Enhanced Video Coding based on Video Analysis and Metadata Information

by

Hyun-Ho Jeon

B.E., Pusan National University, Korea 1988
M. Sc, Korea Advanced Institute of Science and Technology, Korea 1990

A Dissertation Submitted in Partial Fulfillment of

the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical and Computer Engineering

Supervisory Committee

Enhanced Video Coding based on Video Analysis and Metadata Information

by

Hyun-Ho Jeon

B.Eng., Pusan National University, Korea 1988

M. Sc, Korea Advanced Institute of Science and Technology, Korea 1990

Supervisory Committee

Dr. Peter F. Driessen, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Andrea Basso, Co-Supervisor
(Department of Electrical and Computer Engineering)

Dr. Pan Agathoklis, Departmental Member
(Department of Electrical and Computer Engineering)

Dr. Nigel Horspool, Outside Member
(Department of Computer Science)

Supervisory Committee

Dr. Peter F. Driessen, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Andrea Basso, Co-Supervisor
(Department of Electrical and Computer Engineering)

Dr. Pan Agathoklis, Departmental Member
(Department of Electrical and Computer Engineering)

Dr. Nigel Horspool, Outside Member
(Department of Computer Science)

ABSTRACT

Achieving high compression ratio without significant loss of quality is the main goal of the most standard video coding systems. Since consecutive frames of a general video sequence have high correlations, the temporal redundancy between frames is removed by using motion estimation and motion compensation techniques. In this thesis, we investigate the use of video content information within the video coding system and propose a new video coding approach that can save significant bit-rates of the compressed video. Main units of the proposed coding scheme include the scene analyzer and image interpolator. The scene analyzer at the encoder extracts scene-modeling parameters from input sequences. The image interpolator at the decoder reconstructs the video frames by using the transmitted modeling parameters.

The scene analyzer consists of the camera motion detector and image-matching module. We propose a new camera motion detection method that directly analyzes the 2-D distribution of inter-frame motion fields. Experimental results show that the proposed method provides higher detection accuracy and faster computation time than the 1-D angle histogram-based method. A robust image-matching method that is invariant to scale changes, rotations, and illumination changes has been presented. Invariance to these

changes is achieved by adopting mutual information as a measure of similarity and adaptively changing the size and orientation of the local matching windows. To reduce ambiguities of the local matching, a global matching technique has been combined with the local matching.

To evaluate the performance of the proposed coding scheme, we have integrated the camera motion detector, the image-matching module, and the image interpolator with the standard MPEG-4 video codec. We compare our method with the standard MPEG-4 codec in terms of bit rates, computation time, and subjective and objective qualities.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ASM	Angular second moment
CBR	Constant bit rate
CD	Compact disk
DCT	Discrete cosine transform
DDL	Description Definition Language
DoG	Difference of Gaussian
DVD	Digital versatile disk
FOC	Focus of Contraction
FOE	Focus of Expansion
GMC	Global motion compensation
HMMD	Hue-max-min-diff color space
HSV	Hue-saturation-value color space
ISO	International standard organization
ITU-T	International Telecommunication Union-Telecommunication Standardization Sector
MI	Mutual information
MPEG	Motion Picture Expert Group
NMI	Normalized mutual information
PSNR	Peak Signal-to-Noise Ratio
QP	Quantization parameter
RGB	Red-green-blue color model
ROM	Read only memory
SIFT	Scale invariant feature transform
VGA	Video graphics array
VBR	Variable bit rate
XML	Extensible markup language
YUV	Luminance and chrominance color

Acknowledgements

I would like to express my deepest gratitude to my supervisors, Dr. Peter F. Driessen and Dr. Andrea Basso, for their academic and financial support, encouragement, and patience throughout my graduate study. I would like to thank Dr. Pan Agathoklis, Dr. Nigel Horspool, and Dr. Hari Kalva for their comments and suggestions in my oral examination.

I also would like to thank Dr. Alexandra Branzan Albu for the comments on chapters 2 and 3 of the thesis.

I would like to express my sincere thanks to: David Kwon, Chulwoong Jeon, Chengdong Zhang, Thomas Huitika, and Karl Nordstrom.

Lastly, I would like to thank my family for their support and sacrifices they made for me.

Dedication

Dedicated to my family

Chapter 1

Introduction

This chapter provides an introduction to video compression, main contributions, and an outline of the thesis.

1.1 Video Compression Techniques

The storage or transmission of video signals in an uncompressed form requires a large amount of data. For example, a color video signal in VGA resolution (640×480) consists of three-color channels (i.e. R, G, and B). If 24-bit is used to represent the color of a pixel, one second of the uncompressed color video clip with 30 Hz frame rate would require 27,648,000 bytes, or around 27 MB storage. For a two-hour movie, the amount of storage would be over 190 GB. Since the capacity of storage devices or transmission channels is limited, it is important to efficiently compress the video signals without significant loss of quality.

In general, video sequences contain redundancy within and between video frames. Two types of data redundancy can be removed: spatial redundancy and temporal redundancy.
Since the values of spatially closed pixels within the frame are highly correlated, it is assumed that the value of a particular image pixel can be predicted from neighboring pixels. To make use of the spatial correlation, 2-D transforms [1], vector quantization [2] etc. can be used. Transform coding transforms a block of pixels into a block of coefficients. Since a block of spatially correlated data can be approximated by a few coefficients, the 2-D transform followed by a quantization achieves high compression rates. Vector quantization jointly quantizes pairs of neighboring pixels. An image is

regularly divided into small blocks (typically 4×4 block) and each block is represented by a code vector. Since a block of pixels can be reconstructed by transmitting the corresponding index word of the codebook, a reduction of the data rate is possible. In addition to the spatial correlation, the correlation between pixels in temporally close frames is also high. Therefore, it is assumed that a particular image pixel can be predicted from pixels in temporally neighboring frames. Several video compression techniques such as three-dimensional coding [3] and motion-compensated prediction coding have been developed to make use of the temporal correlation between images. Three-dimensional coding extends spatial domain prediction techniques into the temporal domain by applying a transform or vector quantization to the temporally neighboring frames [3, 4]. Motion-compensated prediction coding estimates motions between consecutive frames and predicts the current frame from previous decoded frames [5].

