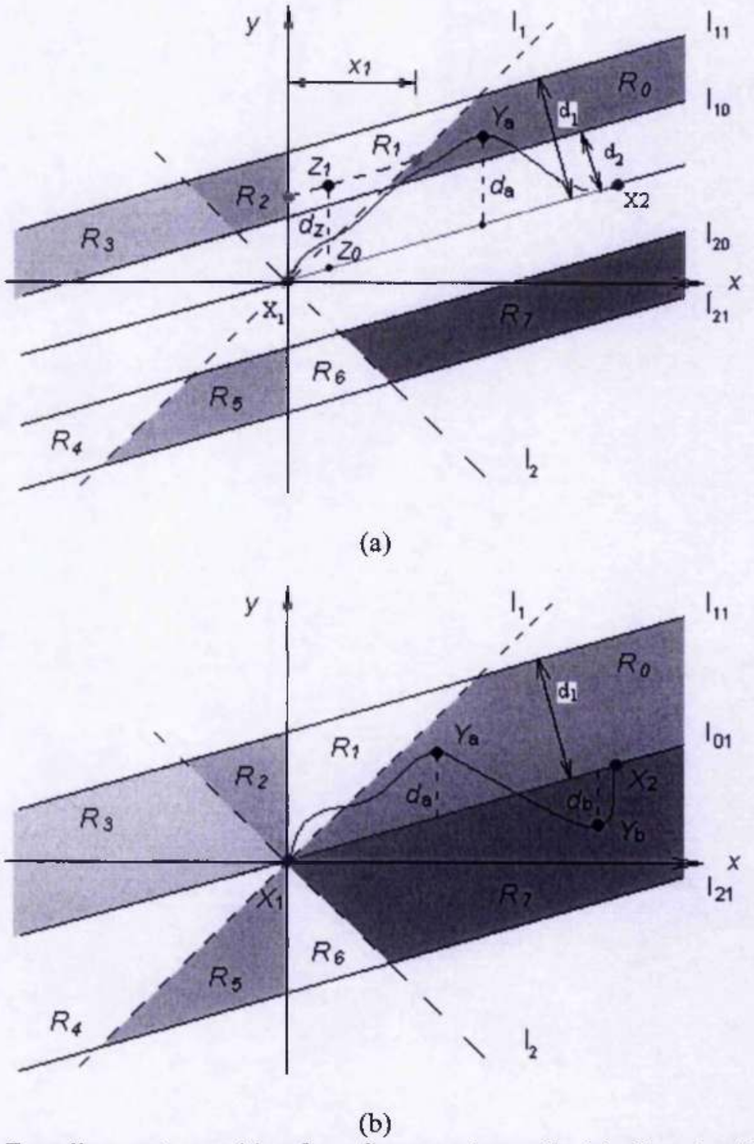


Figure 5.8 – Encoding vertex position for refinement layers for (a): *Case 1* with one vertex locating along the segment $\overline{X_1X_2}$; (b): *Case 2* with two or more vertices locating along the segment $\overline{X_1X_2}$

- *Case 1:* If $n_j = 1$ and the octant of this vertex is 0, 3, 4, or 7, its x component is encoded by using *indicator*$_x$ of the current layer. Its y component is decided by value d_a in Figure 5.8 (a) and is encoded by VLC coder. If its octant is 1, 2, 5, or 6, such as

Z_1 in Figure 5.8 (a), its y component is decided by the distance $d_Z = \left|Z_1Z_0\right|$, which is no larger than *indicator*$_y$ of current layer, and encoded using VLC codec. Its x component is encoded using $\lceil \log_2(\min\{indicator_x, \lceil \log_2(x_1+1)\rceil\})\rceil$ bits, where x_1 can be deduced from value $d = \left|Z_1Z_0\right|$ and the information from l_1 and $\overline{X_1X_2}$.

- *Case 2*: If $n_j \geq 2$, as shown in Figure 5.8 (b), the first vertex is encoded by using the same method as described above for *Case 1*, except that different VLC table is used. For other vertices, the method in [CONNELL-1997] is used except that the error band is also used to decide the number of bits. For example, for Y_a in Figure 5.8 (b), its x component is encoded using $\lceil \log_2(indicator_x)\rceil$ bits. Its y can be deduced from d_a (encoded using variable length coder), and l_{01}. For Y_b in Figure 5.8 (b), its x component can be deduced from the x component of Y_a and the line $\overline{Y_aY_b}$ (encoded using $\lceil \log_2(indicator_x)\rceil$ bits). Its y component is deduced from d_b (encoded using variable length coder), and l_{01}.

For every refinement layer, two VLC tables are designed for the above two cases. For *Case 1*, the discrete set of d is $\{2,3,4,5,6\}$, $\{1,2,3\}$, $\{0,1\}$ for layers 1, 2, 3, respectively. For *Case 2*, the discrete set of d is $\{0,1,2,3,4,5,6\}$, $\{0,1,2,3\}$, $\{0,1\}$ for layers 1, 2, 3, respectively. For layers 1 and 2, about 75% of total vertices satisfy *Case 1*, the value d can be encoded using 1-2 bits. For layer 3, the value d can also be encoded using 1 bit as the discrete set of d is $\{0,1\}$. Therefore, the compression performance can be improved especially for (near-) lossless shape coding. More experimental results are presented in Section 5.4.

5.2.3 Scalable predictive shape coding

The coding efficiency achieved by the intra-shape coding scheme cannot satisfy the requirement of low bitrate video coding, even though current state-of-the-art compression ratio is high. Since a contour sequence has very high correlation in the temporal domain, as shown in Figure 5.9 for Foreman sequence, motion estimation and compensation can be used to achieve further compression. The contour in the current frame can be predicted from the contour obtained in the previous frame. Only the contour segment that cannot be predicted from previous frame must be encoded by using the intra-shape coding technique. Compared with intra-shape coding, this predictive coding method can reduce the bit rate of the shape coding drastically.

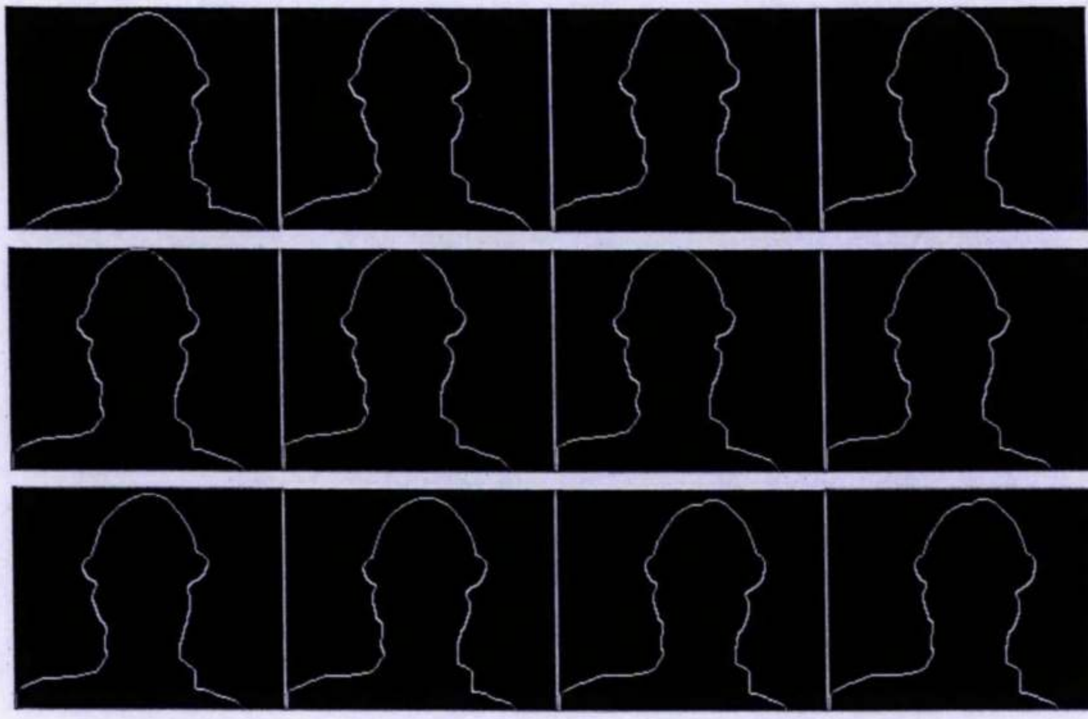
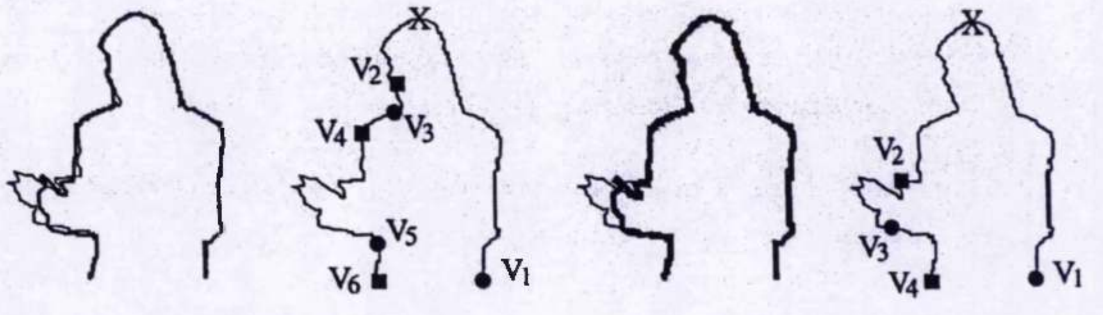


Figure 5.9 – Example of object contour sequence (12 consecutive frames from the Foreman Sequence)

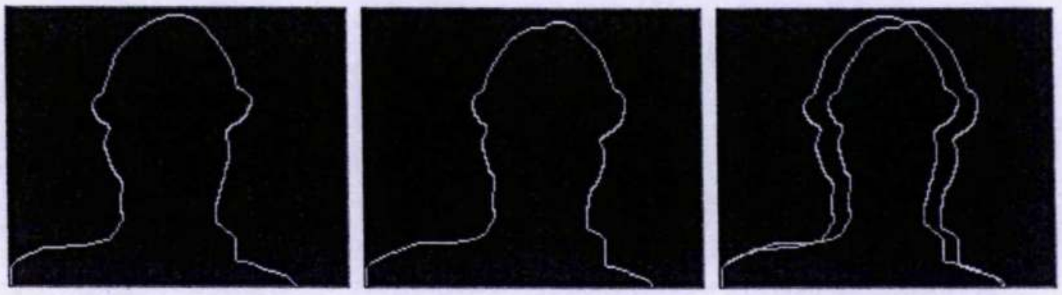
Several motion compensated contour coding schemes have been proposed [CHO-1999] [GU-1995] [KIM-2000]. In these contour motion estimation schemes, the object contour is assumed to undergo a translational motion. A global motion vector is searched according to the number of matched contour points between two contours. The whole contour is segmented into *global motion success segments* and *global motion failure segments,* as shown in Figure 5.10. The *global motion success segment* is the segment that can be correctly predicted by the global motion vector. Otherwise, it is a *global motion failure segment. The global motion failure segment* is encoded by chain coding method. Only the segment length of the *global motion success segment* is required for the transmission. It can be reconstructed from the contour in the previous frame and the global motion vector. In [CHO-1999] and [KIM-2000], a second stage motion, local motion vectors are searched for each global motion failure segment. The *global motion failure segment* is further split into *local motion success* and *local motion failure* segments. The *local motion success* segment can be represented by its length and the local motion vector.

The main assumption of the above contour motion estimation/compensation schemes is that the global motion of the object contour is translational. When there is more complex motion such as zoom and/or rotation, the contour cannot be well compensated. In Figure 5.11, the object contours in two neighbouring frames are overlapped. We can find that different motion patterns exit for different contour segments. It is difficult to use one motion model to describe them.



(a) Lossless motion estimation; (b) Lossy motion estimation

Figure 5.10 – Matched and mismatched segments, and matched (or mismatched) start-end points after motion estimation by traversing in a counter-clockwise direction [CHO-1999].



(a) (b) (c)

Figure 5.11 – Illustration of contour motion pattern for different segments: (a) and (b) are contours in different frames. (c) shows the overlapping of (a) and (b)

In [LU-2002], an affine global motion compensation scheme is investigated. The following six-parameter affine motion model is used as a global motion model.

$$\begin{cases} \hat{x} = a_0 x + a_1 y + a_3 \\ \hat{y} = a_4 x + a_5 y + a_6 \end{cases} \tag{5.1}$$

where $\hat{x}$ and $\hat{y}$ are the coordinates of contour points of the current frame. x and y are the coordinates of contour points in the previous frame.

The vector $[a_1, a_2, a_3, a_4, a_5, a_6]$ is estimated according to two available contours. First, the corner points of each contour are detected according to their curvature values. Then, the corner points are matched by a corner matching process. The motion vectors are calculated from the matched corner pairs. The affine parameters are estimated by a Least Median Square (LMS) algorithm, and then are encoded. One of the problems with this method is the coding of affine parameters. As the affine parameters are floating point, 10-12 bits are required to represent each parameter. Therefore, for most sequences, compression ratios are not high. The other problem is the contour corner matching. Sometimes, it cannot be matched accurately due to the shape distortion. The parameter used to represent the shape feature is not robust enough. Furthermore, this method cannot provide scalable shape coding.

In our research, a novel layer-adaptive scalable predictive coding scheme is proposed, which can achieve higher compression efficiency than state-of-the-art shape coding methods due to the use of temporal prediction. In order to achieve scalable predictive coding, it is necessary to represent and estimate the contour motion hierarchically. In our proposed scheme, the contour motions in level i are first estimated. They are predicted from the MVs of the previously transmitted levels and/or the encoded MVs of the current level. Contour matching in CSS image domain is applied to find the correspondence of two contours during contour motion estimation, which can achieve more accurate motion estimation of object contours. Figure 5.12 shows the diagram of scalable predictive shape coding scheme. The novelties of this method are twofold:

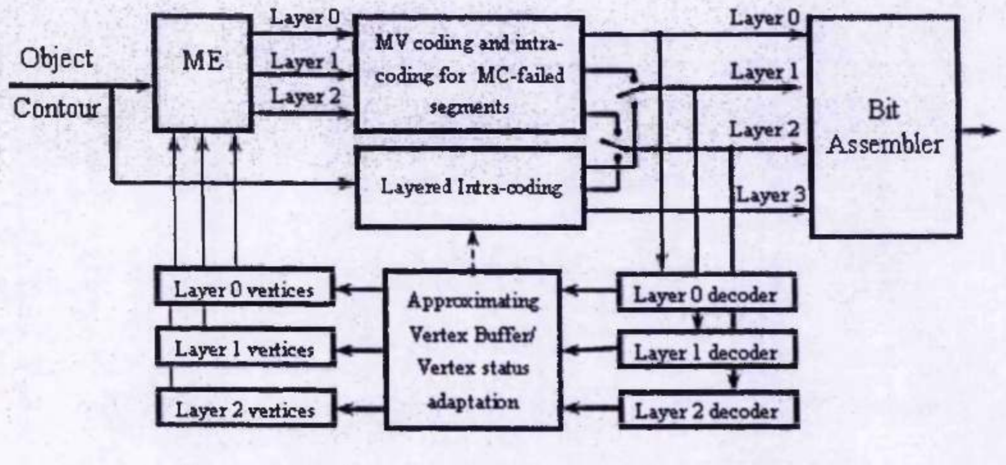


Figure 5.12 – Diagram of scalable predictive shape coding

- First, we propose an efficient contour motion estimation scheme, which is based on the curvature information of an object contour and is used to predict the motion vectors of vertices in the coarser level;
- Second, a scalable encoding scheme is proposed, in which the motion of each contour is estimated hierarchically. A multi-model encoding scheme is included to improve the compression efficiency.

The proposed scheme consists of two steps: contour motion estimation and scalable predictive shape coding, which will be discussed in detail in the following sections.

5.2.3.1 Contour motion estimation

Instead of using the corner matching method for contour motion estimation described in [LU-2002], in our proposed scheme, CSS images and curvature information are used in contour matching for contour motion estimation. CSS images are currently used for shape indexing and retrieving, and have been selected as shape descriptors for the MPEG-7 standard [MPEG7-2002].

5.2.3.1.1 CSS image calculation

The CSS image is computed by first convolving a path-based parametric representation of the contour with a Gaussian function, as the standard deviation of the Gaussian varies from a small to a large value. Next, curvature is computed on each smoothed contour. As a result, curvature zero-crossing points and curvature extremes can be recovered and mapped to the CSS image in which the horizontal axis represents the arc length parameter on the original contour, and the vertical axis represents the standard deviation of the Gaussian filter. The CSS image has the properties that it is invariant under rotation, uniform scaling, and translation of the contour.

Given a planar curve

$$\Gamma = \{(x(w), y(w)) \mid w \in [0,1]\} \tag{5.2}$$

where w is the normalized arc length parameter, and its evolved version is defined by:

$$\Gamma_\sigma = \{(X(u,\sigma), Y(u,\sigma)) \mid u \in [0,1]\} \tag{5.3}$$

where

$$\begin{aligned} X(u,\sigma) &= x(u) \otimes g(u,\sigma) \\ Y(u,\sigma) &= y(u) \otimes g(u,\sigma) \end{aligned} \tag{5.4}$$

$g(u,\sigma)$ denotes a Gaussian of width σ defined by:

$$g(u,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-u^2/2\sigma^2} \tag{5.5}$$

The curvature of Γ_σ can be calculated by:

$$\kappa(u,\sigma) = \frac{X_u Y_{uu} - X_{uu} Y_u}{\left(X_u^2 + Y_u^2\right)^{3/2}} \tag{5.6}$$

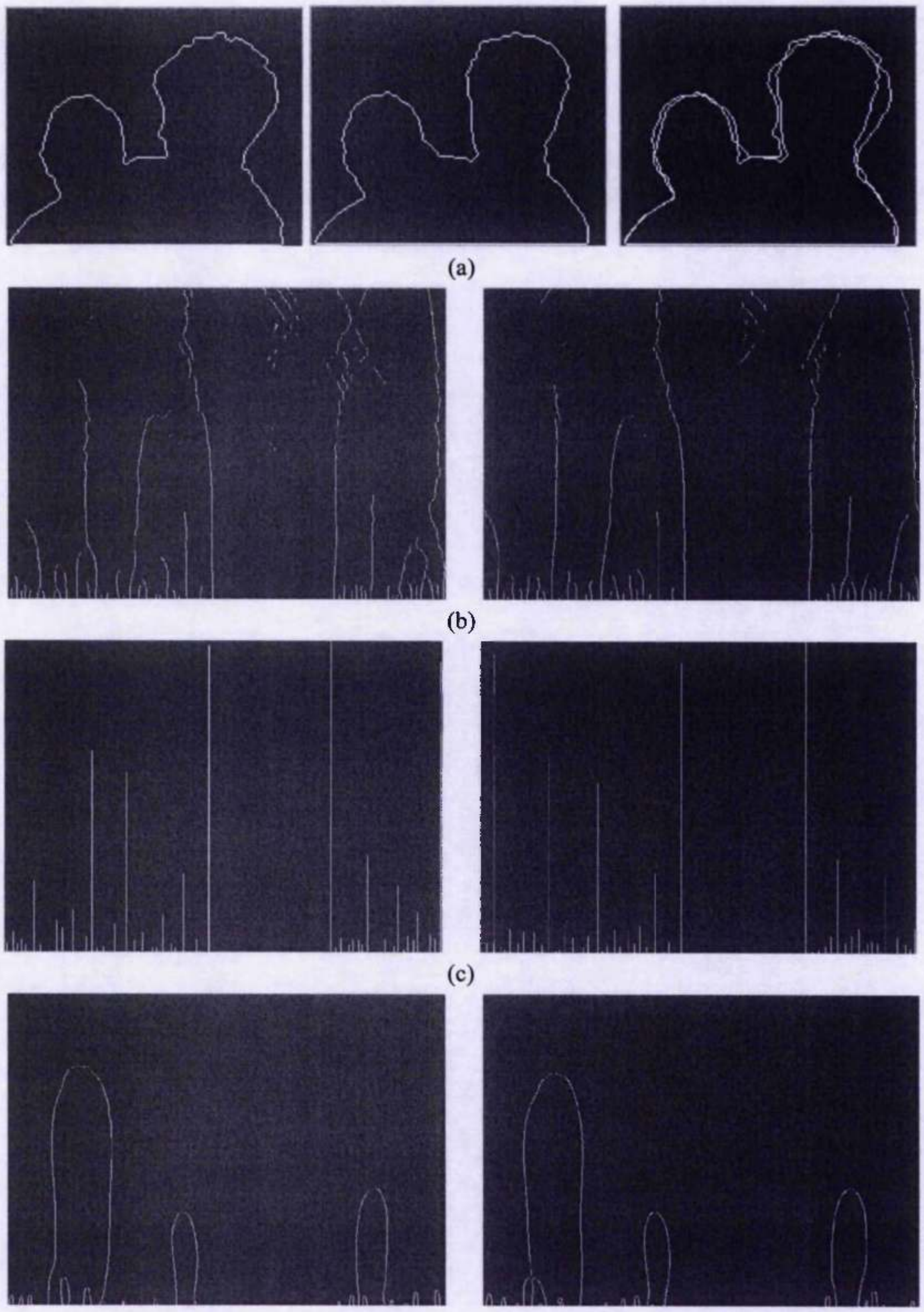
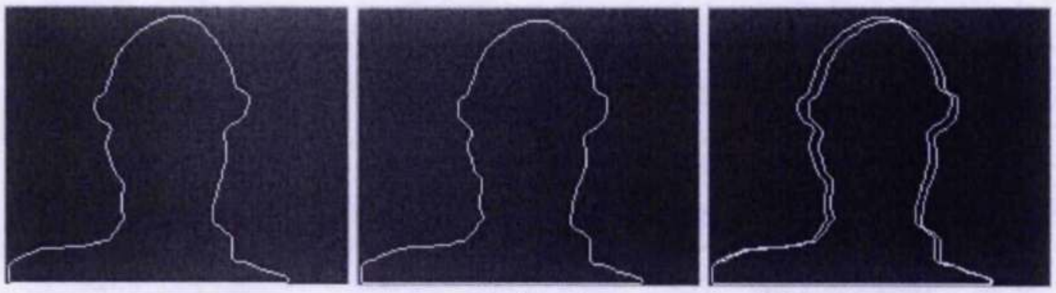
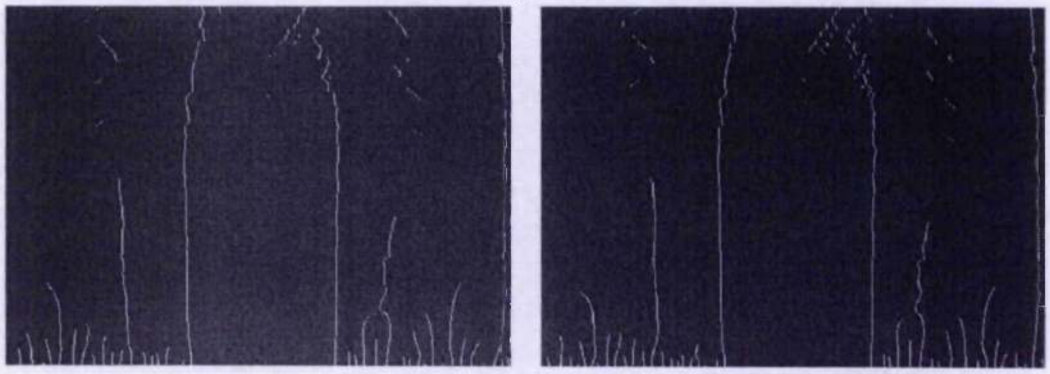
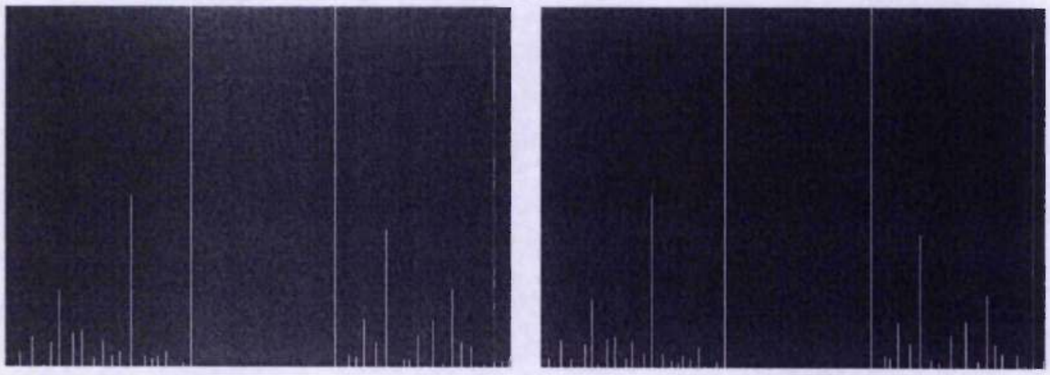


Figure 5.13 – Two object contours from Motr_dhtr sequence, their extremes and zero crossing of contour curvature image with the scale σ. (a) Original object contours and their motion; (b) The extremes curve of curvature across the scale σ; (c) is the length of extremes curve after tracking process; (d) is the zero crossing of curvature across the scale σ.


(a)


(b)


(c)

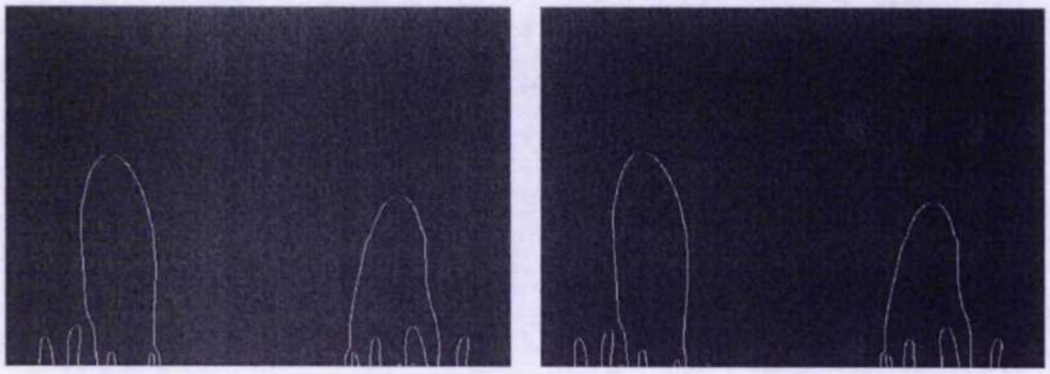

(d)

Figure 5.14 – Two object contours from Foreman sequence, their extremes and zero crossing of contour curvature image with the scale σ. (a) Original object contours and their motion; (b) The extremes curve of curvature across the scale σ; (c) is the length of extremes curve after tracking process; (d) is the zero crossing of curvature across the scale σ.

Using equation (5.6), the main feature points can be detected at any scale. By finding the local curvature extremes, we are able to identify where sharp changes in the contour direction occur. The curvature extremes of lower resolutions also exist in the higher resolutions, and they have similar positions. In our research, all of the curvature extremes are defined as the Curvature Scale Space (CSS) image of Γ.

Figure 5.13 shows the two object contours in the Motr_dhtr sequence, their extremes and zero crossing of contour curvature image with the scale σ. It can be found that the extremes and zero crossing of object contour curvature can identify important geometric properties of contour and two object contours have large similarity in CSS images.

Figure 5.14 shows the object contours and their CSS images for Foreman sequence. The same conclusions can be achieved. Therefore, in the proposed algorithm, the extremes of the contour curvature will be used to find the corresponding features of two object contours.

5.2.3.1.2 Contour matching of CSS image

Figure 5.13 and Figure 5.14 show the CSS images of object contours of two consecutive frames. It can be found that the matching of two object contours in CSS images is much simpler than matching object contour directly. As the two contours have the same direction (counter-clockwise in our experiments), the matching of two CSS images just tries to find the optimal horizontal shift of the maxima in one of the CSS images that would yield the best possible overlap with the maxima of the other CSS image. The basic idea behind the contour matching based on CSS image is to obtain a coarse-level match using the structural features of the input curves. Such a match can be found quickly and reliably since at the high levels of CSS image, there are relatively few features to be matched. The actual features used for matching are the maxima of the curvature local extremes contours since they are the most significant points of these contours: the CSS coordinates of a maximum convey information on both the location and the scale of the corresponding contour whereas the body of the contour is, in general, similar in shape to those of other contours. Furthermore, the maxima are isolated point features and therefore solving the feature corresponding problems is relatively simple.

So the task of the matching algorithm is to find the correct correspondence between two sets of maxima: one from each CSS image. The allowed transformation from one set to the other is mere horizontal translation. The translation parameter is computed when the first image curve CSS maximum is mapped to the first model curve CSS maximum and used to map each of the remaining image curve CSS maxima to the model curve CSS. The corresponding model curve

CSS maximum should then be the closest model curve CSS maximum. Many candidatures may have to be considered since the correspondence between the first pair of maxima can be made in possibly many ways. This matching problem can be solved using a best-first matching strategy which will gradually expand a number of candidatures matched in parallel until the lowest-cost complete match is found. The contour-matching algorithm [MOKHT-2003] (in Chapter 2) has been employed to achieve contour matching in CSS image. The contour-matching algorithm can be summarised as follows:

1. Calculate the CSS images of object contours in the current frame and previous frame, which are named as image CSS and model CSS respectively. Normalise these coordinates so that the horizontal coordinate u varies in the range [0, 1]. Segment the object contours into several segments by using CSS images.

2. Create a number of nodes corresponding to the possible match of the contour segment of current frame with the highest-scale maximum of the image CSS and the contour segments of previous frame, in which the maximum has a σ coordinate close (within 90%) to that of the highest image CSS maximum. Initialise the cost of each node to zero.

3. For each node created in the previous step, compute a CSS shift parameter α using $u_m = u_i + \alpha$, where u_m is the horizontal coordinate of the model CSS maximum and u_i is the horizontal coordinate of the image curve CSS maximum.

4. Create two lists for each node created in step 2. The first list will contain the image CSS maxima matched within that node at any point during program execution and the second list will contain the corresponding mode CSS maxima. Initialise the first list of each node to contain the highest-scale image CSS maximum. Initialise the second list of each node to contain the corresponding mode CSS maximum determined in step 2.

5. Expand each node created in step 2 one step using the procedure described in the next step.

6. To expand the node, select the highest-scale, image CSS maximum (which is not in its first list) and apply that node's CSS shift parameter computed in step 3 to map that maximum to the model CSS image. Locate the nearest model Curve CSS maximum (which is no in the node's second list). The cost of match is defined as the straight line distance in the model CSS image between the two maxima. If there are more image CSS

maxima left, define cost of match as the height of the highest model curve CSS maximum not in the node's second lost. Likewise, if there are no more model CSS maxima left, define cost of match as the height of the selected image CSS maximum. Add the match cost to the node cost. Update the two lists associated with the node.

7. Select the lowest-cost node. If there are no more mode or image CSS maxima that remain unmatched within that node, then return that node as the lowest-cost node. Otherwise go to step 6 and expand the lowest-cost node.

Please refer to [MOKHT-2003] for the detailed description of this algorithm. Once the nodes' correspondence are decided, the contour motion can be represented and estimated by the motion of feature points of the object contour with salient features. In fact, based on these motion vectors, the global motion parameters can be estimated after selecting the proper motion model. If a translation model is used, this method is the same as that in [KIM-2000]. If an affine motion model is used, the proposed method is the same as that in [LU-2002]. The complexity of our proposed method is comparable with the contour motion estimation method in [LU-2002].

5.2.3.2 Scalable predictive shape coding

The proposed scalable predictive shape-coding scheme used different encoding schemes for different approximation levels. Figure 5.12 shows the diagram of scalable predictive shape coding scheme. For each level except the finest one, the motion vectors of vertices are estimated and the contour of the current frame is predicted by motion compensation. For the motion failure segments, where the approximation error band is larger than the predefined threshold, new vertices are inserted to make it satisfy the error band. Their coordinates are intracoded and are transmitted to the decoder. As an adaptive update scheme is proposed and used in the codec, the order of the maintained and rejected vertices need not be coded and transmitted to the decoder. This is different from the method in [CONNELL-1997].

The predictive scalable shape-coding algorithm can be described as follows:

- For the vertices of level 0 in the previous frame, which have the maximal curvature, their corresponding point in the CSS of current frame is estimated based on the contour matching algorithm in Section 5.2.3.1. For other vertices of level 0, which cannot be selected using curvature information in CSS image, they try to make the approximated contour locate in the pre-defined error band and their motion vectors are estimated trying to minimise the approximation error. This process is illustrated in Figure 5.15. Image (a) shows the object contour of the previous frame, in which three vertices belonging to level 0 are indicated as A, B, and C. The vertex A and B locate on the high curvature position and their corresponding positions are estimated, indicated by A' and B' as shown in image (b). In order to find the corresponding vertex of C, the following two steps are conducted:

1. First, the search region S is decided along the contour of the current frame as shown in Figure 5.15 (c). This is mainly to reduce the number of possible search positions and computational complexity;

2. Then, in this region, find the corresponding position C' of vertex C, which has the minimal prediction error of the contour segment between A' and B', as shown in image (d);

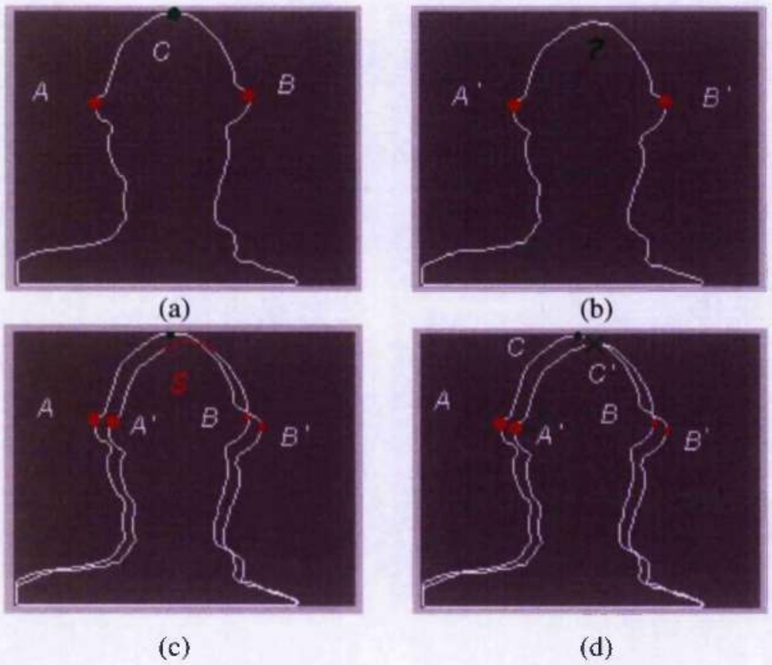


Figure 5.15 – Motion estimation of the vertex without locating on high curvature position. (a) Object contour of the previous frame; (b) Object contour of current frame; (c) Search region along the contour of current frame for the vertex C in the previous frame (indicated by green colour); (d) the final estimated position C' (indicated by green 'x'), which tries to minimise the contour approximation error.

After estimating the motion vectors, these motion vectors are differentially encoded by using a variable length-coding scheme (VLC). In our experiments, adaptive arithmetic coder is used [WITTEN-1987].

For the motion failure segments, where the contour approximation cannot satisfy the predefined error band; new vertices are inserted based on the CSS image and the maximal error distance. The coordinates of these vertices are intra-coded by using the method described in Section 5.2.2, and transmitted to the decoder after the motion vectors.

- If the vertices in levels 1 and 2 in the previous frame are located on the salient points with high curvature, their motion vectors are estimated by using the method described in Section 5.2.3.1. Otherwise, the motion vectors are estimated by using the method illustrated in Figure 5.15, which tries to minimise the approximation error of object contour.

During motion vectors encoding, the motion vectors are first predicted from the MVs of the coarser levels and/or the encoded MVs of the current level, as shown in Figure 5.16. In this Figure, vertices A, B, C and D belong to the coarser levels. Vertex 1, 2, 3, and 4 belong to the current level. Their MVs are estimated as follows:

- The MV of vertex 1 is estimated from the MVs of vertex A and B;
- The MV of vertex 2 is estimated from the MVs of vertex B and C;
- The MV of vertex 3 is estimated from the MVs of vertex 2 and C;
- The MV of vertex 4 is estimated from the MVs of vertex C and D.

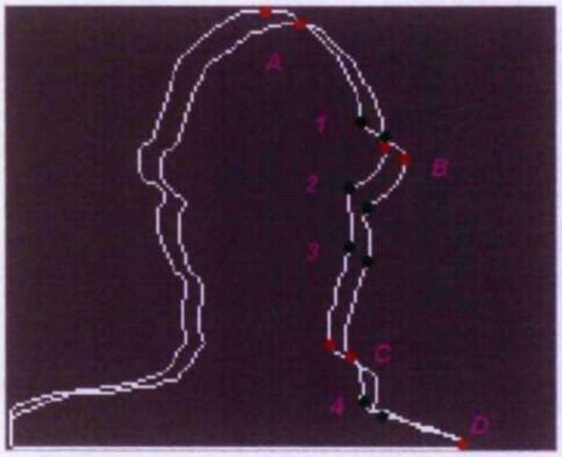


Figure 5.16 – illustration on predicting motion vectors from the coarser levels and/or current levels.

The prediction error of MV is encoded using a VLC scheme. However, some video objects have very complicated shapes, requiring more vertices in levels 1 and 2 to represent them. Here, the MV estimation and differential coding method is not efficient enough as more vertices are needed to represent them, which was also discussed in [HOTT-1989].

Therefore, in our proposed scheme, the multi-model selection method is used in these two levels, which includes two coding methods:

- MV estimation/differential coding method;
- Scalable intra-shape coding method;

The method, which generates the shorter bitstream, is selected and encoded, together with the encoded bitstream.

- For level 3, as there are more approximation vertices, the size of the list update information is significant. Furthermore, it is difficult to estimate the correspondence of two contour segments based on the curvature information, which provides a lossless approximation of the contour. The performance of the MC-based method is not satisfactory [HOTT-1989]. Therefore, in our proposed method, the scalable intra-encoding scheme described in Section 5.2.2 is used. That is, the vertices are first selected progressively to satisfy the error band condition. Then, they are intra-encoded by using the method in Section 5.2.2. Therefore, for the vertices of level 3, no motion estimation is conducted, as the motion estimation of vertices is less efficient for shape coding.

Intensive experiments have been conducted to test the proposed scalable predictive shape-coding scheme, which are illustrated in section 5.4 in detail.

5.3 Proposal for scalable mesh model design and coding

5.3.1 Scalable mesh model design

Model design is vital for the performance of 2-D scalable model-based video coding. The model should represent the movement of the object precisely. Furthermore, in order to overcome the aperture problem commonly encountered during motion estimation, the node points of model should be allocated on the positions that contain sufficient grey-level variation, such as corners. At the same time, based on the results in [NAKAYA-1994], the area of the patches that contain the motion discontinuities should be as small as possible. In this way, the number of pixels with erroneous motion estimation can be reduced. The best way to achieve this is to let the edge of the mesh triangles conform to the object boundary.

In our proposed algorithm, the physical characters (such as edge and texture) and motion character of the object surface have been taken into account, similar to the method in [IZQUIER-

1999]. Before 2-D model construction, a set of feature points and edge lines from the intensity image are first extracted. Then an initial constrained 2-D mesh can be constructed to reduce the triangular patch area with motion discontinuities. The difference from the proposed method to those, such as [ALTUNB-1997], [BEEK-1999] and [CELASUM-2000], is that object surface characteristics are employed as the mesh constraints in order to achieve accurate motion modeling.

Before scalable mesh model design, we assume that object shape has been approximated and transmitted to the decoder progressively and losslessly by using the methods in Section 5.3. The proposed algorithm can be described as follows:

1. The object is analysed and segmented into texture and motion homogeneous patches.

   Certain objects in the video sequence commonly undergo both rigid and non-rigid motion. Therefore, the interior node should be allocated to represent the movement precisely, especially for non-rigid motion. In order to achieve this, the object is analysed and interior edges are first detected since the errors during warping are more likely to occur at the interior edges due to the non-rigid object motion. At the same time, in order to estimate the motion of object precisely, the node should be located on the features with high texture content [SHI-1994].

2. The nodes of the coarsest layer (layer 0) are allocated on the intersection points and the interior edge lines, trying to approximate the patch contours achieved in section 1.

   To generate linear approximations for each interior edge, the method in [DUNHAM-1986] is applied. That is, a symmetric narrow band δ along the curve of concern is first defined. Then the shortest polygonal path lying in the strip defined by the band is then chosen as linear approximation for the curve. To avoid the creation of very small triangles during the generation of the initial mesh, sides of the polygonal path whose length does not exceed a predefined threshold are removed. The resulting lines are designed to form the constrained mesh vertices of the coarsest layer. Please note that, some of the nodes in this layer may be located on the object contour because the interior texture information is not considered during the object contour approximation. But it is far from the control points in layer 0 of object contour.

3. More nodes are allocated on the object interior to obtain meshes with nodes regularly distributed over the object area.

   First, the video object is split into rectangular blocks of moderate size. For each block that does not intersect an edge and the layer 0 nodes (both object contour and interior), the point that is most clearly distinguished from its neighbour is also considered as a candidature for the mesh nodes. SUSAN corner detector [SMITH-1997] is used to detect the corner and estimate the gradient. By adjusting the size of rectangular blocks, different approximation layers can be generated.

In our research, a three-level scalable object model for each object is used. The number of points in every level is decided by the size of object, texture and motion of the frame. On completion of node point allocation, constrained Delaunay triangulation algorithm is used to build the mesh structure of the object model [BERG-1997]. As the node points in level 0 are located on the patch contours, they are considered as the constraints to guarantee the edges of mesh triangles conform to the contour of motion discontinuity.

5.3.2 Model evaluation algorithm

In order to evaluate the performance of the designed model, a hierarchical motion estimation scheme has been proposed to estimate the motion vectors of node points. The PSNR after warping is calculated and used as the criterion of model performance. The evaluation algorithm includes the following steps:

1 Foreword / backward motion estimation of video object

In order to estimate the motion of a video object, a number of points are allocated in the interior of object. These points may be different from those used to represent object model and have good features for tracking [SHI-1994]. Then, both forward and backward motion vectors of these points between frame $I(\bar{x}, t-1)$ and frame $I(\bar{x}, t)$ are estimated using Shi-Tomasi feature tracking algorithm in [SHI-1994]. That is, the forward motion vector of the i th point location V_i in frame $t-1$, moves to location $V_i^{'}$ in frame t. Then the backward motion vector at the location $V_i^{'}$ in frame t maps back to $V_i^{''}$ in frame $t-1$.

2 Reliability evaluation

The motion "reliability" is estimated based on both forward and background motion vectors. The "reliability" is evaluated by the Equation (3.3) in Chapter 3. The smaller the difference between V_i and V_i'', the more reliable the motion vector of i th node. The nodes whose reliability is smaller than a threshold (0.3 is chosen in our experiments) are not considered during model vertices prediction.

3 MV prediction of node points of scalable object model

After estimating the motion vectors of these points, the motion vectors of node points representing the object model are predicted from their m nearest surrounding motion vectors. m is chosen as 6 in our experiments. The weighted least squares (WLS) estimation in [ROUSS-1987] is used to determine the affine parameters of motion for the control points. During estimation, each motion vector is weighed according to its "reliability".

4 MV refinement by hexagonal matching algorithm

As the estimated MVs using feature tracking algorithm has high precision, the MVs are then refined to 1/4-pixel resolution with lower warping error. During refinement, hexagonal matching algorithm in [NAKAYA-1994] is exploited which is efficient for mesh-based motion estimation and can also keep the mesh structure during estimation.

Some results are listed in Section 5.4.1 for the performance evaluation of the designed scalable object model.

5.3.3 Scalable mesh model compression

Scalable mesh model compression includes both intra-coding (coding of node position) and inter-coding (coding of node motion vectors). In this section, our research will just focus on intra-coding. Scalable coding of node motion vectors will be discussed in detail in Chapter 7 of this thesis.

Before model compression, we assume that the mesh topology remains fixed during mesh tracking. The mesh can be constructed using Delaunay triangulation so that the mesh triangular topology need not be coded. The Delaunay triangulation [BERG-1997] is used as a pre-agreed triangulation method, such that the mesh can be reconstructed at the receiver.