To achieve high compression rates, current video compression standards from ISO/IEC or ITU-T adopt hybrid transform coding structure, i.e. a DCT transform coding for spatial redundancy reduction and block-based motion compensation for temporal redundancy reduction. The motion compensated hybrid coding structure relies on the assumption that motions between consecutive frames can be represented by 2-D translational vectors and one motion vector can represent the motion of a block of pixels. Thus, a single video frame is divided into rectangular blocks and each block is predicted from the previously decoded frame by motion estimation and compensation. The differences between original image blocks and motion compensated prediction blocks are coded by DCT transforms and sent to the decoder with motion vectors.

Although these video coding standards provide high compression rates, there have been approaches that aim to improve the coding efficiency by considering semantic information about a scene [6]. If we know that a scene contains a face, model-based coding can be applied to the specific face object. Model-based coding uses a 3-D model of the human face and modeling parameters to efficiently describe and animate the human face in a scene. When combined with the H.263 video codec, this approach shows about 35 % of bit-rate savings compared to the standard H.263 coded streams [7]. However, the

model-based method requires an *a priori* known 3-D model at the encoder and decoder, which may not be available for general video sequences. Region-based coding (also called object-based coding) segments a video frame by a set of regions and applies different encoding strategies to each region [8, 9]. For example, the quality of the foreground region in a scene can be enhanced by adaptively assigning the given available bit rates. The capability of coding sub-regions in the scene is also a part of the MPEG-4 video standard [10]. The recognition of a face in the input video sequence, a 3-D model, or the information of scene segmentation should be available before the start of the encoding.

Recently, the introduction of the MPEG-7 metadata standard [15] has motivated people to expect that future audio-visual material will be available with metadata [11, 12]. The MPEG-7 metadata specifies the standard way of describing the content of multimedia data for indexing purpose. Some approaches have addressed the problem of using video metadata for video coding applications such as complexity reduction for transcoding [13], coding efficiency for multiple reference frames [12] or packet dropping for adaptive transmission [14].

Some video scenes contain motions that are hard to efficiently encode using traditional block-based motion compensation. If these specific motions can be detected in video sequences and modeled by compact modeling parameters, they can be effectively reproduced at the decoder by simulating the temporal variation of the original motion. In particular, we propose to employ MPEG-7 motion descriptors to render the motion of a video shot. We describe a new video coding scheme that extends the block-based video coding toward content-information based approach. The work presented in this thesis investigates the image and video analysis tasks that identify specific camera motions and extract modeling parameters of the motions. We also consider the reconstruction of motion from the modeling parameters. All these analysis and reconstruction tools are integrated into the MPEG-4 video codec.

1.2 Contributions

The goal of this thesis is to develop new video coding schemes and tools that exploit video metadata for the efficient representation of video sequences. The specific research area and general contributions are summarized as below.

A new camera motion detection method based on 2-D motion cooccurrence matrices has been developed. A modified cooccurrence matrix has been proposed for the compact representation of global motion in images. In contrast to 1-D histogram-based approaches, this algorithm does not require additional image features for the analysis of dominant camera motions in video sequences. The proposed method was compared with the angle histogram-based method and shown to have a better performance in terms of the detection accuracy and computation time. This is an extension of the work in [49].

A robust image-matching method that is invariant to scale changes, rotations, and illumination changes has been developed [50]. To achieve illumination invariance, local mutual information has been used as a measure of similarity between local windows. Since mutual information is sensitive to image rotations and scale changes, we introduced a new mutual information-based feature descriptor that is invariant to scale and rotation changes. The proposed method was compared with one of the state-of-the-art feature descriptors for a set of test images. Experimental results showed that the proposed method is particularly effective to find matching points between images that have many similar regions or large illumination changes.

The developed scene analysis tools have been integrated into the hybrid DCT-based codec and a new video coding scheme has been developed. For a set of test videos, the proposed coding scheme was compared with the standard MPEG-4 codec in terms of bit rates, computation time, and subjective and objective qualities. This work builds on that initially presented in [72].

1.3 Thesis Outline

The thesis is divided into six chapters. The first chapter provides introductory material and an outline of the thesis. The remaining chapters are organized as follows.

Chapter 2 describes the proposed camera motion detection technique. Modified 2-D cooccurrence matrices are introduced and the proposed detection algorithms are presented. Experimental results obtained with the proposed method are compared with those of the existing method.

Chapter 3 presents a new robust image-matching method for images containing large scaling changes, rotations, and illumination changes. The traditional mutual information has been extended to the invariant local descriptor. A global matching technique is used to reduce ambiguities of local feature matching. Experimental results are compared with results obtained with the state-of-the-art method.

Chapter 4 describes the brief introduction of the MPEG-7 metadata standard. Recent video coding applications that use video metadata for transcoding, reference frames selection, and packet transmission are introduced. For the metadata that are extracted by the proposed scene analysis tools, encoding scheme has been presented within the MPEG-7 structure.

Chapter 5 presents a new video coding scheme that exploits scene content information for the reduction of bit rate of the compressed stream. The proposed coding scheme is tested for a set of videos and compared with the standard MPEG-4 video codec.

Chapter 6 discusses conclusions and suggestions for future research.

Chapter 2

Camera Motion Detection and Characterization

Camera motion is an important feature for video coding, video indexing and retrieval purposes. In this chapter, we present a new camera motion detection method that can detect camera motions and identify shot boundaries of the detected camera motion shots in video sequences. This is achieved by analyzing a sequence of two dimensional cooccurrence matrices of motion vectors or optical flow vectors computed from successive pairs of images. We evaluate the performance of the proposed method using test video sets and compare it with the existing method.

2.1 Introduction

The estimation of camera motion is important for video coding, video indexing and retrieval purposes. In video coding, global motion is estimated to compensate camera motion and get more precise local motion estimation. A key element of video indexing and retrieval systems is the segmentation of the video sequence into shots and their characterization on the basis of their motion characteristics such as static shot, panning or zooming. A shot is a sequence of frames that were captured from a single camera operation and it is an elementary unit in video segmentation.

A typical camera operation set consists of pan, tilt, zoom, track, boom and dolly [15]. We focus on pan, tilt and zoom only because these three types of camera motion are the most commonly considered in many applications. Fig.2.1 shows camera panning and tilting operations. Camera panning is a horizontal rotation of the camera around the vertical axis.

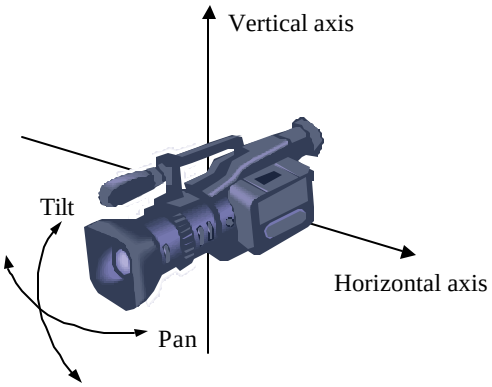


Figure 2.1. Camera panning and tilting

Tilting is a vertical rotation of the camera around the horizontal axis. These two operations are often used to track a moving object or to provide a wider view of the scene. Zooming is a change of the camera focal length. Zoom-in or zoom-out is used to provide more details or general sight of a scene.

The method is based on the analysis of motion cooccurrence matrices that are able to express how pairs of motion vector directions are spatially distributed in the image. For each frame of an input video sequence, we estimate an optical flow field. Then, we build a motion cooccurrence matrix for each frame using the estimated optical flow field. The camera motion type of the current frame can be identified by examining the structure of the cooccurrence matrix between two images because it provides compact representation of spatially homogeneous motion distribution.

We first present a brief review of the work related to the recognition of the camera motion. Then, we present the proposed approach and experimental results. Finally, we discuss the conclusion and future work.

2.2 Related Work

Many methods have been developed and they can be generally classified into two categories [16]: global motion model estimation and direct analysis of motion vector distribution.

Camera operation usually induces a global motion in video sequences. Global motion model based approaches estimate parametric motion model between images. Model parameters are estimated starting from a dense optical flow field or block-based motion vector field. Depending on the number of parameters, different models such as the 6-parameter affine model [26] and the 8-parameter perspective model [18] can be used. Simpler models are often used by assuming more constrained camera motion. For example, the 3-parameter model [17] assumes that the camera dose not rotate around the axis of the camera lens and only considers pan, tilt and zoom. Since the motion of large objects or noisy motion vectors may reduce the reliability of the global motion estimation, robust estimation techniques [18, 25] are needed. During the robust estimation, motion vectors that do not fit well with the estimated global motion are removed to reduce the effect of outliers, such as non-camera motions, and the estimation is repeated. Therefore, they are computationally very expensive. After obtaining model parameters for each frame, thresholding is applied to the estimated parameters to detect the camera motion of each frame [17, 26], which requires the proper selection of multiple thresholds.

Camera motion can also be detected by directly examining the motion vector fields. These approaches rely more on a statistical measurement of global motion distribution, such as angle histograms [28, 29], moments [30] etc. Angle histograms have been widely used because they can be computed efficiently. For a motion vector field, an angle histogram is constructed by quantizing the angle of motion vectors into discrete directions and counting the number of motion vectors in each direction. Consequently, a pan or tilt can be detected by identifying the highest bin of the histogram. But the detection of a zoom requires additional analysis of images such as Focus of Expansion (FOE)/Focus of Contraction (FOC) tests or inter-frame intensity changes [19, 28]. The presence of large moving objects or a low textured background may disturb motion distributions within an image, which often results in incorrect detections. To avoid the influence of moving objects, background motion templates are defined in motion vector fields [20]. However, this approach is only effective when moving objects are confined to the pre-defined areas in the picture. These approaches also depend on the thresholding of the statistical

measurements. To obtain threshold values automatically, adaptive thresholding [19] was proposed in which a threshold at each frame is calculated from the local statistics of neighbouring frames. However, this thresholding technique also requires two parameters: one is the size of the local window and the other is a parameter related to the local statistics, which should be specified by users.

In this chapter, we propose a new method for the detection of camera motions based on the second-order statistics of motion fields. We aim at a detection of dominant camera motion in each video sequence. Our method is qualitative detection in the sense that we estimate which camera operations have occurred in a given sequence. The proposed method uses the 2-D motion cooccurrence matrix that is derived from the motion field between two images. We exploit the relation of the 2-D motion cooccurrence matrix to camera motions. The motion cooccurrence matrix captures spatial correlation between neighboring motion vectors, and it enables the detection of camera motions (pan, tilt and zoom) without additional image feature measurements. Our detection method relies on the thresholding of statistical parameters extracted from motion cooccurrence matrices. To help the selection of threshold values, we use simplified models of motion distribution for the moving object and smooth background, which allows us to avoid manually searching multiple threshold values.

2.3 Qualitative Camera Motion Detection Using Motion Cooccurrences

The proposed method consists of three steps. In the first step, for each frame of the input sequence, we compute the motion cooccurrence matrix using the optical flow field. In the second step, we calculate the statistical features of the cooccurrence matrix for each frame because each camera motion is characterized by a certain pattern in the motion cooccurrence matrix. The statistical features are analyzed to identify the type of current frame's camera motion.