The model compression method in [BEEK-1999] has been extended to achieve scalable mesh model compression, which consists of three steps:

1 Unique ordering of the node points;

Compression of node-point locations assumes a unique ordering of the node points. This ordering is computed on the original finest level and is defined as follows. First, the top left node is defined to be the first in the node ordering. The node is considered as top left with minimum $x + y$, assuming that the origin of the local coordinate system is at the top left. If there is more than one node with the same value of $x + y$, use the y value to break ties. Other interior node points are ordered using a greedy nearest neighbour strategy, starting from the first node. The nearest neighbour strategy identifies the node that has not already been ordered and has minimum $|x - x_{n-1}| + |y - y_{n-1}|$, where (x_{n-1}, y_{n-1}) represents the coordinates of the previously ordered node. If necessary, use the y and x values to break ties for the case when there is more than one such node. This continues until the entire set of node points is ordered. The difference of the ordering process from that in [BEEK-1999] is that the points along the object boundary are not included during the ordering process.

2 Encoding of base-layer mesh geometry

Each node point location of base layer is coded differentially using the coordinates of the previously processed node as predictors. That is, the difference between the x-coordinates of the present node and the previous node is coded using a variable-length coder, as is the difference between the y-coordinates of the present node and the previous node. The total number of node points of base-layer is encoded before coding the actual locations. Thus it is able to reconstruct the mesh of this layer. The decoder finally applies constrained Delaunay triangulation to obtain the topology of the base layer mesh.

3 Encoding of enhancement-layer mesh geometry

The basic predictive schemes discussed above are also used to code the locations and motion vectors of nodes in the enhancement layers. However, coding of the detail information at successively finer levels is performed with respect to already encoded information of the current or coarser levels.

In our study, a scalable method is used to encode the ordering information in such a way that it can be reconstructed layer by layer. The encoding process for ordering information is illustrated in Figure 5.17.

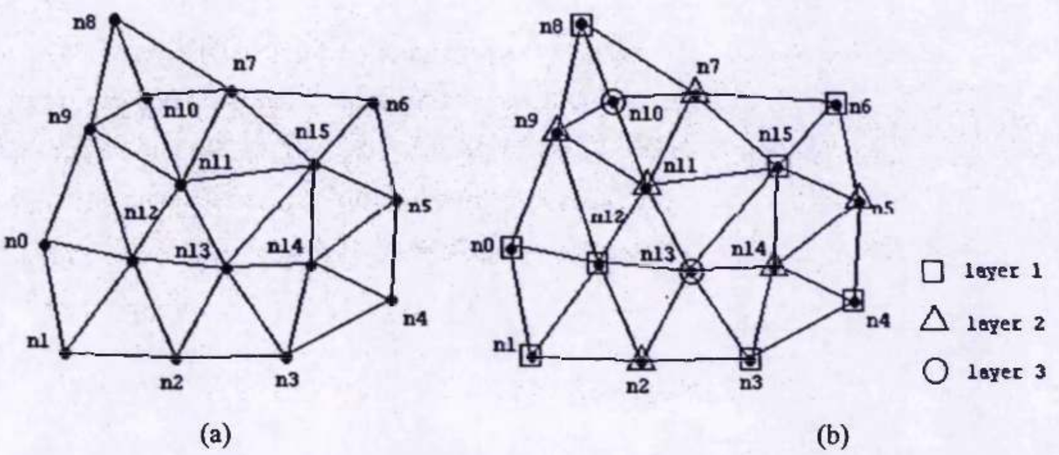


Figure 5.17 – Illustration of scalable model compression. (a) Ordering of model vertices; (b) The layer representation of the object model.

In Figure 5.17 (a), the ordering of node points of partial object model is illustrated. First, the boundary nodes are visited in a counter clockwise direction. Then the interior nodes are visited according to proximity, i.e., the next node is always the nearest node that has not been visited. In Figure 5.17 (b), two layers are used for object boundary and three layers for object interior. The list of layer labels of nodes in the predefined order illustrated in (a) is as follows: 1 1 2 1 1 2 1 2 1 2 3 2 1 3 2 1. This string can be encoded directly by using variable length coding although it is not a scalable coding. In order to achieve scalable coding, the string is arranged as:

```
•   First Layer:    1 1 0 1 1 0 1 0 1 0 0 0 1 0 0 1
•   Second Layer:       1     1   1   1 0 1   0 1
•   Third Layer:                        1     1
```

(* no need to transmit the third layer as the number of layers has been transmitted to the decoder as header information)

To encode the above strings, Run-length coding (RLC) is used to represent the above string as (*a (b): a is the number of 1; b is the number of 0*):

First Layer:	2 (1) 2(1) 1(1) 1(1)
Second Layer:	0 (2) 1(2) 1(0)
Third Layer:	0(1) 1(1) 1(0)

5.4 Experimental results

5.4.1 Results on scalable model design and compression

In Chapter 4, some scalable object models for head-shoulder sequences, such as Claire, Miss_am, Carphone, and Akiyo sequences, have been illustrated and their performance has been evaluated. Figure 4.17 and Figure 4.18 show the designed scalable model for Carphone and Akiyo video objects. Table 4.3 gives the average PSNR values of Carphone, Claire, Miss_am and Akiyo sequence. Compared with the results in [BEEK-1999], for the Akiyo sequence, it shows that the proposed method can achieve about 2-5 dB improvements and represent the object motion precisely.

For non-face video objects, extensive experiments have been conducted to evaluate the efficiency of the proposed modelling algorithm. Figure 5.18 shows the designed scalable object model for the Motr_dhtr video sequence.

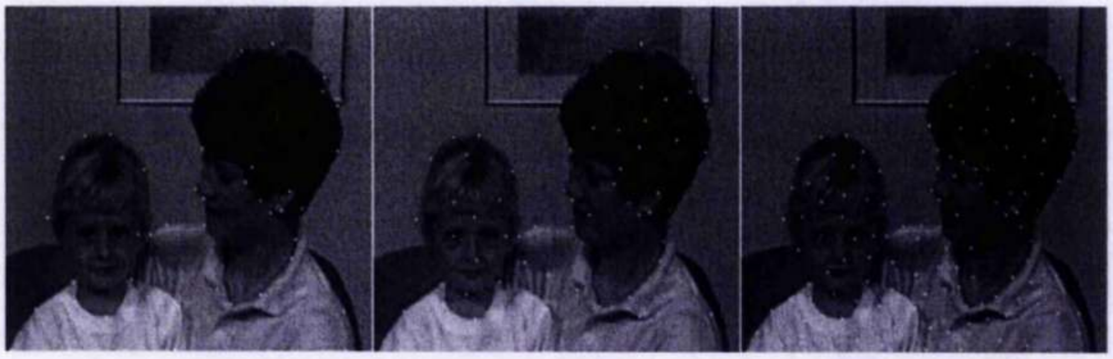


Figure 5.18 – Scalable object models for Motr_dhtr sequence (considered as a non-face video object). Image (a) is the model of level 0; (b) is the model of level 1; (c) is the model of level 2. In (a), (b) and (c), the node points in level 0 of object contour are included.

The performance of the model design scheme is tested by the model evaluation algorithm in Section 5.3.2. The motion vectors of node points have ¼ pixel resolutions and frames 2, 4, 6 and 8 are selected during the test, which is the same as that in Chapter 4. Table 5.1 gives the average PSNR values of non-Carphone, Claire, Miss_am and Akiyo sequence. It shows that good warping performance can be achieved using the proposed scalable modelling algorithm.

Some compression results for scalable models are listed in Table 5.2. These scalable object models are for Carphone, Motr_dhtr, Claire, and Akiyo objects. From the results, it is shown that only 1-2 kbits is required to encode each object model progressively, which only occupy a small portion of the total bits for coding the whole sequence. In fact, further improvement is possible through the optimisation of ordering and prediction of node points as in [TOUMA-1998] to reduce the bits of node positions.

Table 5.1 – Average warping PSNR values (dB) of four sequences for different levels

	Average PSNR Value (dB)		
	Level 0	Level 1	Level 2
Motr_dhtr (QCIF)	24.08	28.98	32.94
Coastguard (CIF)	24.04	27.41	31.79
Container (CIF)	28.71	30.64	33.28
News (CIF)	29.55	32.93	35.41

Table 5.2 – Scalable model compression results by using the proposed scheme

			Test Sequences			
			Carphone	Motr_dhtr	Akiyo	Claire
Node number			150	130	140	110
Proposed method	Node Position (bit)		1672	1315	1369	1086
	Ordering information (bit (points))	Layer 0	97 (63)	55 (90)	81 (54)	66 (40)
		Layer 1	73 (37)	66 (75)	62 (35)	62 (30)
		Layer 2	38 (50)	40 (55)	41 (51)	35 (40)
	Total (bit)		1880	1476	1553	1249

5.4.2 Results on scalable shape representation and coding

5.4.2.1 Results on scalable shape representation

Intensive experiments have been conducted to test the performance of the proposed algorithm. Several video objects are selected, such as Coastguard, Kids, News, Weather sequences. Their binary shape images are illustrated in Figure 5.19. After scalable shape approximation using the proposed approximation scheme, the required average number vertices under different error criteria for different video objects are listed in Figure 5.20. Compared with the methods in [GERKIN-1997] and [JORDAN-1998], the proposed method can achieve up to 30-80% and 20-30% respectively of the total number of vertices for lossless reconstruction of test video objects. The proposed method can also achieve less number of approximating vertices at the coarser layers although reduction is not significant. However, the positions of the allocated vertices are different from those generated by the methods in [GERKIN-1997] and [JORDAN-1998.

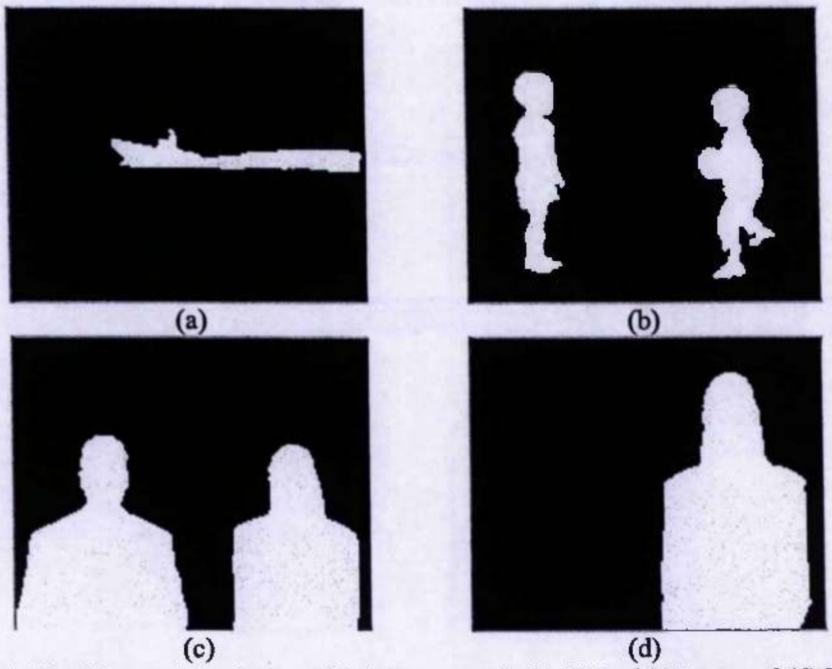


Figure 5.18 – Binary shape image of (a) Coastguard, (b) Kids, (c) News and (d) Weather sequence.

5.4.2.2 Results on scalable intra-shape coding scheme

We test the performance of the proposed intra-shape coding algorithm by coding several widely used MPEG-4 shape sequences: "Weather" and "Kids" sequences. We evaluate our algorithm by comparing with the CAE technique because the CAE technique has already been employed by the MPEG-4, and other vertex-based shape coding methods. Of the various ways to measure distortion, we utilise the following additive distortion metric per frame, which has also been used in the MPEG-4 standardised process to evaluate the performance of competing algorithms:

$$D_n = \frac{Number\ of\ Pixels\ in\ Error}{Number\ of\ Interior\ Pixels} \tag{5.7}$$

where a pixel is said to be in error if it belongs to the interior of the original object and the exterior of the approximating object, or vise-versa.

A number of experiments have been conducted to evaluate the performance of the proposed intra-shape encoding scheme. The first experiment is to compare the proposed method with those existing shape encoding schemes in [CONNELL-1997] and [JORDAN-1998], which also belong to vertex-based shape coding scheme. The corresponding results are illustrated in Figure 5.21. It is shown that the proposed scalable intra-shape coding scheme can provide 25-60% gain in bit rate over the scalable encoding method in [JORDAN-1998]. For some sequences, it can achieve 5-10% gain over conventional non-scalable vertex-based coding [CONNELL-1997] in bit rate.

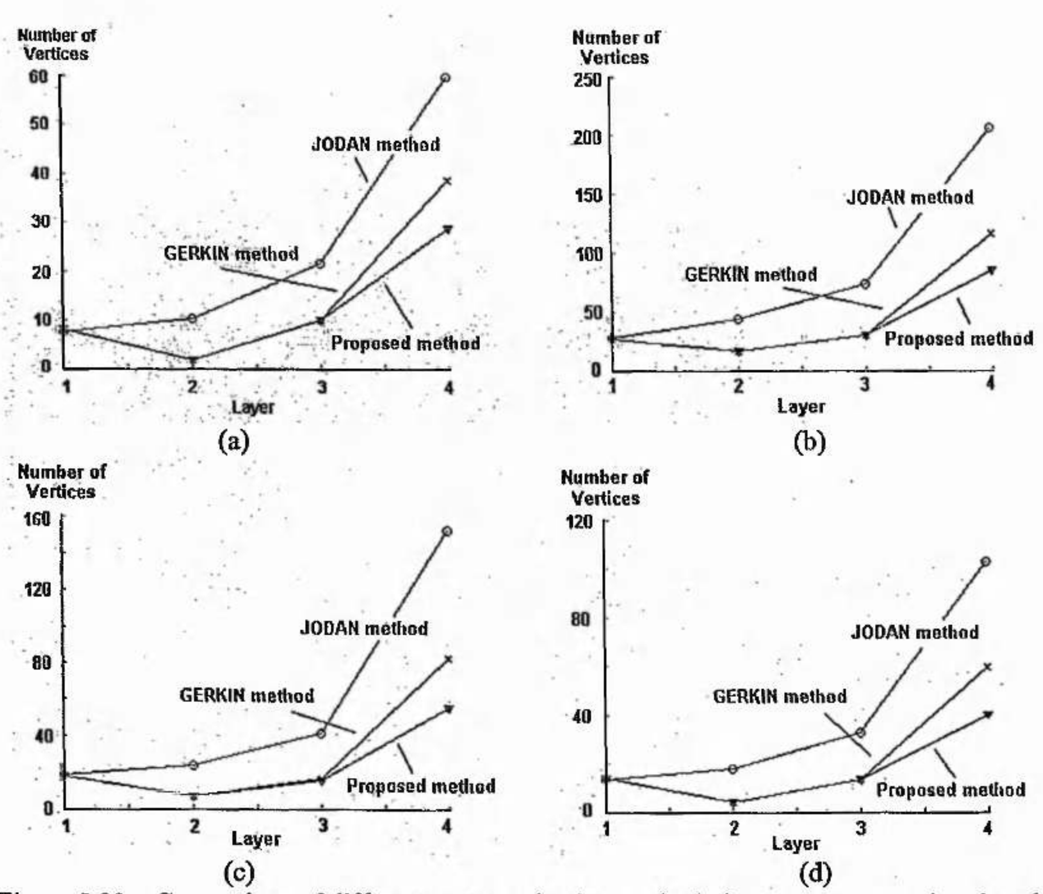


Figure 5.20 – Comparison of different vertex-selection methods for contour approximation. (a) Coastguard; (b) Kids; (c) News; and (d) Weather sequence

Another experiment is conducted to compare the proposed method with non-scalable vertex-based shape coding method [CHUNG-2000] and CAE in MPEG-4 [MPEG4-2001], in terms of R-D performance. The results are shown in Figure 5.22. From this figure, it is found that the proposed method has better R-D performance than these shape-coding methods. The R-D performance of the proposed method is also comparable to the recently developed non-scalable shape coding method in [WANG-2003] and scalable shape coding algorithm in [MELNIKOV-2000a] by compared the RD curves in Figure 5.22 with those reported results (see Fig.17 and Fig.20 in [WANG-2003] for "Kids" and "Weather" sequence respectively, and see Figure 3 in [MELNIKOV-2000a] for "Kids" sequences). One of the most important characteristics of our proposed method is that it can achieve scalable shape coding. The R-D performance of our proposed algorithm shown in Figure 5.22 is achieved by decoding the same shape bitstream. Table 5.3 lists the average bit usage per frame of four layers for the proposed intra-shape coding scheme. All of the shape sequences are with the frame rate of 10 frames per second (fps). The success of the proposed scalable intra-shape coding scheme is the efficient shape representation scheme and the layer-adaptive intra-coding scheme.

Table 5.3 Average bit usage per frame for the proposed scalable intra-shape coding scheme

	Layer 0	Layer 1	Layer 2	Layer 3 (lossless)
Weather	197	208	257	403
Kids	415	532	733	1471
News	290	385	496	927
Forman	261	344	471	676

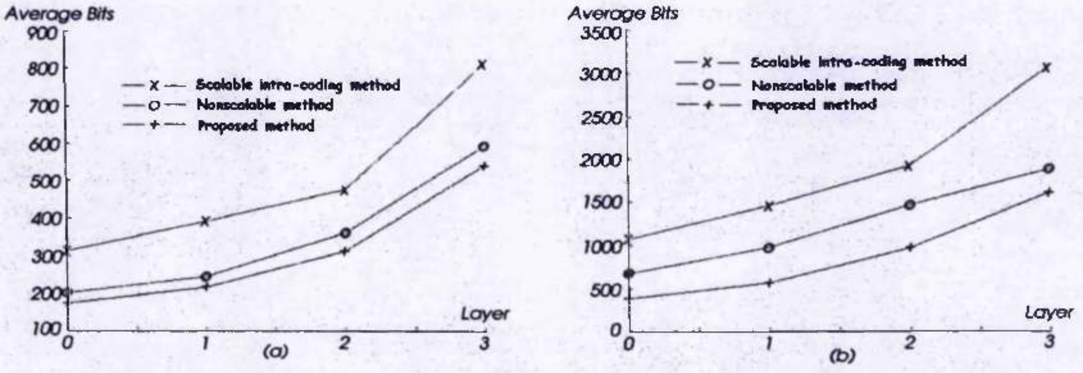


Figure 5.21 – Comparison of the performance (bits/layer) of intra-shape coding methods corresponding to: (a) Weather, (b) Kids sequence.

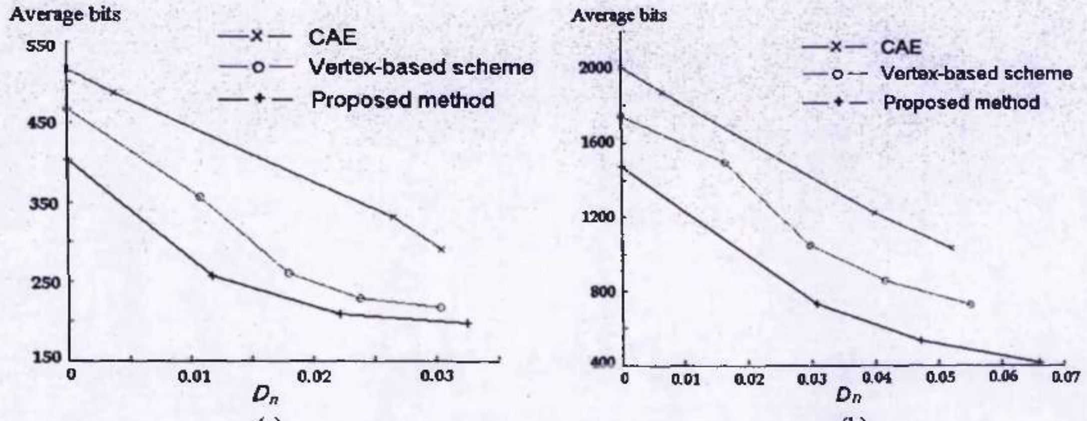


Figure 5.22 – Comparison of R-D performance of the proposed intra-shape coding method with other coding scheme for (a) Weather, (b) Kids sequence.

5.4.2.3 Results on scalable predictive shape coding scheme

The performance of the proposed scalable predictive shape coding algorithm has been tested using the "Weather" and "Kids" sequences. The same distortion metric as Equation (5.7) is used to measure the shape coding distortion. The performance is also compared with the generalised predictive shape-coding scheme (GPSC) in [KIM-2000] and CAE in MPEG-4. Figure 5.23 presents the bit distortion curves of the proposed algorithm for (a) Weather and (b) Kids sequence. It shows that our proposed algorithm can achieve better R-D performance than that of CAE and GPSC techniques. Table 5.4 shows the average bit usage per frame for the proposed scalable predictive shape coding scheme.

The success of our proposed predictive shape coding is due to the adaptive coding for different layers and the accurate motion estimation by using CSS image matching of the object contour.

Table 5.4 Average bit usage per frame for the proposed scalable predictive shape coding scheme

	Layer 0	Layer 1	Layer 2	Layer 3 (lossless)
Weather	24	52	153	325
Kids	241	359	548	1281
News	51	86	277	569
Forman	83	157	201	406

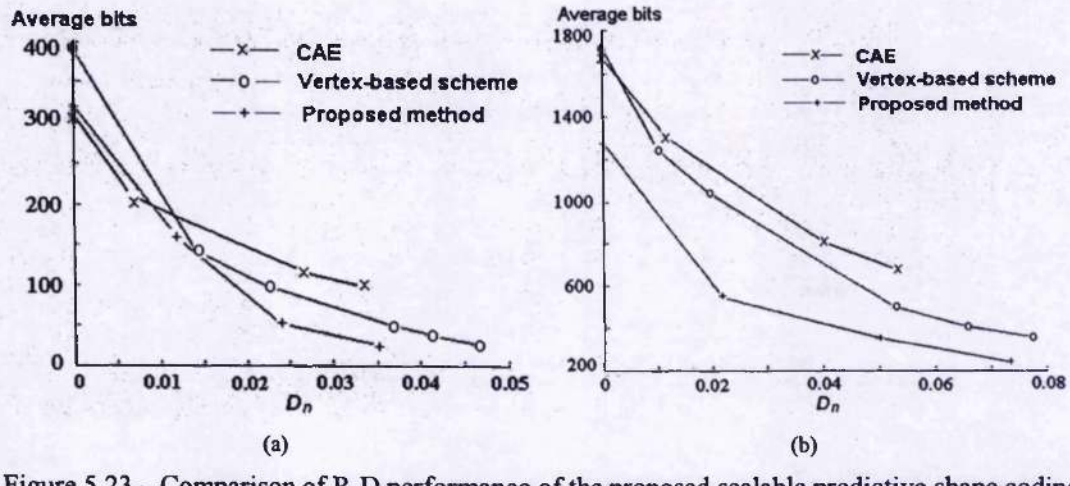


Figure 5.23 – Comparison of R-D performance of the proposed scalable predictive shape coding method with those of other coding scheme for (a) Weather and (b) Kids sequence.

5.5 Conclusions

In this chapter, an extensive study has been conducted for scalable object modelling and model compression. For 2-D mesh modelling and representation, the vertices of object mesh model are separated into two parts: vertices for shape and vertices for object motion (or vertices of the interior object). Different compression schemes are proposed for these two parts. This is because shape information sometimes has other functionalities, such as video object index and retrieval.

After reviewing the main techniques on mesh model design and shape coding, some algorithms have been proposed and described for scalable shape representation, scalable intra-shape coding and scalable inter-shape coding. Experimental results show that the proposed algorithms have better performance than the existing published methods. After discussing the scalable shape coding, scalable object modelling and model compression techniques are investigated. A model design algorithm has been proposed, in which the physical and motion characteristics of the object surface have been taken into account. After that, scalable mesh models are compressed efficiently. Extensive study and experiments have been conducted to test the proposed algorithms. For scalable object modeling, it is shown that the designed scalable object models can represent the object motion more precisely than the method in [BEEK-1999]. The designed object model can be compressed by using 1-2 kbits.

Scalable shape representation and coding are also investigated and discussed in this Chapter. In both shape representation and coding, curvature scale space (CSS) image is employed to detect the salient feature of object contour and to estimate the contour motion. For scalable shape representation, the proposed method can achieve up to 30-80% and 20-30% of the total number of vertices for lossless reconstruction of test video objects while compared with the methods in [GERKIN-1997] and [JORDAN-1998].

The proposed scalable shape coding method can achieve great improvement of compression performance by exploiting the geometrical knowledge of coarser levels and statistical entropy coding, although more computation is needed. For example, the proposed intra-coding scheme can provide 25-60% gain in bit rate over the scalable encoding method in [JORDAN-1998]. For some sequences, it can achieve 5-10% gain over conventional non-scalable vertex-based coding [CONNELL-1997] in bit rate. Experimental results also demonstrate that the proposed scalable intra-shape coding algorithm results in a significant improvement in rate-distortion efficiency with respect to other existing scalable and non-scalable shape coding algorithms.

Experimental results also show that the proposed scalable predictive shape-coding scheme can achieve better R-D performance than an existing predictive shape coding method and CAE method of MPEG-4. The proposed scalable shape coding method can achieve great improvement in compression performance by exploiting the geometrical knowledge of coarser levels, statistical entropy coding, and novel contour motion estimation scheme. Most importantly, the proposed scheme can achieve scalable shape coding, which facilitates error protection and error concealment of shape information. It also facilitates achieving other functions, such as progressive shape retrieval.

Chapter 6

Scalable Texture Intra-coding of Video Objects

6.1 Introduction

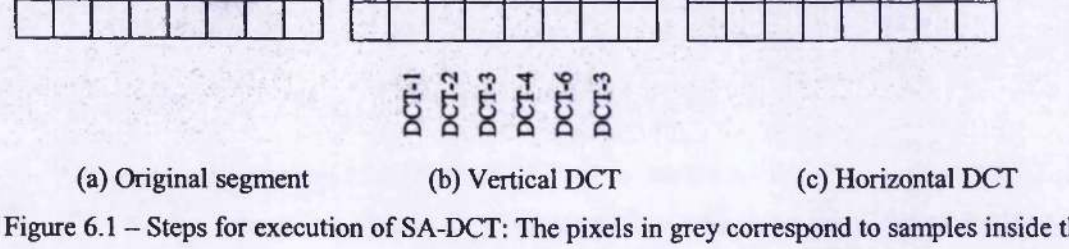
In recent years, as MPEG-4 enables object-based video coding for the coding and representation of semantic units of image and video content called "video objects" [MPEG4-2001], coding techniques have to be developed for the description of image regions, which are no longer squared as in conventional rectangular image coding, but may be of arbitrary shape.

Commonly, before video compression, a mathematical transform is used for the reduction of a large amount of statistical redundancy in video frames. Various mathematical transforms have been employed. Among them, the discrete cosine transforms (DCT) and the discrete wavelet transform (DWT) are two widely used transforms. DCT has good de-correlative properties and is simple for VLSI implementation, so it is applied in most image and video coding standards. In recent years, DWT is becoming promising due to its several important properties for image coding:

- Offering flexible multi-resolution image representations;
- Avoiding "block effects" associated with the block based transform due to its global decomposition characteristic.

The DWT has been chosen by JPEG2000 [JPEG-2000] and used for intra-mode texture-coding in MPEG-4 [MPEG4-2001]. However, for scalable texture coding of arbitrarily shaped video objects, the conventional mathematical transform should be extended in order to handle the problem that arbitrarily shaped video objects have arbitrary numbers of lines and columns. Several modifications have been proposed [SIKORA-1995] [KAUFF-1997] [SHIPENG-2000], which can be classified into three classes. The first class is polynomial fitting, named as padding-based algorithm [SIKORA-1995] [EGGER-1996] [KATATA-1997]. In these approaches, the contour blocks from arbitrarily shaped video object planes (VOPs) are first padded into block regions. The padded blocks are then just handled by conventional transforms, such as DCT and DWT. However, this method often yields more signal samples to encode after the transforms and therefore is inefficient in compression. The coding artefacts are commonly seen along the object boundaries due to the signal padding.

The second class is shape-adaptive DCT (SA-DCT) based scheme [KAUFF-1997]. Shape adaptive DCT based scheme presented in [KAUFF-1997] is the most popular scheme and included by MPEG-4 Version 2. An attractive feature of this scheme is that the number of transform coefficients is exactly the same as that of the input samples. The steps for execution of SA-DCT are shown in Figure 6.1. When applied to a block not fully occupied by objects, SA-DCT first moves all pixels toward the upper block boundary. A variable basis DCT is applied to each column with the number of DCT basis functions equal to the number of coefficients in each column. The pixels are then moved toward the left block boundary, and a similar variable basis DCT on each row is applied horizontally. Since the SA-DCT always flushes samples in an arbitrarily shaped block to a certain edge of a rectangular bounding block before performing row or column DCT transforms, some spatial correlation may be lost, which reduces coding efficiency.



(a) Original segment (b) Vertical DCT (c) Horizontal DCT

Figure 6.1 – Steps for execution of SA-DCT: The pixels in grey correspond to samples inside the video object.

The third class is shape-adaptive discrete wavelet-based (SA-DWT) scheme [BARNARD-1993] [SHIPENG-2000]. One of the most popular SA-DWT algorithms is presented in the paper

[SHIPENG-2000]. In this algorithm, when the data length was longer than the filter length, it was first truncated to the next available even length and transformed directly with a circular wavelet transform. The extra data point in case the data length was odd was directly copied into the low frequency band. When the data length was shorter than the filter length, the Haar transform was applied. This scheme generates the exactly same number of coefficients as that of the original object.

Commonly, for image and video coding, SA-DWT should satisfy the following basic conditions [BARNARD-1993]:

1. The number of coefficients after SA-DWT should be identical to the number of pixels contained in the arbitrary shaped image region, which is a necessary condition for an efficient coding method.

2. The spatial correlation and other wavelet transform properties, such as locality and the self-similarity across subband should be well preserved. The subbands of all the regions should fit together without overlaps or gaps

3. For a rectangular region, the SA-DWT should be identical to a conventional wavelet transform.

The objective of this Chapter is to investigate scalable texture intra-coding of still arbitrarily shaped video objects. To achieve this, we will begin with a review of subband/wavelet analysis, and shape-adaptive discrete wavelet transform. Then, we will review some popular wavelet-based texture coding algorithms, such as Set Partitioning in Hierarchical Trees (SPIHT) algorithm [SAID-1996], Set-Partitioning Embedded Block (SPECK) algorithm [ISLAM-1999], and Embedded Block Coding with Optimized Truncation (EBCOT) algorithm [TAUB-2000], and their extensions for object-based texture coding. After reviewing, an improved shape-adaptive SPECK algorithm will then be presented and discussed in detail. Their performance will be evaluated and compared through extensive experiments. Some algorithms, such as SA_DWT, object-based SPIHT and object-based SPECK algorithms are attached in Appendix B.

6.2 Subband/Wavelet analysis

Subband/wavelet analyses have been widely used for image and video compression. For discrete wavelet transform (DWT), there are two kinds of implementation schemes: two-band filterbank convolution scheme and lifting scheme. The wavelet coefficients computed with these two schemes are identical. In this section, these two kinds of implementation schemes will be reviewed and discussed.

6.2.1 Two-band filterbank convolution scheme

The subband/wavelet filter banks (FBs) used for image/video compression applications should have the following properties: perfect reconstruction (PR), linear phase, finite impulse response (FIR), real coefficient, maximally decimated, and uniform band [VILLASENOR-1995]. Here are several of our justifications:

- The PR property is highly desirable since it provides a lossless signal representation and it simplifies the error analysis significantly.
- For image and video compression, it is also crucial that all analysis and synthesis filters have linear phase. Besides the elimination of the phase distortion, linear phase systems allow us to use simple symmetric extension methods to accurately handle the boundaries of finite-length signals.
- The filter length should be relatively short to prevent ringing artefacts in the reconstructed images and to keep the transform fast.
- For image and video compression, especially at low bit rates, we prefer maximally decimated FBs that do not expand the input signals.

There are also several solutions [VETTERLI-1986] for the filterbank design in order to realize perfect reconstruction (PR) for a two band split scheme. However, this Chapter does not want to discuss much on subband/wavelet FIR filterbank design. For review, please refer to some famous papers and books [ANTONINI-1992] [VETTERLI-1992] [VETTERLI-1995]. Here, we just want to review the standard 1D two-band subband analysis / synthesis scheme, as shown in Figure 6.2. The input signal is split into two parts by filtering with the low-pass filter H and the high-pass filter G. Both generated signals are then downsampled by a factor 2. We assume that these two signals are coded in a lossless manner and transmitted error free over the channel to the receiver side. There, the signals are decoded, upsampled by a factor 2, and filtered with the synthesis filter $\widetilde{H}$ and $\widetilde{G}$. After multiplication by two to restore the amplitude, both signals are added to obtain the reconstructed signal.

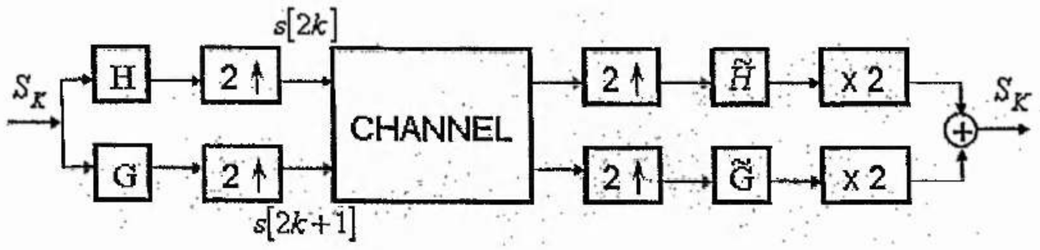


Figure 6.2 – 1D two-band analysis/synthesis system

To use the wavelet transform for image processing, we must implement a 2D version of the analysis and synthesis filter banks. Commonly, 2D separable wavelet decomposition is conducted to reduce the computation. In this case, the 1D analysis filter bank is first applied to the columns of the image and then applied to the rows. Suppose the image has N_1 rows and N_2 columns. After applying the 1D analysis filter bank to each column, we have two subband images, each having $N_1/2$ rows and $N_2/2$ columns. Then, applying the 1D analysis filter bank to each row of both of the two subband images, we have four subband images, each having $N_1/2$ rows and $N_2/2$ columns. The original image can be reconstructed perfectly from these four subband images by using synthesis process.

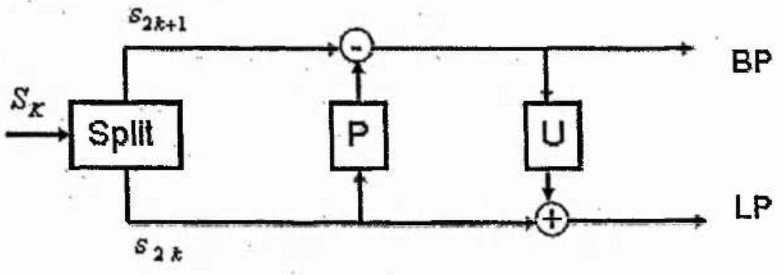
6.2.2 Lifting scheme

Recently, an alternative implementation of the subband decomposition or discrete wavelet transform (DWT) has been proposed, which is named as "lifting scheme" [CALDERBANK-1998]. The generic lifting analysis scheme consists of three steps, the *polyphase decomposition*, *the prediction step*, and the *update* step, as depicted in Figure 6.3 (a). At first, polyphase decomposition is conducted to separate the even and the odd samples of a given signal $S[k]$. For simplicity, it is assumed that $S[k]$ are scalar value. Since the correlation structure typically shows a local characteristic, the even and odd polyphase components are highly correlated, and therefore, in a subsequent step, a prediction of the odd samples from the even samples is performed. The corresponding prediction operator P for each odd sample $S_{odd}[k] = S[2k+1]$ is a linear combination of its neighbouring even samples:

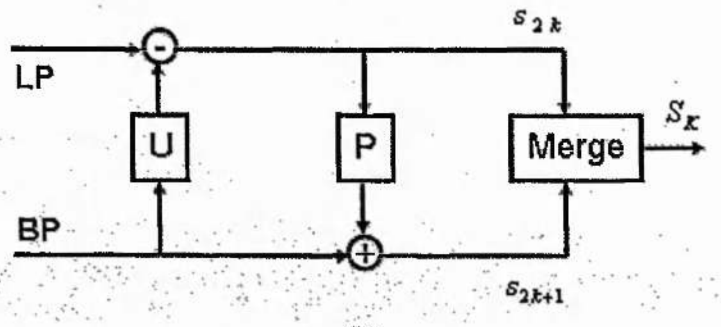
$$S_{even}[k] = S[2k] \tag{6.1}$$

and $$P(S_{even})[k] = \sum_{l} p_l S_{even}[k+l]. \tag{6.2}$$

where p_l is the lifting prediction coefficient.



(a)



(b)

Figure 6.3 – Wavelet analysis and synthesis scheme based on lifting: (a) analysis process and (b) synthesis process.

As a result of the prediction step, we replace the odd samples by their corresponding prediction residual $h[k] = S_{odd}[2k] - P(S_{even})[k]$. Note that the prediction step is equivalent to applying a high-pass filter of a two-channel filterbank [DAUBECHIES-1998] and, in case of video sequence coding, it is similar to motion-compensated prediction, as described in [WIEGAND-2003]. Finally, the update step of the lifting scheme is conducted in which a low-pass filtering is performed by updating the even samples $S_{even}[k]$ with a linear combination of the prediction residuals $h[k]$. The corresponding update operator U is given by:

$$U(h)[k] = \sum_l u_l h[k+l]. \tag{6.3}$$

$$l[k] = S_{even}[k] + U(h)[k] \tag{6.4}$$

where u_l is the lifting update coefficients.

By replacing the even samples with $l[k]$, the given signal $S[k]$ can finally be represented by $l[k]$ and $h[k]$, each at half sampling rate as $S[k]$. In lifting scheme, the lifting coefficients p_l and u_l decide the properties of wavelet transform.

Since both the update and the prediction step are fully invertible, the corresponding transform can be interpreted as a critically sampled perfect reconstruction filterbank. In fact, it has been shown that any biorthogonal family of FIR filters can be realized with a sequence of prediction and update steps [DAUBECHIES-1998]. Figure 6.3 (b) shows the synthesis process of the lifting scheme. It simply consists of the application of the prediction and update operator in the reversed order with inverted signs on the summation process, followed by the reconstruction process using the even and odd polyphase components.

From the structure in Figure 6.3, it can be found that the lifting scheme provides in-place computation of wavelet coefficients by overwriting the memory locations that contain the input sample values. This provides a significant reduction in the memory usage. Lifting scheme also decreases the computational complexity to achieve DWT. Because of these advantages, lifting implementation scheme has been included in JPEG 2000 standard [JPEG-2000].

At present, two DWT kernels in terms of the lifting coefficients are popularly used for image/video coding. They are Haar wavelet and the 5/3 bi-orthogonal spline wavelet, which are presented as follows:

The Haar wavelet: in the case of the Haar wavelet, the prediction operator P_{Haar} and update operator U_{Haar} are simply given by:

$$P_{Haar}(S_{even})[k] = S[2k] \tag{6.5}$$

$$U_{Haar}(h)[k] = \frac{1}{2}h[k], \tag{6.6}$$

such that $h[k] = S[2k+1] - S[2k]$ (6.7)

and $l[k] = S[2k] + \frac{1}{2}h[k] = \frac{1}{2}(S[2k] + S[2k+1])$. (6.8)

where $l[k]$ and $h[k]$ correspond to the low-pass and high-pass (analysis) output of the Haar filter, respectively.

The 5/3 bi-orthogonal spline wavelet: The low- and high-pass analysis filters have 5 and 3 taps, respectively, for the 5/3 spline wavelets. Its simplicity, together with a remarkable good performance in still image coding (like JPEG 2000), makes its popular use for image and video subband coding. In the lifting framework, the corresponding prediction operator $P_{5/3}$ and update operator $U_{5/3}$ of 5/3 transform are given by:

$$P_{5/3}(S_{even})[k] = \frac{1}{2}(S[2k] + S[2k+2]) \tag{6.9}$$

$$U_{5/3}(h)[k] = \frac{1}{4}(h[k] + h[k-1]), \tag{6.10}$$

such that:

$$h[k] = S[2k+1] - \frac{1}{2}(S[2k] + S[2k+2]), \tag{6.11}$$

$$\begin{aligned} l[k] &= S[2k] + \frac{1}{4}(h[k] + h[k-1]) \\ &= \frac{3}{4}S[2k] + \frac{1}{4}(S[2k+1] + S[2k-1]) - \frac{1}{8}(S[2k+2] + S[2k-2]) \end{aligned} \tag{6.12}$$

6.3 Shape-adaptive discrete wavelet-transform (SA-DWT)

This section gives a review of the shape-adaptive DWT algorithm. There are several advantages of SA-DWT over the conventional DWT for image/video compression. First, the subband analysis of a region is totally independent of neighbouring regions, so quantisation errors only have local effect. Second, the high frequency subbands generated by SA-DWT contain less energy than those of the conventional DWT. Commonly, the energy in the high frequency subbands of the conventional DWT mainly corresponds to the strong transitions in the original image, and the transition information is spread over subbands in several levels and several directions. Such a spread results in a significant cost for the coding system, but SA-DWT decreases such transition coming from the object boundaries. Since filtering across the sharp edges located at the objects' contour is avoided, ringing artefacts near these edges will be almost eliminated. Finally, the bit allocation would be able to favour important regions in a video frame, since bits can be assigned to each object and each subband segment separately.

Shape adaptive discrete wavelet transform is needed for efficiently coding arbitrarily shaped visual objects, which is essential for object-oriented multimedia applications. In fact, shape-adaptive discrete wavelet transform is a more delicate work. Its subsampling techniques and extension methods should be treated attentively so as not to lessen the performance of final coding. Commonly, the selected filters and the corresponding subsampling / extension techniques affect the final coding performance [SHIPENG-2000].

Much research has been conducted on SA_DWT [BARNARD-1993] [SHIPENG-2000]. In [BARNARD-1993], the authors gave overview of what needed to be done and how to do for SA-DWT, but they do not provide a good solution. A comprehensive description of shape-adaptive

DWT algorithm is presented in Li's paper [SHIPENG-2000]. In his paper, wavelet decomposition methods of using different wavelet filters, such as orthogonal filters, even symmetric biorthogonal filters, and odd symmetric biorthogonal filters in the SA-DWT are discussed in details. Much research has shown that odd symmetric biorthogonal wavelet filters can achieve better compression performance [VILLASENOR-1995]. Another advantage of odd symmetric biorthogonal wavelets is that they can provide linear phase filters, hence, eliminating the phase distortion caused by magnitude distortion of transformed coefficients. This is very important when they are applied for image and video compression where magnitudes of the transformed coefficients are mostly likely to be quantised. Therefore, odd symmetric biorthogonal wavelet filters have been selected as the default wavelet filters in MPEG-4 standard [MPEG4-2001]. In our research, only the case of arbitrary length wavelet decomposition using odd symmetric biorthogonal wavelets is investigated and employed. Refer to [SHIPENG-2000] for more details of other two cases.

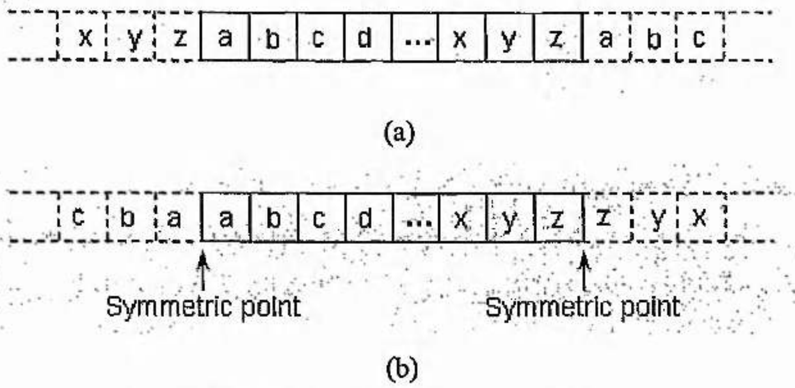


Figure 6.4 – Example of (a) periodic extension and (b) symmetric extension

Two issues should be clarified related to SA-DWT. The first one in applying wavelet decomposition for finite-length signal segment is how to deal with the boundary of the signal segment. In order to maintain the perfect reconstruction property of wavelet transform, boundary extension is necessary, such as periodic extension and symmetric extension as shown in Figure 6.4. If the signal segment is long, the correlation between the end of the signal and the start of the signal is small. There could be a good chance of a sharp change at the transition from the end of previous signal period to the start of the next signal period if the periodic extension method is used. In the symmetric extension scheme, the signal is extended symmetrically at the leading and trailing boundaries of a signal segment. The neighbouring samples with such symmetric extensions have the same close correlation as in the original signal segment. Therefore, shape transitions are avoided.