2.3.1 Motion Cooccurrence Matrix

The concept of the cooccurrence matrix has been introduced for the purpose of texture analysis [21]. Given an image to be analyzed, the cooccurrence matrix with a distance *d*

consists of four 2-dimensional $N \times N$ matrices, where N is the number of quantization levels for the pixel value. Let r be the spatial position of a pixel in a picture I, s be the position of the neighboring pixel with distance d, and $I(r)$ be the intensity at r. Then, for each of the four directions (horizontal, vertical, positive diagonal, and negative diagonal), the element $C(i,j,d,f)$ of the matrix is defined as follows:

$$C(i,j,d,f) = \#\{(r,s) \mid I(r) = i, I(s) = j, d(r,s) = d\}, \; f = \{0^o, 45^o, 90^o, 135^o\} \qquad (2.1)$$

Eq. (2.1) specifies the number of occurrences of the two neighboring pixels with value j at a f directional distance d from a pixel of value i in the image. Here, i and j represent the quantized level of intensities, $d(r,s) = |r-s|$ represents the spatial distance between two positions, r and s. The cardinal of a set '#' counts the number of neighbouring point pairs that have quantized intensities i and j over the whole picture. For $d = 0$, the cooccurrence matrix reduces to a general intensity histogram. For $d > 0$, the matrix represents the spatial correlation of pairs of points as a function of the distance between image pixels. This correlation usually decreases with distance. The idea of the cooccurrence matrix was extended to motion description. In [22], the normal flow in which the component of the flow vector is parallel to the spatial gradient is computed for each frame pair. Since the ratio of the same motion to the different motion is extracted from the cooccurrence matrices of the normal flow direction, this approach can only measure the spatial homogeneity of the flow in video sequences. In [23], temporal cooccurrence matrices of local motions are introduced to characterize the global motion of a whole video shot. For each frame pair with a temporal distance d, the temporal cooccurrence matrix between two frames is obtained by counting the occurrence of quantized motion magnitude for all motion vector pairs. Then, a set of global motion features are extracted from temporal cooccurrence matrices to classify dynamic motion properties of the video sequences.

Since cooccurrence matrices are useful tools for representing global motion characteristics such as homogeneity and coherency, we use them for the detection of dominant camera motions. Our method is different from the above approaches. In [22, 23], each shot was assumed to be previously segmented and cooccurrence matrices of each shot were extracted in spatial or temporal domains to describe dynamic motion contents.

We aim to temporally segment the video sequences and identify camera motion types by using global features extracted from cooccurrence matrices.

For each frame of an input video sequence, we compute an optical flow field to estimate the motion between two consecutive images. Optical flow field is the velocity field that represents the three-dimensional motion of an object on a two dimensional image [31]. An example of two consecutive images and a corresponding optical flow field are shown in Fig. 2.2.

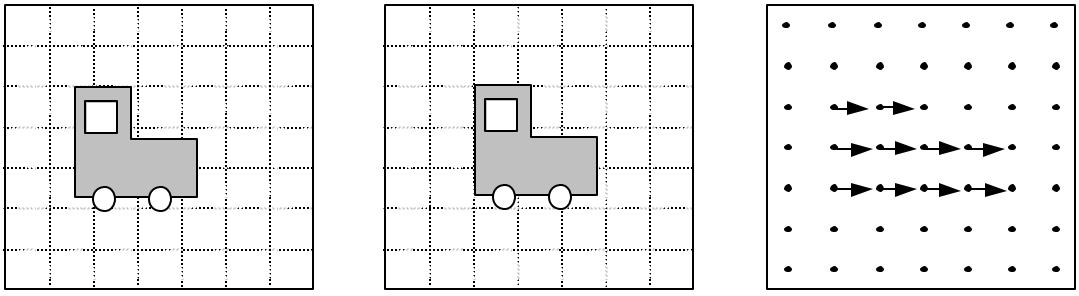


Image I(t) Image I(t+1) Optical flow field

Figure 2.2. Example of an optical flow field

Optical flow is the apparent motion of brightness patterns in the image. To calculate the optical flow at an image point, it is assumed that the intensity does not change along the trajectory of the moving point in the image. Let $I(x, y, t)$ be the image intensity at time t at the image point (x,y), and (u,v) be the optical flow vector at that point. By assuming that the intensity will be the same at time $t + dt$ at the point $(x+dx, y+dy)$, we obtain the well-known optical flow constraint equation

$$\frac{\partial I}{\partial x}\cdot u+\frac{\partial I}{\partial y}\cdot v+\frac{\partial I}{\partial t}=0 \qquad (2.2)$$

Since there are two unknown components, u and v, in Eq.(2.2), further constraints such as local smoothness constraints or global smoothness constraint are necessary to estimate flow vectors [24]. Conditions such as the absence of texture and the presence of large

motions decrease the accuracy of the optical flow. Reliability measures and multi-resolution techniques can be used to handle these problems [24].

To compute optical flow, we use Lukas and Kanade's algorithm. Lukas and Kanade's algorithm is known to provide the lowest error rate [24] and its multi-resolution structure can search for large displacements. As we do not need a dense motion field for this work, we calculate the optical flow for uniformly sampled image points. Since flow vectors for each 16×16 or 8×8 image block provide enough motion information of an image, we measure optical flow vectors for each discrete point separated by 8 pixels in both horizontal and vertical directions. For an optical flow (u, v) at the image point (x, y), the magnitude and angle are given by

$$r = \sqrt{u^2 + v^2}\ ,\ q = atan\left(\frac{v}{u}\right) \tag{2.3}$$

Given a flow field, the computation of the motion cooccurrence matrix requires a quantization of the flow vector angle. In this work, the angle is quantized into 12 directions with 30^o intervals, as illustrated in Fig.2.3. The motivation for using the orientation of flow vectors is that the direction of flow vectors has a more direct relationship to the type of camera motion. We consider only flow vectors with non-zero magnitudes, which are detected by a global thresholding of 0.2. The cooccurrence matrix in Eq. (2.1) is simplified to represent the number of motion vector pairs having specific orientations. For each flow vector, the orientation of its four neighboring vectors with distance d is counted in a 2–dimensional flow field. Then, the simplified motion cooccurrence matrix $C(i,j,d)$ is defined as

$$C(i,j,d) = \#\{(r,s) \mid \hat{q}(r) = i, \hat{q}(s) = j, d(r,s) = 8\} \tag{2.4}$$

where r and s represent spatial positions of the current and neighboring flow vector pairs in the flow field, $d(r,s)$ represents the spatial distance between two positions, r and s, and $\hat{q}(r)$ and $\hat{q}(s)$ represent quantized angles of two flow vectors.

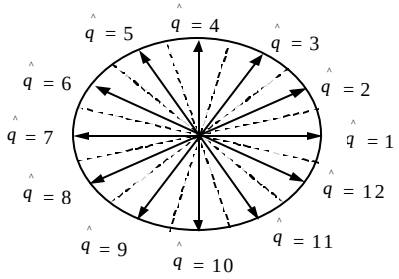
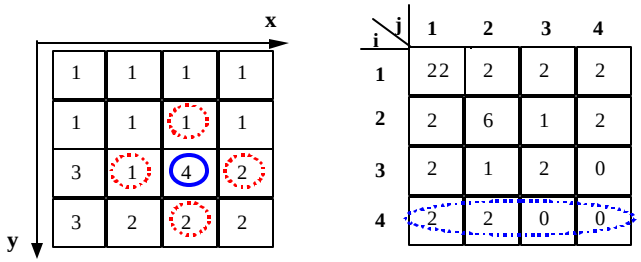


Figure 2.3. Quantized flow vector angles.



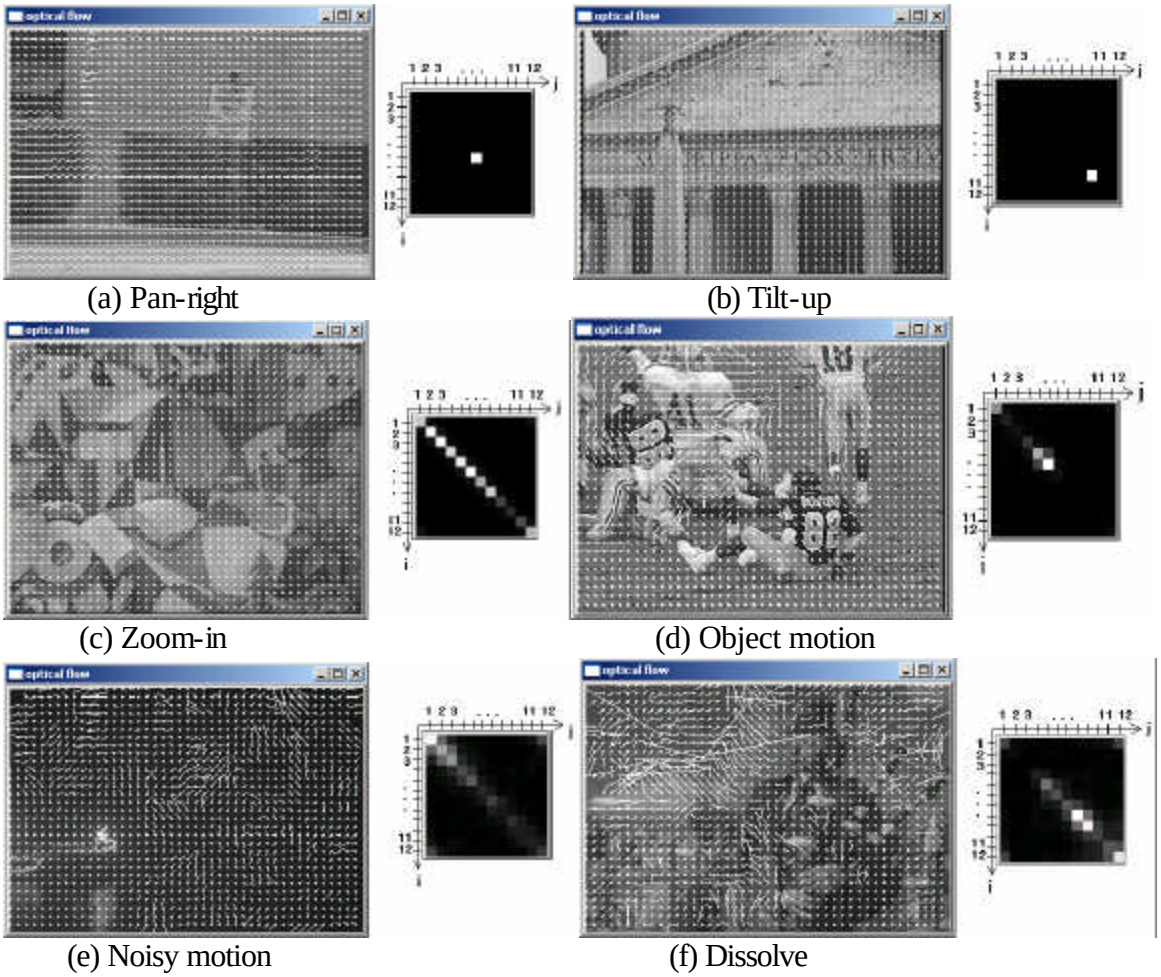
(a) (b)

Figure 2.4. Construction of a motion cooccurrence matrix.(a) Quantized angles of a 4×4 flow field. (b) Motion cooccurrence matrix of the given flow field with quantized angles 1-4.