The other issue is the subsampling for arbitrary length signal segment at arbitrary locations. A proper subsampling method is important for the SA-DWT. One consideration for the subsampling is that it should keep the spatial correlation, locality, and self-similarity across subbands so that 2-D separable wavelet decompositions and pyramid wavelet decomposition can still be applied to the arbitrarily shaped image region without loss of spatial correlation. The subsampling strategy can also affect the efficiency of zerotree coding. Commonly, the subsampling process is decided by the employed wavelet filters. Let $\{g(i), i = 0, \cdots, L_g - 1\}$, $\{h(i), i = 0, \cdots, L_h - 1\}$, $\{\bar{g}(i), i = 0, \cdots, L_h - 1\}$, and $\{\bar{h}(i), i = 0, \cdots, L_g - 1\}$ be the impulse response of the low-pass analysis filter, high-pass analysis filter, low-pass synthesis filter and high-pass synthesis filter, respectively. The filter lengths, both L_g and L_h, are odd numbers. Let $x(i)$ be the input signal with the finite length with appropriate extensions at the leading and trailing boundaries. The relations between these filters and wavelet analysis/synthesis process can be summarized in Appendix B.1

Assuming a signal segment $\{x(j), j = 0, \cdots, N-1\}$, with length N, and combining symmetric extensions, filtering and subsampling together, the arbitrary length wavelet decomposition using odd symmetric wavelet transforms can be described as follows (s is defined in Appendix B.1 to indicate subsampling position):

1. If $N = 1$, this isolated sample is repeatedly extended and the low-pass wavelet analysis filter is applied to obtain a single low-pass wavelet coefficient. The synthesis process simply scales this single low-pass coefficient by a factor of $1/K$ ($K = \sum_{i=0}^{L_g-1} g(i)$), and puts it in the correct position in the original signal domain.

2. If N is greater than 1, and $p = 0$ if N is even, $p = 1$ if N is odd, the signal segment is extended using the type shown in Figure 6.4 (b). The $(N/2 + p(1-s))$ low-pass wavelet coefficients $C(i), i = s, \cdots, (N/2 - (1-p)(1-s))$, are generated by Equation (B_1.5), (Equation in (B_1.7) in Appendix B.1). The $(N/2 + ps)$ high-pass wavelet coefficients $D(i), i = 0, \cdots, (N/2 - 1 + ps)$, are generated by (B_1.6), (Equation in (B_1.8) in Appendix B.1). The synthesis process begins with upsampling the low- and high-pass wavelet coefficients using (B_1.9) and (B_1.10), respectively. As a result, an upsampled low-pass segment $P(j)$ and an upsampled high-pass segment $Q(j)$ are obtained, where $j = 0, \cdots, N-1$. The up-sampled low- and high-pass segments $P(j)$ and $Q(j)$ are then extended using the type in Figure 6.4 (b). The extended low-pass and high-pass signal

$P(j)$ and $Q(j)$ are then synthesized using (B_1.11) and (B_1.12) to reconstruct the signal segment $r(j), j = 0, \cdots, N-1$.

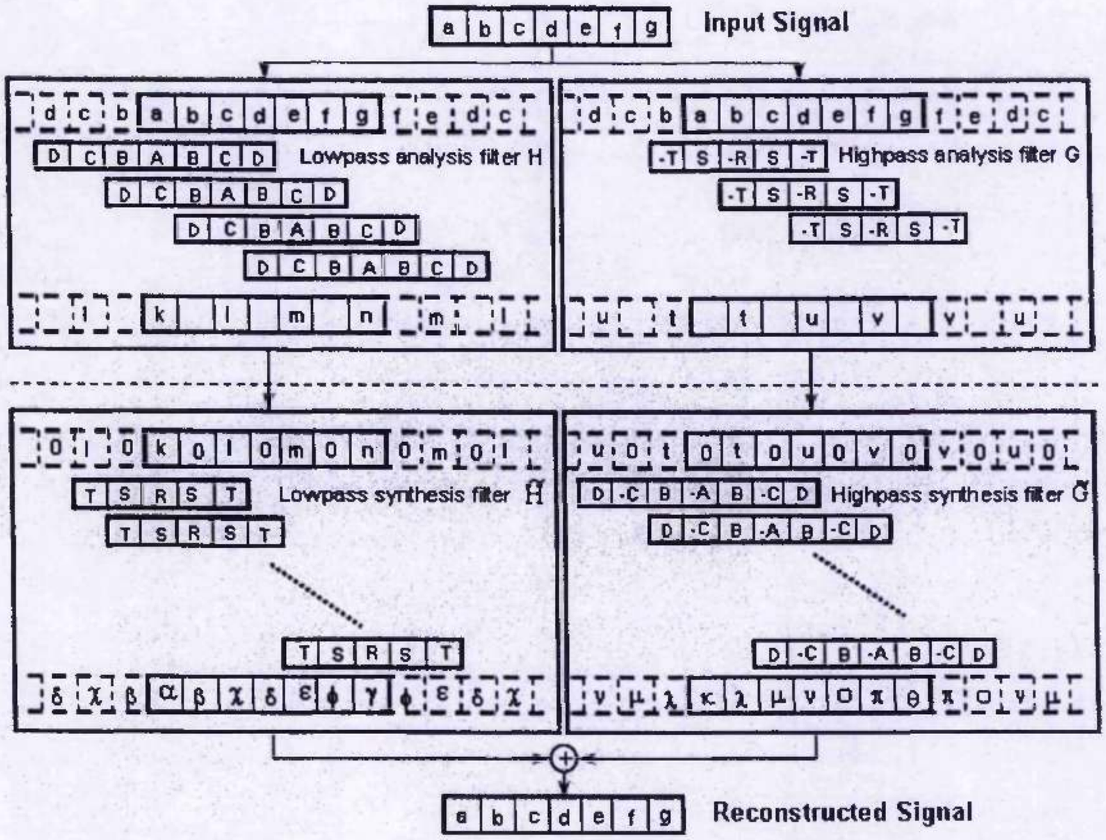

Input Signal
Lowpass analysis filter H
Highpass analysis filter G
Lowpass synthesis filter H̃
Highpass synthesis filter G̃
Reconstructed Signal

Figure 6.5 – Example of SA-DWT with even-sampling for the low-pass wavelet coefficients and odd-sampling for the high-pass wavelet coefficients for an odd length signal segment [SHIPENG-2000].

Figure 6.5 examplifies the analysis and synthesis process with even-sampling for the low-pass wavelet coefficients and odd-sampling for the high-pass wavelet coefficients for an odd length signal segment.

6.4 Wavelet-based texture coding of video frame

For image and video compression, DWT-based coding algorithms have been proved to outperform DCT-based coding techniques by a wide margin, in terms of compression efficiency and enhanced feature such as scalability. That is the reason why both MPEG-4 and JPEG2000 have selected wavelet-based scheme as the basis for coding still texture and images [MPEG4-2001] [JPEG-2000]. In this section, the main wavelet-based coding algorithms are reviewed. They are SPIHT algorithm [SAID-1996], SPECK algorithm [ISLAM-1999] and EBCOT algorithm [TAUB-2000]. All of these algorithms employ some kind of significant testing of sets or groups of pixels, in which the set is tested to determine whether the maximum magnitude in it is above a certain threshold. The results of these significant tests determine the path taken by the encoder to code the source samples. These significance testing schemes are based on some very simple principles which allow them to exhibit excellent performance. Among these principles is the partial ordering of magnitude coefficients with a set-partitioning sorting algorithm, bit plane transmission in decreasing bit plane order, and exploitation of self-similarity across different scales of an image wavelet transform.

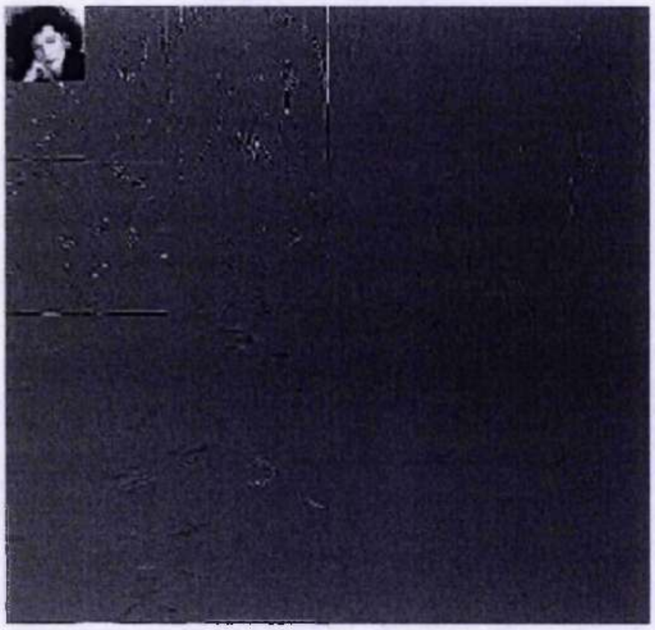


Figure 6.6 – Three-layer wavelet decomposition of "Lady" image

All of these algorithms consist of three steps: (1) Discrete wavelet transform (DWT); (2) quantisation; and (3) entropy coding. DWT is used to generate the subband samples. Figure 6.6 shows the wavelet decomposition result of "Lady" image. The original image is represented in terms of a collection of subbands, which are organised into increasing resolution levels. The wavelet coefficients are typically organised into a hierarchical data structure, so that bit allocation and data compaction can be employed more efficiently.

6.4.1 SPIHT

SPIHT algorithm (Set partition in hierarchical tree algorithm) was proposed by A. Said and W. A. Pearlman [SAID-1996]. It is the refinement of embedded zerotree wavelet (EZW) algorithm of J.

Shapiro [SHAPIRO-1993], which imposes a hierarchical quadtree structure on wavelet-transformed image. But in this algorithm, the crucial parts of the coding process are fundamentally different from EZW technique in that the subset partitioning of SPIHT algorithm is so effective that even un-encoded binary bit stream can achieve better performance than EZW algorithm.

SPIHT algorithm can achieve progressive image coding and transmission easily. The encoding algorithm orders the coefficients by magnitude and transmits the most significant bits first. It can be stopped at any allocated number of bits, or at any peak signal to noise ratio (PSNR) value. In the SPIHT algorithm, the parent-children dependencies of subbands are the same as that in EZW algorithm, which are shown in Figure 6.7 (a). The coefficient at the coarse scale is called the ***parent***. All coefficients corresponding to the same spatial location at the next finer scale of the similar orientation are called ***offsprings***. For a given parent, the set of all coefficients at all finer scales of similar orientation corresponding to the same location are called ***descendants***. Similarly, for a given child, the sets of coefficients at all coarser scales of similar orientation corresponding to the same location are called ***ancestors***. The ***tree*** is defined in such a way that each node has either no children (the leaves) or four children.

Given a threshold level T, a coefficient x is said to be an element of a ***zerotree*** for threshold T if itself and all of its descendents are insignificant with respect to T. An element of a zerotree for threshold T is a ***zerotree root*** if it is not the descendant of a previously found zerotree root for threshold T, which is encoded with a special symbol indicating that the insignificance of the coefficients at finer scale is completely predictable.

In Figure 6.7 (a), the arrows are oriented from the parent node to its four children. The pixels in the highest level of the pyramid are the tree roots and are grouped in 2 by 2 adjacent pixels. However, the children branching rule is different, and in each group, one of them has no descendants. In Figure 6.7 (b), it is shown that the parents must be scanned before children, and all positions in a given subband are scanned before the scanning of the next subband.

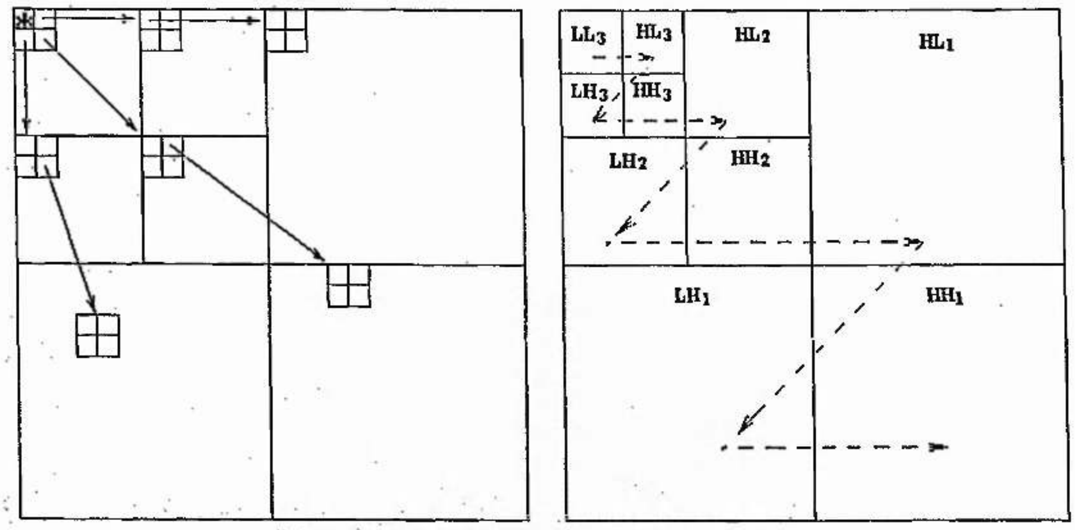


Figure 6.7 – (a) the parent-children dependencies of subbands; and (b) scanning order during coding significant map of SPIHT algorithm

The significant test order is very important for the synchronization between encoder and decoder without any header information. There are three lists in SPIHT algorithm. They are the list of significant pixels (LSP), the list of insignificant pixels (LIP), and the list of insignificant sets (LIS). They are initialized by different sets. We refer to the set of root node and corresponding descendants as a spatial orientation tree (SOT). The LSP is initially empty, the LIP is initialized with the elements of the lowest frequency subband, and the LIS is initialized with the root of each SOT.

During encoding, a threshold is needed to decide the significance. Commonly, it is chosen as $T(0)=2^{n_0}$, where n_0 is selected such that the largest pixel magnitude, say M, satisfies $2^{n_0} \le M < 2^{n_0+1}$. The encoding process is progressive in pixel magnitude, using a sequence of thresholds $T(n)=2^{n_0-n}$, $n=1,2,\cdots$. At stage n, all pixels with magnitudes satisfying $T(n) \le |x| < 2T(n)$ are identified as significant and their positions and sign bits are encoded.

The encoding process includes two passes: sorting pass and refinement pass. During the sorting pass, the pixels in the LIP --- which were insignificant in the previous pass --- are tested, and those that become significant are moved to the LSP. Similarly, sets are sequentially evaluated following the LIS order, and when a set is found to be significant it is removed from the list and partitioned. The new subsets with more than one element are added back to the LIS, while the single-coordinate sets are added to the end of LIP or the LSP, depending on whether they are significant or insignificant, respectively. In the refinement pass, the LSP contains the coordinates of the pixels that are visited and refined by encoding the n-th most significant bit. Refer to the paper [SAID-1996] for the more detailed description on SPIHT algorithm.

6.4.2 SPECK

The SPECK coding algorithm belongs to a class of embedded tree structured significance mapping schemes [SHAPIRO-1993] [SAID-1996] [ISLAM-1999]. It exploits two fundamental characteristics of a wavelet-transformed image – the well-defined hierarchical structure, and energy clustering in frequency and in space. However, SPECK algorithm differs from SPIHT and EZW algorithms in that it does not use trees which span and exploit the similarity across different subbands; rather it makes use of sets in the form of blocks. It mainly exploits the clustering of energy in frequency and space in hierarchical structures of transformed images. In the SPECK algorithm, the quadtree is formed by successive recursive splitting of a subband block (parent) into four quadrants children. The pixels are grouped together in sets which comprise of regions in the transformed image. The transformed image is partitioned into two sets: set S and set I, as shown in Figure 6.8 (a). Only two linked lists are maintained in the SPECK algorithm: LIS – List of insignificant sets, and LSP – List of significant pixels. The former list contains sets of type S of varying sizes which have not yet been found significant against a threshold n, while the latter list obviously contains those pixels which have tested significant against n. Alternatively, as will become obvious later on, an array of smaller lists of type LIS is used, each containing sets of type S of a fixed size, instead of using a single large list having sets S of varying sizes. Use of multiple lists will speed up the encoding/decoding process.

The SPECK coding algorithm consists of the initial step, the sorting pass and the refinement pass. The algorithm starts by partitioning the image into two sets: set S, which is the root of the pyramid (or the coarsest level), and set I which is everything that is left of the image after taking out the root (see Figure 6.8 (a)). The dimension of set S depends on the dimension of the original image and the subband level of the pyramidal structure at which the set lies. To start the algorithm, set S is added to LIS.

In the sorting pass, when data set S is significant, it is partitioned into four subsets $O(S)$, as shown in Figure 6.8 (b); each of these four child sets is further tested and partitioned until all the significant coefficients are found. The significance test results of the four subsets are all coded together before further processing the subsets. For example, the significant test result of the first set is coded without any context, while the significant test result of the second subset is coded using the context of the first coded subset, and so on. In this way, previously coded subsets form the context for the subset being currently coded. Once all sets of type S are processed, the set I is processed by testing it against the same threshold n. If it is found to be significant, it is partitioned by another partition scheme as shown in Figure 6.8 (c).

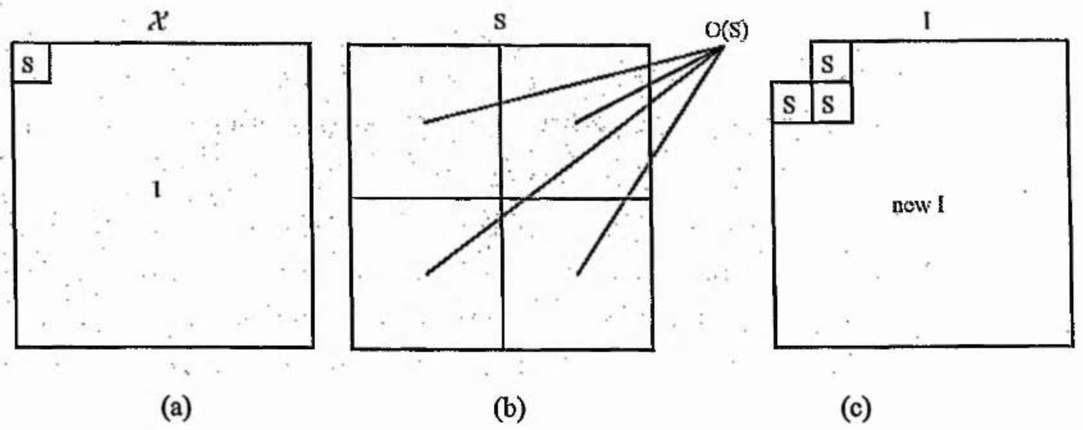


Figure 6.8 – Set partitioning in SPECK algorithm: (a) Partitioning of image X into set S and I; (b) Partitioning of set S; and (c) Partitioning of set I

In the refinement pass, the significant coefficients found in the sorting pass are refined and transmitted to decoder according to the bit-plane transmission. Once the refinement pass has been done, the threshold is lowered and the sequence of sorting pass and refinement pass is repeated against the lower threshold. This process is repeated until the desired rate is achieved. For detailed information about the SPECK algorithm, please refer to the original papers [ISLAM-1999].

6.4.3 EBCOT

EBCOT algorithm is proposed by Taubman [TAUB-2000], which is related in various degrees to much earlier work on scalable image compression, such as Shapiro's EZW algorithm [SHAPIRO-1993], Said and Pearlman's SPIHT algorithm [SAID-1996] and Taubman and Zarhor's LZC algorithm [TAUB-1994]. It has been adopted in JPEG2000 [JPEG-2000] due to its high compression efficiency as well as other excellent features, including resolution and SNR scalability. Unlike SPIHT algorithm, which uses spatial orientation trees, EBCOT algorithm partitions each subband into relatively smaller code blocks. Every code block is encoded separately so that a highly embedded bitstream is generated separately for each code block. Given a target bit-rate R^{max}, each of the independent code-block bit-streams can be truncated in an optimal way so as to minimise distortion subject to the bit-rate constraint, which is referred as post-compression rate-distortion (PCRD) optimisation.

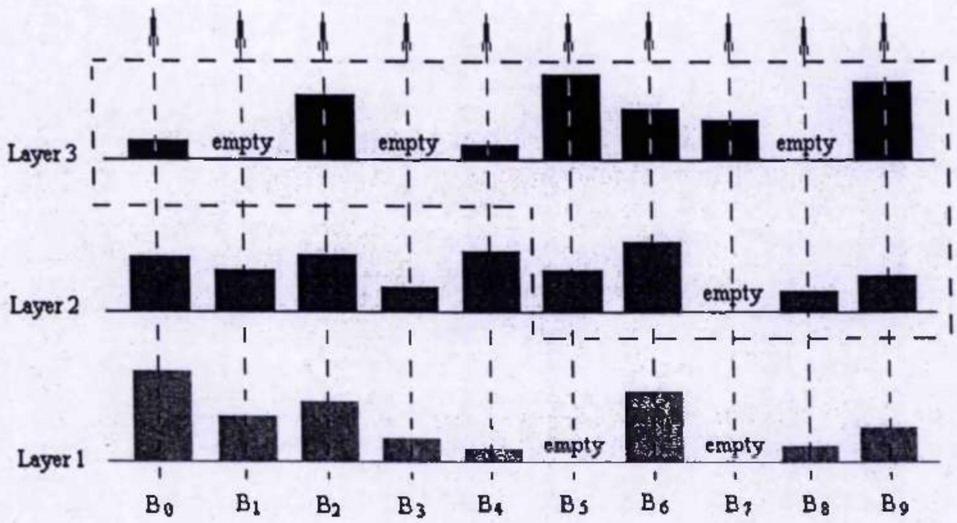


Figure 6.9 – Progressive representation of embedded code-block bit-streams in quality layers. The blocked region identifies the block contributions, which are discarded by truncating the bit-stream between layer1 and 2.

Figure 6.9 illustrates the layered bitstream concept. It also illustrates the effect of truncating the bit-stream between the first and second layers. Each quality layer must include auxiliary information to identify the size of each code-block contribution to the layer. When the number of layer is large, only a subset of the code-blocks will contribute to any given layer, introducing substantial redundancy in the auxiliary information. EBCOT introduces a two-tiered coding strategy to compress the auxiliary information for each quality layer, as shown in Figure 6.10.

During the embedded block coding, four different primitive coding operations form the foundation of the embedded block coding strategy of EBCOT algorithm. The primitives are used to code new information for a single sample in some bit-plan. If the sample is not yet significant, a combination of the "Zero coding" (ZC) and "run-length coding" (RLC) primitives is used to code whether or not the symbol becomes significant in the current bit-plane; if so, the "Sign coding" (SC) primitive must be invoked to identify the sign. If the sample is already significant, the "magnitude refinement" (MR) primitive is used to encode the bit for the magnitude of current bit-plane. In every case, a single binary-values symbol must be coded using the arithmetic coder.

The probability model used by an arithmetic coder involves the following contexts: nine for the zero coding primitive, one for the RLC primitive, five for the sign coding primitive, and three for the magnitude refinement primitive.

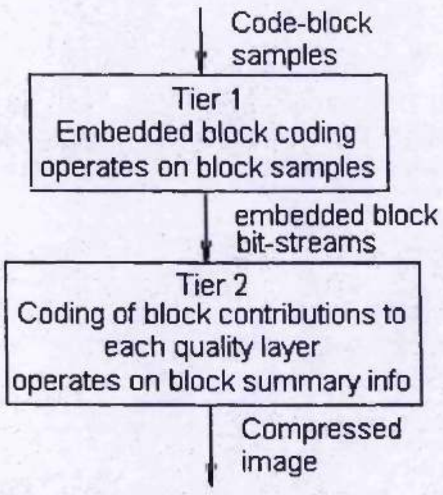

Code-block samples
Tier 1
Embedded block coding
operates on block samples
embedded block bit-streams
Tier 2
Coding of block contributions to each quality layer
operates on block summary info
Compressed image

Figure 6.10 – Two-tiered coding structure of the EBCOT image compression algorithm

- Zero coding (ZC): This primitive is used to code whether an insignificant coefficient changes to a significant one in the current bit-plane. Eight immediate neighbours of the current coefficient are used to determine the context mode, which are decided by the number of significant neighbours in horizontal direction h, vertical direction v and diagonal direction d, shown in Figure 6.11.

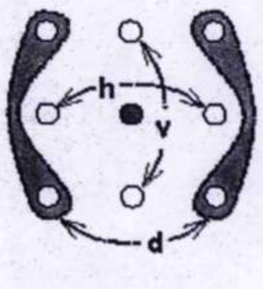


Figure 6.11 – Encoded sample and its neighbours

- Run-length coding (RLC): This primitive is used to reduce the average number of binary symbols which must be processed by the arithmetic coding engine. It is invoked in place of the ZC primitive when a horizontal run of insignificant samples is encountered and whose immediate neighbours are all insignificant.

- Sign coding (SC): This primitive is used to code the sign of a coefficient when it becomes significant.

- Magnitude Refinement (MR): The objective of this primitive is to code the value of a significant coefficient in the current bit-plane. The contexts are decided by the significance of its neighbours.

Four encoding passes are defined in EBCOT algorithm which formulates the possible truncation points for each bit-plane. They are "Forward significant propagation pass", "Reverse significant propagation pass", "Magnitude refinement pass", and "Normalisation pass". Paper [TAUB-2000] gives a detailed description on how to process these passes and generate the bitstream.

After generating the bitstream of all code blocks, the second tier coding engine is ignited, which is responsible for efficiently identifying the contribution of each code-block to each bit-stream layer, along with other summary information for the code-blocks. Rate-distortion optimisation algorithm is developed to allocate the bits among the code-blocks [TAUB-2000]. The inter-block redundancy is exploited in the second tier coding engine, which only operates on the summary information for the whole code-blocks, rather than individual samples.

EBCOT algorithm also supports region-of-interest (ROI) coding although in this algorithm the transform and coding of the ROIs and the background are not done separately in the wavelet domain. For detailed information about the EBCOT algorithm, please refer to the original papers [TAUB-2000].

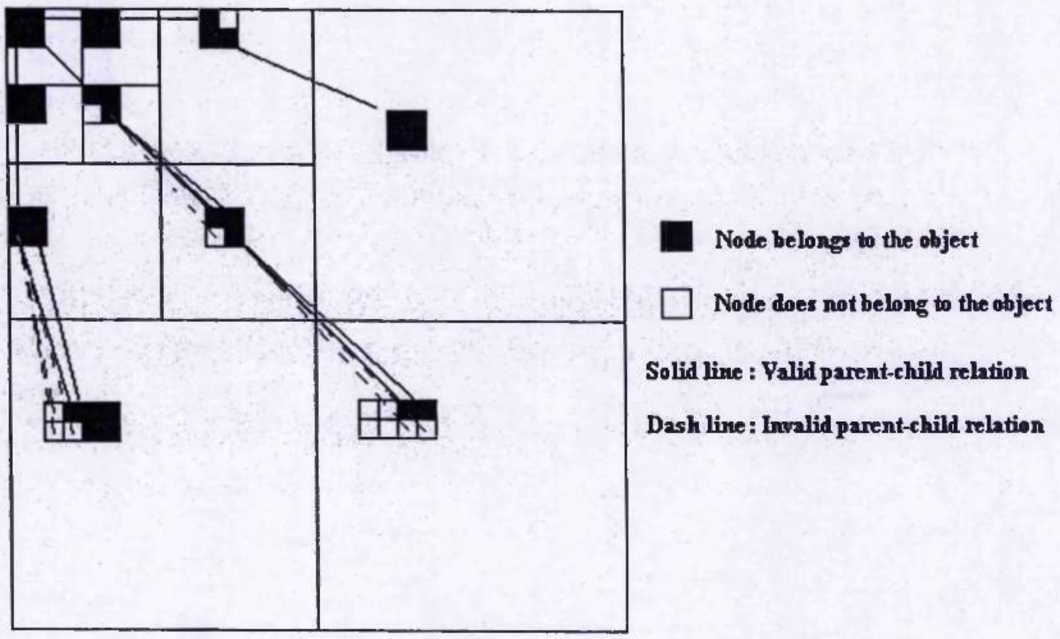
6.5 Wavelet-based texture coding of arbitrarily-shaped video objects

In this section, a number of shape-adaptive texture coding techniques are reviewed, which are the extensions of the methods discussed in Section 6.4. A straightforward approach is to consider transparent regions to be permanently "insignificant", such that significant pass (or sorting pass) and refinement pass of the encoder can process these transparent regions in a manner identical to that of other insignificant coefficients. Although most approaches are based on this general idea, depending on the particular method of significant-map coding involved, additional refinements are often possible to increase performance. In this section, the main algorithms for texture coding of arbitrarily-shaped video objects are first reviewed. Then, a proposal for an improved shape-adaptive SPECK algorithm will be presented and discussed.

6.5.1 Extension of existing algorithms for object-based texture coding

6.5.1.1 Shape-adaptive SPIHT algorithm (SA-SPIHT)

Much research has been conducted to extend the SPIHT algorithm for arbitrarily shaped object coding (e.g. [MINAMI-2001]). The aim is to modify the original SPIHT algorithm so that extraneous coefficients, which are not in the arbitrary region-of-support (AROS), are not encoded. Similar to shape-adaptive wavelet decomposition presented in Section 6.3, the shape image is also decomposed into a pyramid of subbands, called shape mask pyramid. In this way, the regions which belong to the object in each subband are known by both the encoder and decoder. We create the shape mask pyramid to represent the shape of the video object in each subband. Each pixel of the shape mask has the 2-bit mask value: 1 bit is used to distinguish whether the current wavelet coefficient is within the object; the other bit is used to tell whether its child branch is within the object. The child branch is defined according to the parent-child relations in the SA-SPIHT algorithm as shown in Figure 6.12.



■ Node belongs to the object
□ Node does not belong to the object
Solid line : Valid parent-child relation
Dash line : Invalid parent-child relation

Figure 6.12 – Parent-child relation of wavelet trees of SA-SPIHT algorithm

For SA-SPIHT algorithm, a spatial orientation tree is skipped if all coefficients in the tree are not in the AROS. This is simply done by not putting the coordinates of the root node (in the lowest frequency band) of the tree in the LIP and the LIS in the SPIHT initialization step.

For a spatial orientation tree with some coefficients not in the AROS, the significance test of a coefficient in the tree is skipped if that coefficient is not in the AROS. Likewise, the significance test of a subset in the tree is skipped if all coefficients in the subset are not in the AROS. As sign bits and refinement bits are only associated with the coefficients in the AROS, no modification are needed in related parts of the original SPIHT algorithm for these bits. Finally, when all coefficients in a spatial orientation tree are in the AROS, the tree is coded in the same way as in the original SPIHT algorithm.

If the AROS is the whole image, the SA-SPIHT algorithm described above will give exactly the same R-D performance as the original SPIHT algorithm does. So the SA-SPIHT algorithm includes the original SPIHT algorithm as a special case. The detail description of SA-SPIHT algorithm is listed in the Appendix B.2.

6.5.1.2 Shape-adaptive SPECK algorithm (SA-SPECK)

A straightforward extension of the SPECK algorithm in [ISLAM-1999] to encoding video objects of arbitrary shape is that we set all the coefficients outside the object in each subband to zero. Then the original SPECK algorithm can be applied just as if the support of the object were rectangular. No modification of the algorithm would be required. This method is inefficient, since one bit must be transmitted to tell the decoder that the node or branch outside the object is insignificant under each threshold.

In order to improve its performance, an efficient SA-SPECK algorithm was proposed [LU-2001]. Similar to SA-SPIHT algorithm, in SA-SPECK algorithm, the shape image is also decomposed into a pyramid of subbands, called the shape mask pyramid. The regions, which belong to the object in each subband, are known by both the encoder and the decoder. When the spatial orientation tree is constructed, which node and/or child branch are inside/outside the video object is known. Before the coding process, the node and branch outside the video object are pruned. During the sorting pass in SA-SPECK algorithm, those nodes and branches are not added to any list of LSP, LIP and LIS. Therefore, no information about these nodes and branches are transmitted. When the encoder and decoder scan these nodes and branches, they will be informed by the shape mask pyramid and skip over them.

The parent-child relation in SA-SPECK algorithm is illustrated in Figure 6.13. In this figure, the branches, which correspond to the nodes outside the object (represented by the dash arrows), are pruned before the encoding process begins. The detail description of SA-SPECK algorithm is listed in the Appendix B.3.

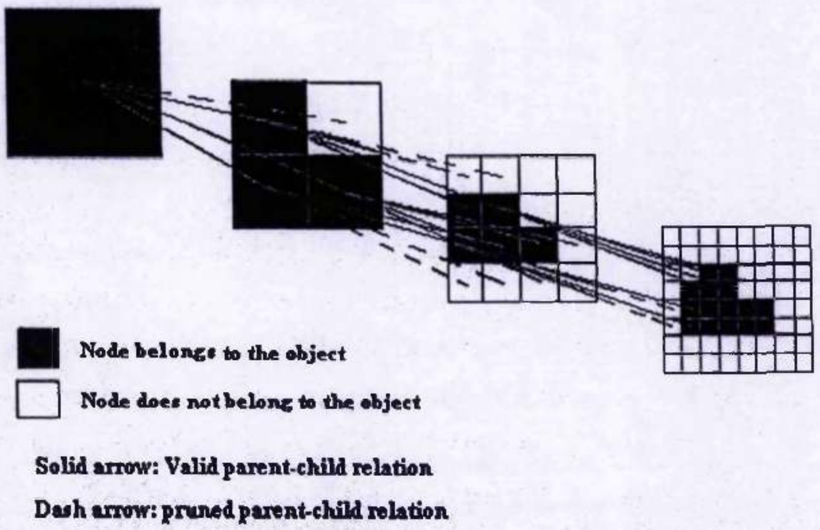

Node belongs to the object
Node does not belong to the object
Solid arrow: Valid parent-child relation
Dash arrow: pruned parent-child relation

Figure 6.13 – Parent-child relationship in the SA-SPECK algorithm

6.5.1.3 Shape-adaptive EBCOT algorithm (SA-EBCOT)

For object-based extension of EBCOT algorithm, a modification is made so that it only scans and codes the coefficients inside the video object after performing the shape-adaptive wavelet decomposition [LIU-2002]. For all the three bit scan passes, namely the significant propagation pass, the magnitude refinement pass and the normalization pass, all coefficients outside the video object are skipped. The same bit-plane coding primitives (ZC, SC, and MR) are used in the extended EBCOT, but they are only applied to wavelet coefficients inside the video object. A direct neighbour outside the object is always treated as an insignificant neighbour. During our research and experiments, this algorithm has not been implemented and tested. But from the reported results in [LIU-2002], good compression results can be achieved.

6.5.2 Proposal for texture coding of arbitrarily-shaped video object

In this section, an improved variant of SA-SPECK algorithm designed specifically for shape-adaptive texture coding has been proposed and discussed. The reason that I choose SPECK algorithm is its good compression performance as well as fast compression speed [ISLAM-1999]. Furthermore, it is easy to implement by using software.

The improved shape-adaptive SPECK algorithm employs two tactics:

1. Aggressive discarding of transparent regions from sets after partitioning as shown in Figure 6.14;

2. Context-adaptive binary arithmetic codec (CABAC) [MARPE-2003] is employed to code the sign and significant map. For the original SA-SPECK algorithm, the significance information, the sign, and the bits during the refinement pass are encoded using adaptive arithmetic codec (AAC). However, it is found the context information around the pixel to be encoded can be exploited to improve the coding performance, just as the EBCOT algorithm [TAUB-2000].

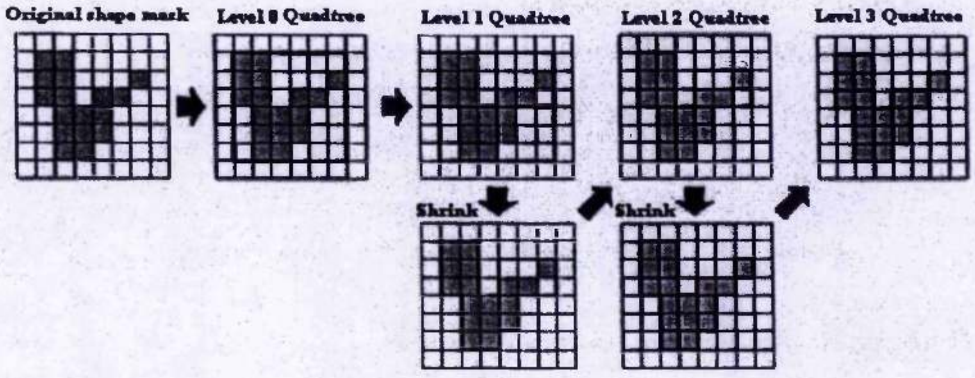


Figure 6.14 – The "shrunk" process to the bounding box of the opaque coefficients in the set

The improved SA-SPECK algorithm starts by splitting the set of transform coefficients λ into individual subbands S which are placed in a list of insignificant sets (LIS). Afterward, the algorithm follows the common bitplane-coding paradigm, such as SPECK algorithm consisting of the sorting and refinement passes.

Similar to the original SA-SPECK, the improved algorithm determines the significance of a set by comparing the largest opaque-coefficient magnitude to the current threshold. Sets without a significant coefficient are placed in the list LIS. During the sorting pass, each set in LIS is tested for significance against the current threshold. If the set becomes significant, it is split into four subsets according to the quadtree decomposition structure illustrated in Figure 6.14. When a set of coefficients S is split during the sorting pass, the four new subsets, S_1, S_2, S_3, and S_4, are placed into an LIS, recursively tested for significance and split again if needed. Additionally, in the sorting pass, before the set S is added to an LIS, the set is "shrunk" to the bounding box of the opaque coefficients in the set as shown in Figure 6.14. Similar to the original SA-SPECK algorithm, the improved algorithm encodes the significance test results of the four subsets jointly before further processing the subsets. If the first three subsets are empty or insignificant, the algorithm can deduce the significance of the fourth subset without sending any bits. During significant coding of set S, the context model is decided from its neighbouring sets, which are shown in Figure 6.16.

In the improved SA-SPECK algorithm, context-adaptive binary arithmetic codec (CABAC) [MARPE-2003] is applied to code the significant map, sign bits and refinement bits, in order to employ the strong dependency among subbands through modelling contexts. By combining an adaptive binary arithmetic coding technique with context modeling, CABAC can achieve a high degree of adaptation and redundancy reduction. It significantly outperforms the baseline entropy coding method of H.264/AVC for the typical area of envisaged target applications. The CABAC encoder block diagram is illustrated in Figure 6.15. The encoding process of CABAC consists of three elementary steps: Binarization; Context modeling; and binary arithmetic coding.

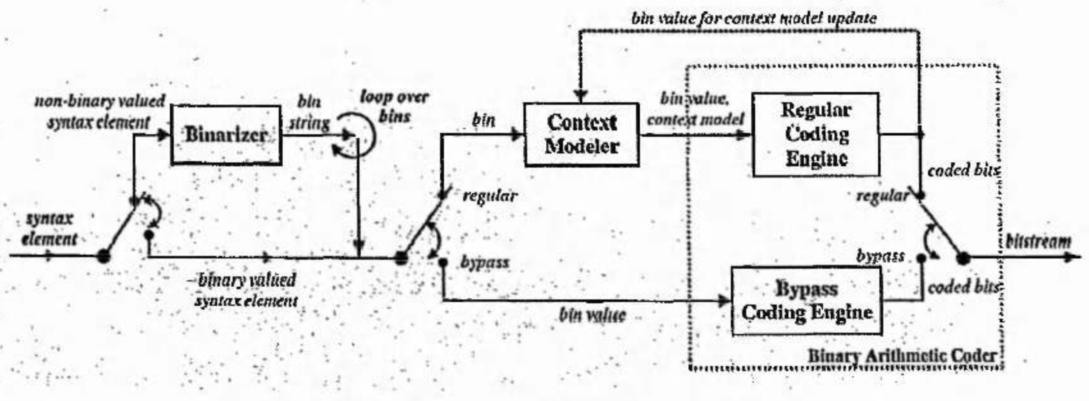


Figure 6.15 – CABAC encoder block diagram [MARPE-2003]

Since it is the context model that determines the coding efficiency in the first place, it is of paramount importance to design an adequate context model that explores the statistical dependencies to a large degree and that this model is kept "up to date" during encoding. In the improved SA-SPECK algorithm, different contexts are used for significant bits, sign bits and refinement bits coding.

For significant bit coding, the neighbouring nodes, shown in Figure 6.16 (a), are included in the modelling contexts. Eight spatial adjacent nodes from the same subband are utilised to exploit intraband correlation. Such a contextual structure has been employed in EBCOT algorithm [TAUB-2000]. In order to exploit the interband correlation, the corresponding node in the next high subband is also employed, as shown in Figure 6.16 (b). This choice is based on the fact that there exits a strong correlation between the coefficients of the two adjacent subbands. Instead of treating the entire resulting context vector (2^9 totally) as different conditional states, we carefully classify into several model classes, similar to the context selection adopted in EBCOT algorithm [TAUB-2000]. The look-up tables are established accordingly to fast map a given context to the assigned model index, as shown in Table B_4.1 in Appendix B.4. Given a pixel(i,j), its significant map can be defined by:

$$\sigma(i,j) = \begin{cases} 1, & \textit{if node } (i,j) \textit{ is significant}, \\ 0, & \textit{otherwise}, \end{cases} . \tag{6.13}$$

Therefore, the variable P, H, V, HV, and D can be decided as follows:

$$H = \sigma(W) + \sigma(E), \text{ such that } 0 \le H \le 2; \tag{6.14}$$

$$V = \sigma(N) + \sigma(S), \text{ such that } 0 \le V \le 2; \tag{6.15}$$

$$HV = H + V, \text{ such that } 0 \le HV \le 4; \tag{6.16}$$

$$D = \sigma(NW) + \sigma(NE) + \sigma(SW) + \sigma(SE), \text{ such that } 0 \le D \le 4; \tag{6.17}$$

$$P = \sigma(F), \text{ such that } 0 \le P \le 1. \tag{6.18}$$

where, the relative positions of node W, E, N, S, NW, NE, SW, SE and F are shown in Figure 6.16 (a) and (b).

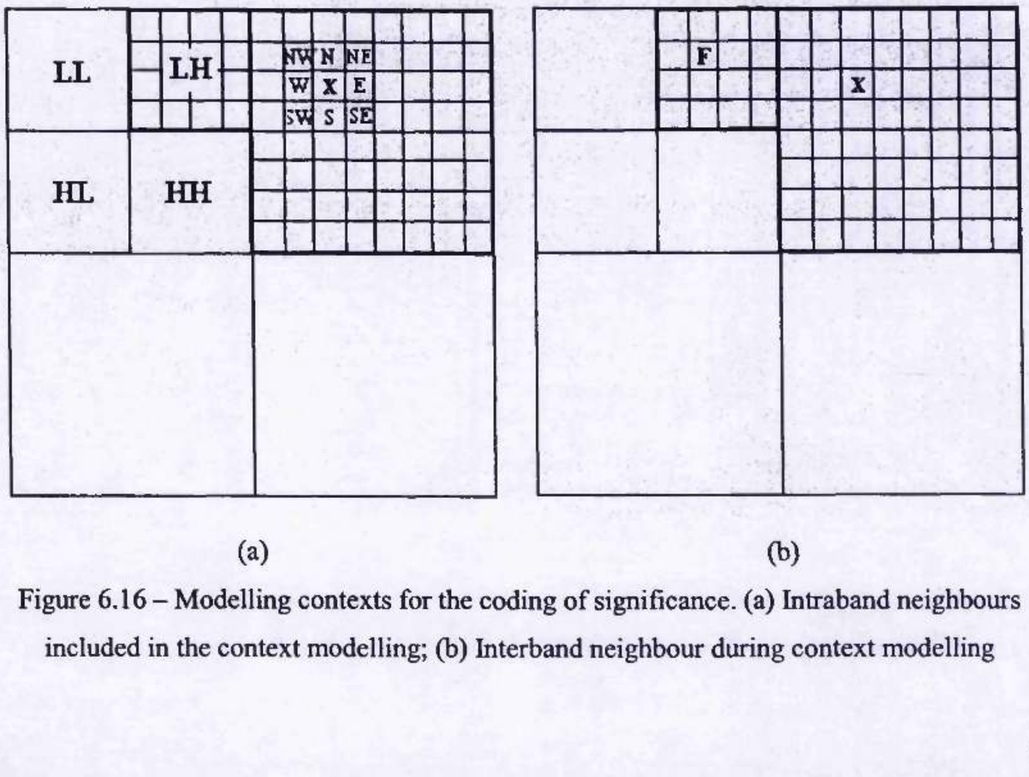


Figure 6.16 – Modelling contexts for the coding of significance. (a) Intraband neighbours included in the context modelling; (b) Interband neighbour during context modelling

In the original SA-SPECK algorithm, compressibility of sign bits of subband coefficients had not been fully employed. Improvement on sign bit coding is possible. In the improved SA-SPECK algorithm, a sign coding scheme similar to EBCOT algorithm is adopted [TAUB-2000]. The sign coding operation follows the three basic steps:

1. Summarise the sign and significant information about the neighbour coefficients in the different orientations.

2. Predict the sign of the current coefficient based on the information collected in the previous step.

3. Encode the correctness of sign prediction.

The contribution from the horizontal direction is formulated in Table B_4.2 of Appendix B.4, where the relative position of the neighbouring nodes is indicated in Figure 6.16 (a). In Table B_4.2, variable h identifies the contribution from the horizontal direction, where the relative position of the neighbouring nodes W and E are indicated in Figure 6.16 (a), and the pair (significant/insignificant, +/-) represents the significant status and the sign of the neighbouring node, respectively.