Fig. 2.4 shows the construction of a motion cooccurrence matrix for a given 4×4 flow field. Each number in Fig. 2.4 (a) represents the quantized angle of a flow vector direction, ranging from 1 to 4. For instance, consider the point surrounded by a solid blue circle. The quantized angle of this point, $q(r)$, is 4. Next, consider four neighboring points. These points are marked by dotted red circles. By counting the number of points for each angle, we get $C(4,1) = C(4,2) = 2$, $C(4,3) = C(4,4) = 0$. Corresponding elements of $C(i,j)$ are marked by a solid blue circle in Fig. 2.4 (b). To obtain a complete cooccurrence matrix, the number of occurrences is accumulated over the entire flow field.

Fig. 2.5 shows some examples of motion cooccurrence matrices for images containing camera motion or object motion. If the camera pans or tilts, most of the motion vector directions will be parallel to the horizontal or vertical direction. For example, when the camera pans, the element of the cooccurrence matrix that corresponds to a quantized angle of 0^o or 180^o will contain the largest value of occurrences depending on pan-left or

pan-right. When the camera tilts, the matrix element corresponding to an angle of 90^o or 270^o will show the largest value for tilt-down or tilt-up. If the camera zooms, motion vectors will be distributed equally over all directions. So all the elements in the main diagonal direction will have similar occurrence values. If the images have large moving objects without camera motion, only the elements corresponding to object motions will contain a large number of occurrences. When the image intensities are very low or it is a gradual overlap of two different scenes, called dissolve, the direction of a flow field is randomly distributed as in Fig. 2.5 (e) and (f). In this case, the elements of the matrices are dispersed along the diagonal direction and non-diagonal components are more noticeable than those of the zoom image.



(a) Pan-right (b) Tilt-up

(c) Zoom-in (d) Object motion

(e) Noisy motion (f) Dissolve

Figure 2.5. Examples of the motion cooccurrence matrices for various camera motions.

2.3.2 Global Feature Extraction

As described in 2.3.1, the motion cooccurrence matrix shows different patterns for various camera motions. Therefore, we use simple but useful global features extracted from these cooccurrence matrices. In cooccurrence matrix based analysis, a set of 14 features is defined to characterize the image [21]. Some of these features relate to specific characteristics of the images such as homogeneity and complexity, thus we select a subset of these features and also define an additional feature for the purpose of global motion analysis between images. As a measure of the importance of motion, we define the *Average* in each frame. Let *p(i,j)* be the normalized (*i,j*) element of a cooccurrence matrix, then the *Average* is defined as in Eq.(2.5). The *Average* reaches values close to its maximum, equal to 1 when there is a motion over the whole image.

$$Average = \sum_{i=1}^{N_q}\sum_{j=1}^{N_q} p(i,j), \quad p(i,j) = C(i,j)/R \tag{2.5}$$

where N_q and R denote the total number of quantized orientation and the total number of flow vector pairs, respectively. For each flow vector, four neighbors are considered. Therefore, R is $4\times N\times M$, where N and M are the total numbers of vectors in horizontal and vertical directions.

Since a panning or tilting typically produces the whole motion vectors aligned with a specific direction, the *Angular Second Moment* (*ASM*) is used as a measure of the motion direction homogeneity. It varies within $[1/(N_q)^2, 1]$ and its maximum value can be obtained when every motion vector has the same orientation.

$$ASM = \sum_{i=1}^{N_q}\sum_{j=1}^{N_q} p(i,j)^2 \tag{2.6}$$

During a zooming, the elements of the matrix are only present on the main diagonal. Two other features, the *Entropy* and the *Difference Entropy*, are used to detect camera zooms. The *Entropy* feature indicates the degree of spread of the motion orientation. It varies

within [0, $\log((N_q)^2)$]. The maximum *Entropy* will be obtained when all the elements of the matrix are distributed uniformly with equal magnitudes.

$$Entropy = -\sum_{i}^{N_q}\sum_{j}^{N_q} p(i,j)\log(p(i,j)) \tag{2.7}$$

Since the *Entropy* feature in Eq.(2.7) exhibits high values for dissolves as well as zooms, we need an additional feature, the *Difference Entropy*, to distinguish them. In Eq.(2.8), the *Difference Entropy* feature measures the degree of spread of the all $p_{x-y}(k)$ in the cooccurrence matrix. The *Difference Entropy* varies within [0, $\log(N_q)$]. It is supposed to be very low for zooms and high for dissolves.

$$Difference\ Entropy = -\sum_{i=0}^{N_g-1} p_{x-y}(i)\log(p_{x-y}(i)), \tag{2.8}$$

$$p_{x-y}(k) = \sum_{i=1}^{N_q}\sum_{j=1}^{N_q} p(i,j), \quad |i-j| = k$$

Fig. 2.6 shows a plot of the *Entropy* and *Difference Entropy* in a video sequence. This sequence contains a dissolve followed by a zoom and another dissolve as shown in Fig. 2.6 (a). The plot of the *Entropy* and *Difference Entropy* is shown in Fig. 2.6 (b) and (c). We can see that the *Entropy* feature is high for both zoom and dissolve, while the *Difference Entropy* feature is high for only dissolves. Therefore, the use of these two features prevents dissolves from being falsely detected as zooms.

2.3.3 Detection of Camera Motion

After extracting the global features for all input frames, we compare each frame's features with corresponding threshold values in order to determine the camera motion type of the frame. Detailed algorithmic steps are described in Table 2.1 and Table 2.2. The problem of threshold selection will be described in 2.3.4.