The contexts for sign coding are included in Table B_4.3 of Appendix B.4. In Table B_4.3, the vertical and diagonal contributions, variable v, d_{45}, and d_{135} are defined in the same way as variable h. The sign bit χ is then predicted as $\hat{\chi}$. The correctness of sign prediction ζ is encoded, which is defined to be 1 if $\chi = \hat{\chi}$ and 0 otherwise.

The same contextual intraband region shown in Figure 6.16 (a) is utilised for conditional coding of the refinement of the significant coefficients. The contextual information is characterised by significant map and the significant status with respect to the quantisation threshold at the previous bitplane level. The related look-up table is given in Table B_4.4, Appendix B.4. In Table B_4.4, HV^p is defined in a similar way to HV except using significance $\sigma^p(i,j)$, which is the significance status with respect to the quantisation threshold at the previous bitplane level.

This improved SA-SPECK algorithm has been fully implemented in software and extensive experiments have been conducted to test the performance of this algorithm. The results are included in the following section.

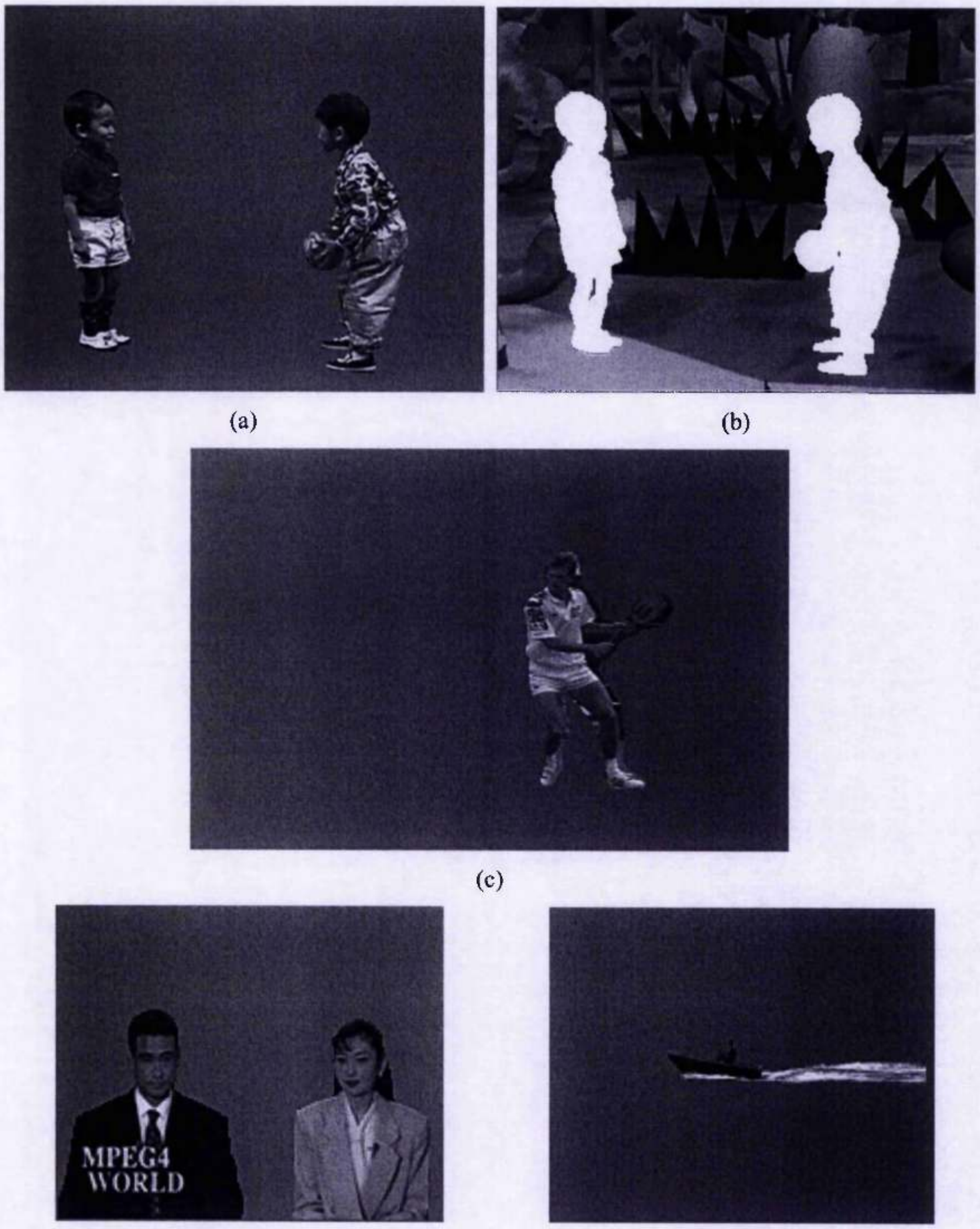
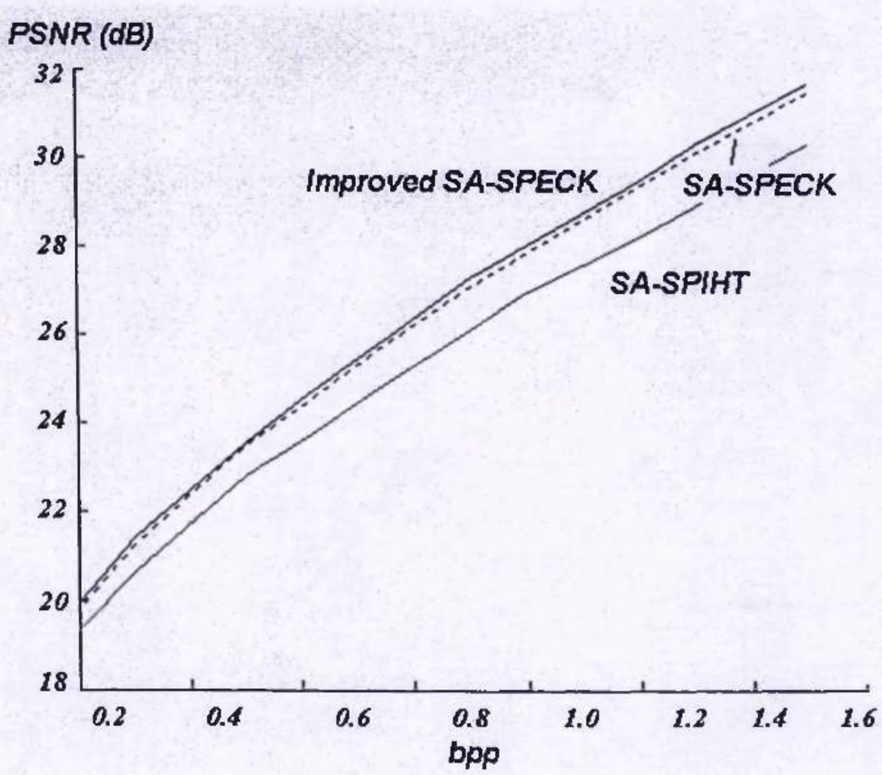


Figure 6.17 – Video objects used for test: (a) Children; (b) Children-Background; (c) Stefan; (d) News; (e) Coastguard

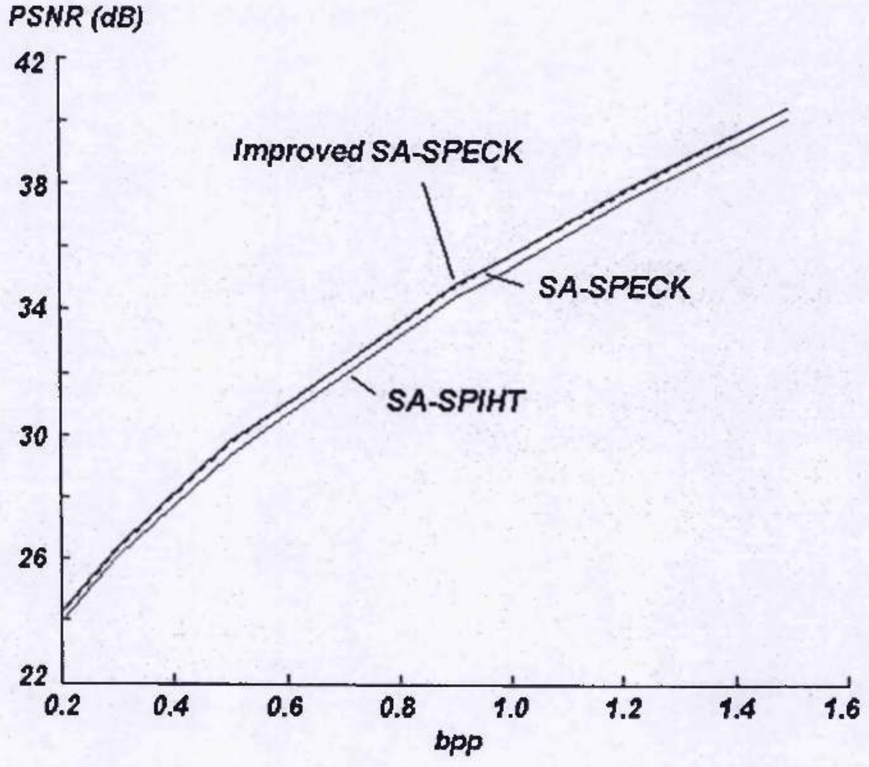
6.6 Experimental results

Extensive experiments have been conducted to study the scalable texture coding algorithms for video objects. They are SA-SPIHT algorithm, SA-SPECK algorithm and the improved SA-SPECK algorithm. The SA-EBCOT algorithm has not been implemented during our research and not compared with above shape-adaptive encoding algorithms. The above algorithms are tested in the intra-coding mode without motion estimation and compensation. Five monochrome video objects are used during the tests which are shown in Figure 6.17. During the encoding and decoding processes of the arbitrarily shaped video objects, the bitrate (bit/pixel) is calculated based on the number of pixels within the object. Here, it is assumed that the object shape has been encoded and the bits used for shape coding is not included. For these five video objects in Figure 6.17, the Rate-distortion curves of SA-SPIHT, SA-SPECK and the improved SA-SPECK algorithms are shown in Figure 6.18.

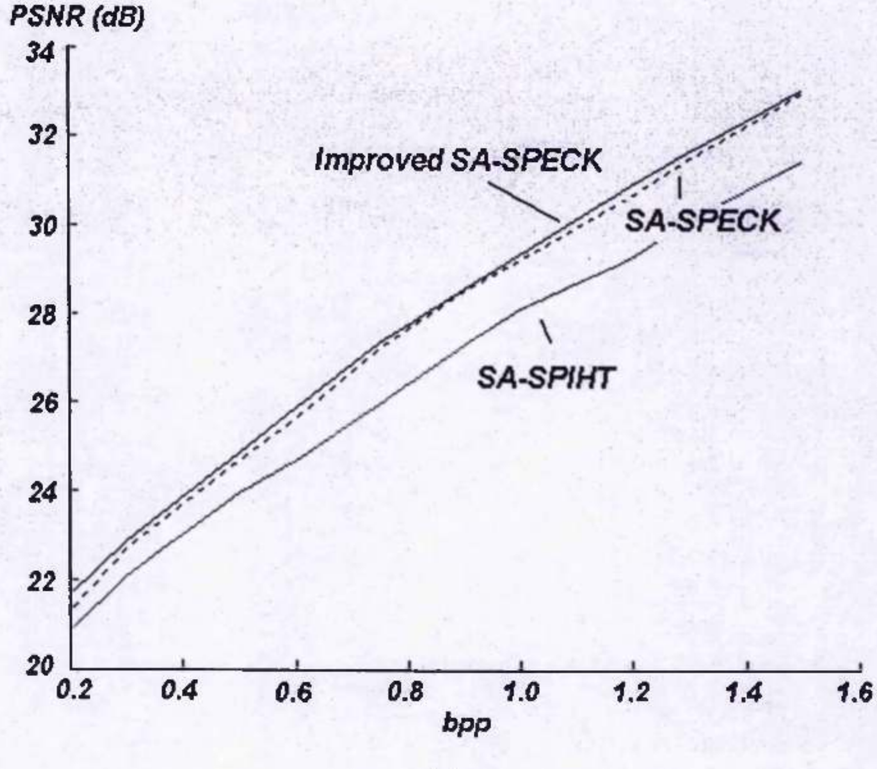
Due to the CABAC, the improved SA-SPECK algorithm has higher complexity than the original SA-SPECK algorithm using arithmetic coder. However, the complexity does not increase much (about 20-40%) as no binarization step is required. The significant bits, sign bits and refinement bits are all binary symbols.



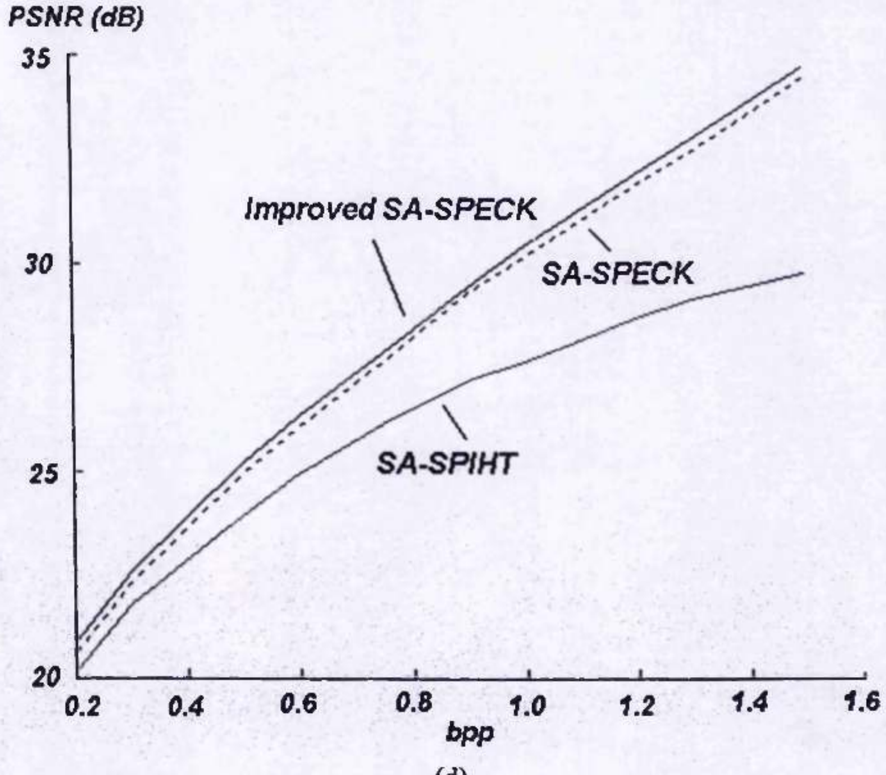
(a)


PSNR (dB)
42
38
34
30
26
22
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
bpp
Improved SA-SPECK
SA-SPECK
SA-SPIHT

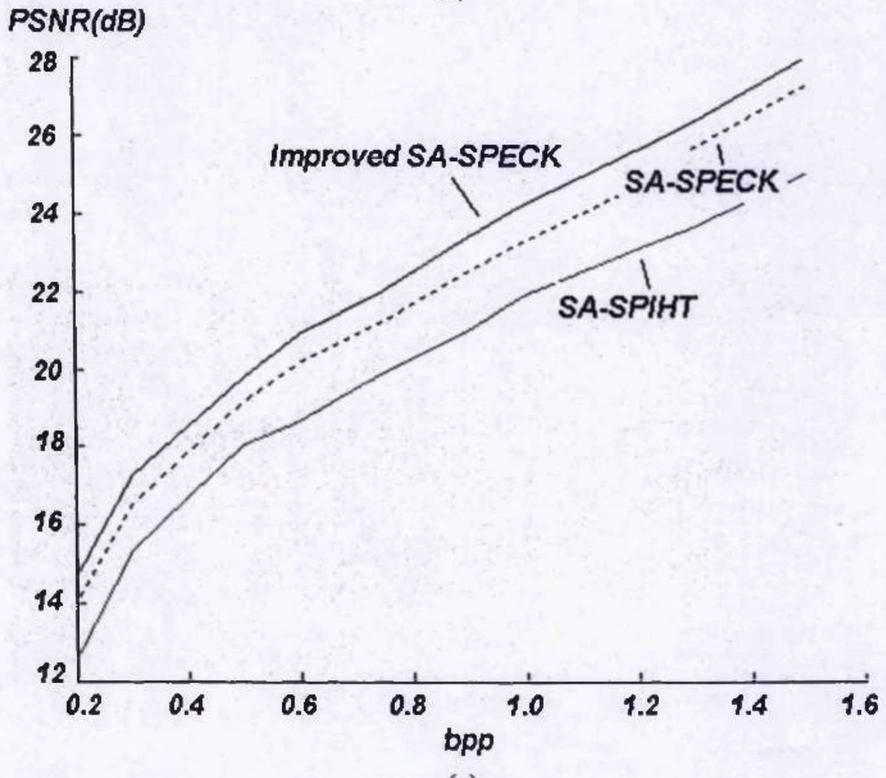
(b)


PSNR (dB)
34
32
30
28
26
24
22
20
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
bpp
Improved SA-SPECK
SA-SPECK
SA-SPIHT

(c)



(d)



(e)

Figure 6.18 – Comparison of R-D performances of different shape-adaptive bit-plane coding algorithms for (a) Children; (b) Children-background; (c) Stefan; (d) News; (e) Coastguard objects

The average PSNR results of the three algorithms on these video objects are also listed in Table 6.1 to Table 6.4 for the bit rates 0.2 bpp, 0.5 bpp, 1.0 bpp and 1.5bpp, respectively, including the improvements of the proposed SA_SPECK algorithm over the original one.

Table 6.1 – Distortion performance for the shape-adaptive coders under 0.2 bpp

Image Object	PSNR(dB)			
	SA-SPIHT	SA-SPECK	Improved SA-SPECK	Δ dB
Children	19.33	19.78	19.95	0.17
Children-Background	23.98	24.21	24.26	0.05
Stefan	20.85	21.30	21.64	0.34
News	20.13	20.60	20.85	0.25
Coastguard	12.60	14.06	14.71	0.65

Table 6.2 – Distortion performance for the shape-adaptive coders under 0.5 bpp

Image Object	PSNR(dB)			
	SA-SPIHT	SA-SPECK	Improved SA-SPECK	Δ dB
Children	22.80	23.47	23.55	0.08
Children-Background	29.34	29.73	29.78	0.05
Stefan	23.95	24.70	24.83	0.13
News	23.94	24.94	25.22	0.28
Coastguard	18.07	19.19	19.80	0.61

Table 6.3 – Distortion performance for the shape-adaptive coders under 1.0 bpp

Image Object	PSNR(dB)			
	SA-SPIHT	SA-SPECK	Improved SA-SPECK	Δ dB
Children	26.99	27.87	28.03	0.16
Children-Background	35.37	35.75	35.80	0.05
Stefan	28.11	29.23	29.36	0.13
News	26.65	30.14	30.42	0.28
Coastguard	21.93	23.36	24.28	0.92

Table 6.4 – Distortion performance for the shape-adaptive coders under 1.5 bpp

Image Object	PSNR(dB)			
	SA-SPIHT	SA-SPECK	Improved SA-SPECK	Δ dB
Children	30.29	31.45	31.67	0.22
Children-Background	40.09	40.42	40.46	0.05
Stefan	31.44	32.94	33.05	0.11
News	29.77	34.42	34.67	0.25
Coastguard	25.04	27.28	27.96	0.68

From above tables and figures, it is found that the coding efficiency of SA-SPECK algorithm is better than that of SA-SPIHT algorithm. The difference of PSNR is about 1.0 – 3.0 dB. The SA-SPECK algorithm also preserves the features of an embedded bitstream and allows exact bitrate control. The improved SA-SPECK algorithm can further improve the efficiency for about 0.1 – 0.4 dB when compared with the original one. Figure 6.19 to Figure 6.23 show the decoded video objects at bitrates of 0.2 bpp, 0.5 bpp, 1.0 bpp and 1.5 bpp for Children, Children-background, Stefan, News and Coastguard objects respectively.

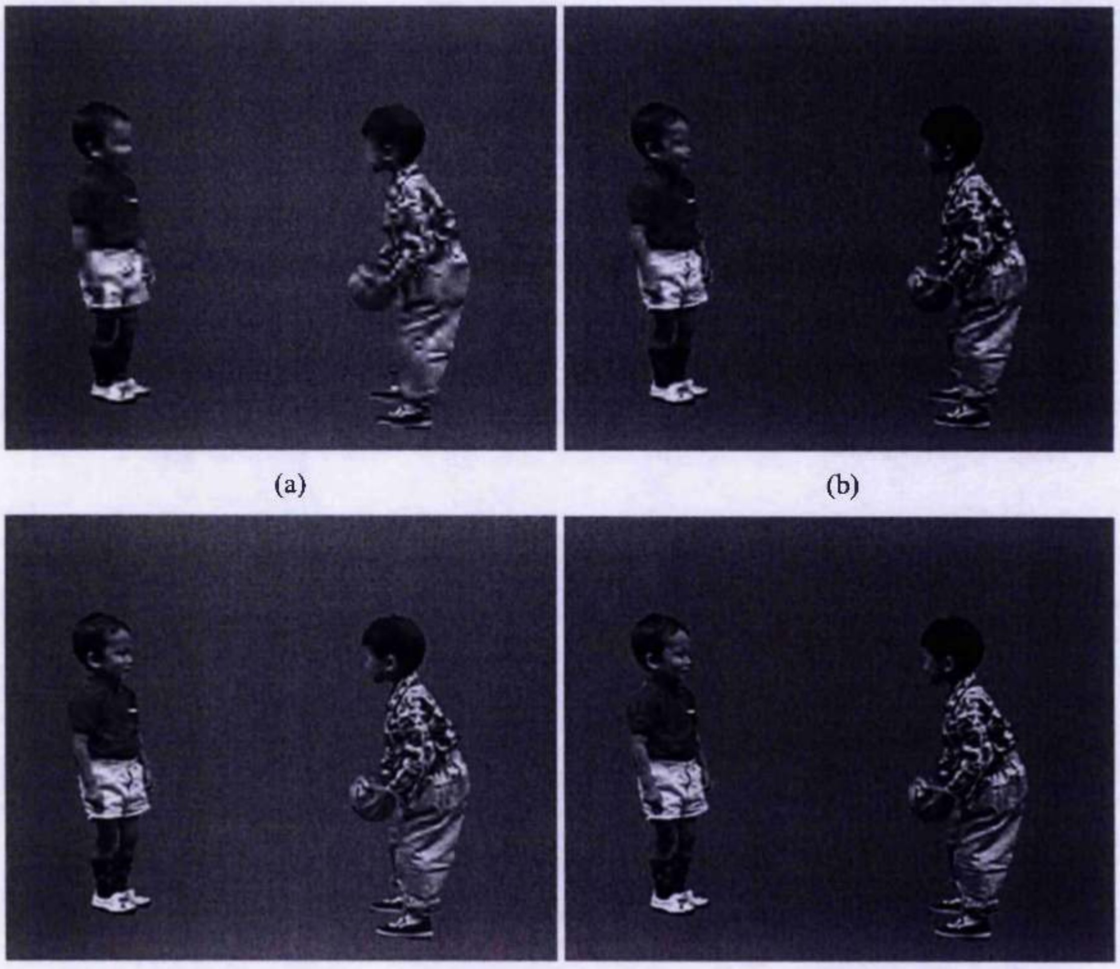
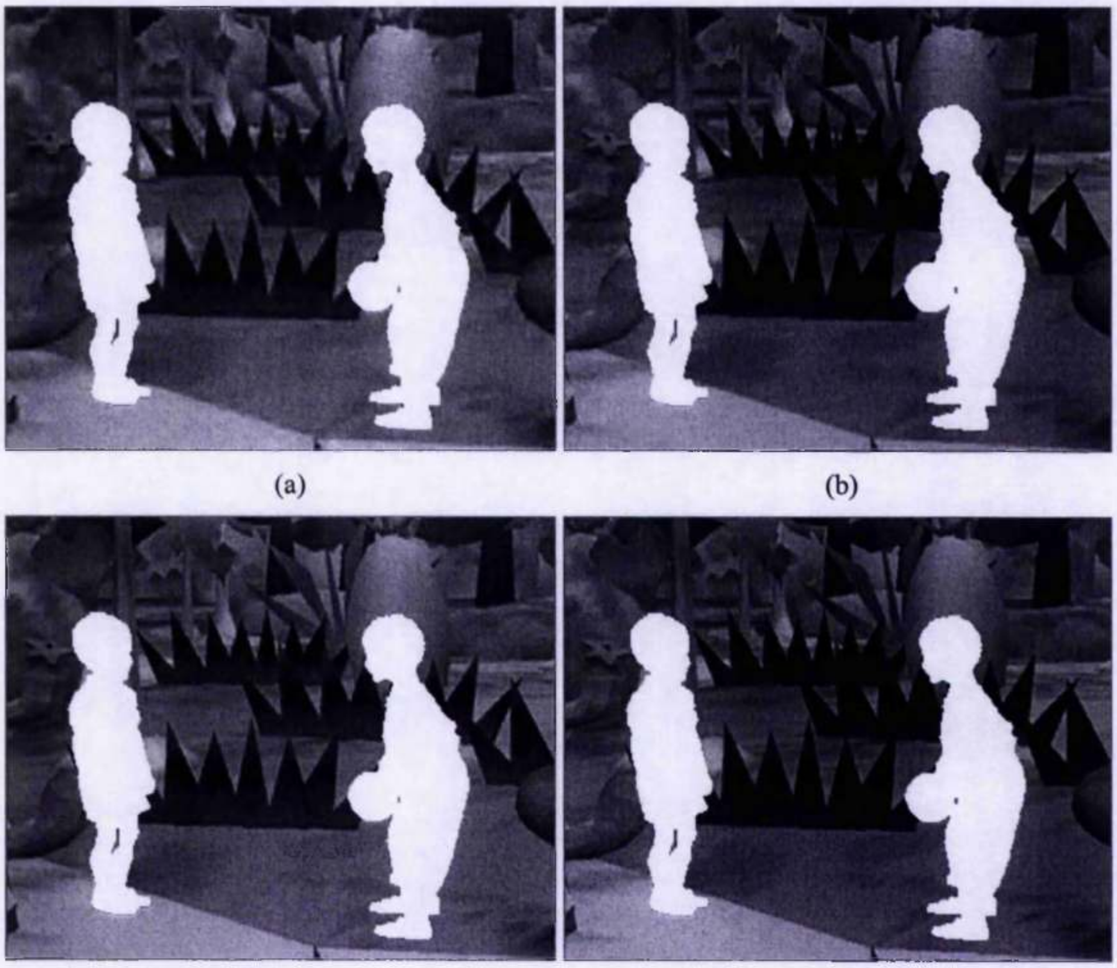
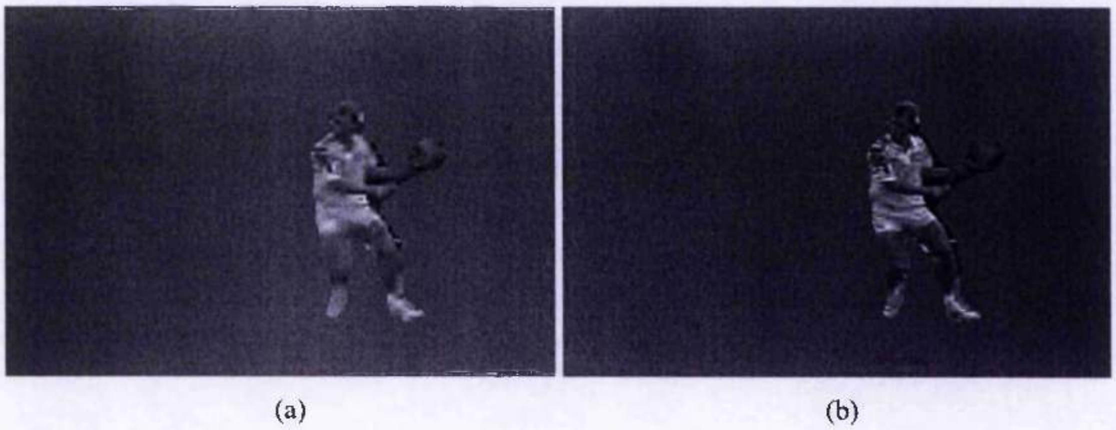


Figure 6.19 – Decoded Children object under (a) 0.2 bpp; (b) 0.5bpp; (c) 1.0bpp; and (d) 1.5bpp by truncating the same pre-coded bitstream

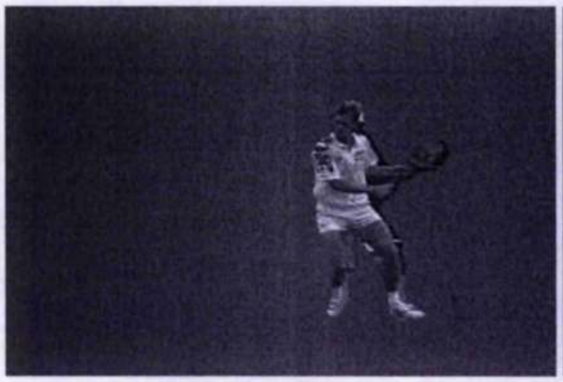
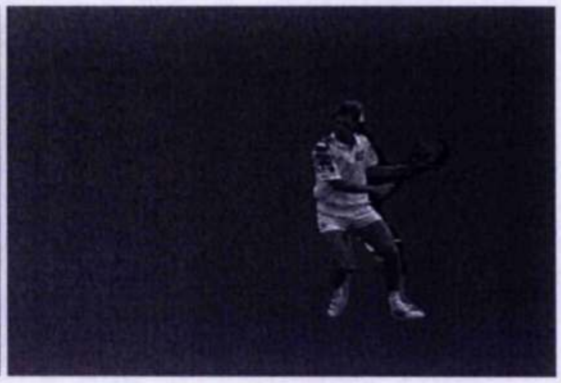


(c) (d)

Figure 6.20 – Decoded Children-background object under (a) 0.2 bpp; (b) 0.5bpp; (c) 1.0bpp; and (d) 1.5bpp by truncating the same pre-coded bitstream

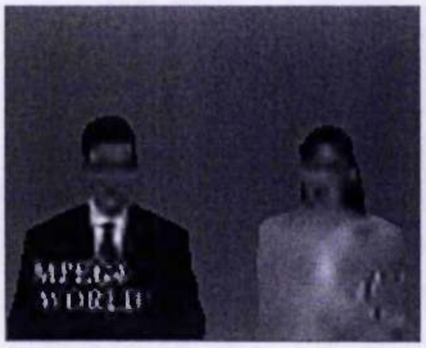


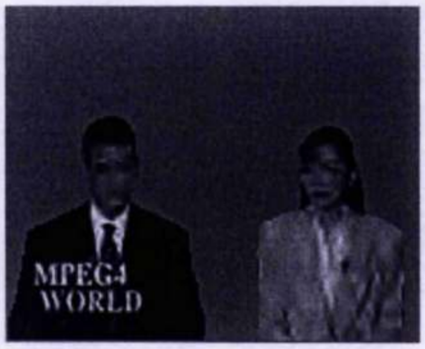
(a) (b)

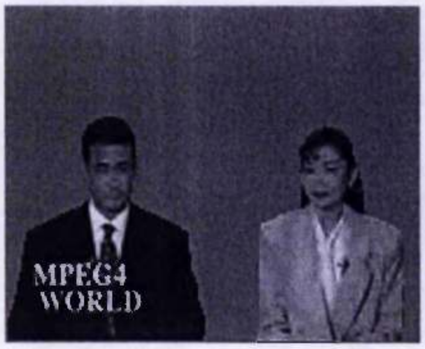
(c) (d)

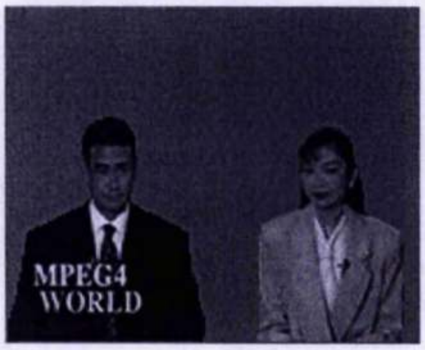
Figure 6.21 – Decoded Stefan object under (a) 0.2 bpp; (b) 0.5bpp; (c) 1.0bpp; and (d) 1.5bpp by truncating the same pre-coded bitstream





(a) (b)





(c) (d)

Figure 6.22 – Decoded News object under (a) 0.2 bpp; (b) 0.5bpp; (c) 1.0bpp; and (d) 1.5bpp by truncating the same pre-coded bitstream

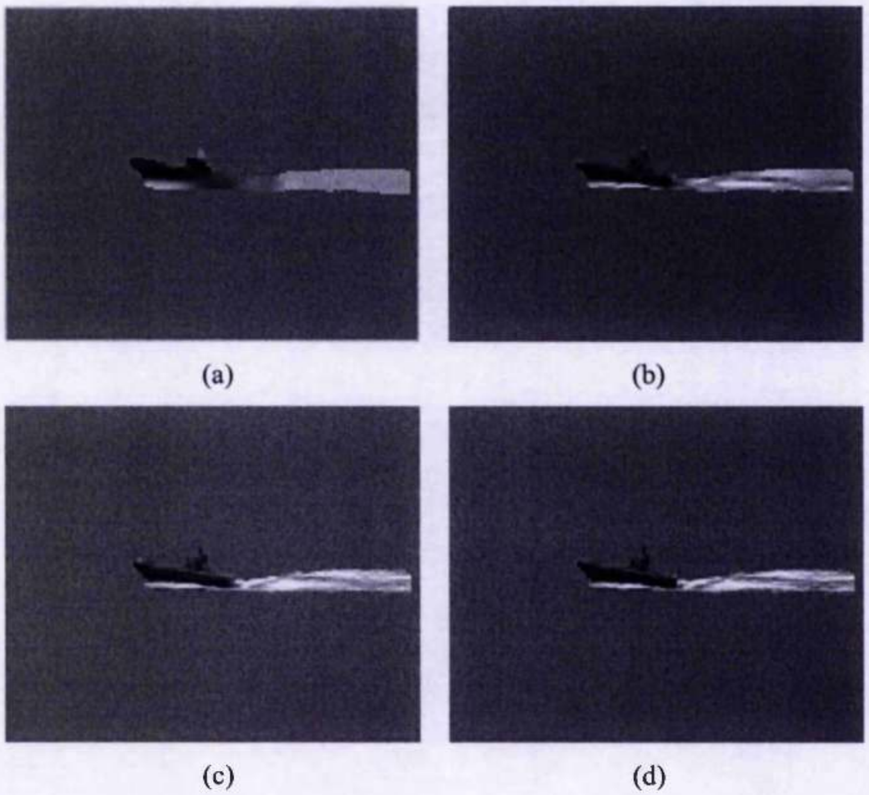


Figure 6.23 – Decoded Coastguard object under (a) 0.2 bpp; (b) 0.5bpp; (c) 1.0bpp; and (d) 1.5bpp, by truncating the same pre-coded bitstream

6.7 Conclusions

In this chapter, the scalable texture intra-coding techniques are presented for video objects. After reviewing texture coding techniques for video objects, the theory and implementation of subband/wavelet analysis are discussed and the shape-adaptive discrete wavelet transform algorithms are reviewed. The main wavelet-based texture coding algorithms, such as SPIHT, SPECK, and EBCOT algorithms, are discussed, followed by their extension to arbitrarily-shaped objects. An improved SA-SPECK algorithm has been proposed that incorporates the context-adaptive binary shape coding (CABAC) into the shape-adaptive SPECK algorithm. Extensive experiments are conducted to code arbitrarily shaped video object in intra mode. The results show that the improved algorithm achieves higher coding efficiency compared with SA-SPIHT and the original SA-SPECK algorithm. This algorithm will be employed in the proposed scalable 2D model-based texture coding scheme, which will be discussed in Chapter 7.

Chapter 7

Scalable 2D Model-based Texture Inter-coding

Excellent compression performance for scalable intra texture coding was exhibited in Chapter 6. However, the intra texture coding scheme can not achieve efficient texture coding of video objects because no temporal information is employed. In this Chapter, a scalable 2D model-based texture coding scheme is proposed, which is a combination of temporal filtering and the improved shape-adaptive SPECK algorithm presented in Chapter 6.

The designed scalable 2D model-based texture coding scheme possesses the following desirable properties:

- Free from DCT blocking artefacts: In comparison to the conventional DCT-based coder, the designed scheme incorporates warping motion compensation and wavelet analysis. As a result, the reconstructed texture does not show the annoying blocking artefacts at very low bit rate video coding.

- Error resilience: Error propagation in the proposed scheme, as well as other scalable video coding scheme based on temporal filtering [OHM-2005], is limited by the length of the temporal synthesis filters. This is the advantage of employing temporal filtering technique.

- Excellent compression efficiency: The redundancy in the source video is efficiently reduced by temporal filtering with warping motion compensation of video objects. The representation of video frames into video objects also improves the motion estimation and compensation. The subband correlation can be effectively exploited through the improved shape-adaptive SPECK algorithm. The experimental results show that the proposed coding system outperforms the nonscalable standard MPEG-4 coder over a wide range of bitrates in PSNR performance. Its performance is also comparable to H.264 at very low bit rates (<10kbits/s).

- Flexible and highly scalable bitstream: The proposed scalable texture coding system can accommodate a wide variety of scalable functionalities utilising the multi-resolution nature innate in temporal and spatial subband filtering, and scalable object modelling. Most importantly, these desirable scalable features are provided without a significant performance loss when compared with H.264, which is commonly seen in traditional hybrid coding for scalable applications.

The organisation of this chapter is as follows. Related works on scalable video coding in the literature are first reviewed and commented in the next section. The discussion covers the hybrid DCT-based scalable video coding systems, wavelet-based scalable video coding systems, and motion compensated 3D wavelet-based video coding techniques. The proposed scalable 2D model-based texture coding scheme is then presented in Section 7.2, which includes scalable motion vector coding and rate-distortion optimised bit truncation. The performance of the proposed scheme is evaluated through extensive experiments, as discussed in Section 7.3. The chapter is summarised and concluded in Section 7.4.

7.1 Overview

With the recent expansion of multimedia applications, video coding systems are expected to become more flexible. In particular, they should be able to adapt a single video bitstream to variable transport conditions (bandwidth and channel error rate) and to varying receiver capability and demands (display size, manipulation and applications) as well. Scalability is the expected functionality to address this issue. Scalable coding methods allow the decoder to partially decode a single compressed bitstream depending on the conditions (bit rate, errors and recourses).

In Chapter 2, much discussion has been conducted for scalable video coding, which has been an active research field over the past decade. A scalable stream can offer adaptivity to varying channel error characteristics, and different kinds of users. For wireless communications, scalability allows the adjustment of the source rate and the application of unequal error protection in response to channel error conditions. For internet transmission, scalability enables variable-bit-rate transmission, selective bit discarding, and the adjustment of the source rate to correspond to different modem rates, and diverse device capabilities. Scalability becomes increasingly important for rich media access from anywhere, by anyone, at any time, with any device and in any form. Due to its importance, scalable video coding (SVC) is currently being intensively investigated

[MPEG-2003] [RADHA-1999] [MARPE-1999] [SCHA-2000b] [LUO-2001] [LI-2001] [SCHWARZ-2004]. Some methods have been submitted for the proposals of MPEG-4 Part 10: SVC standard. Commonly, these methods can be classified into three classes: DCT-based hybrid scalable video coding, wavelet-based scalable video coding, and motion compensated (MC) 3D wavelet-based video coding. In this section, several scalable video coding techniques will be reviewed and analysed.

7.1.1 Hybrid scalable video coding

Current standards like H263 or MPEG-4 are based on block DCT in coding of displaced frame difference (DFD). In these hybrid coders, scalability is achieved through additional layers of the single-scale prediction loop that delivers one base, and one or more enhancement video streams [LI-2001] [WU-2001] [SCHA-2001] [SCHA-2002], which is named as Fine-Granular-Scalability (FGS). These proposed solutions are not very granular except for the quality (or SNR) scalability, as provided in MPEG-4 FGS algorithm [RADHA-1999], where the decoding process can be stopped at any point of the enhancement layer. Temporal scalability is obtained at a reasonable cost by sending some of the B and P frames in the enhancement layer, where spatial and SNR scalable schemes have a very limited efficiency. Experiments with MPEG-4 and H.263 using scalability modes show that generally the coding efficiency would lose 0.5-1.5 dB with every layer, compared with a non-layered coding scheme [SCHA-2000b]. It is difficult for these schemes to achieve scalability efficiently since there is always a potential drifting problem associated with predictive coding [WU-2001].

Recently, the researchers in HHI have proposed a scalable extension of the H.264/AVC video coding standard [SCHWARZ-2004]. To achieve an efficient scalable bit-stream representation of a video sequence, the temporal dependencies between pictures are exploited by using an open-loop subband approach. The related temporal analysis-synthesis filter band structure is generalised to facilitate an adaptive block-based choice between the motion-compensated lifting representations of the Haar filter (uni-directional prediction) and the 5/3 filter (bi-prediction), both coupled with multiple-reference frame capabilities. Furthermore, in this method, an intra model can be chosen on a block basis to efficiently represent blocks that cannot be reasonably predicted using motion compensation. In order to provide spatial scalability, a pyramid structure is employed. Although motion compensated temporal filtering (MCTF) is independently applied in each spatial layer, a large degree of inter-layer prediction is incorporated. Intra macroblocks and residual macroblock representing temporal high-pass signals can be predicted using the corresponding interpolated reconstruction signals of the previous layers. The motion description of each MCTF layer can be used for a prediction of the motion description for the following enhancement layers. A remarkable feature of this hybrid scalable video coding scheme is that most components of H264/AVC are used, while only a few have been adjusted to the motion compensated temporal filtering structure. Experimental results indicate that this hybrid scalable coding method is capable of providing a coding efficiency nearly comparable to that of an original H264/AVC encoding [SCHWARZ-2004].

7.1.2 Wavelet-based scalable video coding

Another kind of scalable video coding technique is based on wavelet analysis. As we know, wavelet transform is an efficient tool for video decomposition, which can pack the energy of video into a small set of wavelet coefficients, and lead to a nice scalability. Applications of the wavelet transform in video coding follow two paths: motion compensated wavelet residual coding [CHENG-1997] [SHEN-1999] [MARPE-1999] [XU-2000] [ASBUN-2000] and 3D wavelet video coding [THAM-1998] [KIMB-2000].

In the motion compensated wavelet residual encoder schemes, the current frame is predicted by the content from the previous frame, subject to the object motion. The prediction residue is then encoded by wavelet encoder. The framework of the coder is very similar to existing video coding standards, such as H264, except that the residual frame is encoded using a wavelet-based encoder instead of DCT-based encoder [SHEN-1999] [MARPE-1999] [ASBUN-2000]. In [SHEN-1999], a scalable adaptive motion compensated wavelet (called SAMCoW) algorithm was proposed, which used motion compensation to reduce temporal redundancy. The intra-coded frames (I-frames) and the residual frames are encoded using an approach similar to the embedded zerotree wavelet (EZW) coder. An adaptive motion compensation scheme is introduced to address error propagation problems. This encoder can achieve comparable performance to the more traditional hybrid video coders, such as H. 263. The scheme in [MARPE-1999] used a modified block-matching algorithm, so called overlapped block motion compensation (OBMC). Like conventional block-based motion compensation, OBMC is a very efficient technique for temporal predictive coding with the advantage of eliminating blocking artefacts in the prediction error signal, which can reduce the efficiency of wavelet-based residual coder. Furthermore, an optimisation activity on the wavelet coder is conducted in this scheme to improve the coding performance. The experimental results demonstrate that this coder can achieve better performance than MPEG4. One of the disadvantages of these motion compensated wavelet residual coding schemes is that they can not achieve highly scalable bitstreams.

An alternative to the predictive approaches in various video coding standards is 3D wavelet video coding, which has been investigated by several researchers [THAM-1998] [KIMB-2000]. 3D wavelet video coding applies wavelet transform both temporally and spatially, and then encodes the transformed coefficients using entropy coding [KIMB-2000]. With a proper entropy coding and bitstream packaging scheme, the generated bitstream can achieve spatial, temporal, and quality scalabilities simultaneously. Experimental results turn out that 3D wavelet-based video coding is competitive with standard motion compensated predictive coding schemes. In [KIMB-2000], Kim reports that their 3D SPIHT coder generates a fully embedded bitstream that can be truncated at points and still decodable to the best quality available. On the other hand, the straightforward 3D wavelet coding scheme does not use motion compensation to remove temporal redundancy. The primary weakness of the existing 3D wavelet video coder lies in the temporal filtering. Although the computationally intensive motion estimation is avoided, this makes the performance of 3D wavelet video coding very sensitive to the motion. Without motion information, motion blur will occur because of the temporal averaging effect of several frames. For the video sequence or objects with large motion pattern, the object motion (such as panning and zooming) causes the object to be misaligned along the temporal direction, and leads to compression inefficiency.

7.1.3 Motion Compensated (MC) 3D wavelet based video coding

For 3D wavelet based video coding, much work has been done to improve the correlation of the video signal along the temporal direction, by employing motion estimation/compensation [ZHANG-1992] [TAUB-1994] [OHM-1994] [WANG-1999] [XU-2000] [XU-2001]. Taubman and Zakhor [TAUB-1994] pan shifted the video sequence before the 3D wavelet transform was applied. Ohm [OHM-1994] and Choi [CHOI-1999] incorporated block matching into the temporal transform by separately handling the covered / uncovered, connected / unconnected regions. Xu et.al proposed a motion threading (MTh) approach so that the pixels along the same motion trajectory are aligned for wavelet filtering [XU-2000]. In this scheme, macroblock-based backward motion estimation is performed from the first frame to the last frame. Pixels along the same motion trajectory are aligned to form non-overlapped motion threads. Afterward, the shape-adaptive wavelet transform is applied along each motion thread. After temporal and spatial decomposition, the coefficients are encoded with embedded entropy coding to form scalable bitstream. This method outperforms MPEG-4 up to 1.5-2.5dB in those sequences with simple motion. However, it is about 0.5-1.8 dB inferior to MPEG-4 in compressing those sequences with complex motion [XU-2000].

Recently, the performance of motion compensated (MC) 3D wavelet video coding (MC-3DSBC) has improved greatly due to the use of lifting-based wavelet transforms, which allow for full adaptability in the selection of reference pictures, MC mode selection, and advanced motion mode for motion estimation [BOTTREAU-2001] [LUO-2001] [SECKER-2001]. Most methods usually apply motion compensated temporal filtering (MCTF) combined with a 2D spatial wavelet transform. The structure of the MCTF-based encoders enables high flexibility for scalability, i.e. a high number of spatial, temporal and quality representations with fine granularity over a large range of bitrates. In MCTF, the combination of lifting wavelet filters with motion compensation enables an open loop implementation which can also improve error resilience and solve the drift problems of the hybrid coding approaches.

Based on the order of the spatial and temporal processing, MC-3DSBC methods can be classified into two major classes:

- **Class 1: Inter-frame wavelet** (t + 2D). In this class, the open loop MCTF is first performed on the temporal axis followed by a 2D wavelet spatial decomposition. So, the original frames are first motion compensated with a lifting wavelet transform to exploit the temporal redundancy. Note that motion estimation and compensation are performed in the time domain. The decomposed temporal frames are then spatially transformed by 2D spatial wavelet filters, and the wavelet coefficients are entropy encoded. Several scalable video coding algorithms fit well in this category, e.g. [SECKER-2001] [WOODS-2002] [LUO-2003] [XU-2004] [CHEN-2004] [WU-2004] [WIEN-2004] [HAN-2004].

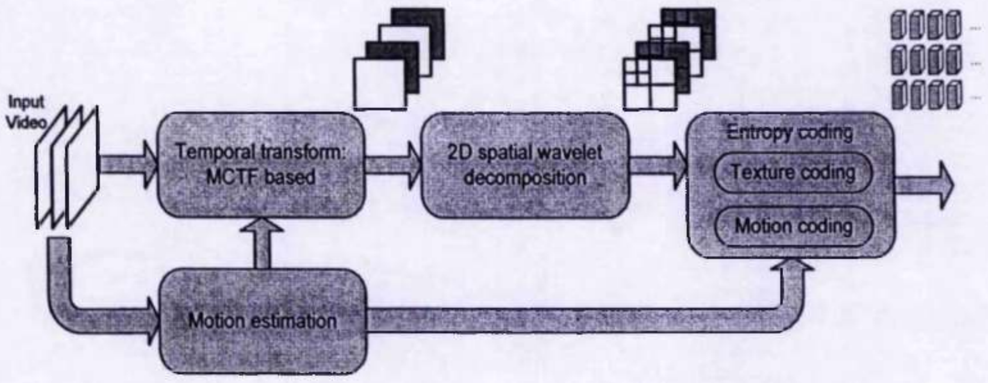


Figure 7.1 – Architecture for the proposed Inter-frame wavelet coders (t+2D).

Figure 7.1 presents the general architecture for inter-frame wavelet coders. In the first step, a temporal decomposition of the input video is performed, followed by a spatial decomposition of each temporal subband. The motion vectors for each spatial resolution level and the wavelet coefficients are then entropy coded.

Although many schemes are based on this architecture, there are some differences among them. In [SECKER-2001], motion-compensated lifting steps are used to implement the temporal wavelet transform, which preserves invertibility, regardless of the motion model. Recently, MC-EZBC scheme proposed by Woods et al [WOODS-2002] has become prominent because of its excellent performance. In MC-EZBC, each pair of frames is first motion estimated with hierarchical block structure, and then decomposed into a high-band frame and a low-band frame by the motion-aligned lift-based Haar filter. MC-EZBC efficiently solves the problems in the fractional-pel motion aligned temporal transform due to the use of lift-based wavelet transform. For example, one of the proposals for MPEG SVC is based on MC-EZBC [WIEN-2004]. Within this scheme, the temporal redundancy is removed by MCTF with a biorthogonal 5/3 filter pair and a sliding window approach, i.e. the temporal filtering is extended beyond the boundaries of the current GOP. The motion compensation is performed using variable block sizes, and a rate distortion motion estimation and mode decision is performed with the definition of different block-modes and a global scene change flag. The precision accuracy is ¼ pel and 8-tap interpolation filters are used. Motion vector data is encoded employing median prediction and Context Adaptive Binary Arithmetic Coding (CABAC) [MARPE-2003]. In order to reduce the blocking artefacts caused by the block based motion compensation, a deblocking filter is applied at the decoder side at each resolution level. Its filter design is similar to the in-loop deblocking filter in H.264/MPEG-4 AVC [H264-2003]. After the temporal processing, the MCTF prediction error is spatially decomposed using 9/7 Daubechies wavelet filters and entropy encoded using Embedded Zero Block Coding (EZBC) [HSIANG-2001]. The EZBC technique provides a high granularity for SNR scalability. The weak point of the scheme in paper [WIEN-2004] is the absence of motion scalability and object scalability.

Luo et al [LUO-2003] proposed an advanced MTh technique, trying to improve the method in [XU-2000] by continuous threading a bi-directional alignment. In this method, block-based motion estimation is first performed between pair of adjacent frames. According to the motion vectors of the blocks they belong to, pixels along the same motion trajectory are linked into motion threads. In this method, an R-D optimized technique is introduced to estimate motion vectors and select proper prediction modes for each block. Promising experimental results have been demonstrated that this method can be competitive with the state-of-the-art H.263 video standard on coding efficiency. The advanced MTh technique has been further improved and submitted as a MPEG-21 SVC proposal [XU-2004], which employs a Barbell lifting implementation of the wavelet transform. Within this technique, both the prediction and the update step of the lifted wavelet transform are modified. In the lifting stage, each output coefficient is calculated from a set of pixels in each input frame, instead of using a single pixel value. The correspondence from the set of pixels to the single coefficient is established by linear functions. They minimize the amount of energy in high-pass frames and remove ghosting artefacts from the low-pass frames. They are also responsible to follow the motion trajectory in an efficient way, i.e. to perform motion compensation between frames that belong to a Group of Pictures (GOP). Both unidirectional prediction, equivalent to an extension of Haar filtering, and bidirectional prediction, equivalent to 5/3 filters, are supported and a method to adaptively choose the best option is also proposed. The motion compensation supports adaptive block size (similar to H.264/MPEG-4 AVC), overlapping block motion compensation (OBMC) and ¼ pel precision for motion vectors. The motion vectors are encoded in an embedded bitstream, in a coarse to fine fashion, i.e. the motion information is scalable, and the bit budget allocated for motion vectors at each layer can be adjusted according to the target bit rate and spatial resolution. After temporal filtering, the wavelet coefficients are encoded using a 3D variant of the EBCOT algorithm [TAUB-2000], already used in the JPEG2000 standard to provide SNR scalability. The advantages of this method are the 3D EBCOT engine, and the motion compensation tools with scalable motion vectors.

For this class of coding architecture, experimental results obtained with SNR-scalable MCTF video coders suggest that this architecture can be comparable or superior in rate-distortion terms to an optimised non-scalable coder that uses the closed-loop structure [CHEN-2004].

- **Class 2: In-band MCTF** (2D + t). For the methods of this class, spatial transform precedes temporal filtering. As a result, the application of temporal prediction and temporal update of the lifting decomposition occurs in the wavelet-domain. In this approach, each video frame is first spatially decomposed into multiple bands using a 2D wavelet spatial decomposition and then the temporal correlation for each band is removed using MCTF. Typically, a complete to overcomplete wavelet transform (CODWT) is used [PARK-2000] to improve the performance of the motion compensation in the wavelet domain. Examples of scalable video coding methods following this architecture are included in papers [BOTTREAU-2001] [ANDREO-2002] [TUBARO-2004] [VIERON-2004] [BAUD-2004].

Figure 7.2 shows a general architecture for the in-band MCTF coders. In this scheme, the 2D spatial wavelet decomposition is performed before temporal filtering, and motion estimation and compensation is achieved in the subband domain. However, since the spatial wavelet transform used is not shift invariant, i.e. spatial shifts of 1 pixel in the time domain can not be translated directly in the frequency domain, an overcomplete wavelet representation is used, which is achieved by a CODWT transform.

In [BOTTREAU-2001], a fully scalable 3D subband video codec is presented. The proposed codec is based on 2D+t subband decomposition. In this codec, groups of frames are first temporally filtered using motion compensation and then spatially decomposed with wavelets. The spatial-temporal coefficients are further scanned and compressed using a new SPIHT-like strategy, together with arithmetic encoding, which provides a combination of temporal, spatial and SNR scalability.

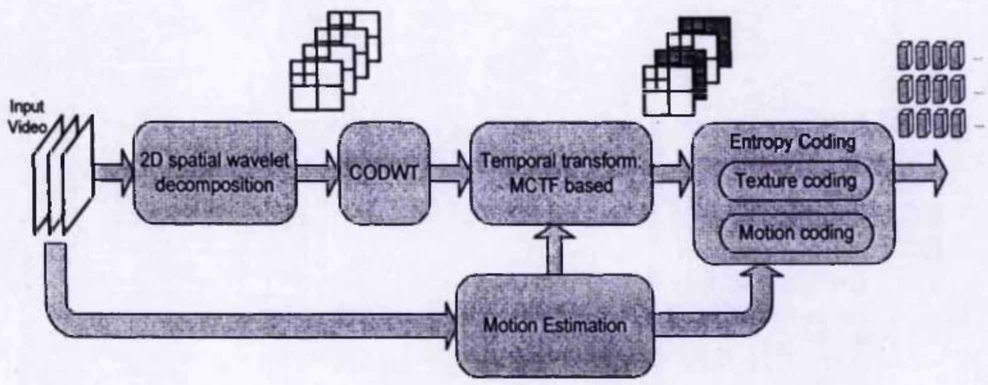


Figure 7.2 – Architecture for the In-band MCTF coders (2D+t).

In [ANDREO-2002], the proposed framework applies the in-band MCTF (IBMCTF) after the DWT is performed in the spatial domain. To overcome the inefficiency of MCTF in the critically-sampled DWT, a complete-to-overcomplete DWT (CODWT) is performed. Furthermore, in order to improve the efficiency of motion compensation, an algorithm for optimised multi-hypothesis temporal filtering is also proposed. Experiments show that the proposed in-band MCTF equipped with multi-hypothesis prediction and update is comparable to the method of class 1 in coding efficiency over a large range of bitrates under the same experimental conditions. The in-band structure additionally permits the independent temporal filtering of each resolution of the input content, which enables many potential developments for multi-resolution decoding.

The in-band MCTF ($2D+t$) approaches present the potential advantage of adaptive tuning of the lifting decomposition across resolution levels according to different criteria for complexity, coding efficiency and scalability, something that is not possible with the conventional $t+2D$ approaches. However, the disadvantage of ($2D+t$) approaches is their limited performance, ranking in the last positions for both scenarios during the MPEG-21 SVC evaluation [ASCENSO-2004]. Further investigations and developments are probably required in order to obtain, with a $(2D+t)$ approach, results comparable to those obtained with the other techniques.

7.2 System description of scalable 2D model-based texture coding

In this section, a scalable 2D model-based texture coding scheme is discussed for arbitrarily shaped video objects. In order to improve the efficiency of temporal prediction, in the proposed scheme, motion compensated temporal filtering (MCTF) is employed, which is similar to MC-3DSBC of class 1. However, the proposed scheme has the following differences compared to the above reviewed methods:

- Mesh-based motion estimation is conducted instead of block-based motion estimation. Currently, most motion compensated 3D wavelet video coding techniques are based on block-based motion estimation and 3D-DWT coding structure, which suffers from the appearance of the so-called "disconnected" pixels occurring in the areas not conforming to the rigid translation model and in occluded/exposed areas [OHM-1994]. Many measures have been proposed to reduce these effects, such as deblocking filtering [WIEN-2004] and overlapping block motion compensation [XU-2004]. However, these effects still exit for very low bit rate coding. Mesh-based motion compensation can overcome above disadvantages due to the existence of unique trajectories (i.e., one-to-one correspondence between all positions in analysed frames).

- Motion compensation is conducted in the object domain, instead of the frame domain. As the video frames are represented into video objects, the effects of motion discontinuity between different video objects on motion compensation are reduced.

- Rate-distortion optimised rate control is achieved easily among video objects and frames. During the motion estimation of layer i, the encoding rate for motion vectors $R_{mv,i}$ and the motion compensation error $D_{mv,i}$ are recoded. At the same time, during spatial bit-plane coding, the encoding rate $R_{spatial,j}$ and approximation error $D_{spatial,j}$ of bit-plane j are also recoded. All of this recoded rate-distortion data is used during the final bit packetising step to achieve optimal bit allocation among video objects, and among object motion and texture.

Figure 7.3 shows the general structure for scalable 2D model-based texture coding scheme. This scheme assumes that the video frame has been segmented into several video objects with different motion patterns. It is also assumed that object shape and model have been encoded. The detailed description of the proposed scheme is included in the following sections.

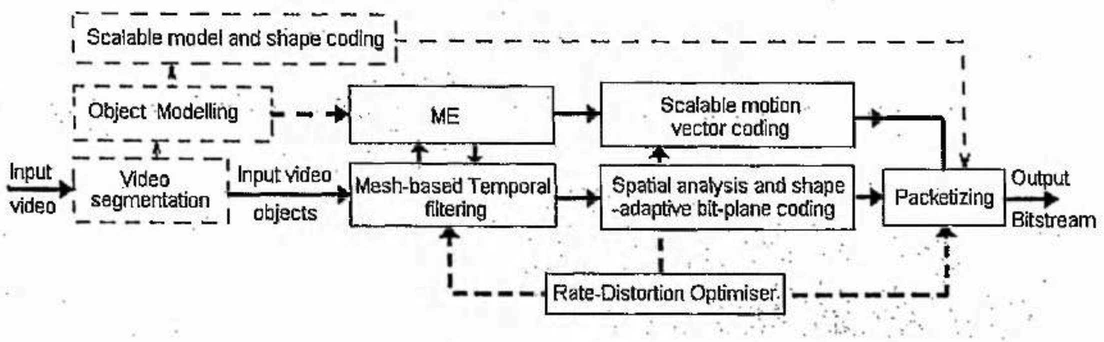

Scalable model and shape coding
Object Modelling
ME
Scalable motion vector coding
Input video
Video segmentation
Input video objects
Mesh-based Temporal filtering
Spatial analysis and shape -adaptive bit-plane coding
Packetizing
Output Bitstream
Rate-Distortion Optimiser

Figure 7.3 – General structure of the proposed 2D model-based texture coding scheme

7.2.1 MC-based lifting scheme for temporal filtering

During the initial research on incorporating motion information into the 3D wavelet video coding [TAUB-1994] [OHM-1994] [HSIANG-1999], filter bank-based filtering is employed during temporal filtering. However, in these approaches, neither the perfect reconstruction can be achieved, nor the motion accuracy is sufficient. The tight coupling between the temporal transformation and the motion models also hampers the use of wavelet kernels other than the Haar wavelet in the temporal domain.

In 2001, several papers have been published independently on the temporal DWT through lifting scheme [SECKER-2001] [LUO-2001] [BOTTREAU-2001]. In these papers, the lifting realisation of temporal DWT with motion compensation applying along the lifting steps has been investigated which can achieve perfect reconstruction for temporal filtering and can also achieve arbitrary motion accuracy. Therefore, in our proposed scalable 2D model-based texture coding scheme, the lifting implementation of DWT has been applied during temporal filtering, which is discussed in details in this section.

In Chapter 6, lifting scheme for DWT has been discussed in detail. The diagram of the lifting scheme for temporal filtering is the same as that in Figure 6.3. Let $s[X,k]$ be a video signal with the spatial coordinate $X=(x,y)^T$ and the temporal coordinate k. The prediction operator P_{Haar} and update operator U_{Haar} for the temporal decomposition using the lifting representation of the Haar wavelet are given by

$$P_{Haar}(s[X,2k+1]) = s[X,2k] \tag{7.1}$$

$$U_{Haar}(s[X,2k]) = \frac{1}{2} h[X,k], \tag{7.2}$$

where $h[X,k] = s[X,2k+1] - P_{Haar}(s[X,2k+1])$. (7.3)

For the 5/3 transform, the prediction operator $P_{5/3}$ and update operator $U_{5/3}$ are given by:

$$P_{5/3}(s[X,2k+1]) = \frac{1}{2}(s[X,2k] + s[X,2k+2]) \tag{7.4}$$

$$U_{5/3}(s[X,2k]) = \frac{1}{4}(h[X,k] + h[X,k-1]), \tag{7.5}$$

where $h[X,k] = s[X,2k+1] - P_{5/3}(s[X,2k+1])$. (7.6)

In the proposed scheme, the lift-based 5/3-filtering structure is employed to achieve temporal filtering.

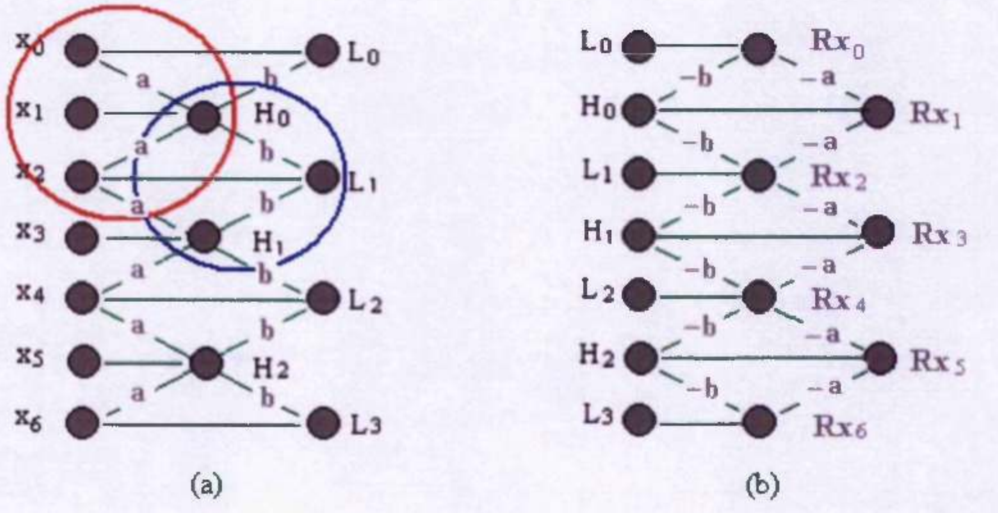


Figure 7.4 – Forward and inverse lifting wavelet and the elementary lifting operations (circled in the forward lifting)

Figure 7.4 illustrates a sample of one-level bi-orthogonal 5/3 lifting wavelet. The original data x_0, x_1, ..., x_6 is input at the left, while the decomposed wavelet coefficients are output at the right two columns. It is observed that the wavelet coefficients are calculated through two stages of computation. The two-stage lifting process can be formulated as follows:

$$\begin{cases} H_i = x_{2i+1} + a \times (x_{2i} + x_{2i+2}) \\ L_i = x_{2i} + b \times (H_{i-1} + H_i) \end{cases}, \text{ where } a = -1/2, \quad b = 1/4. \tag{7.7}$$

With normalisation, the lifting calculation is equal to the traditional bi-orthogonal 5/3 convolution kernel:

$$\begin{cases} H_i = \frac{\sqrt{2}}{2}\left(x_{2i+1} - \frac{1}{2} \times (x_{2i} + x_{2i+2}) \right) \\ L_i = \sqrt{2}\left(\frac{3}{4} x_{2i} + \frac{1}{4} \times (x_{2i-1} + x_{2i+1}) - \frac{1}{8} \times (x_{2i-2} + x_{2i+2}) \right) \end{cases} \tag{7.8}$$

Circled in Figure 7.4 (a), the elementary lifting operation unit involves only three nodes, the updated value can be saved in the same memory of the original pixel, which is called in-place calculation. Each elementary forward lifting unit can be straightforwardly inversed to an inverse lifting unit. The inverse wavelet lifting structure is shown in Figure 7.4 (b).

Within the temporal lifting structure, the frames go through the lifting stages step by step. The calculation process first upgrades the odd frames to the high pass wavelet coefficient frames, and upgrades the even frames to the low pass coefficient frames. For each pixel or patch in the odd frame, motion estimation is conducted to find the corresponding pixels and patches from its left or right frame, or both frames.

Figure 7.5 shows the lifting-based 5/3 wavelet temporal filtering structure [LUO-2003], where each column is a frame and each block represents a pixel. Block-based or mesh-based motion estimation is always from an odd frame to an adjacent even one. For the pixels in the odd frame, motion compensation is conducted to find their corresponding matched pixels in their neighbouring even frame or frames. Some criteria can be used to decide whether the pixels are predicted from one frame (forward or backward prediction) or from two frames (bi-directional prediction). For the pixels which have corresponding matched (one or two) pixels in the neighbouring frames, lifting step is employed to get the high-pass part. If no pixel matches this pixel, intra-prediction is conducted. Following the lifting step, update step is conducted for the even frames. The pixels which are originally terminated in many-to-one mapping can continue the temporal filtering without being stopped, as shown in Figure 7.5. Within the elementary lifting operation, the original terminated pixel in Frame$_1$ can be upgraded using both its left and right matching pixels instead of being stopped at the right side. When the anchor pixel in Frame$_2$ is to be lifted, though many pixels in Frame$_1$ are pointing to it, it is only calculated with the first scanned one according to the motion scan order. For a non-referred pixel in an even frame, it is still linked on both sides using the motion vectors of the adjacent motion threads. It is very easy to generate the invertible fractional-pel accuracy motion threading with lifting structure. As indicated with the dashed arrows in Figure 7.5, all the motion estimation directions are from an odd index frame to an even one, either forwardly or backwardly. An elementary temporal lifting operation can be regarded as a bidirectionally motion compensated prediction process among three consecutive frames.

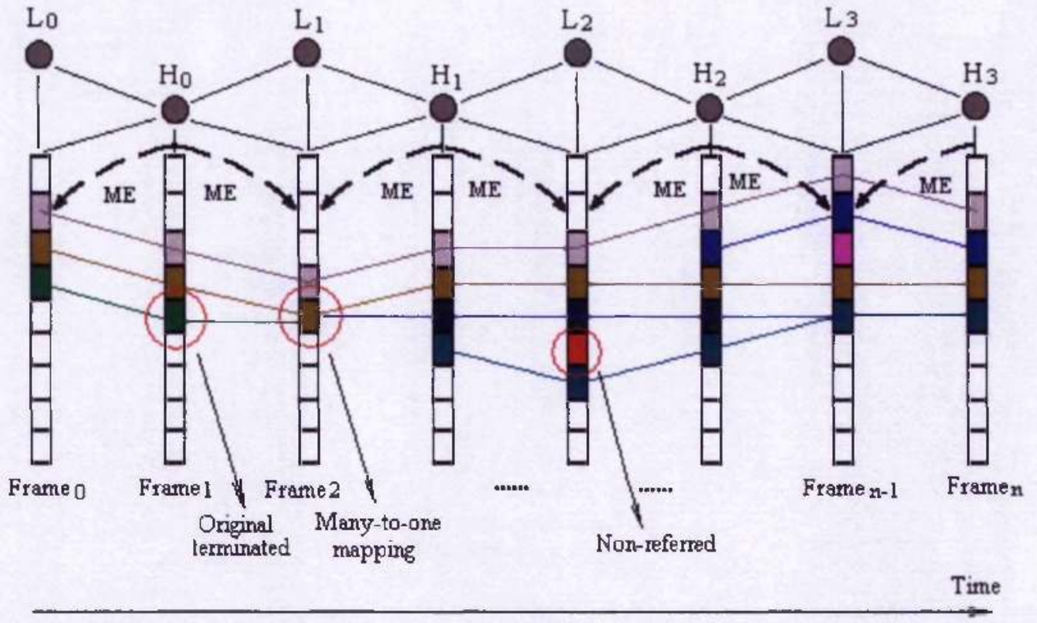


Figure 7.5 – Lifting-based temporal filtering based on bi-directional motion search [LUO-2003]

Figure 7.6 illustrates a first-stage elementary lifting operation in which frame F_{2n+1} is lifted to a high pass coefficient frame. The solid curve with arrow represents the pixel motion vector generated from the block motion estimation, and the dashed curve represents the pixel motion vector which is directly inversed from the solid one. As shown in Figure 7.6, in this lifting stage where frame F_{2n+1} is to be upgraded to a high pass frame, if pixel x_2 in frame F_{2n+1} refers to the half pel between x_1 and x_2 in F_{2n}, then in the next lifting stage where F_{2n} is to be upgraded to a low pass frame, x_2 in F_{2n} will accordingly refer to the half pel between x_2 and x_3 in F_{2n+1}. In other words, the counterpart motion vectors are strictly kept with inverse direction. The quarter-pel operation resembles the similar method. Based on the elementary lifting operation, the reference frames in each lifting stage can be reproduced in the decoder side, thus the perfect reconstruction of the wavelet synthesis is guaranteed.

Many algorithms have employed the lifting scheme for temporal filtering [SECKER-2001] [LUO-2003] [XU-2004]. However, the difference of the proposed scalable 2D model-based texture coding from these methods is that warping motion compensation is applied instead of block-based motion compensation. Warping motion compensation using scalable object mesh model will be discussed in the following sections.

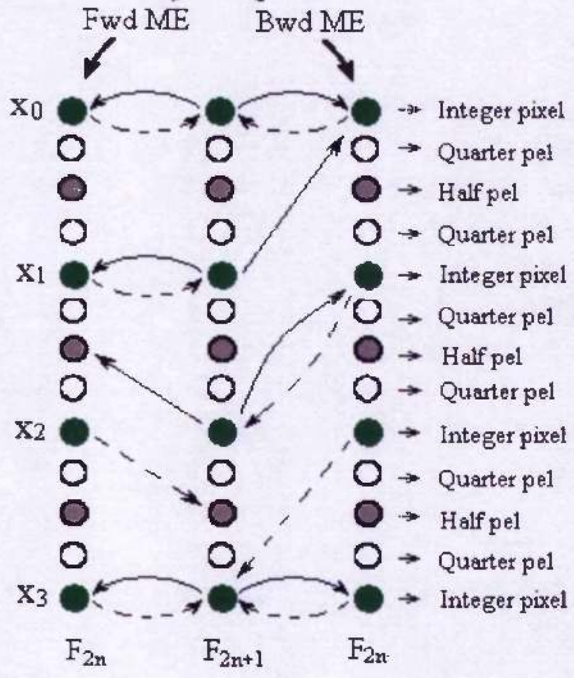


Figure 7.6 – Quarter-pel elementary lifting operation

7.2.2 Warping motion compensation using object mesh model

Block-based motion compensation (MC) for temporal filtering has some inherent effects that can degrade the visual quality of decoded video sequences. One of the effects is blocking artefacts, which are clearly related to the use of block-based MC. If the object motion cannot be represented properly by the block-based motion model, motion-compensated images tend to have a visually noticeable block structure. Following the perfect reconstruction property of the lifting scheme, the block structure of the temporal subband frames can be compensated at the high bit rate. However, when employing a coarser quantisation at lower rates, the block structure becomes visible in the reconstructed frames.

In order to mitigate blocking artefacts, a deblocking mechanism can be applied during reconstruction and can be incorporated into the MC-based temporal lifting scheme. For example, in [XU-2004], overlapped block motion compensation (OBMC) is adopted to improve the performance of motion compensation. Alternatively, warping motion compensation can be employed [HEIS-2001]. In the proposed scheme, warping motion compensation is employed during temporal filtering. Furthermore, as the video frames have been segmented into several video objects, different warping motion compensation schemes have been employed for different video objects.

To help matching every pixel in the target video object, a pre-processing step of padding is applied to the reference video object planes (VOPs) prior to warping motion estimation and compensation for arbitrarily-shaped video object. The boundary pixel padding technique in MPEG-4 has been employed in the proposed scheme [MPEG4-2001]. Only the pixels of the current VOP are considered for matching in motion compensation.

7.2.2.1 Temporal filtering of foreground objects using context-adaptive scalable model

Temporal filtering of foreground objects consists of two basic steps: **scalable model tracking** and **model refinement**.

- **Scalable model tracking:** the scalable model is tracked along the video frames, from the highest temporal layer to the lowest temporal layer, as shown in Figure 7.7.

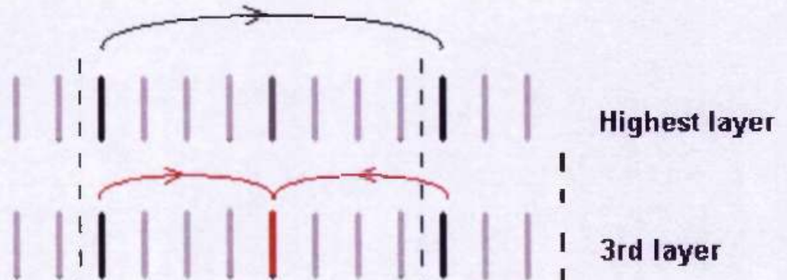


Figure 7.7 – Scalable model tracking along the video frames for video object

For the foreground objects, a context-adaptive scalable mesh model is used during temporal filtering for motion compensation, which is designed through the algorithms discussed in detail in Chapter 5. The advantage of using a context-adaptive scalable mesh model is its high efficiency in representing the object motion. However, compared with the regular triangular mesh model [WANG-1994a], content-adaptable scalable models require more computation during model design. Furthermore, more bits are required to compress these models. In order to reduce the complexity and coding bits in the proposed scheme, only the scalable models of the foreground objects in an "I-frame" are designed and compressed. For the other frames, their scalable models are achieved and updated through tracking. As the object shape has been encoded before texture coding, only the interior vertices of object models need to be tracked.

The detailed tracking algorithm is presented as follows, which is similar to the proposed model evaluation algorithm in Chapter 5:

1 Foreword / backward motion estimation of video object from two frames

In order to estimate the motion of video object, a number of feature points are selected in the interior of object, which may be different from the interior vertices of the object model and have good features for tracking [SHI-1994]. Then, both forward and backward motion vectors of these points between frame $I(\vec{x},t-1)$ and frame $I(\vec{x},t)$ are estimated using Shi-Tomasi feature tracking algorithm in [SHI-1994], which is indicated by V_i and $V_i^{"}$, respectively.

2 Reliability evaluation

The "reliability" of the estimated motion vectors is evaluated based on both forward and background motion vectors through Equation (3.3) in Chapter 3 (page 55). The smaller the difference between V_i and $V_i^{"}$, the more reliable the motion vector of i th node.

3 MV prediction of model vertices

After achieving the motion vectors of interior feature points, the motion vectors of model vertices are predicted from their m nearest surrounding motion vectors. m is chosen as 6 in our experiments. The weighted least squares (WLS) estimation in

[ROUSS-1987] is also used to determine the affine parameters of motion for the control points. During estimation, each motion vector is weighed according to its "reliability".

Unlike the model evaluation algorithm in Chapter 5, the proposed model tracking algorithm does not include object model refinement by iterative hexagonal matching algorithm in [NAKAYA-1994]. A more detailed scalable model refinement step will be introduced in the lifting step of temporal filtering.

- **Scalable model refinement:** The scalable model is refined during the lifting step, from the lowest temporal layer to the highest temporal layer, as shown in Figure 7.8.

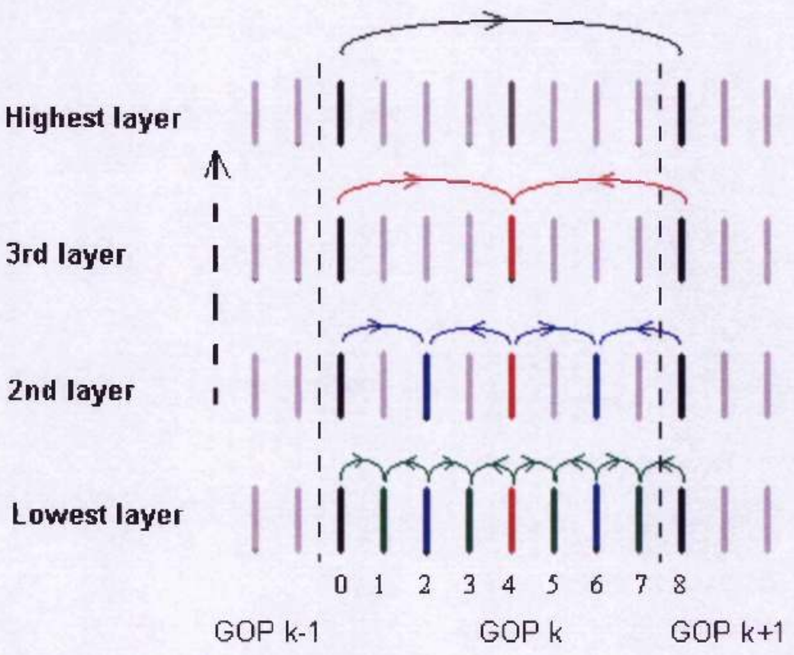


Figure 7.8 – lifting-based temporal filtering process

Even though the motion of the object model can be derived from above tracking algorithms, the motion vectors obtained are not optimal due to the update process of the low-pass frames in the lifting scheme. For example, during temporal filtering of the 2nd layer shown in Figure 7.8, Frame 2 and Frame 6 are first predicted from Frame 0, Frame 4 and Frame 8 as the high-pass temporal frames. Then, Frame 0 and Frame 4 are updated. Due to the updating step, Frame 0 and Frame 4 are different from those in Figure 7.7, thus the tracked scalable model is not optimally related to the updated frames. Therefore, a refinement step is needed in order to get the optimal motion vectors corresponding to the object model.

Due to the object motion, some pixels of the object have just one correspondence from either left or right frame. These pixels can be considered as the original terminated pixels, as shown in Figure 7.5. For example, some newly-appeared pixels in frame n can only be estimated from frame $n+1$, as shown in Figure 7.9. Therefore, during model refinement, optimal prediction model (Bi-directional, Uni-left, or Uni-right models) is also decided. In the proposed scheme, the optimal prediction model is object-based, instead of block-based, due to the warping motion compensation.

The motion estimation is optimised under rate-distortion criteria, which are shown in Equation (7.9), (7.10) for different estimation model.

For Uni-left and Uni-right model, the motion vectors are optimised subject to:

$$MV_d^{opt} = arg \min_{MV_d^j \in SR_d} \left(\sum_{p \in NR_d} \left| B(p) - MC\left(A_d, MV_d^{Neighbor}, MV_d^j\right)(p) \right| + \lambda^{mod\,e} \cdot R\left(MV_d^{Neighbor}, MV_d^j\right) \right) \tag{7.9}$$

For Bi-directional model, the motion vectors are optimised subject to:

$$\begin{pmatrix} MV_l^{opt} \\ MV_r^{opt} \end{pmatrix} = arg \min_{\substack{MV_l^j \in SR_l \\ MV_r^j \in SR_r}} \left(\begin{array}{l} \sum_{p \in NR_d} \left| B(p) - \frac{1}{2}\left(MC\left(A_r, MV_r^{Neighbor}, MV_r^j\right)(p) + MC\left(A_l, MV_l^{Neighbor}, MV_l^j\right)\right)(p) \right| \\ + \lambda^{mod\,e} \cdot \left(R\left(MV_r^{Neighbor}, MV_r^j\right) + R\left(MV_l^{Neighbor}, MV_l^j\right)\right) \end{array} \right) \tag{7.10}$$

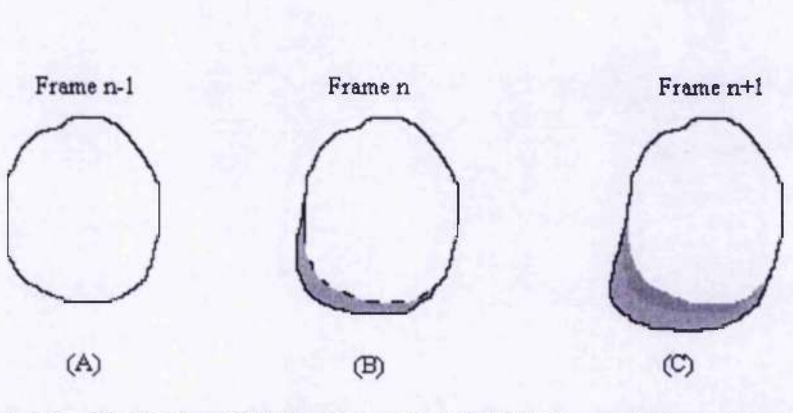


Figure 7.9 – Illustration of video object along the three consecutive video frames

In Equation (7.9) and (7.10), B and A_d $(d \in \{r, l\})$ are the current frame and reference frame during motion compensation, respectively. p is the pixel located in the neighbouring region NR_d of current vertex, as shown in Figure 7.10. MV_r^{Opt} and MV_l^{Opt} are the final estimated motion vectors from the right (forward) and left (backward) motion estimation process respectively. $MC\left(A_d, MV_d^{Neighbor}, MV_r^j\right)$ is the result of warping motion compensation process of current frame, given the reference frame and motion vectors of model vertex and its neighbouring model vertices. $R\left(MV_d^{Neighbor}, MV_r^j\right)$ is a bit rate term representing the expected number of bits for encoding the motion vectors MV_r^j given its neighbouring motion vectors. The search region SR_d can be adapted depending on the temporal layer.

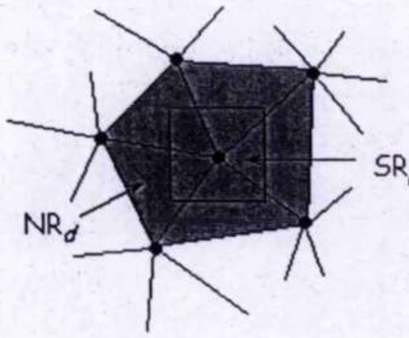
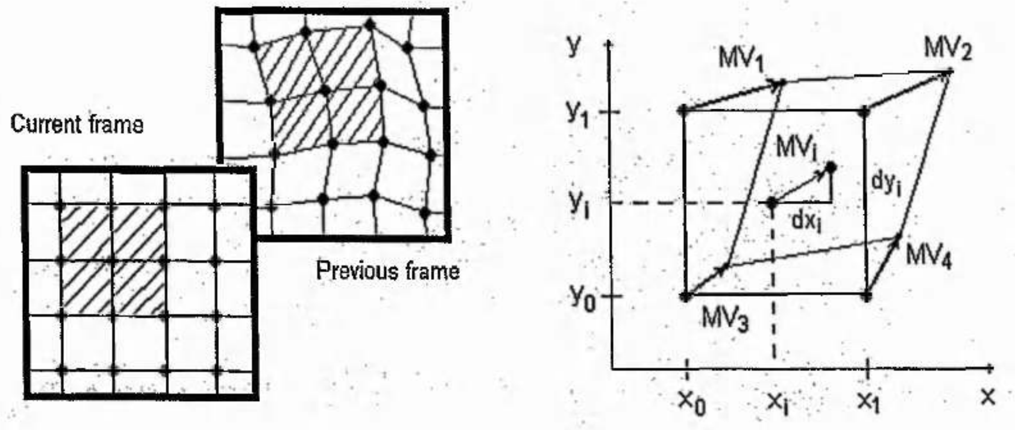


Figure 7.10 – Illustration of neighbouring region NR_d and search region SR_d of model vertex

To approximate the optimal solution for the two-dimensional optimisation problem given in Equation (7.10), we first optimise MV_l using one dimensional optimisation. We then fix MV_l and optimise MV_r. By subsequently fixing MV_r and re-optimising MV_l, this process can be iteratively continued. During the iteration process, the positions of its neighbouring vertices are fixed. After the above refinement, the rate-distortion optimised motion vectors are achieved and will be compressed by using the scalable coding method presented in Section 7.2.3.

7.2.2.2 Temporal filtering of background objects using an adaptive quadrangular mesh model

For the background object of the video frame, it is assumed that only simple motion occurs. Therefore, the simple scalable quadrangular mesh model is employed, instead of context-adaptive scalable model. One of the advantages of using such an adaptive model is that it need not be compressed and transmitted to the decoder. Furthermore, less computation is required to build the adaptive quadrangular mesh models of the background objects.



(a) (b)

Figure 7.11 – Warping motion compensation and motion vector interpolation. (a) Warping motion compensation and (b) Motion vector interpolation

Figure 7.11 illustrates warping motion compensation (MC) using quadrangular meshes. During motion estimation (ME), the positions (or motion vectors) of the grid points in the previous frame are optimised to reduce the warping error between the current frame and the previous frame. It can be achieved through the iterative hexagonal matching algorithm in [NAKAYA-1994]. For the points inside each quadrangle, the motion vectors (MV) are linearly interpolated from the four MVs of the surrounding grid points.

As we know, the drawback of warping MC technique is that it suffers from strong inhomogeneous motion, e.g. very fast moving objects [OHM-1996], leading to "warping artefact". Even though motion discontinuities are reduced through the representation of video frame into objects, overlapped block motion compensation (OBMC) and an adaptive quadtree grid with variable density according to the varying motion activity are investigated and incorporated into the proposed scheme in order to further improve the performance of motion compensation.

OBMC can achieve better prediction by means of a superposition of overlapped displaced blocks from the reference frame, each weighted by a smooth cosine window [HEIS-2001]. According to Figure 7.11, the background object of current frame k is divided into squares of 16 by 16, thus obtaining a regular grid. A dense motion vector field is achieved by using bilinear geometric transform which smoothly varies over the image. The motion vector MV_i of point (i, j) inside a square block is interpolated from the four surrounding control point motion vectors MV_1, ..., MV_4 using the equation (7.11):

$$MV_i = \begin{pmatrix} dx_i \\ dy_i \end{pmatrix} = \begin{pmatrix} dx_1 \\ dy_1 \end{pmatrix} \cdot (\tilde{y}_i - \tilde{x}_i \cdot \tilde{y}_i) + \begin{pmatrix} dx_2 \\ dy_2 \end{pmatrix} (\tilde{x}_i \cdot \tilde{y}_i) + \begin{pmatrix} dx_3 \\ dy_3 \end{pmatrix} (1 - \tilde{x}_i - \tilde{y}_i + \tilde{x}_i \cdot \tilde{y}_i) + \begin{pmatrix} dx_4 \\ dy_4 \end{pmatrix} (\tilde{x}_i - \tilde{x}_i \cdot \tilde{y}_i) \tag{7.11}$$

where $\tilde{x}_i = \frac{x_i - x_0}{x_1 - x_0}$, $\tilde{y}_i = \frac{y_i - y_0}{y_1 - y_0}$ (7.12)

and x_0, y_0, x_1, y_1 are the coordinates of four surrounding points, as shown in Figure 7.11 (b).