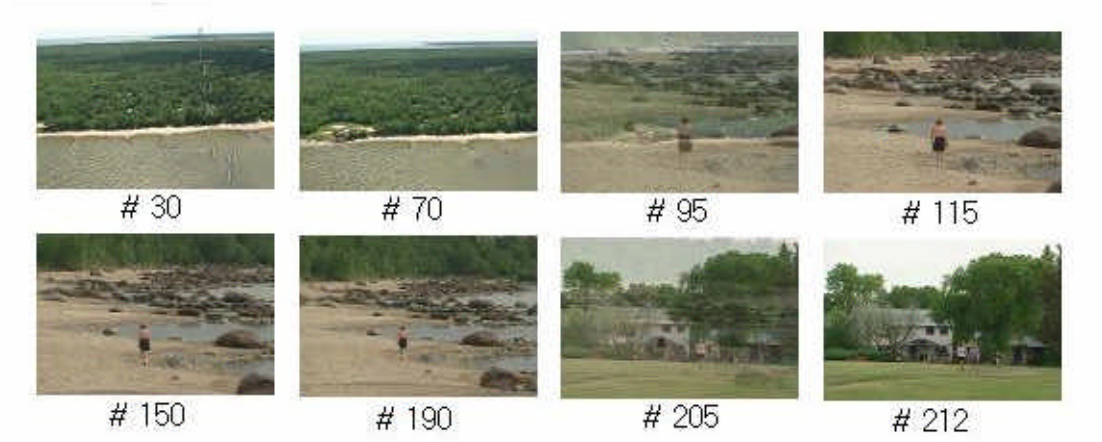
Assume that the total length of the video sequence is *N* and the global features for *N*-1 cooccurrence matrices are given. For *i-th* frame, *Average*(*i*) and *ASM*(*i*) are compared with their thresholds. If both of them are higher than the thresholds, the current frame is

marked as a potential pan/tilt frame. In addition, the index of the largest bin in each cooccurrence matrix is saved into an array that will be used to determine the direction of a pan or tilt. If any of the above two features is lower than its threshold, postprocess() is called to check the end of the motion and refine the detected boundaries. In postprocess(), over-segmentation due to the instantaneous drop of the feature values is avoided by looking ahead to the next frame's feature values. When the end of a potential camera motion shot is detected, as in line 20, only shots that last longer than the minimum duration are declared as pan/tilt shots. The direction of the detected pan/tilt is determined by taking the majority of the largest bin indices within the detected shot period.

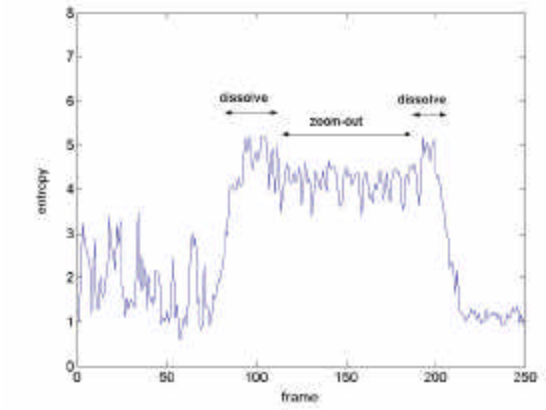
Table 2.1. Algorithm for pan/tilt detection

```
 1: M_det = 0;  /* length of the detected shot.*/
 2: min_duration=10;
 3: for i=0 to N-2
 4:   if Average(i)>T_avg and ASM(i)>T_asm
 5:      Mark current frame as a potential detected frame;
 6:      Save the index of the largest bin;
 7:      Increase M_det by 1;
 8:   else
 9:      if M_det > 0
10:         postprocess();
11:      end
12:   end
13: end
14:
15: postprocess():
16: if Average(i+1)>T_avg and ASM(i+1)>T_asm
17:     Mark current frame as a detected frame;
18:     Save the index of the largest bin;
19:     Increase M_det by 1;
20: else
21:     if M_det > min_duration
22:        Determine the direction of a pan/tilt;
23:        Output detected boundary data;
24:     else
25:        Reset previous detection result;
26:     end
27:     Reset M_det;
28: end
```

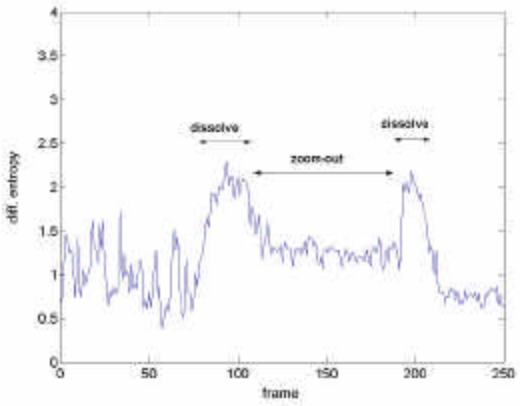
To detect zooms, three features of *i-th* frame are compared with their thresholds. If they meet the conditions in line 4 of Table 2.2, *i-th* frame is marked as a potential zoom frame. In addition, the divergence of the current frame is saved into an array that will be used to determine the direction of the zoom. The divergence of a frame is defined in Eq.(2.9) and it is the mean of the divergences for all flow vectors within the frame. Zoom-in or zoom-out can be identified by the sign, i.e. positive or negative, of the mean divergence. In postprocess(), the steps to avoid over-segmentation are conducted.



(a)



(b)



(c)

Figure 2.6. Plot of the *Entropy* and *Difference Entropy* in a video sequence containing a zoom and dissolves.(a) Sample frames of the video sequence. (b) The *Entropy* of the sequence. (c) The *Difference Entropy* of the sequence.