This warping prediction leads to a motion vector field without motion discontinuities. For high motion part of background, in order to combat the motion discontinuities, overlapped block motion compensation is employed by superimposing four predicted intensity values using nonlinear weighting functions w_1, w_2, w_3 and w_4.

$$\begin{aligned} p^{obmc}(x_i, y_i, k) &= \sum_{j=1}^{4} p_j^{trans}(x_i, y_i, k) \cdot w_j(x_i, y_i) \\ &\equiv p_1^{trans} \cdot (\hat{y}_i - \hat{x}_i \cdot \hat{y}_i) + p_2^{trans} \cdot (\hat{x}_i \cdot \hat{y}_i) \\ &+ p_3^{trans} \cdot (1 - \hat{x}_i - \hat{y}_i + \hat{x}_i \cdot \hat{y}_i) \\ &+ p_4^{trans} \cdot (\hat{x}_i - \hat{x}_i \cdot \hat{y}_i) \end{aligned} \tag{7.13}$$

where

$$\hat{x}_i \equiv \frac{1}{2}\left(1 - \cos\left(\pi \cdot \frac{x_i - x_0}{x_1 - x_0}\right)\right) \tag{7.14}$$

$$\hat{y}_i \equiv \frac{1}{2}\left(1 - \cos\left(\pi \cdot \frac{y_i - y_0}{y_1 - y_0}\right)\right) \tag{7.15}$$

and $p_j^{trans} \equiv I(x_i + dx_j, y_i + dy_j, k-1)$, $j = 1,\ldots,4$ (7.16)

Thus, the four predicted values p_1^{trans}, p_2^{trans}, p_3^{trans} and p_4^{trans} are computed by employing the translational motion model with one of the motion vectors of the four surrounding vertices for each prediction.

In order to further improve the motion prediction and achieve scalable coding, an adaptive quadtree grid is employed. In contrast to the method in [HUANG-1994] which uses the local variance of a given frame difference as a criterion to decide whether block should be split into smaller one, the splitting in our proposed scheme is based on the rate-distortion criterion. Commonly, the use of a regular grid of block size 16x16 pels imposes a severe constraint on the motion model in highly active and quasi-stationary regions. In order to adapt the motion model to such kind of scenes, an irregular grid is employed with block sizes of 16x16 and 8x8.

As shown in Figure 7.12, the control points (CPs) in the grid are classified into three types: coarse grid CP, fine grid CP and boundary vertex (BV). They are defined as follows:

- Coarse grid CP: Control points located at the 16x16 grid, which can move freely.
- Fine grid CP: Control points located at the 8x8 grid, which can move freely.
- Boundary vertex (BV): control points located at the 8x8 grid and forming a T-shaped connection with coarser grid CP and fine grid CP. Its motion vector is just bilinearly interpolated from the MVs of neighbouring CPs and hence need not be transmitted.

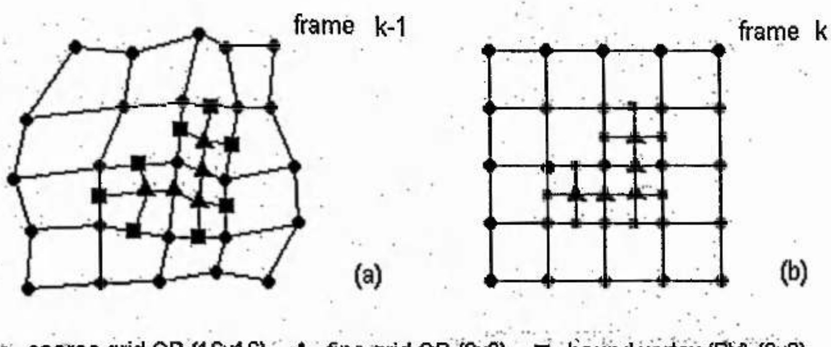


Figure 7.12 – Description of hierarchical control grid interpolation

Iterative hexagonal matching algorithm in [NAKAYA-1994] is used to estimate and refine the motion vector of the control points. It includes the following steps:

- Motion vector of control points on the grid of size 16x16 is obtained.
- Rate-distortion theory is applied to decide whether the 16x16 block should be split or not.
- If the decrease of MSE of 16×16 block is larger than $\lambda^{split}\cdot R(\Delta MV)$, the block is split and a fine grid CP is inserted. Otherwise, it is not split.

For background objects, only Bi-directional prediction is employed. That is, during the motion estimation, two motion vectors are estimated for every control point (vertex of a square) from the adjacent frames, based on rate-distortion criterion. That is, a Lagrange multiplier λ^{model} is used to choose the best control point motion vector, considering the prediction error and the local motion vector variance between the candidature vector $\left(MV_l^j, MV_r^j\right)$ and the eight motion vectors $\left(MV_r^{Neighbor}, MV_l^{Neighbor}\right)$ of its neighbouring control points, which is similar to equation (7.10).

To estimate the motion vector of the vertex, the motion vectors of its eight neighbouring control points are fixed and only the motion vector of the centre control point is changed. Because of their interdependence, the motion vectors are iteratively refined. During the iteration, the control points are scanned from top left to bottom right of the image. The estimated MVs of the quadrangular mesh model, together with the overall structure of the grid described by a quadtree, are encoded progressively, which will be discussed in Section 7.2.3.

7.2.3 Scalable coding of motion vectors

Before scalable coding of motion vectors for foreground and background objects, the scalable mesh structure of video object is compressed. The detailed technique for the compression of scalable mesh model has been discussed in Chapter 5, which also indicates the scanning sequence and layer information of vertices. Based on the frame prediction model, such as bi-directional, uni-left, and uni-right prediction model, one or two motion vectors are needed for every vertex of an object model. For background objects, the overall structure of a hierarchical grid can be described by a quadtree [HUANG-1994]. In this section, motion vector prediction is discussed, followed by scalable motion vector coding.

7.2.3.1 Motion vector prediction

For the motion vector prediction of the foreground objects, the motion vector prediction method is similar to the prediction method for scalable model compression discussed in Chapter 5. For the vertices in the first layer, motion vectors are predicted from the preceding vertex based on the pre-decided connectivity during scalable model compression. For other layers, the motion vector of the vertex is predicted from its two neighbouring vertices of the current layer and/or the previously coded layers, according to the pre-determined connectivity information.

The motion vector prediction of background objects is different from that of foreground objects. The scanning sequence during motion vector prediction is from upper left to bottom right. Based on the hierarchical control grid in Figure 7.12, the motion vector of the vertex in current layer can be predicted from the neighbouring vertices of the previously encoded layers and the vertices the current layer which have already been encoded, as shown in Figure 7.13.

Assume that the fine grid control points (CPs) belong to a different layer from the coarse grid CPs. The coarse grid CP is predicted from its neighbouring encoded vertices, as shown in Figure 7.13 (a). For example, CP_a is predicted from CP_b and CP_c. The fine grid CP is predicted from its neighbouring coarse grid CPs, neighbouring bound vertex, and encoded fine grid CPs. For example, in Figure 7.13 (b), fine grid CP_e is predicted from CP_b, CP_d and CP_f. While fine grid CP_h is predicted from CP_e, CP_g and CP_i. As the motion vector of boundary vertex (BV) is bilinearly interpolated from the MVs of neighbouring CPs, it need not be compressed and transmitted to the decoder.

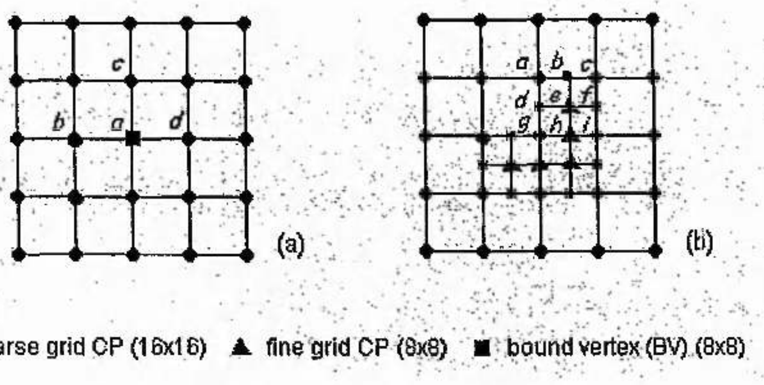


Figure 7.13 – Illustration of motion vector prediction for a background object

7.2.3.2 Scalable coding

In MPEG-4 and H.263, the differential MV components are encoded using adaptive arithmetic coding (AAC), described by Witten et al [WITTEN-1987]. In these standards, one probability model is used for all the motion vector symbols in a given frame and updated adaptively at the encoder and decoder. As the number of symbol increases, this scheme faces the zero frequency problem, i.e. even the unused symbols must be assigned some initial probability.

In order to improve the coding efficiency of MV prediction errors, H.264 adopted context-adaptive binary arithmetic codec (CABAC) [H264-2003], which is also employed for scalable motion vector coding of our proposed scheme. This method consists of two steps: binarization and binary coding. In the binarization step, each motion vector symbol is represented by a unique binary pattern as shown in Table 7.1. The resulting code words are then encoded using a binary arithmetic coder and context model according to Table 7.2.

Table 7.1 – Binarization of motion vector prediction residual component

Prediction residual component	Binary code
0	1
+/- 0.25	0 1
+/- 0.5	0 0 1
+/- 0.75	0 0 0 1
....	....
Bin number	1 2 3 4 5 ......

Table 7.2 – Bin numbers and corresponding context number for binarised residual motion vector components

Bin number	Context number
1	$CTX^c(dE) \in \{0,1,2\}$
2	3
3	4
4, 5, 6, ...	5
Sign	6

Motion vector differences are prediction residuals, for which a context model is established in CABAC that is based on the local prediction error. Let $mvd(X,cmp)$ denotes the value of a motion vector difference component of direction $cmp \in (horizontal, vertical)$. Then, the related context for encoding the first bin is determined by its preceding neighbour (for layer 0) or neighbours (for other layers). Three different context models $CTX^c(dE)$ are selected depending on the motion vector residuals of the neighbouring control points, which have been scanned and encoded. Let $c \in \{x,y\}$ denote the vector component. Then, $CTX^c(dE)$ is defined as follows:

$$CTX^c(dE) = \begin{cases} 0, & e^c(dE) < 2 \\ 1, & e^c(dE) > 8 \\ 2, & else \end{cases} \tag{7.17}$$

where $e(dE)=\sum_i |dA_i|$ and i is the number of its neighbours.

7.2.4 Rate-distortion optimised bit stream truncation

After temporal filtering of arbitrarily shaped video object, scalable motion vector coding is conducted using the method in Section 7.2.3, and the residual texture images are encoded by the improved shape-adaptive SPECK algorithm which has been discussed in Chapter 6. After the encoding process, the compressed bitstream can be further truncated at a later stage to form a fully scalable bitstream in the temporal and quality level.

In the proposed scheme, rate-distortion optimised bit truncation scheme used in JPEG2000 [JPEG-2000] has been extended to decide the optimal truncation points of the bit stream. Based on the rate distortion data of all video objects and the given available bit rate, the optimal truncation points can be decided within each GOP. During temporal filtering, the warping error $D_i^{l,j_{mv}}$ and the expected number of bits $R_i^{l,j_{mv}}$ for object i in motion layer j_{mv} of frame l are recoded. At the same time, for residual texture bit-plane coding, the coding rate $R_i^{l,j_{sp}}$ and distortion $D_i^{l,j_{sp}}$ for object i in bit plane j_{sp} of frame l are also recoded.

Based on rate-distortion theory, the coding performance of each video object i (including motion and residual texture for video coding) is characterised by a rate-distortion curve. Therefore, optimal trade-off between rate and distortion can be achieved by minimising:

$$J=\sum_l J_l=\sum_l\sum_i\sum_j\left(D_i^{l,j}+\lambda_{l,i}R_i^{l,j}\right) \qquad (7.18)$$

where $j\in\{j_{mv},j_{sp}\}$ and is selected from the candidature truncation points. Lagrange parameters $\{\lambda_{l,i}\}$ are chosen for all temporal frames and video objects. If only one frame is encoded and each block is considered as one video object, Equation (7.18) can be simplified as the rate-distortion optimisation criterion used in EBCOT algorithm [TAUB-2000].

Given a specific bit-rate R_{max}, the objective of bit truncation is to find the optimal truncation point so that $\sum_l\sum_i\sum_j R_i^{l,j}\le R_{max}$, and construct a bit stream that satisfies the bit-rate constraint and with minimal distortion. Before deciding the optimal truncation points, we should select the candidature truncation points so that we can find a convex hull of the $R-D$ curve. The truncation can only take place at these candidate truncation points so as to guarantee that at every truncation point, the bitstream is rate-distortion optimised. For object i, the end of each MV layer is the feasible truncation points followed by the feasible truncation points that are located at the end of each coding pass for texture coding.

Let $j_0 < j_1 < \cdots < j_{N-1}$ be an enumeration of these feasible truncation points for object i in frame l, where $\{j_0, j_1, \cdots, j_k\} \in j_{mv}$ and $\{j_{k+1}, j_{k+2}, \cdots, j_{N-1}\} \in j_{sp}$. Let the corresponding distortion rate "slope" be given by:

$$S_i^{j_k} = \Delta D_i^{j_k} / \Delta R_i^{j_{k-1}} \tag{7.19}$$

where $\Delta R_i^{j_k} = R_i^{j_k} - R_i^{j_{k-1}}$ and $\Delta D_i^{j_k} = D_i^{j_{k-1}} - D_i^{j_k}$.

Evidently, the slopes must be strictly decreasing. If $S_i^{j_{k+1}} \geq S_i^{j_k}$, the truncation point j_k could never be selected as the candidature truncation points. After selection, the number of candidature truncation points is N_i^l.

After determining the candidature truncation points for all video objects and all frames within current GOP, the determination of optimal truncation points $\vec{n}_i = \{n_i^0, n_i^1, \cdots, n_i^{K-1}\}$, for any given λ_i^l, may be performed very efficiently. Due to the restriction of the set of candidature truncation points whose slopes are strictly decreasing, the algorithm for determining optimal truncation points is reduced to the selection of $\vec{n}_i$ so that its component

$$n_i^l = max\left\{j_k \in N_i^l \middle| S_i^{l,j_k} > \lambda\right\}, \text{and } 0 \leq l < K. \tag{7.20}$$

Suppose there are O objects in the frames and K frames in current GOP. Then, the detailed algorithm for determining truncating points is described as follows:

1. Construct the rate vector $\vec{R} = \{R_i^{l,j} \mid 0 \leq i < O, 0 \leq l < K, 0 \leq j < N_i^l\}$ and slope vector $\vec{S} = \{S_i^{l,j} \mid 0 \leq i < O, 0 \leq l < K, 0 \leq j < N_i^l\}$ for all candidature truncation points. Therefore, for rate and slope vector, there are in total $N = \sum_{i=0}^{O-1} \sum_{l=0}^{K-1} N_i^l$ elements.

2. Rank the elements of $\vec{S} = \{S_i^{l,j}\}$ and adjust the element sequence of rate vector correspondingly so that $\vec{S} = \{S_0, \cdots, S_{j-1}, S_j, S_{j+1} \cdots, S_{N-1}\}$ and $S_{j-1} \le S_j \le S_{j+1}$. The corresponding rate vector is $\vec{R} = \{R_0, \cdots, R_{j-1}, R_j, R_{j+1} \cdots, R_{N-1}\}$.

3. Decide the truncation point n based on rate vector $\vec{R}$ so that $\sum_{i=0}^{n-1} R_i \le R_{max} < \sum_{i=0}^{n} R_i$.

4. Retrieving the optimal truncation points $\vec{N} = \{n_l^0, n_l^1, \cdots, n_l^{K-1} \mid 0 \le i < O\}$ for all video objects and all frames with current GOP from the selected n elements of rate and slope vectors in step 3.

After deciding the optimal truncation points, the bit number $R_i^{l,j}$ and the slope $S_i^{l,j}$ for each $j \in n_i^l$ are kept in the header along with the embedded bit stream.

The proposed rate truncation scheme can easily achieve temporal, quality and object scalabilities. Since the video objects are encoded independently, the bitstream of video objects is separable. The decoder can easily extract special video objects and decode them, so the manipulation of video object is natural. Furthermore, as the video frames are encoded independently, temporal scalability can be easily achieved by throwing the bits from all video objects of high temporal level frames.

To achieve quality scalability, a multi-layer bitstream is formed and each layer indicates a certain quality level. To make a N-layer bitstream, we first select $\lambda_{l,i}^1 > \lambda_{l,i}^2 > \cdots > \lambda_{l,i}^N$ which satisfy $\sum_l \sum_i R_{l,i}^{\lambda_{l,i}^N} \le R_{max}$. With every threshold, a truncation point and a layer of bitstream can be achieved for each video object. The corresponding layers from all video frames and all video objects constitute the layers of the final bitstream. According to the available bandwidth and the computation capacity, the decoder can select first few layers to be decoded. The bit-plane coding and multiple video objects ensure that the bitstream is embedded with fine granularity.

Above algorithm can also be used to achieve object scalability easily by allocating the bits among different objects optimally. The rate-distortion theory indicates that optimal coding performance can be achieved if all of the video objects operate on the same R-D curve. The functionality of the algorithm is thus to find the common rate-distortion slope of all video objects, and calculate the number of included bits for each object. The final bitstream consists of the truncated block bitstream and the bitstream length of each video object.

Above algorithm can also achieve rate control among the video objects easily. Since each object and each frame in the GOP are encoded independently, the bitstream of each object is separable. The decoder can easily extract only a few video objects and decode them. For example, if we are more interested in some special video objects, we can assign smaller Lagrange multipliers $\lambda_{l,j}$ to these video objects according to their importance during R-D optimization for multiple video objects, and code these regions by operating at points of less negative slope on the $D(R)$ curve. Less interesting regions are assigned larger Lagrange multipliers so that the operating points on the $D(R)$ curve have more negative slopes.

In general, according to the requirement of applications, the final bitstream can be constructed in order to meet the requirement. The preceding multi-layer bitstream construction method enables the bitstream with quality scalability. To obtain resolution or temporal (frame rate) scalability, the bitstream can be assembled subband-by-subband, with the lower resolution or low temporal subband in the beginning. Moreover, as the bit number $R_i^{l,j}$ and the slope $S_i^{l,j}$ for each $j \in n_i^l$ are included in the header of the bitstream, the final bitstream can be rearranged to meet further requirements. This property makes the final bitstream very flexible to be reused for all sorts of applications without re-encoding them.

7.3 Experimental results

The proposed scalable 2D model-based texture coding scheme has been fully implemented. Its performance has been investigated and compared with state-of-the-art video coding standards, H.264 [H264-2003] and MPEG-4 [MPEG4-2001]. Currently, H.264 is the best available video codec, which can match the best possible MPEG-2 quality at up to half the data rate. H.264 also delivers excellent video quality across the entire bandwidth spectrum - from 3G to HD and everything in between (from 40 Kbps to upwards of 10 Mbps). However, H.264 does not support object-based video coding. MPEG-4 can support both frame-based and object-based video coding. Therefore, these two available codecs are selected during experiments.

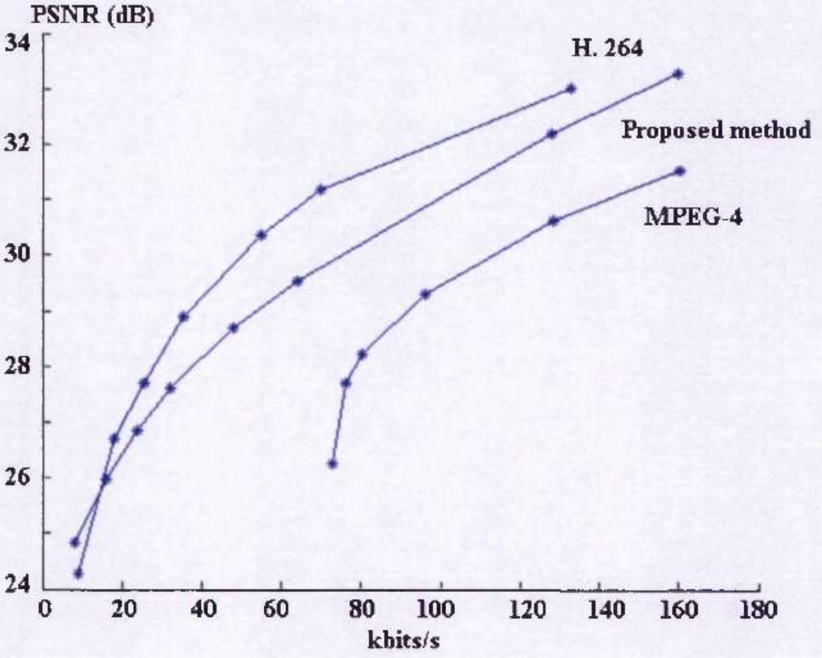
In order to compare the performance of the proposed scheme with H. 264, the video frame is segmented into two video objects: one foreground object and one background object. Two video objects are encoded and decoded separately, and then used to reconstruct the decoded video frame. This coding process is named as *frame-based texture coding*. If only one video object is encoded and decoded so as to compare it with the object-based coding of MPEG-4, this coding process is named as *object-based texture coding*.

7.3.1 Comparison with MPEG-4 and H. 264 for frame-based texture coding

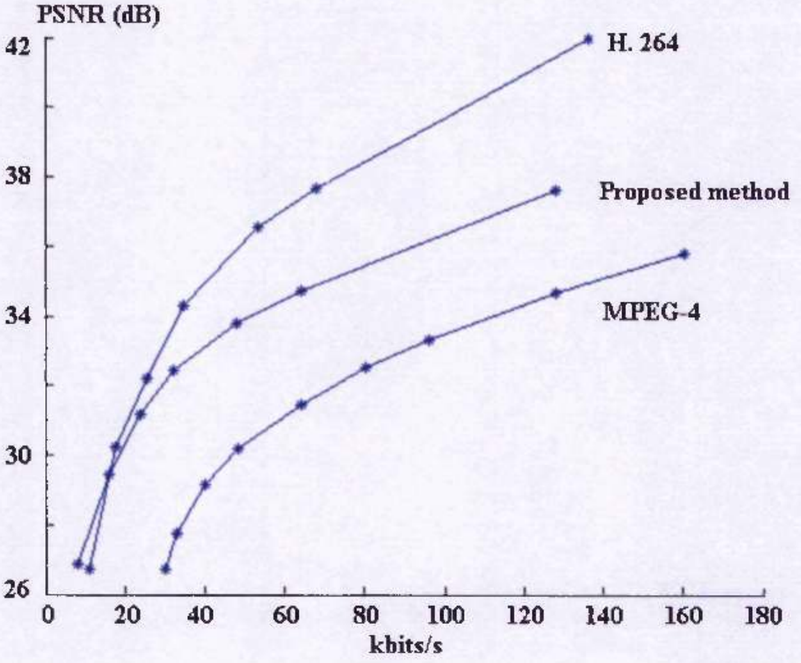
Extensive experiments have been conducted to evaluate the performance of the proposed scalable 2D model-based texture coding scheme for frame-based texture coding, and compare it with that of MPEG-4 and H 264 standards. Three test video clips, including Coastguard, News, and Motr_dhtr sequence in QCIF resolution (with 10 fps, 160 frames), were used in the experiments. For MPEG-4, Microsoft version is employed. For H. 264, version JM82 is applied. During the tests, rate control is enabled for both MPEG-4 and H. 264. For MPEG-4, TM5 is selected as the rate control method.

Figure 7.14 illustrates the PSNR performance of Y-component for different encoding bit rates. Readers are reminded that the experimental results presented here for each sequence are decoded from a single embedded bitstream for the proposed encoding method and from different bitstreams corresponding to individual target coding rates for MPEG-4 and H. 264. From the results, it is found that the proposed method is 1 - 4 dB superior to the MPEG-4 coder for a wide range of bit rates. When compared with H.264, the proposed scheme can achieve better compression performance at the low bit rate. However, it is inferior to H. 264 at medium and high bit rates. The success is due to the use of scalable MV coding in the proposed scheme. When the target bit rate is very low, which is not enough to encode the full MV information, only the first MV layers are encoded and some bits are saved to encode the first frame of GOP. In the proposed scheme, the amount of bits used for encoding MV information is decided automatically by the rate-distortion optimised bit truncation algorithm discussed in Section 7.2.4. Most importantly, the proposed scheme can achieve highly scalable bit stream, which is important for the new applications such as UMA. It can achieve temporal, object and quality scalabilities simultaneously.

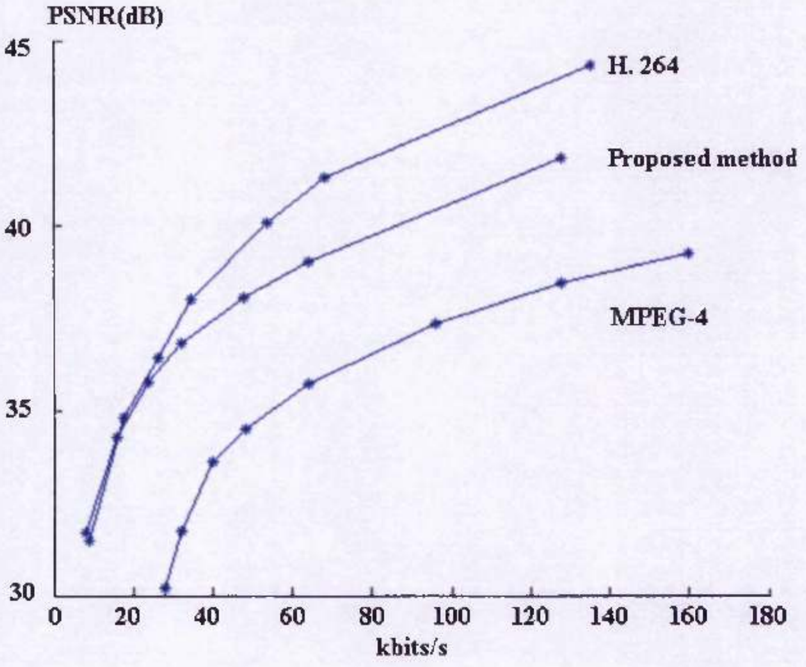
Regarding to the computational complexity, the proposed scheme has higher complexity than MPEG-4 and H.264 due to the video segmentation, object modelling and MC-based temporal filtering. During texture coding, the complexity of MC-based temporal filtering is larger than ME/MC in MPEG-4 and H.264 as mesh-based motion estimation and compensation is employed.


PSNR (dB)
34
32
30
28
26
24
0
20
40
60
80
100
120
140
160
180
kbits/s
H. 264
Proposed method
MPEG-4

(a)


PSNR (dB)
42
38
34
30
26
0
20
40
60
80
100
120
140
160
180
kbits/s
H. 264
Proposed method
MPEG-4

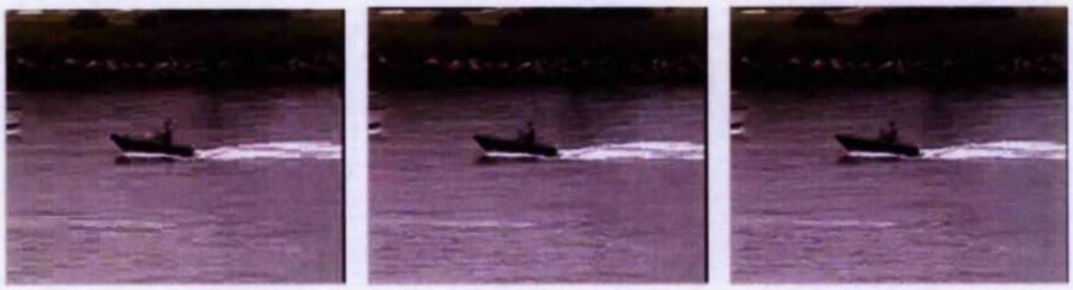
(b)



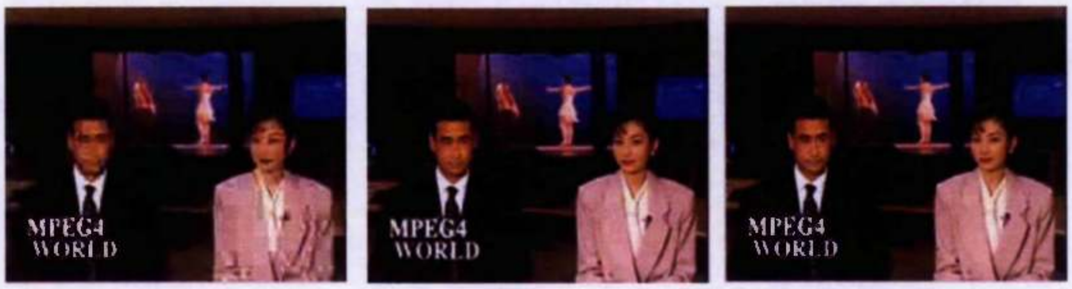
(c)

Figure 7.14 – PSNR performance comparison for Y-component of (a) Coastguard; (b) News; and (c) Motr_dhtr sequence

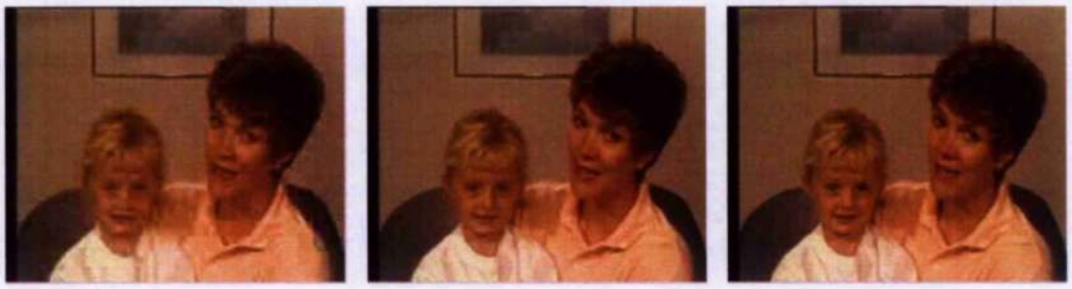
Figure 7.15 shows the decoded video frames for MPEG-4, the proposed scheme and H. 264. It is shown that MPEG-4 produces visually annoying blocking artefacts. The decoded images through the proposed scheme and H. 264 have better subjective performance.



(a)



(b)



(c)

Figure 7.15 – Encoding performance comparison for (a) Coastguard with 128kbits/s; (b) News with 64kbits/s; and (c) Motr_dhtr sequence with 64kbits/s. The left image is encoded by MPEG-4. The middle image is encoded by our proposed scheme. The right one is encoded by H. 264 encoder

Figure 7.16 shows the PSNR distribution of Y component for Coastguard sequence under the bit rate of 128kbits/s, as well as the results of MPEG-4 and H.264 codec (for MPEG-4 and H.264, the actual bit rate is 128.17 kbits/s, 133kbits/s respectively).

Figure 7.17 shows the PSNR distribution of Y component for News sequence under the bit rate of 64kbits/s, as well as the results of MPEG-4 and H.264 (for MPEG-4 and H.264, the actual bit rate is 64.16 kbits/s, 67.97 kbits/s respectively).

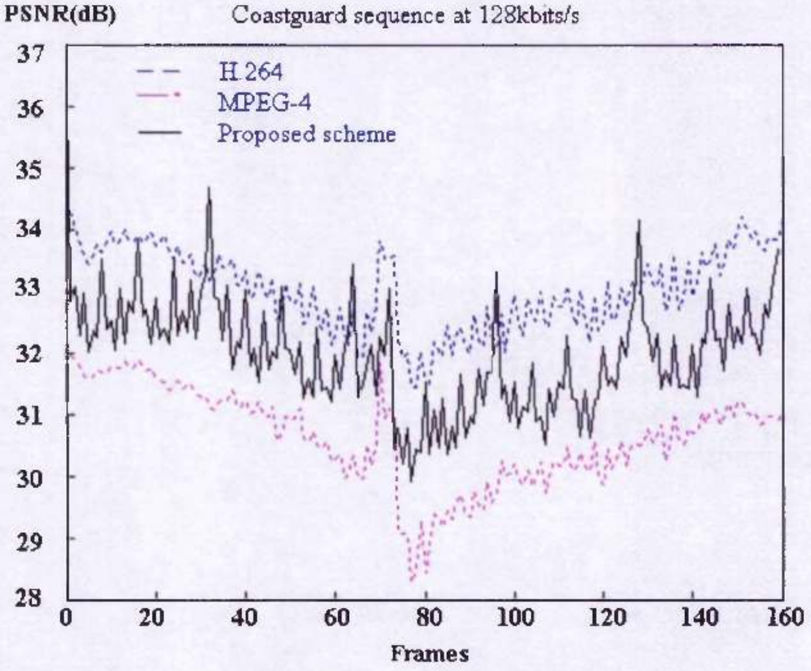


Figure 7.16 – PSNR performance (Y component) of H.264, MPEG-4 and the proposed scheme

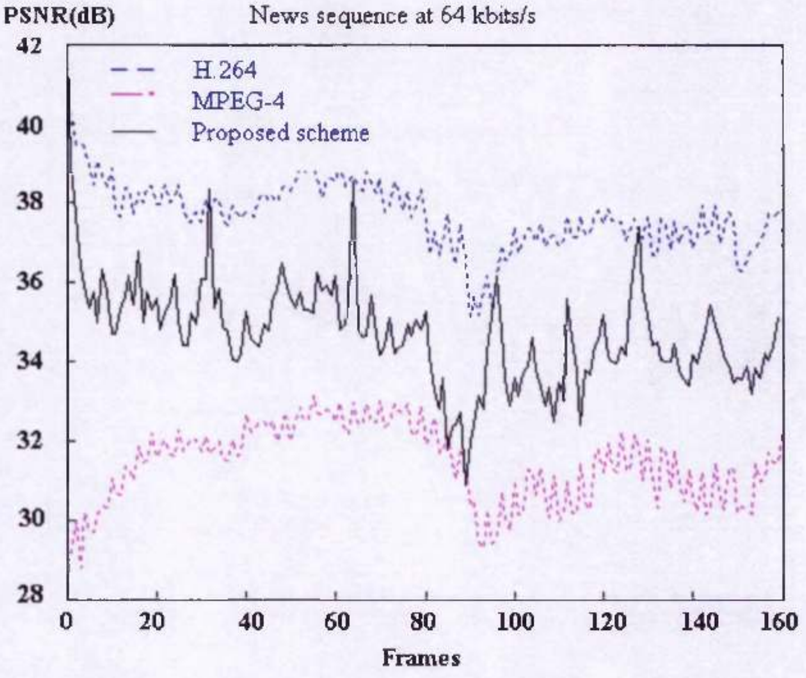


Figure 7.17 – PSNR performance (Y component) of H.264, MPEG-4 and the proposed scheme

Figure 7.18 shows the PSNR distribution of Y component for Motr_dhtr sequence under the bit rate of 64kbits/s, as well as the results of MPEG-4 and H.264 codec (for MPEG-4 and H.264, the actual bit rate is 64.12 kbits/s, 68.36 kbits/s respectively).

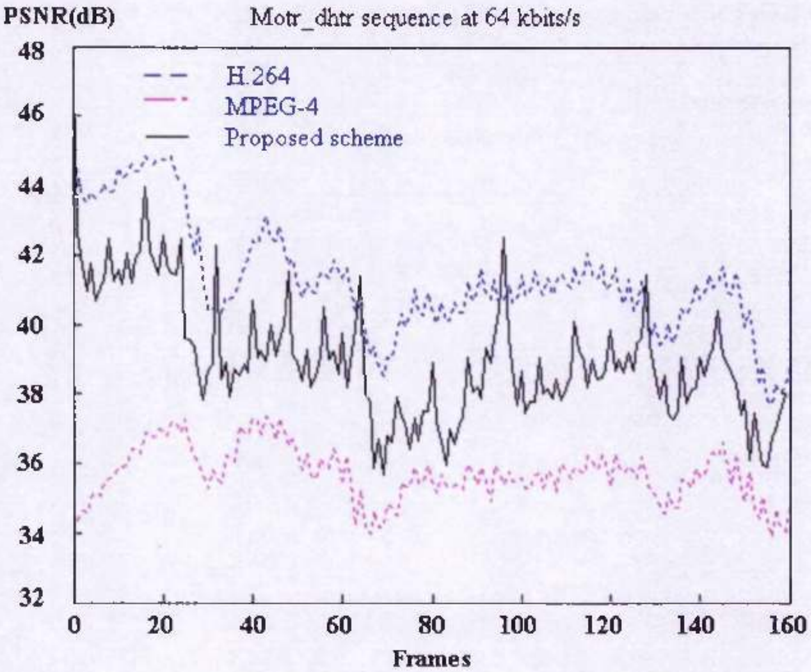


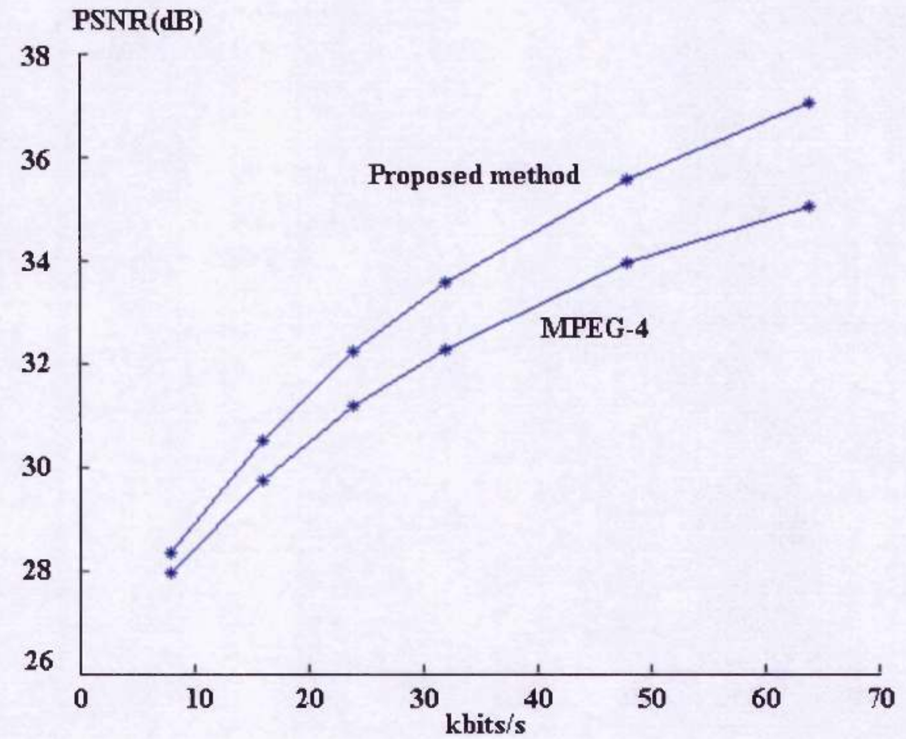
Figure 7.18 – PSNR performance (Y component) of H.264, MPEG-4 and the proposed scheme

7.3.2 Comparison with MPEG-4 for object-based texture coding

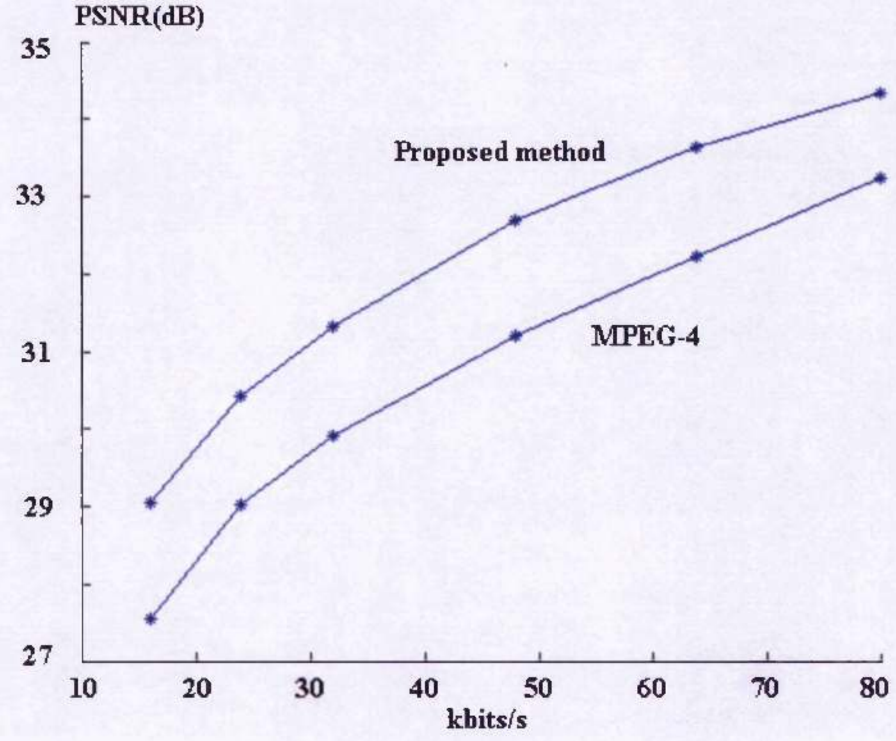
The proposed algorithm is compared with an MPEG-4 coder that uses object-based texture coding. Four test video clips with 100 frames are used: Claire, Motr_dhtr, News, and Foreman sequence in QCIF (10fps). For each test sequence, only the foreground video object is encoded. During test, "MP4" rate control model is selected for MPEG-4, which is the only model to support object-based video coding.

Figure 7.19 shows the PSNR values for Y component at different coding bitrates for MPEG-4 and the proposed scheme. It is shown that the proposed coder outperforms MPEG-4 (object-based) by 1-2dB while providing desirable multi-rate features. The experimental results presented here for each sequence are decoded from a single embedded bitstream for the proposed encoding method and from different bitstreams corresponding to individual target coding rates for MPEG-4.

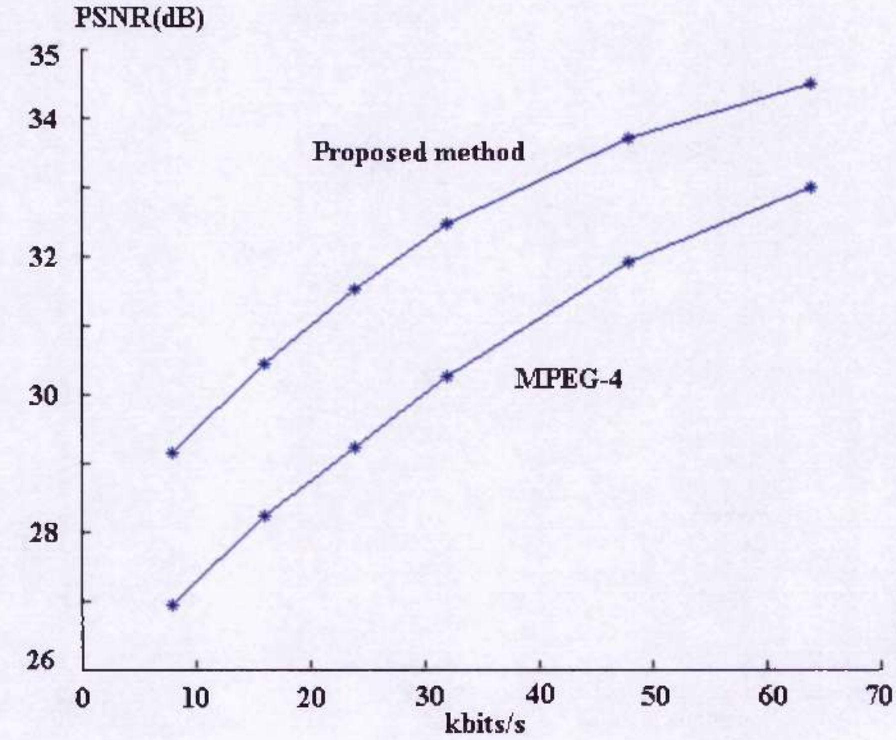
Figure 7.20 shows the performance comparison for different video objects under different video encoding rates. The left image is encoded by MPEG-4. The right one is encoded by the proposed scheme. It can be shown that our proposed scheme can achieve better visual performance.



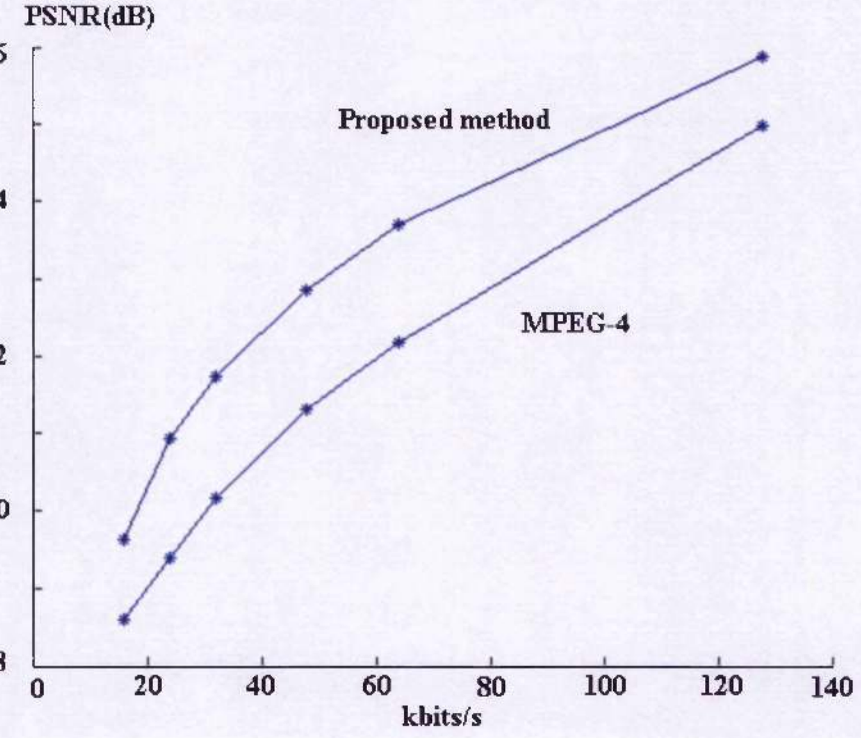
(a)


PSNR(dB)
35
33
31
29
27
10
20
30
40
50
60
70
80
kbits/s
Proposed method
MPEG-4

(b)

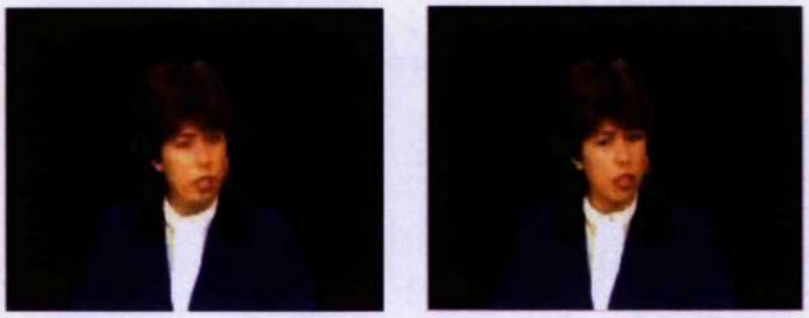

PSNR(dB)
35
34
32
30
28
26
0
10
20
30
40
50
60
70
kbits/s
Proposed method
MPEG-4

(c)



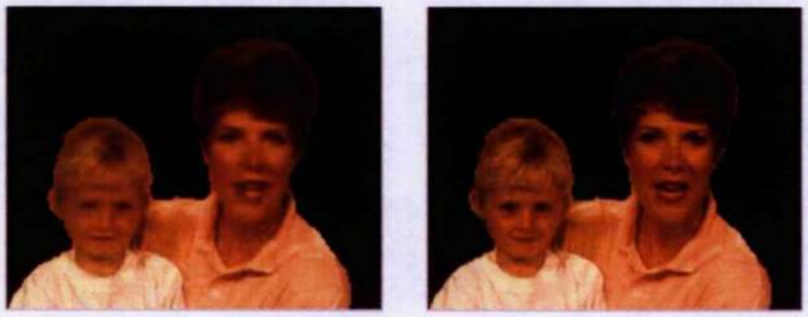
(d)

Figure 7.19 – PSNR performance comparison for Y-component of (a) Claire; (b) Foreman; (c) Motr_dhtr, and (d) News sequence.

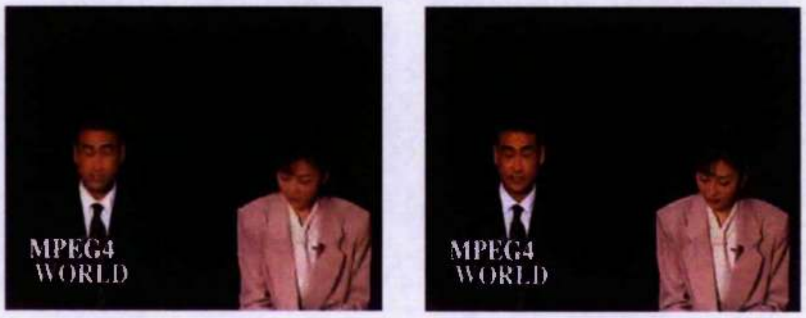


(a)

(b)

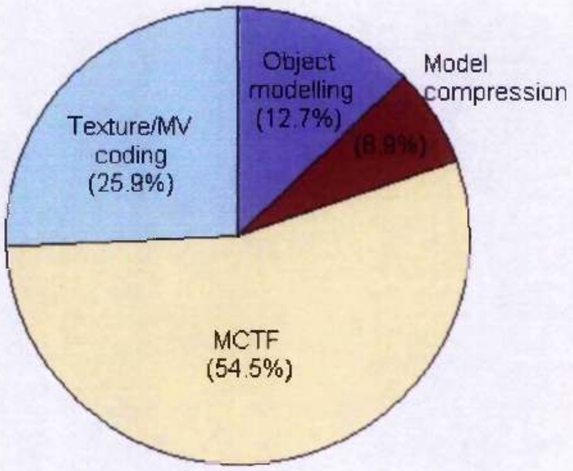


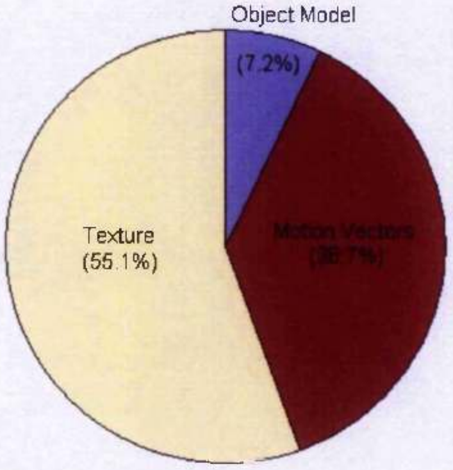
(c)



(d)

Figure 7.20 – Encoding performance comparison for: (a) Claire with 24kbits/s; (b) Foreman with 48kbits/s; (c) Motr_dhtr with 24kbits/s and (d) News with 48kbits/s. (Left) MPEG-4; (right) the proposed method





(a) Complexity of each component (b) Bit usage of each component

Figure 7.21 – Complexity and bit usage of the proposed scalable 2-D model-based video coding scheme

The complexity and the bit usage of each component in the whole video coding system have also been evaluated. Figure 7.21 shows the complexity and bit usage of the proposed scalable 2D model-based video coding system for Motr_dhtr sequence. Currently, video segmentation is a time-consuming part of the proposed system. It almost occupies 40% computation complexity. Figure 7.21 (a) shows the average proportion of the computational time for the system. However, video segmentation is not included here. It is shown that speeding the MCTF is necessary to achieve real-time video compression. Figure 7.21 (b) shows the bit usage of the components in the proposed system Motr_dhtr sequence under bit rate 32 kbits/s. It is shown that only few percentages (7.2%) of the total bits are used for scalable model and shape compression.

7.4 Conclusions

This chapter presents a scalable 2D model-based texture coding scheme. We first reviewed some scalable video coding methods, some of which have just emerged for the proposal of MPEG scalable video coding (SVC). The properties and limitations of these techniques are also mentioned.

In order to avoid the disadvantages of existing techniques, we proposed a scalable 2D model-based texture coding scheme. In the proposed scheme, video sequence is first segmented into several video objects with different motion patterns. This makes the temporal filtering easy for the occlusions and the newly appeared patches. The foreground video object is represented by 3-layer scalable content-adaptive object model before texture encoding. The background object is represented by adaptive quadrangular mesh. In the proposed texture coding technique, warping motion compensated temporal filtering is conducted before wavelet-based residual image coding. An improved shape-adaptive SPECK algorithm is used to code the "I-frame" and residual frames. After encoding the motion vectors and texture frames progressively, the bit allocation and optimal truncation scheme in EBCOT algorithm have been extended to facilitate bit allocation among video objects, among video frames and among motion and texture components within GOP.

The experimental results show that the proposed scalable 2D model-based texture coding scheme outperforms the nonscalable MPEG-4 standard in both objective and subjective evaluation over a wide range of bitrates and for both frame-based and object-based texture coding. Although its performance is inferior to H.264 standard in middle to high bit rate ranges, the proposed scheme is superior to H.264 for the low bit rate coding, due to the scalable coding of motion vectors. Most importantly, the proposed scheme provides highly scalable bit stream, which can achieve temporal, quality and object scalabilities simultaneously. This property is very important for achieving new functionalities, such as Universal Multimedia Access (UMA).

Chapter 8

Conclusions

We round off the thesis with a summary of contributions of this work and some directions for future research.

8.1 Contributions of the Thesis

The overall goal of this research work is to design a scalable 2D model-based video coding system. The original work is set out in Chapters 2-7. The main contributions of the research are summarised below:

- A novel scalable 2D model-based video coding system is presented in this thesis. The proposed system consists of video segmentation and object modelling (including face detection and modelling), scalable model compression and scalable texture compression. The proposed system is capable of achieving scalable video coding with good compression performance at very low bit rates. It can also achieve temporal, quality and object scalability simultaneously.

- A new video segmentation scheme targeted at reducing human interaction during video segmentation is presented in Chapter 3, which is based on the proposed complexity-scalable contour tracking algorithm. First, watershed transform is used to segment the video frame into uniform and homogeneous patches with respect to colour. The motion and user input information is used to merge the patches into objects with semantic meaning. Next, subsequent frames are segmented by using the proposed contour tracking algorithms. It is experimentally demonstrated that the proposed contour tracking algorithm is robust for tracking the object contour with non-rigid and large motion, even with partial occlusion. The tracking results of each step can be used for some special applications with different accuracy requirements. As object motion and texture are used in different steps of tracking process, the proposed algorithm can be considered as a hybrid feature-based, texture-based and contour-based tracking algorithm.

- A robust face detection and facial feature extraction scheme is presented in Chapter 4. This scheme unitises the luminance-adaptive skin colour model and Bayesian detection / relaxation for face extraction, which makes face extraction robust to different skin colour and lighting conditions. After localising the human face, a simple and reliable facial feature detection scheme is developed for eye and mouth detection. A robust chin detection algorithm is also proposed with the combination of active snake with prior shape model. In the chin detection algorithm, gradient vector flow (GVF) of a binary edge map is used as the external force of active snake model, which can enlarge the convergence range. Experimental results show that the prior shape model can improve the robustness against the weak chin edges and partial occlusion. After facial feature extraction, a heuristic face modeling scheme is developed using the detected facial features and facial muscular distribution. The experimental results show that *a priori* knowledge of human face can improve the accuracy of 2D model design and motion representation.

- New algorithms for scalable shape coding are presented in Chapter 5, which include scalable shape representation, scalable intra-shape coding and scalable predictive shape coding. In both shape representation and coding, curvature scale space (CSS) image is employed to detect the salient feature of object contour and to estimate the contour motion. For scalable shape representation, the proposed algorithm can achieve up to 20-80% of the total number of vertices for lossless reconstruction of test video objects when compared with the state-of-the-art methods [GERKIN-1997] [JORDAN-1998] [MELNIKOV-2000b]. For scalable intra shape coding, the proposed coder exhibits excellent compression performance for both lossy and lossless shape coding as compared to both state-of-the-art vertex-based shape coding algorithms and CAE algorithm in MPEG-4. For example, the proposed intra-shape coding scheme can provide 25-60% gain in bit rate over the scalable encoding method in [JORDAN-1998], and it can achieve 5-10% gain over conventional non-scalable vertex-based coding [CONNELL-1997] in bit rate. For scalable predictive shape coding, motion compensation in the coarser layers and intra-coding for the finest layer can improve the compression performance further. The reasons for the success of our proposed scalable shape coding algorithms are:

    1. The intrinsic image grid quantisation is taken into account during the contour approximation, which can reduce up to 30-80% the number of approximating vertices for lossless representation.

2. CSS image is used to detect the salient feature of object contour and used to match the contours during motion estimation;

3. During shape coding, the information of the encoded coarser layers are employed to encode the vertices of the current layer.

- An improved shape-adaptive SPECK algorithm is proposed in Chapter 6. SPECK algorithm is an efficient wavelet-based image coding algorithm and it has been extended successfully for arbitrarily shaped texture coding [LU-2001]. In our research, the improvement of shape adaptive SPECK algorithm focuses on the two aspects: aggressive discarding of transparent regions and employing CABAC coder to compress the significance map, refinement information, and sign information. An improvement over the state-of-the-art algorithms in the literature, such as SA-SPIHT and the original SA-SPECK, is exhibited in extensive simulation results.

- A new highly scalable 2D model-based texture coding scheme is presented in Chapter 7. We demonstrate in the experimental results that a variety of coding bit rates and temporal resolution can be decoded from a single compressed file using the new scalable coding method. It is demonstrated that no blocking artefact is encountered even with very low bit rate coding as the warping motion compensation, together with the scalable object model, is employed. It is further demonstrated that scalable motion vector coding can achieve better compression performance at very low bit rates. The proposed scheme can achieve high scalability without a significant loss in compression when compared with H. 264. Its performance is superior to MPEG-4 objectively and subjectively, in both frame-based and object-based video coding.

  The designed scalable 2D model-based texture coding scheme possesses the following desirable properties:

  1. Free from DCT blocking artefacts due to the warping motion compensation and wavelet analysis. The decoded texture does not show the annoying blocking artefacts of DCT coding, even at very low bit rates.

  2. Error resilience: Error propagation in the proposed scheme is limited by the length of the temporal synthesis filters. This is the advantage of employing lifting-based temporal filtering technique.

3. Excellent compression efficiency: The redundancy in the source video is efficiently reduced by temporal filtering with warping motion compensation of video objects. The representation of video frames into video objects also improves the motion compensation. The subband correlation can be effectively exploited through the improved shape-adaptive SPECK algorithm. The experimental results show that the proposed coding system outperforms the nonscalable standard MPEG-4 coder over a wide range of bitrates in PSNR performance. Its performance is also comparable to H.264 at very low bit rates (<10kbits/s).

4. Flexible and highly scalable bit streams: The proposed scalable texture coding system can accommodate a wide variety of scalable functionalities utilising the multi-resolution nature innate in temporal and spatial subband filtering. Most importantly, these desirable scalable features are provided without a significant performance loss when compared with H.264, which is commonly seen in traditional hybrid coding for scalable applications.

8.2 Future Research Directions

This section describes some possibilities for further development and research, and outlines the author's view of the future development of scalable model-based video coding. Some suggestions for future work are given below:

- The more immediate research activities can be carried out with the video segmentation described in Chapter 3. Here, a novel video segmentation approach is proposed, which is based on a complexity-scalable contour tracking scheme. Intensive experiments have been conducted to prove the efficiency of this approach. However, video segmentation technique is still immature. Future research can be focused on:

    1. In order to reduce the human interaction for the initial object contour and improve the segmentation efficiency, more sophisticated mathematical methods, such as Graph cut [BOYKOV-2001], Markov random field [GEMAN-1984], and Level sets algorithm [PARAG-2000], can be investigated and employed.

2. For the contour tracking of the subsequent frames, particle filtering, Level sets and Bayesian network are the possible techniques to incorporate into the proposed algorithm and thus improve the accuracy and robustness of contour tracking. Furthermore, multiple object segmentation and tracking is also one of the interesting topics for future research.

3. Future research is necessary for performance evaluation, both objectively and subjectively, of video segmentation. Although some research has been conducted [WOLLBORN-1998] [CORREIA-2003], a satisfying solution is not yet available in the literature.

These research topics have been included as part of our current research tasks in EU VISNET project (www.visnet-noe.org).

- Chapter 4 discusses the issues regarding face detection and scalable modelling. This work can be extended to 3D domain. The main future research will be head modelling from a single video sequence, as well as the 3D head model transmission. One idea is that face model is constructed from the first several frames, which are encoded using traditional object-based video coding techniques. Then the subsequent frames are encoded using the 3D model-based video coding technique. To achieve this, for example, shape-adaptive face texture and shape information from the first several frames is encoded and transmitted to the decoder. Then, content-adaptive 3D face model is constructed by both encoder and decoder. The constructed 3D face model is used to compensate the face motion in the subsequent frames. In this way, only 2D shape and texture need to be sent. Furthermore, the use of content-adaptive 3D face model can avoid the computationally complex step of 3D face model adaptation.

- Chapter 5 examines the issues related to scalable shape coding and scalable model compression. For shape coding, the generated bit stream is very sensitive to the channel error during coding. Although scalable shape coding can improve its robustness against channel error through unequal error protection (UEP), the investigation of error concealment techniques is necessary. Currently, some research has been conducted for intra error concealment of shape information [SHIRANI-2000] [SOARES-2004]. However, it is possible to use the correctly decoded object shape in the previous frames to cancel the shape error in the current frame, that is, to employ temporal information for error concealment. Not much research has been conducted in this topic. Furthermore, for scalable shape coding, the correctly decoded coarser layers can also be employed to conceal the error in the current layer.

- Chapter 6 mainly investigates the scalable intra texture coding of arbitrarily shaped video objects. An improved shape-adaptive SPECK algorithm is proposed for high compression efficiency. Experimental results verify the improvement. However, as demonstrated in Figure 6.6, a variety of statistical dependencies can be observed in the dual pyramidal structure established from quadtree representations of a decomposed image. The context modelling scheme presented in this work (Section 6.6) for exploitation of such statistical redundancies is primarily based on some local texture features. It is expected that improved compression can be achieved further by more sophisticated context modelling strategies, e.g., advanced context selection and quantisation methods in [WU-1997]. Furthermore, both "I-frame" and the residual frames are currently encoded using the same context model. Further research can improve the coding performance by introducing different context models for different temporal frames.

- Scalable 2D model-based texture coding is discussed in Chapter 7. At the current stage, dyadic subband decomposition and a fixed GOP size have been employed during temporal filtering. Such a subband structure may not be efficient for image sequence with low temporal correlation. Further improvement of encoding performance can be made by adopting adaptive subband decomposition structure and variable GOP size. Moreover, further research can be conducted for joint source-channel coding related to model-based source codec. Very little work has been performed that looks at the joint source-channel coding related to model-based video codec.

Appendix A

List of Publications and Patents

Journal Papers:

1. M. Hu, A. H. Sadka, A. M. Kondoz, "Scalable shape coding of video objects", *submitted to IEEE Transactions on Image Processing,* June 2005
2. M. Hu, A. H. Sadka, S. T. Worrall, A. M. Kondoz, "Highly scalable 2D model-based video coding scheme using temporal filtering", *submitted to IEEE Trans. on Multimedia,* April 2005
3. M. Hu, S. T. Worrall, A.H. Sadka and A.M. Kondoz, "Automatic scalable face model design for 2-D model-based video coding", *Elsevier Signal Processing: Image communication*, Vol.19, no.5, May 2004, pp. 421-436
4. M. Hu, S. T. Worrall, A.H. Sadka and A.M. Kondoz, "Model design for scalable two-dimensional model-based video coding", *IEE Electronics Letters*, Vol.38, No.24, Nov. 2002, pp.1513-1515

Conference Papers:

5. M. Hu, A. H. Sadka, S. T. Worrall, A. M. Kondoz, "A scalable predictive shape coding scheme for video object", *In Proceedings of IEEE International Conference on Multimedia & Expo(ICME-2005), Amsterdam, Netherlands,* 6-8 July 2005
6. M. Hu, S. T. Worrall, A. H. Sadka, A. M. Kondoz, "Highly scalable 2D model-based video coding scheme using warping motion compensated temporal filtering", *In Proceedings of International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS-2005),* Montreux/Switzerland, April 13-15, 2005
7. M. Hu, S. T. Worrall, A. H. Sadka, A. M. Kondoz, "An efficient scalable object contour tracking scheme and its application for video segmentation", *In Proceeding of IEEE Workshop of multimedia signal processing*, Siena, Italy, Sept. 2004, pp. 155-158

8. M. Hu, S. T. Worrall, A. H. Sadka, A. M. Kondoz, "A scalable vertex-based shape intra-coding scheme for video objects" *In Proceedings of IEEE ICASSP-2004*, Vol. 3, Montréal, Canada, May 2004, pp. 273-276

9. M. Hu, S. T. Worrall, A. H. Sadka, A. M. Kondoz, "A fast and efficient chin detection method for 2-D scalable face model design", *In Proceedings of IEE International Conference on Visual Information Engineering*, Guildford, UK, 7-9, July 2003, pp. 121-124

10. M. Hu, S. T. Worrall, A. H. Sadka, A. M. Kondoz, "Face feature detection and model design for 2-D model-based video coding", *In Proceedings of IEE International Conference on Visual Information Engineering*, Guildford, UK, 7-9, July 2003, pp. 125-128

11. M. Hu, S. T. Worrall, A. H. Sadka, A. M. Kondoz, "A Scalable and Low Bit-Rate Video Codec Based on Two-Dimensional Mesh Motion Compensation", *Workshop on Visual Media Standards for Today and Tomorrow*, 25 April 2002, Savoy Place, London, UK

Patents:

12. "A scalable predictive shape-coding scheme for video objects", filed in May 2004, UK Patent No. 0410191.1

13. "Highly scalable object texture coding based on MC-3DSBC", filed in May 2004, UK Patent No. 0410190.3

Appendix B

B.1 Shape-adaptive discrete wavelet transform

Assuming odd symmetric biorthogonal wavelet filters with L_g (odd) taps for the low-pass filter and L_h (odd) taps for the high-pass filter, let the low-pass analysis filter taps be $g(i), i = 0, \cdots, L_g - 1$ and the high-pass analysis filter taps be $h(i), i = 0, \cdots, L_h - 1$. They have the following properties:

$$g(i) = g(L_g - 1 - i), \textit{ for } \ i = 0, \cdots, (L_g - 1)/2 \qquad \text{(B\_1.1)}$$

$$h(i) = h(L_h - 1 - i), \textit{ for } \ i = 0, \cdots, (L_h - 1)/2 \qquad \text{(B\_1.2)}$$

Let the low-pass synthesis filter be $\overline{g}(i)$, then

$$\overline{g}(i) = (-1)^{i+1} h(i), \textit{for } \ i = 0, \cdots, L_h - 1 \qquad \text{(B\_1.3)}$$

Let the high-pass synthesis filter be $\overline{h}(i)$, then

$$\overline{h}(i) = (-1)^{i} g(i), \textit{for } \ i = 0, \cdots, L_g - 1 \qquad \text{(B\_1.4)}$$

The analysis filtering process is given by:

$$T(i) = \sum_{j=0}^{L_g - 1} x(i + j - (L_g - 1)/2) g(L_g - 1 - j) \text{ (Low pass)} \qquad \text{(B\_1.5)}$$

$$S(i) = \sum_{j=0}^{L_g - 1} x(i + j - (L_h - 1)/2) h(L_h - 1 - j) \text{ (High pass)} \qquad \text{(B\_1.6)}$$

where $T(i)$ and $S(i)$ are the low-pass and high-pass band filter outputs before subsampling, respectively.

The wavelet coefficients from the analysis are obtained by subsampling the above filtering results by a factor of two. Subsampling can be at either even position or odd positions. However, in order to use the symmetric extensions, the subsampling of low-pass coefficients and that of high-pass coefficients always have one sample shift. If the subsampling positions of low-pass coefficients are even, then the sub-sampling positions of high-pass coefficient should be odd, or vice versa.

The subsampling process is described as follows:

$$C(i) = T(2i - s) \tag{B\_1.7}$$

$$D(i) = S(2i + 1 - s) \tag{B\_1.8}$$

The subsampling of high-pass coefficients always has one sample advance.

To perform synthesis, these coefficients are first upsampled by a factor of two. The upsampling process is given as follow:

$$P(2i - s) = C(i); \qquad P(2i + 1 - s) = 0; \tag{B\_1.9}$$

$$Q(2i + 1 - s) = D(i); \quad Q(2i + s) = 0; \tag{B\_1.10}$$

where $P(k)$ and $Q(k)$ are upsampled low-pass and high-pass coefficients, respectively. Then the synthesis filtering process is given as follows:

$$u(i) = \sum_{j=0}^{L_g - 1} P\left(i + j - (L_h - 1)/2\right)\overline{g}(L_h - 1 - j) \text{ (Low pass)} \tag{B\_1.11}$$

$$v(i) = \sum_{j=0}^{L_g - 1} Q\left(i + j - (L_g - 1)/2\right)\overline{h}(L_g - 1 - j) \text{ (High pass)} \tag{B\_1.12}$$

$$r(i) = u(i) + v(i) \tag{B\_1.13}$$

where $r(i)$ is the reconstructed signal.

B.2 SA-SPIHT Algorithm

The SA-SPIHT algorithm can be summarised as follows:

H = roots of the spatial orientation trees
$O(i,j)$ = offsprings of pixel (i,j).
$D(i,j)$ = descendants of pixel (i,j).
$L(i,j) = D(i,j) - O(i,j)$.
$S_n(T)$ = significance of set T w-r-t n, 1 means significant, 0 means insignificant.

```
1. Initialisation
     • output $n = \lfloor \log_2(\max_{(i,j)}\{|c_{i,j}|\}) \rfloor$, where $c_{i,j}$ is the wavelet coefficient of point $(i,j)$
     • set $LSP = \phi$
     • set $LIP = (i,j) \in H$ if $(i,j)$ is within the object
     • set $LIS = (i,j) \in H$ if $(i,j)$ is within the object and with descendants

2. Sorting Pass
     (a) for each $(i,j) \in LIP$,
          • output $S_n(i,j)$
          • if $S_n(i,j) = 1$, move $(i,j)$ to $LSP$ and output the sign of $c_{i,j}$

     (b) for each $(i,j) \in LIS$, if $(i,j) \in Type\ A$
          • output $S_n(D(i,j))$
          • if $S_n(D(i,j)) = 1$
               – for each $(k,l) \in O(i,j)$ and $(k,l) \in shape\ mask$
                    o output $S_n(k,l)$
                    o if $S_n(k,l) = 1$, add $(k,l)$ to $LSP$ and output its sign
                    o if $S_n(k,l) = 0$, add $(k,l)$ to $LIP$
               – if $L(i,j) = \phi$, remove $(i,j)$ from $LIS$, otherwise change $(i,j)$ to $Type\ B$

          • if $(i,j) \in Type\ B$ and $(i,j) \in shape\ mask$
               – output $S_n(L(i,j))$;
```

- if $S_n(L(i,j))=1$, add each $(k,l)\in O(i,j)$ to *LIS* as *Type A* entry, remove (i,j) from *LIS*

3. Refinement Pass

   For each $(i,j)\in LSP$, except hose included in the latest sorting pass, output the n th MSB of $|c_{i,j}|$.

4. Quantisation Update Step

   Decrement n by 1 and go back to step 2.

B.3 SA-SPECK Algorithm

The SA-SPECK algorithm can be summarised as follow:

```
1. Initialisation
    • partition image transform $X$ into two sets: $S \equiv root$ and $I \equiv X - S$ (see Figure 6.8 (a) )
    • output $n = \left\lfloor \log_2\left( \max_{\forall(x,y)\in X} |c_{i,j}| \right) \right\rfloor$
    • add $S$ to $LIS$
    • set $LSP = \phi$
    • set $\max lengthinLIS = length(S)$

2. Sorting Pass
    • for $l = 1$ to $l = max\,lengthLIS$
        – for each $S \in LIS$, with $length(S) = l$ and $S \in shape\ \ mask$,
            o $\Pr ocessS(S)$
    • $\Pr ocessI(\ )$

3. Refinement Pass
    • for each $(i, j) \in LSP$, except those included in the last sorting pass, output the $n\,th$ MSB of $|c_{i,j}|$

4. Quantisation Step Update
   Decrement $n$ by 1 and go back to step 2.
```

The functions $\Pr ocessS(S)$, $CodeS(S)$, $\Pr ocessI(\ )$ and $CodeI(\ )$ are presented below:

```
Function $\Pr ocessS(S)$:
    • output $S_n(S)$
    • if $S_n(S) = 1$
        – if $S$ is a pixel
            o output sign of $S$
            o add $S$ to LSP
        – else
            o $CodeS(S)$
    – if $S \in LIS$
        o remove $S$ from $LIS$
• else
    – if $S \notin LIS$, add $S$ to $LIS$

Function $CodeS(S)$:
• partition $S$ into four equal subsets $O(S)$ (see Figure 6.8 (b))
• for each $O(S) \in shape\ mask$
    – output $S_n(O(S))$
    – if $S_n(O(S)) = 1$
        o if $O(S)$ is a pixel
            ✓ output sign of $O(S)$
            ✓ add $O(S)$ to $LSP$
        o else
            ✓ $CodeS(O(S))$
    – else
        o add $O(S)$ to $LIS$
Function $ProcessI(\ )$:
• output $S_n(I)$
• if $S_n(I) = 1$
    – $CodeI(\ )$
Function $CodeI(\ )$:
• partition $I$ into four sets: three $S$ and one $I$ (see Figure 6.8 (c))
• for each of three sets $S$
    – if $length(S) > \max lengthinLIS$
        o $\max lengthinLIS = length(S)$
    – $ProcessS(S)$
• $ProcessI(\ )$
```

B.4 Context tables of improved shape adaptive SPECK algorithm

Table B_4.1 – Look-up table for significant coding of each quadtree node

LL, LH, and HL bands						HH bands					
Label	P	H	V	HV	D	*Label*	P	H	V	HV	D
0	0	0	0	0	x	0	0	0	0	0	x
0	0	x	x	1	<3	0	0	x	x	1	<3
1	0	x	x	1	≥ 3	1	0	x	x	1	≥ 3
1	0	x	x	2	x	1	0	x	x	2	x
2	0	x	x	>2	x	2	0	x	x	>2	x
3	1	0	0	0	1	3	1	0	0	0	1
4	1	0	0	0	>1	4	1	0	0	0	>1
4	1	0	1	1	x	4	1	x	x	1	<3
5	1	1	0	1	x	4	1	1	1	2	<2
6	1	x	x	2	x	5	1	x	x	1	≥ 3
7	1	x	x	>2	x	5	1	0	2	2	x
						5	1	2	0	2	x
						5	1	1	1	2	≥ 2
						6	1	x	x	>2	x

Table B_4.2 – Contribution from the horizontal neighbours

W	E	h
Significant, +	Significant, +	1
Significant, -	Significant, +	0
Insignificant	Significant, +	1
Significant, +	Significant, -	0
Significant, -	Significant, -	-1
Insignificant	Significant, +	-1
Significant, +	Insignificant	1
Significant, -	Insignificant	-1
Insignificant	Insignificant	0

Table B_4.3 – Look-up table for sign coding

h	v	d_{45}	d_{135}	$\hat{\chi}$	*Label*
1	1	x	x	1	4
1	0	x	x	1	3
1	-1	x	x	1	2
0	1	x	x	-1	1
0	0	0	0	1	0
0	-1	x	x	1	1
-1	1	x	x	-1	2
-1	0	x	x	-1	3
-1	-1	x	x	-1	4

Table B_4.4 – Look-up table for context selection in refinement coding of significant coefficients

LL, LH, and HL bands		HH bands	
Label	HV^p	*Label*	HV^p
0	0	0	0
1	<3	1	>0
2	≥ 3		

Appendix C

List of Abbreviations

2-D	Two-dimensional
3-D	Three-dimensional
AAC	Adaptive Arithmetic Coding
AROS	Arbitrary Region-Of-Support
AVO's	Audio-Visual Objects
B-VOPs	Bi-directional VOPs
bpp	bit per pixel
BTBC	Background To Be Covered
BV	Boundary Vertex
CABAC	Context-Adaptive Binary Arithmetic Codec
CAE	Context-based Arithmetic Encoding
CIF	Common Intermediate Format
CODWT	Complete to Overcomplete Discrete Wavelet Transform
CPs	Control Points
CSS	Curvature Scale Space
DCT	Discrete Cosine Transform
DFD	Displaced Frame Difference
DWT	Discrete Wavelet Transform
EBCOT	Embedded Block Coding with Optimised Truncation
EM	Expectation Maximization
EOL	End-of-layer
EZBC	Embedded Zero Block Coding

EZW	Embedded Zerotree Wavelet
FAP's	Facial Animation Parameters
FB	Filter Banks
FEC	Forward Error Correction
FGS	Fine-Granularity Scalability
FIR	Finite Impulse Response
FLD	Fisher Linear Discriminant
fps	frame per second
GMM	Gaussian Mixture Models
GOP	Group of Pictures
GPSC	Generalised Predictive Shape Coding
GVF	Gradient Vector Flow
IBMCTF	In-band MCTF
ITU	International Telecommunication Union
JBIG	Joint Bi-level Image experts Group
JPEG	Joint Photographic Experts Group
Kbps	Kilobits per second
LIP	List of Insignificant Pixels
LIS	List of Insignificant Sets
LMS	Least Mean Square
LSP	List of Significant Pixels
MAP	Maximization of A posteriori Probability
Mbps	Megabits per second
MC	Motion Compensation
MC-EZBC	Motion Compensated Embedded Zero Block Coding
MCTF	Motion Compensated Temporal Filtering
ME	Motion Estimation
MF	Motion-Failure regions

ML	Maximal Likelihood
MORF	Morphological Open by Reconstruction Filter
MPEG	Motion Picture Experts Group
MR	Magnitude Refinement
MRF	Markov Random Field
MSB	Most Significant Bit
MTh	Motion threading
MV	Motion Vector
NN	Neural networks
OAVE	Object Adaptive Vertex Encoding
OBMC	Overlapped Block Motion Compensation
OTS	Object-based Temporal Scalability
PCA	Principal Component Analysis
PCRD	Post-Compression Rate-Distortion
PR	Perfect Reconstruction
PSNR	Peak-to-peak Signal to Noise Ratio
QCIF	Quarter Common Intermediate Format
QP	Quantisation Parameter
RLC	Run-Length Coding/Coder
ROI	Region-of-interest
SA-DCT	Shape-adaptive DCT
SA-DWT	Shape-adaptive Discrete Wavelet Transform
SA-EBCOT	Shape-adaptive EBCOT algorithm
SA-SPECK	Shape-adaptive SPECK algorithm
SA-SPIHT	Shape-adaptive SPIHT algorithm
SC	Sign Coding
SIF	Storage Intermediate Format
SNR	Signal to Noise Ratio

SOT	Spatial Orientation Tree
SPECK	Set Partitioning Embedded BlocK
SPIHT	Set Partitioning in Hierarchical Trees
SVC	Scalable Video Coding
SVM	Support Vector Machine
UB	Uncovered Background
UEP	Unequal error protection
UMA	Universal Multimedia Access
VLC	Variable-Length Coding/Coder
VLD	Variable-Length Decoding/Decoder
VO	Video Object
VOL	Video Object Layer
VOP	Video Object Plane
WLS	Weighted Least Squares
ZC	Zero Coding

Bibliography

[AACH-1993a] – Aach T., Kaup A., Mester R., "Statistical model-based change detection in moving video", Signal processing, Vol. 31, pp. 165-180, 1993

[AACH-1993b] – Aach T., Kaup A., Mester R., "Change detection in image sequences using Gibbs random fields", *IEEE Internal. Works. Intell. Signal Processing Com. Sys.*, Sendai, Japan, pp.56-61, Oct. 1993

[AIZA-1989] – Aizawa K., Harashima H., Saito T., "Model-based analysis synthesis image coding for a person's face", Image communication, ", Vol.1, No.2, pp.139-152, 1989

[AIZA-1995] – Aizawa K., Huang, T. S., "Model-based image coding: advanced video coding techniques for very low bit-rate applications", *Proceedings of the IEEE*, Vol.83, no.2, pp.259-271, Feb.1995

[ALTUNB-1997] – Altunbasak Y., and Tekalp A. M., "Occlusion-adaptive, content-based mesh design and forward tracking", IEEE Trans. IP, Vol.6, no.9, pp.1270-1280, Sept. 1997

[AMINI-1990] – Amini A.A., Weymouth T. E., and Jain R. C., "Using dynamic programming for solving variational problems in vision", IEEE Trans. on PAMI, Vol.12, no.5, pp.867-885, 1990

[ANDREO-2002] – Andreopoulos Y., Munteanu A., Auwera G. V. der, Schelkens P., and Cornelis J. and Schelkens P., "In-band motion compensated temporal filtering", *Signal Processing: Image Communication* (special issue on "Subband/Wavelet Interframe Video Coding"), vol.19, no.7, pp.653-673, Aug. 2004

[ANSARI-1991] – Ansari N., E. J. Delp, "On detecting dominant points", Pattern Recognition, Vol.24,no.5, May 1991, pp.441-451

[ANTONINI-1992] – Antonini M., Barlaud M., Mathieu P. and Daubechies I., "Image coding using wavelet transform," IEEE Trans. Image Processing, vol.1, pp.205-220, Apr. 1992

[ASBUN-2000] – Asbun E., Salama P., Delp E. J., "A rate-distortion approach to wavelet-based encoding of predictive error frames", in Proc ICIP-2000, Vol.3, pp.150-153, Sept. 2000

[ASCENSO-2004] – Ascenso J., Tagliasacchi M., Tubaro S., Pereira F., "MPEG-21 Scalable Video Coding: From Applications to Technologies", Report for EU VISNET Project, June 2004

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

[BARNARD-1993] – Barnard H. J., Weber J. H., and Biemond J., "Efficient signal extension for subband/wavelet decomposition of arbitrary length signals", SPIE Vol.2094 Visual Communications and Image Processing, pp.966-975, Nov. 1993

[BARNARD-1994] – Barnard H. J., Weber J. H., Biemond J., "A region-based discrete wavelet transform for image coding", Chapter in *Wavelet in Image Communications*, Ed. M. Barlaud, Elsevier Science B. V., 1994

[BAUD-2004] – Baud G., Duvanel M., Reichel J., Ziliani F., "Low latency video codec using in-band prediction and block arithmetic", ISO/IEC JTC1/SC29/WG11 M10569/S20, Munich MPEG meeting, Germany, March 2004

[BEEK-1999] – Beek P. van, Tekalp A. M., Ning Zhuang, Celasum I., Minghui Xia, "Hierarchical 2-D mesh representation, tracking, and compression for object-based video", IEEE Trans. on CSVT, Vol.9, no.2, pp.353-369, 1999

[BENTLEY-1975] – Bentley J. L., "Multidimensional binary search trees used for associative searching", Comm. ACM, Vol.18, no.9, pp.509-517, Sept. 1975

[BERG-1997] – Berg M. de, Kreveld M. van, Overmars M., Schwarzkopf O.: *Computational Geometry: algorithms and applications*, Springer-Verlag, Berlin, 1997

[BEUC-1993] – Beucher S. and Meyer F., "The morphological approach to segmentation: the Watershed transformation", in *Mathematical Morphology in Image Processing*, Edited by E. Dougherty, M. Dekker, 1993

[BORMANS-2003] – J. Bormans, J. Gelissen, A. Perkis, "MPEG-21: The 21st century multimedia framework", IEEE Signal Processing Magazine, Vol.20, no.2, March 2003, pp. 53-62

[BOTTREAU-2001] – Bottreau V., Benetiere M., Felts B., and Pesquet-popescu B., "A fully scalable 3D subband video codec", in Proc. ICIP-01, Vol.2, pp.1017-1020, 2001

[BOUT-1993] – Bouthemy P., and Francois E., "Motion segmentation and qualitative scene analysis from an image sequence", International Journal of Computer Vision, Vol.10, no.2, pp.157-182, 1997

[BOYKOV-2001] – Boykov Yuri Y., Jolly Marie-Pierre, "Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images", Proc. IEEE int'l Conf. CV, Vancouver, Canada, pp.105-112, July 2001

[BOZD-1994] – Bozdagi G., Tekalp A. M., Onural L., "3-D motion estimation and wireframe adaptation including photometric effects for model-based coding of facial image sequence", *IEEE Trans. CSVT*, Vol.4, no.3, pp.246-256, June 1994

[BRADY-1996] – Brady N. and O'Connor N. E., "Object detection and tracking using an EM-based motion estimation and segmentation framework", Proc. IEEE Int'l Conf. IP, pp.17A2, 1996

[BRADY-1997] – Brady N., Bossen F., and Murphy N., "Context-based arithmetic encoding of 2D shape sequences", in Special session on shape coding, ICIP'97, 1997

[CALDERBANK-1998] – Calderbank R., Daubechies I., Sweldens W., and Yeo B.-L, "Wavelet transforms that map integers to integers", Applied and Computational Harmonic Analysis, Vol.5, pp.332-369, July 1998

[CALLAGHAN-2005] – O'Callaghan Robert J., Bull David R., "Combined morphological-spectral unsupervised image segmentation", IEEE Trans. on image processing, Vol.14, no.1, pp. 49-62, Jan. 2005

[CANNY-1986] – Canny J., "A computational approach to edge detection", IEEE Transactions on PAMI, Vol.8, no.6, pp 679-698, Nov 1986

[CELASUM-2000] – Celasum I., Tekalp A. M., "Optimal 2-D hierarchical content-based mesh design and update for object-based video", IEEE Trans. on CSVT, Vol.10, no.7, pp.1135-1153, Oct. 2000

[CHAI-1999] – Chai D., Ngan K. N., "Face segmentation using skin-color map in videophone applications", IEEE Trans. CSVT, Vol.9, no.4, pp.551-564, June 1999

[CHAL-1996] – Chalom E., and Bove V., "Segmentation of an image sequence using multi-dimensional image attributes", in Proceedings of the IEEE international Conference on Image Processing, ICIP'96, Lausanne, pp.525-528, Sept. 1996

[CHAN-2001] – Chan T., Vese L., "Active contours without edges", IEEE Trans. on Image Processing, Vol.10, no.2, pp.266-277, Feb. 2001

[CHEN-2002] – Chen P., Woods J. W., "Improved MC-EZBC with quarter-pixel motion vector", JVT proposal, ISO/IEC JTC1/SC29/WG11, MPEG2002/M8366, Fairfax, VA, May 2002

[CHEN-2004] – Chen P., and Woods J. W., "Bidirectional MC-EZBC with lifting implementation", IEEE Trans. on CSVT, Vol.14, no.10, pp.1183-1194, Oct. 2004

[CHENG-1997] – Cheng P., Li J., and Kou J., "Rate control for embedded wavelet video coding", IEEE Trans. on CSVT, Vol.7, no.4, pp.696-702, Aug. 1997

[CHO-1996] – Cho Y. S., Lee S. H., Shin J. S., and Seo Y. S., "Shape coding tool: Using polygonal approximation and reliable error residue sampling method", in ISO/IEC JTC1/SC29/WG11 MPEG 95/565, Munich, Germany, Jan. 1996

[CHO-1999] – Cho Sung Ho, Kim R. C., Oh Seung S., and Lee Sang U., "A coding technique for the contours in smoothly perfect eight-connectivity based on two-stage motion compensation", IEEE Trans. On CSVT, Vol.9, no.1, pp.59-69, Feb. 1999

[CHOI-1994] – Choi C S., Aizawa K., Harashima H., Takebe T., "Analysis and synthesis of facial image sequence in model-based image coding", *IEEE Trans CSVT*, Vol.4, no.3, pp.257-274, June 1994

[CHOI-1997] – Choi J. G., Lee S., Kim S., "Spatial-temporal video segmentation using a joint similarity measure", IEEE Trans. on CSVT, Vol.7, no.2, pp.279-286, April 1997

[CHOI-1999] – Choi S. -j. and Woods J. W., "Motion-compensated 3D subband coding of video", IEEE Trans. on IP, Vol.8, no.2, pp.155-167, Feb. 1999

[CHOWD-1994] – Chowdhury M. F., Clark A. F., Downton A. C., and Pearson D. E., "A switched model-based coder for video signals", IEEE Trans. on CSVT, Vol.4, pp.216-227, June 1994

[CHUNG-2000] – Chung Jae-won, et.al., "A new vertex-based binary shape coder for high coding efficiency", Signal Processing: Image Commun., Vol.15, pp.665-684, 2000

[COLMEN-1997] – Colmenarez A. J., Huang T. S., "Face detection with information-based maximum discrimination", in Proc. CVPR, pp.782-787, 1997

[CONNELL-1996] – O'Connell K. J., "S4 vertex coding results and comparison to VM", ISO/IEC JTC1/SC29/WG11 MPEG96/1135, Tempere, July 1996

[CONNELL-1997] – O'Connell K. J., "Object-adaptive vertex-based shape coding method", ~~IEEE Trans. on CSVT, Vol.7, no.1, pp.251-255, Feb. 1997~~

[COOTES-1995] – Cootes T. F., Taylor C. J., Cooper D.M., and Graham J., "Active shape models-their training and applications", Computer Vision and Image Understanding, Vol.61, no.1, pp.38-59, 1995

[CORREIA-2003] – Correia P. L., Fernando Pereira, "Objective evaluation of video segmentation quality", IEEE Trans. on IP, Vol.12, no.2, pp.186-200, Feb. 2003

[CORT-1995] – Cortez D., Nunes P., Sequeira M., and Pereira F., "Image Segmentation towards new image presentation methods", Signal processing: Image communications, Vol.6, pp.485-498, 1995

[CRAW-1992] – Craw I., Tock D., and Bennett A., "Finding face features", in Proc. 2nd ECCV, pp.92-96, 1992

[CROWLEY-2000] – Crowley J., Coutaz J., and Brard F., "Things that see: Machine perception for human computer interaction", Commun. ACM, Vol.43, no.3, pp.54-64, 2000

[CZEREPINSKI-1997] – Czerepinski, P.J.; Bull, D.R, "Enhanced interframe coding based on morphological segmentation", IEE Proceedings of Vision, Image and Signal Processing, Vol.144, no.4, pp.220-226, Aug. 1997

[DAI-1996] – Dai Y. and Nakano Y., "Face texture model based on SGLD and its application in face detection in a colour scene", Pattern Recognition, Vol.29, no.6, pp.1007-1017, 1996

[DAUBECHIES-1998] – Daubechies I. and Sweldens W., "Factoring wavelet transforms into lifting steps", *Journal Fourier Analysis and Applications*, Vol.4, no.3, pp. 247-269, 1998

[DENG-1997] – Deng J. Y. and Lai Feipei, "Region-based template deformation and masking for eye feature extraction and description", Pattern Recognition, Vol.30, no.3, pp.403-419, 1997

[DIEH-1991] – Diehl N., "Object-oriented motion estimation and segmentation in image sequence", *Signal Processing: Image Communications*, Vol.3, no.1, pp.23-56, Jan.1991

[DUNHAM-1986] – Dunham J. G., "Optimum uniform piecewise linear approximation of planar curves", IEEE Trans. on PAMI, Vol.8, no.1, pp. 67-75, Jan. 1986

[ECKERT-1997] – Eckert M., Ronda J. V., Pacheco A., Garcia N., "Irregular triangle mesh based motion compensation for very low bit-rate coding supporting discontinuous motion fields", Proceedings of ICSIP'97, New Orleans Louisiana USA, pp.13-18, Dec 4-6 1997

[EGGER-1996] – Egger O., Ebrahimi T., and Kunt M., "Arbitrarily-shaped wavelet packets for zerotree coding", in Proc. ICASSP, Vol.4, pp.2335-2338, May 1996

[EISERT-2000] – Eisert Peter, Wiegand Thomas, Girod Bernd, "Model-aided coding: a new approach in incorporate facial animation into motion compensated video coding", IEEE Trans. CSVT, Vol.10, no.3, pp.344-358, April 2000

[ELEF-1994] – Eleftheriadis A., and Jacquin A., "Model-assisted coding of video teleconferencing sequences at low bit rates", IEEE ISCAS'94, Vol.3, pp.177-180, May 1994

[ERYU-1995] – Eryurtlu F., Kondoz A. M., Evans B. G., "Very low bit rate segmentation-based video coding using contour and texture prediction", *IEE Proc. VISP*, Vol.142, no.5, pp.253-261, 1995

[ETOH–1997] – Etoh M., Boon C. S., Kadono S., "Template-based video coding with opacity representation", IEEE Trans. on CSVT, Vol.7, no.2, pp.172-180, Feb. 1997

[FACE-2004a] – http://www.uk.research.att.com:pub/data/att_faces.zip

[FACE-2004b] – http://www.humanscan.de/support/downloads/facedb.php

[FORSYTH-2003] – Forsyth D.A., Ponce J., *Computer Vision: A Modern Approach*, Prentice Hall, New Jersey, USA, 2003

[FOWLER-2004] – Fowler J. E., "Shape-adaptive coding using binary set splitting with K-d trees", in Proceedings of the 2004 IEEE International Conference on Image Processing, Singapore, Oct. 2004

[FREEMAN-1961] – Freeman H., "On the encoding of arbitrary geometric configurations", IRE Trans. on Electronic Computing, EC-10, pp.260-268, June 1961

[FUKUN-1990] – Fukunaga K., *Introduction to statistical pattern recognition* (2nd edition) Academic Press, INC., New York, 1990

[GELGON-2000] – Gelgon M., Bouthemy P., "A region-level motion based graph representation and labelling for tracking a spatial image partition", Pattern Recognition, Vol.33, pp.725-740, April 2000

[GERKIN-1994] – Gerkin P., "Object oriented analysis-synthesis coding based on moving two-dimensional objects", *IEEE Trans. on CSVT*, Vol.4, no.3, pp.228-236, Sept. 1994

[GERKIN-1997] – Gerkin P., "Object-based analysis-synthesis coding of image sequences at very low bit rates", IEEE Trans. on CSVT, Vol. 7, no.2, pp.251-255, Feb. 1997

[GEMAN-1984] – Geman S., and Geman D., "Stochastic relaxation, gibbs distributions and the Bayesian restoration of images", *IEEE Trans. PAMI*, Vol.6, pp.721-741, 1984

[GNSEL-1998] – Gnsel B., Tekalp A. M., and Beek P. J. V., "Content-based access to video objects: Temporal segmentation, visual summarization and feature extraction", Signal Processing, Vol. 66, no. 2, pp.261-280, 1998

[GOKCE-2000] – Gokcetekin M., Celasun I., Tekalp A.M., "Mesh based segmentation and update for object based video", in Proc. ICIP, Vol.1, pp.10-13, Sept. 2000

[GOTO-2002] – Goto T., Lee W.S., and Nadia M.T., "Facial feature extraction for quick 3D face modeling", Signal Processing: Image Communication, Vol.17, pp.243-259, 2002

[GOVIND-1996] – Govindaraju V., "Locating human faces in photographs", International Journal of Computer Vision, Vol.19, no.2, pp.129-146, 1996

[GRAF-1995] – Graf, H. P. Chen T., Petajan E., and Cosatto E., "Locating faces and facial parts", *in Proceedings of Int'l Workshop Automatic Face and Gesture Recognition*, pp.41-46, 1995

[GU-1995] – Gu C. and Kunt M., "Contour simplification and motion compensated coding", signal processing: Image communications, Vol.7, pp.279-296, Nov. 1995

[GU-1998] – Gu C. and Lee M., "Semiautomatic segmentation and tracking of semantic video objects", IEEE Trans. on CSVT, Vol.8, no.5, pp.572-584, Sept. 1998

[H264-2003] – *ISO/IEC JTC1/SC29/WG11, ISO/IEC FDIS 14496-10 (AVC): Information Technology-Coding of Audio-Visual Objects-Part 10: Advanced Video Coding*, 2003

[HAN-2004] – Han W.-J., "Responses for call-for-proposals for scalable video coding", ISO/IEC JTC1/SC29/WG11 M10569/S17, Munich MPEG meeting, Germany, March 20004

[HARA-1992] – Haralick R. and Shapiro L., "Computer and Robot Vision", Vol.1 Addison-Wesley Publication Company, 1992

[HARA-1994] – Haralick R. and Shapiro L., "*Glossary of Computer Vision Terms*" in "Digital Image Processing Methods", Edited by E. Dougherty, Dekker, pp.415-467, 1994

[HEIS-2001] – Heising G., Marpe D., Cycon H. L., and Petukhov A. P., "Wavelet-based very low bit-rate video coding using image warping and overlapped block motion compensation", *IEE Proc. VISP*, Vol.48, no.2, April 2001

[HILL-2003] – Hill Paul R., Canagarajah C. Nishan, and Bull David R., "Image segmentation using a texture gradient based watershed transform", IEEE Trans. on Image Processing, Vol.12, no.12, pp.1618-1633, Dec. 2003

[HORO-1976] – Horowitz S., Pavlidis T., "Picture segmentation by a tree traversal algorithm", Journal of the association for computing machinery, Vol.23, no.2, pp.368-388, April 1976

[HOTT-1988] – Hotter M. and Thoma R., "Image segmentation based on object oriented mapping parameter estimation", *Signal Processing*, Vol.15, pp.315-348, 1988

[HOTT-1989] – Hotter M., Ostermann J., "Analysis synthesis coding based on planar rigid moving objects", In Workshop on 64 kbps Coding of Moving Video, Hanover, Germany, 1988

[HOTT-1990] – Hotter M., "Object-oriented analysis-synthesis coding based on moving two dimensional objects", *Image Commun.*, Vol.2, no.4, pp.409-428, Dec. 1990

[HSIANG-1999] – Hsiang S.-T., and Woods J. W., "Invertible three-dimensional analysis/synthesis system for video coding with half-pixel-accurate motion compensation ", in SPIE Conference on Visual Communications and Image Processing, Vol. 3653, (San Jose, CA), pp. 537-546, Jan. 1999

[HSIANG-2001] – Hsiang S.-T., "Embedded image coding using zeroblocks of subband/wavelet coefficients and context modelling", in Proc. 2001 IEEE Data Compression Conference, (Snowbird, Utah), Mar. 2001, pp. 83-92

[HSU-2001] – Hsu Rein-Lien, Mottaleb M. A., and Jain A. K., "Face detection in colour images", IEEE Trans. PAMI, Vol.24, no.5, pp.696-706, May 2001

[HU-2003] – Hu M., Worrall S., Sadka A. H., Kondoz A. M., "A fast and efficient chin detection method for 2-D scalable face model design", In Proceedings of IEE International Conference on Visual Information Engineering, Guildford, pp.121-124, UK, 2003

[HUANG-1994] – Huang C. L., and Hsu C. Y., "A new motion compensation method for image sequence coding using hierarchical grid interpolation", IEEE Trans. CSVT, Vol.4, no.1, pp.72-85, Jan. 1994

[ISARD-1998] – Isard M., and Blake A., "CONDENSATION – conditional density propagation for visual tracking", Int. J. Computer Vision, Vol.29, no.1, pp.5-28, 1998

[ISLAM-1999] – Islam A. and Pearlman W. A., "An embedded and efficient low-complexity hierarchical image coder," in *Proc. of SPIE (3653) on Visual Communications and Image Processing*, vol. 2, pp.294-305, 1999

[IZQUIER-1999] – Izquierdo E., Xiaohua Feng, "Modeling arbitrary objects based on geometric surface conformity", IEEE Trans. CSVT, Vol.9, no.2, pp.336-352, March 1999

[JAIN-1989] – Jain A. K., "*Fundamentals of Digital Image Processing*", Prentice-Hall, 1989

[JBIG-1993] – *ISO/IEC International Standard 11544, ISO/IEC/, 1TC1/SC29/WG9, also ITU-T Recommendation T.82, Progressive Bi-level Image Compression*, 1993

[JONG-2000] – Jong Il Kim, Bovik Alan C., and Evans Brian L., "Generalized prediction binary shape coding using polygon approximation", Signal processing: Image communications, Vol.15, pp.643-663, 2000

[JORDAN-1998] – Jordan C.L. Buhan, Ebrahimi T., and Kunt M., "Progressive content based shape compression for retrieval of binary images", Computer Vision and Image Understanding, Vol.71, No.2, pp.1126-1154, Aug. 1998

[JPEG-2000] – *ISO/IEC JTC1/SC29/WG1.FCD 15444-1: Information technology – JPEG2000 image coding system*, March 2000

[KAMP-1997a] – Kampmann M., "Estimation of the chin and cheek contours for precise face model adaptation", in: Proc. ICIP'97, pp.300-304, 1997

[KAMP-1997b] – Kampmann M., and Ostermann J., "Automatic adaptation of a face model in a layered coder with an object-based analysis-synthesis layer and a knowledge-based layer", Signal Processing: Image Communications, Vol.9, no.3, pp.201-220, March 1997

[KANEKO-1985] – Kaneko T., Okudaira M., "Encoding of arbitrary curves based on the chain code representation", IEEE Trans. on Communications, Vol.33, pp.697-707, July 1985

[KATATA-1997] – Katata H., Ito N., Aono T., and Kusao H., "Object wavelet transform for coding of arbitrarily shape image segments", IEEE Trans. on CSVT, Vol.7, pp.234-237, Feb. 1997

[KATSAGGELOS-1998] – Katsaggelos A. K., Kondi L., Meier F. W., Ostermann J., Schuster G. M., "MPEG-4 and rate distortion based shape coding techniques", Proc. IEEE, pp.1126-1154, June 1998

[KAUFF-1997] – Kauff P., Makai B., Rauthenberg S., Golz U., Lameillieure J. and Sikora T., "Functional coding of video using a shape-adaptive DCT algorithm ad an object-based motion prediction toolbox", IEEE Trans. on CSVT, Vol.7, no.1, pp.181-196, Feb. 1997

[KAWANAKA-1999] – Kawanaka A., and Algazi V. R., "Zerotree coding of wavelet coefficients for image data on arbitrarily shaped support", in Proc DCC, Snowbird, UT, pp.534, March 1999

[KIM-1999] – Kim M., Choi J., Kim D., Lee H., Lee M., Ahn C., and Ho Y., "A VOP generation tool: automatic segmentation of moving objects in image sequences based on spatial-temporal information", IEEE Trans. on CSVT, Vol.9, no.8, pp.1216-1226, Dec. 1999

[KIM-2000] – Kim J. I., Bovik A. C., and Evans B. L., "Generalized predictive binary shape coding using polygon approximation", Signal Processing: Image Communication, pp.643-663, July 2000

[KIM-2002] – Kim J., Fisher J W., Yezzi A., Certin M., and Willsky A. S., "Nonparametric methods for image segmentation using information theory and curve evolution", in Proc. ICIP, 2002

[KIMB-2000] – Kim Beong-Jo, Xiong Z., Pearlman W.A., "Low bit-rate scalable video coding with 3D set partitioning in Hierarchical trees (3D SPIHT)" IEEE Trans. CSVT, Vol.10, no.8, pp.1374-1387, Dec. 2000

[KIMJ-1998] – Kim Jong-Han; Lee Jae-Yong; Kang Eui-Sung; Ko Sung-Jea, "Region-based wavelet transform for image compression", IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing, Vol.45, no.8, pp.1137-1140, Aug. 1998

[KIMOTO-2004] – Kimoto T., Miyamoto Y., "Multi-resolution multi-compensated temporal filtering for 3D wavelet video coding", ISO/IEC JTC1/SC29/WG11 M10569/S09, Munich MPEG meeting, Germany, March, 2004

[KJELD-1996] – Kjeldsen R., and Kender J., "Finding skin in colour images", *in Proc. 2nd Int'l Conf. Automatic Face and Gesture Recognition*, pp.312-317, 1996

[KOCH-1993] – Koch R., "Dynamic 3-D scene analysis through synthesis feedback control", *IEEE Trans. PAMI*, Vol.15, no.6, pp.556-568, 1993

[KOTROP-1997] – Kotropoulos C., and Pitas I., "Rule-based face detection in frontal views", in Proceedings of ICASSP, Vol.4, pp.2537-2540, 1997

[KRUS-1996] – Kruse S., "Scene segmentation from dense displacement vector fields using randomized Hough Transform", Signal Processing: Image Communications, Vol.9, pp.29-41, 1996

[KRUS-1999] – Kruse S., Graffunder A., Askar S., "A new tracking system for semi-automatic video object segmentation", Proceedings of the workshop on image analysis for multimedia interactive services, WIAMIS'99, Berlin, Germany, May 1999

[KUO-2002] – Kuo C. J., Ruey-Song Huang, Tsang-Gang Lin, "3-D facial model estimation from single front-view facial image", IEEE Trans. CSVT, Vol.12, no.3, pp.183-192, 2002

[KWAK-1998] – Kwak J., Kim M., Jeon J., Lee M., Anh C., "User-assisted object segmentation by object boundary tracking with graphical user interface", DOC. ISO/IEC JTC1/SC29/WG11 MPEG98/M3936, Oct. 1998

[LANITIS-1995] – Lanitis A., Taylor C. J. and Cootes T. F., "An automatic face identification system using flexible appearance models", Image and Vision Computing, Vol.13, no.5, pp.393-401, 1995

[LEE-1999] – Lee S. H., Cho D. –S, Cho Y.-S, Jang E. S., Shin J.-S., and Seo Y. S., "Binary shape coding using baseline-based method", IEEE Trans. on CSVT, Vol.9, no.2, pp.44-58, Feb. 1999

[LEUNG-1995] – Leung T. K., Burl M. C., and Perona P., "Finding faces in cluttered scenes using random labelled graph matching", in Proc. 5th ICCV, pp.637-644, 1995

[LI-1993] – Li H., Roivainen P., and Forchheimer R., "3-D motion estimation in model-based facial image coding", *IEEE Trans. PAMI*, Vol.15, no.6, pp.545-555, June 1993

[LI-2001] – Li, W., "Overview of fine granularity scalability in MPEG-4 video standard", *IEEE Trans. on CSVT*, Vol.11, no.3, pp.301-317, March 2001

[LIU-2002] – Liu Z., Hua J., Xiong Z., and Castleman K., "Lossy-to-lossless ROI coding of chromosome images using modified SPIHT and EBCOT", in Proceedings of Int. Symp. Biomedical Imaging, Washington, DC, pp.317-320, July 2002

[LU-1991] – Lu C., J.G. Duham, "Highly efficient coding schemes for contour lines based on chain code representations", IEEE Trans. on Communications, Vol.39, pp.1511-1514, Oct. 1991

[LU-2001] – Lu Z. and Pearlman W. A., "Wavelet video coding of video object by object-based SPECK algorithm", in Proceedings of PCS, Seoul, Korea, pp 413-416, April 2001

[LU-2002] – Lu Z., and Pearlman W. A., "Motion Compensated Two-link Chain Coding for Binary Shape Sequence," *Visual Communications and Image Processing 2002*, Proceedings of SPIE, Vol. 4671, pp.356-362, Jan. 2002

[LUO-2001] – Luo L., Jin Li, Shipeng Li, Zhuang Z., and Zhang Y. -Q., "Motion compensated lifting wavelet and its application in video coding", IEEE International Conference on Multimedia and Expo (2001), Tokyo, Japan, Aug. 2001

[LUO-2003] – Luo L., Wu F., Li S., and Zhang Z., "Advanced lifting-based motion-threading (MTh) techniques for 3D wavelet video coding", Proc. of SPIE Vol. 5150, VCIP2003, Lugano, Switzerland, pp.707-718, July 2003

[MAIO-2000] – Maio D. and D. Maltoni, "Real-time Face Location on Gray-Scale Static Image", Pattern Recognition, Vol.33, no.9, pp.1525-1539, Sept. 2000

[MARC-1999a] – Marcotegui B., Correia P., Marques F., Mech R., Rosa R., Wollborn M., Zanoguera F., "VOGUE: The MomuSys Video Object Generator with User Environment" in Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS'99, Berlin, Germany, pp.25-28, 1999

[MARC-1999b] – Marcotegui B., Correia P., Marques F., Mech R., Rosa R., Wollborn M., Zanoguera F., "A video object generation tool allowing friendly user interaction" in Proceedings of IEEE International Conference on Image Processing, ICIP'99, Kobe, Japan, Oct 25-28, 1999

[MARPE-1999] – Marpe D., Cycon H., "Very low bit-rate video coding using wavelet-based techniques", IEEE Trans. on CSVT, Vol.9, no.1, pp.85-94, Feb. 1999

[MARPE-2003] – Marpe D., Schwarz H., Wiegand T., "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard", IEEE Trans. on CSVT, Vol.13, no.7, pp.620-636, July 2003

[MARQ-2000] – Marques F., Marcotegui B., Zanoguera F., Correia P., Mech R., Rosa R., Wollborn M., "Partition-based image representation as basis for user-assisted segmentation", " in Proceedings of IEEE International Conference on Image Processing, ICIP'00, Canada, Sept.10-13, pp.312-315, 2000

[MCKEN-1998] – McKenna S., Raja Y., and Gong S., "Tracking colour objects using adaptive mixture models", *Image and Vision Computing*, Vol.17, nos.3/4, pp.223-229, 1998

[MECH-1998] – Mech R., Wollborn M., "A noise robust method for 2-D shape estimation of moving objects in video sequence considering a moving camera", Signal Processing – Special Issue on "Video sequence segmentation for content-based processing and manipulation", Vol.66, no.2, Elsevier, pp.203-217, 1998

[MEIER-1998] – Meier T. and Ngan K., "Automatic segmentation of moving objects for video object plane generation", IEEE Trans. on CSVT, Vol.8, no.5, pp.525-538, 1998

[MELNIKOV-1999] – Melnikov G., Schuster G. M., and Katsaggelos A. K., "Inter mode vertex-based optimal shape coding," in Proc. ICASSP-1999, Vol.6, pp.3125-3128, 1999

[MELNIKOV-2000a] – Melnikov G. and Katsaggelos A. K., "A rate-distortion optimal scalable vertex based shape coding algorithm," in Proc. ICASSP-2000, Vol.4, Turkey, June 5-9, pp.1947-1950, 2000

[MELNIKOV-2000b] – Melnikov G., and Katsaggelos A. K., "Shape approximation through recursive scalable layer generation", In Proc. IEEE ICIP 2000, Vol.2, pp.915-918, Sept. 2000

[MEYE-1990] – Meyer F., Beucher S., "Morphological segmentation", Journal of Visual Communications and Image Representation, Vol.1, no1, pp.21-46, Sept. 1990

[MINAMI-2001] – Minami G., Xiong Z., Wang A., Mehrota S., "3-D wavelet coding of video with arbitrary regions of support", IEEE Trans. CSVT, Vol.11, no.9, pp.1063-1068, Sept. 2001

[MOES-2001] – Moeslund T., and Granum E., "A survey of computer vision-based human motion capture", Computer Vision and Image Understanding, Vol.81, no.3, pp.231-268, 2001

[MOKHT-1992] – Mokhtarian F., Mackworth Alan K., "A theory of multiscale, curvature-based shape representation for planar curves", IEEE Trans. on PAMI, Vol.14, no.8, pp.789-805, Aug. 1992

[MOKHT-2003] – Mokhtarian F and Bober M., *Curvature Scale Space Representation: Theory, Applications & MPEG-7 Standardization*, Kluwer Academic, ISBN: 1-4020-1233-0, 2003

[MOSC-1998] – Moscheni F., Bhatacherjee S., Kunt M., "Spatial-temporal segmentation based on region merging", IEEE Trans. on PAMI, Vol.20, no.9, pp.897-915, Sept. 1998

[MPEG4-2001] – *ISO/IEC 14496-2: 2001, "Information Technology – Coding of audio-visual objects- Part 2: Visual"*, 2001

[MPEG7-2002] – *ISO/IEC 15938-3:2002, "Information Technology – Multimedia Content description Interface- Part 3: Visual"*, 2002

[MURR-1987] – Murray D. H., and Buxton H., "Scene segmentation from visual motion using global optimization", IEEE Trans. on PAMI, Vol.9, no.2, pp.220-228, March 1987

[MUSM-1989] – Musmann H., Hotter M., and Ostermann J., "Object-oriented analysis-synthesis coding of moving images", Signal Processing: Image Communications, Vol.1, pp.117-138, 1989

[MUSM-1995] – Musmann H., "A layered coding system for very low bit rate video coding", Signal Processing: Image Communications, Vol. 7, No.4-6, pp.267-278, Nov. 1995

[NAKAYA-1994] – Nakaya Y., Harashima Hiroshi, "Motion compensation based on spatial transformation", IEEE Trans. on CSVT, Vol. 4, no.3, pp.339-356, June 1994

[OCON-1997] – O'Connor N. E., Brady N., and Marlow S., "Supervised image segmentation using EM-based estimation of mixture density parameters", Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS'97, Louvain-la-Neuve, Belgium, pp.27-32, June 1997

[ODOB-1996] – Odobez J. M., and Bouthemy P., "Separation of moving regions from background in an image sequence acquired with a mobile camera", in "Video Data Compression for Multimedia Computing", Edited by H. Li, S. Sun, and H. Derin, Kluwer Academic Publisher, 1996

[OHM-1994] – Ohm J. R., "Three-dimensional subband coding with motion compensation", IEEE Trans. IP, Vol.3, no.5, pp.572-588, Sept. 1994

[OHM-1996] – Ohm J. –R., "Motion grid interpolation with simple contour adaptation", in Proceedings of PCS'96, 1996

[OHM-1999] – Ohm J.-R., Makai B., "Feature-similarity retrieval of face images based on 2D model description", in: Proceedings of *WIAMIS'99*, Berlin, May 1999

[OHM-2005] – Ohm J. –R., "Advances in scalable video coding", Proceedings of the IEEE, Vol.93, no.1, pp.42-56, Jan. 2005

[OSUNA-1997] – Osuna E. Freund R. and Girosi F., "Training support vector machines: an application to face detection", in Proc. IEEE Conf. CVPR, pp.130-136, 1997

[PARAG-2000] – Paragios N., and Deriche R., "Geodesic active contours and level sets for the detection and tracking of moving objects", IEEE Trans. on PAMI, Vol.22, pp.266-280, March 2000

[PARAS-2001] – Patras I., Hendriks E. A., Lagendijk R. L., "Video segmentation by MAP labelling of watershed segments", IEEE Trans. PAMI, Vol.23, no.3, pp.326-332, March 2001

[PARK-2000] – Park H. W., and Kim H. S., "Motion estimation using low-band-shift method for wavelet-based moving picture coding", IEEE Trans. Image Processing, Vol.9, no.4, pp.577-587, April 2000

[PEAR-1995] – Pearson D. E., "Developments in model-based video coding", Proceedings of the IEEE, Vol.83, no.6, pp.892-906, June 1995

[PEREZ-2004] – Perez P., Vermaak J. and Blake A., "Data fusion for visual tracking with particles", Proc. of IEEE, Vol.92, no.3, pp.495-513, March 2004

[PEREZA-1987] – Perez A., Gonzalez Rafael C., "An iterative thresholding algorithm for image segmentation", IEEE Trans. on PAMI, Vol.9, no.6, pp.742-751, Nov. 1987

[PHUNG-2002] – Phung S. L.,"ECU face detection database," Edith Cowan University, School of Engineering and Mathematics, 2002, Available at: http://www.soem.ecu.edu.au/~sphung/face_detection/database/

[PRAT-1991] – Pratt W., *Digital Image Processing*, 2nd Edition, Wiley, 1991

[QIAN-1997] – Qian R. J., Sezan M. I., "Content-scalable shape representation and coding", Proc. ICIP97, pp.II-819-822, 1997

[RADHA-1999] – Radha H, Chen Y., Parthasarathy K., and Cohen R., "Scalable internet video using MPEG-4", Signal processing: Image communication, Vol.15, no.1-2, pp.95-126, Sept.1999

[RAJAGO-1998] – Ragagopalan A., Kumar K., Karlekar J., "Finding faces in photographs", in Proc. 6th ICCV, pp.640-645, 1998

[REDN-1984] – Redner R.A., Walker H.F., "Mixture densities, Maximum likelihood and the EM algorithm", SIAM Review, Vol.26, no.2, April 1984

[ROTHER-2004] – Rother C., and Kolmogorov V., and Blake A., "Interactive Foreground Extraction using Iterated Graph Cuts", Proc. ACM Siggraph, 2004

[ROUSS-1987] – Rousseeuw P. J., Leroy A. M., *Robust Regression and Outlier Detection*, John Wiley and Sons, New York, December 1987

[ROWLEY-1998] – Rowley H., Baluja S., and Kanade T., "Neural network-based face detection", IEEE Trans. on PAMI, Vol.20, no.1, pp.23-38, 1998

[RUDIA-1996] – Rudianto R. L., Ngan K.N., "Automatic 3D wireframe model fitting to frontal facial image in model-based video coding", in: Picture Coding Symposium (PCS'96), pp.585-588, 1996

[SABER-1998] – Saber A. and Tekalp A. M., "Frontal-view face detection and facial feature extraction using colour, shape and symmetry based cost functions", Pattern Recognition Letters, Vol.19, no.8, pp.669--680, 1998

[SAID-1996] – Said A. and Pearlman W. A., "A new, fast, and efficient image codec based on set partitioning in hierarchical trees", IEEE Trans. CSVT, Vol.6, no.3, pp.243-250, 1996

[SALE-1994] – Salembier P., Pardas M., "Hierarchical morphological segmentation for image sequence coding", IEEE Trans. on Image Processing, Vol.3, No.5, pp.639-651, Sept.1994

[SALE-1995] – Salembier P., Torres L., Meyer F., Gu C., "Region-based video coding using mathematical morphology", *Proceedings of the IEEE*, Vol.83, no.6, pp.843-857, June 1995

[SALE-1996] – Salembier P., Marques, Gasull A., "Coding of partition sequences", *Video Coding: The Second Generation Approach (L. Torres and M. Kunt, eds.)*, 1996

[SCHA-2000a] – Schaar van der M., Radha H., "A novel MPEG-4 based hybrid temporal-SNR scalability for Internet video", in Proceedings of IEEE Int. Conf. Image Processing, Vol.3, pp.348-351, Sept. 2000

[SCHA-2000b] – Schaar van der M., "All FGS temporal-SNR-Spatial scalability", Contribution to 54th MPEG meeting m640, La Baule France, Oct. 2000

[SCHA-2001] – Schaar van der M., Radha H., "A hybrid temporal-SNR fine-granular scalability for Internet video", *IEEE Trans. on CSVT*, Vol.11, no.3, pp.318-331, March 2001

[SCHA-2002] – Schaar van der M., Radha H., "Adaptive motion-compensation fine-granular-scalability (AMC-FGS) for wireless video", IEEE Trans. on CSVT, Vol.12, no.6, pp.360-371, June 2002

[SCHUSTER-1998] – Schuster G. M., Katsaggelos A. K., "An optimal polygonal boundary encoding scheme in the rate-distortion scene", IEEE Trans. on Image Processing, Vol.7, no.1, pp.13-26, Jan. 1998

[SCHWARZ-2004] – Schwarz H., Marpe D., and Wiegand T., "Scalable Extension of H.264/AVC", ISO/IEC JTC1/SC29/WG11, Doc. M10569/S03, Munich, Germany, Mar. 2004

[SCLAR-1998] – Sclaroff S., and Isidoro J., "Active blobs", Proc. ICCV, Jan. 1998

[SECKER-2001] - Secker A. and Taubman D., "Motion-compensated highly scalable video compression using an adaptive 3-D wavelet transform based on lifting", In proceeding of ICIP, 2001, Thessalonica, GR, Vol.2, pp.1029-1032, Oct. 2001

[SHAPIRO-1993] – Shapiro J., "Embedded image coding using zerotree of wavelet coefficients", IEEE Trans. on Signal Processing, Vol. 41, no. 12, pp.3445-3462, Dec. 1993

[SHEN-1999] – Shen K. and Delp E. J., "Wavelet Based Rate Scalable Video Compression," *IEEE Trans on CSVT*, Vol.9, no.1, pp.109-122, February 1999

[SHI-1994] – Shi J., and Tomasi C., "Good features to track", Proc. IEEE Int. Conf. CVPR, pp.593-600, 1994

[SHI-2000] – Shi J., Malik J., "Normalized cuts and image segmentation", IEEE Trans. on PAMI, Vol.22, no.8, pp.888-905, 2000

[SHIPENG-2000] – Shipeng Li, Weiping Li, "Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding", IEEE Trans. on CSVT, Vol.10, no.5, pp.725-743, Aug. 2000

[SHIRANI-2000] – Shirani S., Erol B., and Kossentini F., "A concealment method for shape information in MPEG-4 coded video sequences", IEEE Trans. on Multimedia, Vol.2, no.3, pp.185-190, Sept. 2000

[SIKORA-1995] – Sikora T., Makai B. "Shape-adaptive DCT for generic coding of video", IEEE Trans. CSVT, Vol.5, pp.59-62, Feb. 1995

[SMEU-1997] – Smeulders A., Olabarriaga S., Boomgaard R., Worring M., "Design considerations for interactive segmentation", Visual Information System 97, San Diego, USA, pp.5-12, 1997

[SMITH-1997] – Smith S.M., Brady J. M., "SUSAN-a new approach to low level image processing", Int. Journal of Computer Vision, Vol.23, no.1, pp.45-78, 1997

[SOARES-2004] – Soares L. D. and Pereira F., "Spatial shape error concealment for object-based image and video coding", IEEE Trans. on Image Processing, Vol.13, no.4, pp.586-599, April 2004

[SOBOTTKA-1996] – Sobottka K. and Pitas I., "Face localization and feature extraction based on shape and colour information", in: Proceedings of Int. Conf. IP (ICIP-96), pp.483-486, Sept. 1996

[SOBOTTKA-1998] – Sobottka K. and Pitas I., "A novel method for automatic face segmentation, Facial Feature Extraction and Tracking", Signal Processing: Image Communications, Vol.12, no.2, pp.263-281, Feb. 1998

[SONK-1993] – Sonka M., Hlavac V., and Boyle R., *Image Processing, Analysis and Machine Vision*, Chapman & Hall, 1993

[STAIB-1992] – Staib L. H., Duncan J. S., "Boundary finding with parametrically deformable models", IEEE Trans. on PAMI, Vol.14, no.1, pp.1061-1075, 1992

[STAU-1993] – Stauder J., "An illumination estimation method for 3-D object-based analysis and synthesis coding", In COST 211 European Workshops, 4.5.1-6, University of Hanover, Germany, Dec. 1993

[STAU-1995] – Stauder, J., "Estimation of point light source parameters for object-based coding", *Signal processing: Image Communications*, Vol.46, no.7, pp.355-379, 1995

[STAUFFER-1999] – Stauffer, C.; Grimson, W.E.L., "Adaptive background mixture models for real-time tracking", IEEE Conf. on CVPR, Vol. 2, pp.246-252, June 1999

[SUNG-1998] – Sung K., and Poggio T., "Example-based Learning for View-based Human Face Detection", IEEE Trans. PAMI, Vol.20, no.1, pp.39-51, Jan. 1998

[SWELDENS-1995] – Sweldens W., "The lifting scheme: a new philosophy in biorthogonal wavelet construction", in wavelet applications in signal and image processing III, Proc. SPIE 2569, 1995, pp.68-79

[TAUB-1994] – Taubman D., Zakhor A., "Multirate 3-D subband coding of video", IEEE Trans. on Image Processing, Vol.3, no.5, pp.572-589, Sept. 1994

[TAUB-2000] – Taubman D., "High performance scalable image compression with EBCOT," *IEEE Trans. on Image Processing*, Vol. 9, no.7, pp.1158-1170, July 2000

[TEKALP-1997] – Tekalp A.M., Altunbasak Y., and Bozdagi G., "Two- versus three-dimensional object-based video compression", IEEE Trans. CSVT, Vol.7, no.2, pp.391-397, Feb. 1997

[THAM-1998] – Tham J. Y., Ranganath S., and Kassim A. A., "Highly scalable wavelet-based video codec for very low bit rate environment", IEEE JSAC, Vol.16, no.1, pp.12-27, Jan. 1998

[THOMA-1989] – Thoma R., Bierling M., "Motion compensating interpolation considering covered and uncovered background" *Signal Processing: Image Communications*, Vol.1, pp.191-212, 1989

[TORRES-1996] – Torres Luis, and Kunt M., *Video coding: The second generation approach*, Kluwer Academic Publishers, London, 1996

[TOUMA-1998] – Touma C. and Gotsman C., "Triangle Mesh Compression", in Proceedings of Graphics Interface, Vancouver, June 1998.

[TSAI-2004] – Tsai C.-Y., Hsu H.-K., Hang H.-M., Chiang T., "A scalable video coding scheme based on interframe wavelet technique", ISO/IEC JTC1/SC29/WG11 M10569/S08, Munich MPEG meeting, Germany, March, 2004

[TUBARO-2004] – Tubaro S., Cordara G., "Fast In-band MCTF with Scalable Motion Vectors", ISO/IEC JTC1/SC29/WG11 M10569/S01, Munich MPEG meeting, Germany, March 2004

[TURK-1991] – Turk M. and Pentland A., "Eigenfaces for recognition", J. Cognitive Neuroscience, Vol.3, no.1, pp.71-86, 1991

[VETTERLI-1986] – Vetterli M., "Filter banks allowing perfect reconstruction," Signal Processing, vol.10, pp.219-244, 1986.

[VETTERLI-1992] – Vetterli M. and Cormac H., "Wavelets and filter banks: theory and design", IEEE Trans. Signal Processing, vol. 40, no. 9, September 1992.

[VETTERLI-1995] – Vetterli M. and Kovajcevic J., *Wavelets and Subband Coding,* Englewood Cliffs, NJ: Prentice-Hall, 1995.

[VIERON-2004] – Vieron J., François E., Bottreau V., Guillemot C., Marquant G., Boisson G., "Fully Scalable Video Coding", ISO/IEC JTC1/SC29/WG11 M10569/S18, Munich MPEG meeting, Germany, March 2004

[VILLASENOR-1995] – Villasenor, J.D.; Belzer, B.; Liao J., "Wavelet filter evaluation for image compression", IEEE Trans. on Image Processing, Vol.4, no.8, pp.1053-1060, Aug.1995

[VINC-1991] – Vincent L. and Soille P., "Watersheds in digital space: an efficient algorithm based on immersion simulations", IEEE Trans. on PAMI, Vol.13, no.6, pp.583-598, June 1991

[WANG-1994a] – Wang Yao, Lee O., "Active mesh-a feature seeking and tracking image sequence representation scheme", *IEEE Trans. Image Processing*, Vol.3, no.5, pp. 610-624, Sept. 1994

[WANG-1994b] – Wang J., Adelson E., "Representing moving images with layers", IEEE Trans. on Image Processing, Vol.3, no.5, pp.625-638, Sept. 1994

[WANG-1998] – Wang D. "Unsupervised video segmentation based on watersheds and temporal tracking", IEEE Trans. on CSVT, Vol.8, no.5, pp.539-546, 1998

[WANG-1999] – Wang A., Xiong Z., Chou P. A., and Mehrotra S., "Three dimensional wavelet coding of video with global motion compensation", in Proc. Data Compression Conference, SPIE, 1999

[WANG-2002] – Wang Yao, Ostermann Jorn, Ya-Qin Zhang, *Video Processing and Communications*, Prentice Hall, ISBN 0-13-017547-1, 2002

[WANG-2003] – Wang H., Schuster G.M., Katsaggelos, A. K., Pappas T. N., "An efficient rate-distortion optimal shape coding approach utilising a skeleton-based decomposition", IEEE Trans. on Image Processing, Vol.12, no.10, pp.1181-1193, Oct. 2003

[WATERS-1987] – Waters K., "A muscle model for animating three-dimensional facial expression", in Computer Graphics (Proceedings of SIGGRAPH 87), Vol.21, no.4, pp.17-24, 1987

[WEIS-1997] – Weiss Y., "Smoothness in layers: Motion segmentation using nonparametric mixture estimation", in IEEE Conference on CVPR, pp.520-526, 1997

[WIEGAND-2003] – Wiegand T. and G. J. Sullivan, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H264 | ISO/IEC 14496-10 AVC)", JVT-G050r1, May 2003

[WIEN-2004] – Wien M., Rusert T., Hanke K., "RWTH Proposal for Scalable Video Coding Technology", *ISO/IEC JTC1/SC29/WG11 M10569/S16*, Munich MPEG meeting, Germany, March 2004

[WITTEN-1987] – Witten I. H., et al, "Arithmetic coding for data compression", Comm. of ACM, Vol. 30, no.6, pp.520-541, 1987

[WOLLBORN-1998] – Wollborn M., Mech R., "Refined procedure for objective evaluation of video object generation algorithms", in *Doc. ISO/IEC JTC1/SC29/WG11 M3448, 43rd MPEG Meeting*, Tokyo, Japan, March 1998

[WONG-2003] – Wong Kwok-Wai, Kin-Man Lam and Wan-Chi Siu, "A robust scheme for live detection of human faces in colour images", signal processing: Image Communication, Vol.18, no.2, pp.103-114, 2003

[WOODS-2002] – Woods J. W., and Chen P., "Improved MC-EZBC with quarter-pixel motion vectors", ISO/IEC JTC1/SC29/WG11, MPEG doc M8366, Fairfax, May 2002

[WORRALL-2002] – Worrall S., Sadka A. H., Kondoz A.M., "3-D facial animation for very low bit-rate mobile video", in Proceedings of the IEE Third International Conference on 3G Mobile Communication Technologies (3G 2002), London, UK, 8-10 May 2002.

[WU-1993] – Wu S., Kittler J., "A gradient-based method for general motion estimation and segmentation", Journal of Visual Communication Image Representation, Vol.4, no.1, pp.25-28, March 1993

[WU-1997] – Wu X., "Lossless compression of continuous-tone image via context selection, quantisation, and modelling", IEEE Trans. on Image Processing, Vol.6, no.5, pp.656-664, May 1997

[WU-2001] – Wu, F. Li S., Zhang Y. –Q., "A framework for efficient progressive fine granularity scalable video coding", IEEE Trans. on CSVT, Vol.11, no.3, pp.332-344, March 2001

[WU-2004] – Wu Y., Golwelkar A., Woods J. W., "MC-EZBC Scalable Video Coder", *ISO/IEC JTC1/SC29/WG11 M10569/S15*, Munich MPEG meeting, Germany, March 2004

[XING-2001] – Xing Guiwei; Jin Li; Shipeng Li; Ya-Qin Zhang, "Arbitrarily shaped video-object coding by wavelet", IEEE Trans on CSVT, Vol.11, no.10, pp1135-1139, Oct. 2001

[XU-1997] – Xu Chenyang and Prince J. L., "Gradient vector flow: a new external force for snakes", in: Proceeding of Int. Conf. CVPR'97, pp.66-71, 1997

[XU-2000] – Xu J. Z., Li S., Zhang Y.-Q, "Three-dimensional shape adaptive discrete wavelet transform for efficient object-based video coding", IEEE/SPIE Visual communications and image processing, (VCIP2000), Perth, June 2000

[XU-2001] – XU J. Z., Li S., Xiong Z., and Zhang Y.-Q., "3D embedded subband coding with optimal truncation (3D-ESCOT)", Applied and Computational Harmonic Analysis, Vol.10, May 2001, pp. 589

[XU-2004] – Xu J., Xiong R., Feng B., Sullivan G., Lee M.-C., Wu F., Li S., "3D sub-band video coding using barbell lifting", ISO/IEC JTC1/SC29/WG11 M10569/S05, Munich MPEG meeting, Germany, March 2004.

[YAKI-1978] – Yakimovsky, Y., and Cunningham R., "A system for extracting three-dimensional measurements from a stereo pair of TV Camera", *Computer Graphics and Image Processing*, Vol.7, no.2, pp.195-210, April 1978

[YAMAG-1997] – Yamaguchi Y., Ida T., and Watanabe T., "A binary shape coding method using modified MMR", in Proceedings of the IEEE International Conference on Image Processing, Santa Barbara, CA, pp.I-504 -508, Oct. 1997

[YANG-1994] – YANG G., Huang T. S. "Human face detection in complex background", Pattern Recognition, Vol.27, no.1, pp.53-63, 1994

[YANG-2002] – Yang Ming-Hsuan, Kriegman D., Ahuja N., "Detecting faces in images: a survey", IEEE Trans. PAMI, Vol.24, no.1, pp.34-58, Jan. 2002

[YILMAZ-2004] – Yilmaz A., Li Xin, and Shah M., "Object contour tracking using level sets", Proceedings of ACCV, Korea, 2004

[YONG-1998] – Yongmei Wang, and Staib L. H., "Boundary finding with correspondence using statistical shape models", in Proceedings of CVPR, pp.338-345, 1998

[YOW-1997] – Yow K. C. and Cipolla R., "Feature-based human face detection", *Image and Vision Computing*, Vol.15, no.9, pp.713-735, 1997

[YULLE-1992] – Yulle A L., Hallinan P. W., and Cohen D. S., "Feature extraction from face using deformable templates", International Journal of Computer Vision, Vol.8, no.2, pp.99-111, 1992

[ZHANG-1992] – Zhang Y. -Q., Zafar S., "Motion-compensated wavelet transform coding for colour video compression", IEEE Trans. CSVT, Vol.2, no.3, pp. 285-296, Sept. 1992