Table 2.2. Algorithm for zoom detection

```
 1: M_det = 0;  /* length of the detected shot.*/
 2: min_duration=10;
 3: for i=0 to N-2
 4:   if Average(i)>T_avg and Entropy(i)>T_ent and Diff_Entropy(i)<T_dent
 5:      Mark current frame as a potential detected frame;
 6:      Compute the divergence of the flow field;
 7:      Increase M_det by 1;
 8:   else
 9:      if M_det > 0
10:         postprocess();
11:      end
12:   end
13: end
14:
15: postprocess():
16: if Average(i+1)>T_avg and Entropy(i+1)>T_ent and Diff_Entropy(i+1)<T_dent
17:     Mark current frame as a detected frame;
18:     Compute the divergence of the flow field;
19:     Increase M_det by 1;
20: else
21:     if M_det>min_duration
22:        Determine the direction of a zoom;
23:        Output detected boundary data;
24:     else
25:        Reset previous detection result;
26:     end
27:     Reset M_det;
28: end
```

$$F_{div} = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} div(u(m,n)), \tag{2.9}$$

$$div(u(m,n)) = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}, \qquad u(m,n) = (u,v)$$

where M and N are the total number of vectors in horizontal and vertical directions.

2.3.4 Threshold Selection

As described in the previous section, our detection method requires thresholding of four feature values. Therefore, we need to determine four threshold values. To help us determine the appropriate threshold values, we use a simple modeling for the distribution of motion vectors in a moving object and smooth background areas. Since moving objects are often characterized by spatially homogeneous motion field except object boundaries in motion based segmentations [77, 78], we assume that the motion vectors inside a moving object have a single direction and the motion vectors in smooth background areas have randomly distributed orientations with uniform distribution. Although moving objects or smooth background areas in real video sequences exhibit more complex motion distributions, these simplified models will provide the relationship between the global feature parameters and non-camera motion distributions. Let a single parameter r denote the percentage of motion vectors belonging to the moving object or smooth background in the image. Then, the normalized elements of a cooccurrence matrix can be defined as in Eq.(A.1). From the Eq.(2.5)~(2.8), the threshold values of the four features can be related to r as shown in Table 2.3. The derivation is given in Appendix A.

When the parameter r is set too low, the thresholds are so strict that many camera motion frames will be missed. When it is set too high, on the other hand, many frames will be falsely detected as camera motions because the thresholds are too relaxed. As we are interested in the detection of dominant camera motions, we can expect that the motion in the scene is dominated by camera movement and other types of motion (i.e. moving objects or unreliable background area) take less than half of the total motions. Therefore, we can expect that the threshold values should be set at $r \leq 0.5$ for the dominant camera motion detection. In the next section, we evaluate the performance of the proposed method over different threshold sets by changing the parameter r.

Table 2.3. The threshold values for the four features (r =[0,1])

T_{avg}	1- r
T_{asm}	$(1-r)^2 + r^2 / N_q^2$
T_{ent}	$-(1-r)\cdot\log(\frac{1-r}{N_q}) - r\cdot\log(r)$
T_{diff_ent}	$-(1-r)\log(1-r)$ $-\{r/N_q\cdot\log(r/N_q) + r\cdot[\sum_{i=1}^{N_q-1} 2\cdot(N_q-i)/N_q^2\cdot\log(2\cdot r\cdot(N_q-i)/N_q^2)]\}$

2.4 Experimental Results

In this section, we describe details of test sequences, evaluation parameters and detection results. The performance of the proposed camera motion detection method has been evaluated using two sets of video sequences.

Test set 1

We have created the first test set, consisting of 50 video clips using a digital camcorder. The video frames are in 352×288 YUV 4:2:0 format and the frame rate is 30 frames/sec. In the detection of camera motion, it is generally assumed that camera motion is the dominant motion in a scene. This assumption is not valid when a large moving object or smooth background is present, since the motion of a significantly large object becomes the global motion of the scene and motion vectors in low textured areas tend to be very unreliable. Therefore, the motion of large object and smooth background are the main sources of erroneous detections. To evaluate the proposed approach under different object motion and noisy motion conditions, our test set contains various level of moving objects and smooth backgrounds. The test sequences are classified into three types. Type A consists of sequences having high textured background with little or no moving objects. In type C, sequences contain a large moving object or smooth background, which takes more than half of the whole image area. Type B consists of scenes of complexity between type A and C. Table 2.4 shows parts of the test set. The number of frames, ground truth of

Table 2.4. Part of the videos in test set 1

Seq. name (Type)	Length (frame)	Manual segmentation	Comments on content	Detection results (r =0.4)
ground001 (A)	375	119-240 : pan-left	A Textured background	118-241 : pan-left
library002 (A)	169	66-151 : zoom-in	A textured background	66-156 : zoom-in
mall008 (B)	425	77-168 : zoom-in 245-398 : pan-left	0-158 : Two walking people 345-399 : A walking person	73-169 : zoom-in 246-399 : pan-left
office003 (B)	181	53-130 : pan-left	35-130 : A passing toy train	55-129 : pan-left
park008 (B)	137	26-86 : zoom-in	0-136 : A group of walking people	25-119 : zoom-in 123-132 : zoom-in
building002(C)	245	16-213 : tilt-up	81-150 : An approaching pedestrian 40-244: A large white wall	33-84 : tilt-up 149-158 : tilt-up 164-188 : tilt-up
office010 (C)	184	18-120 : zoom-in	69-88 : A passing toy train	18-176 : zoom-in
park013 (C)	270	67-230 : pan-left	0 – 269 : A large smooth background (sky)	No detected frames

temporal segmentation, short descriptions of each sequence, and detection results of the proposed method are summarized in the table. Detection results in the table were obtained by using the threshold values at r =0.4. Sample frames of these sequences are shown in Fig. 2.7. The description of the full data set is given in Appendix B.

Figure 2.7. Sample frames of the sequences in Table 2.4

The effectiveness of the proposed method is evaluated from two frequently used parameters: *Recall* and *Precision*. Let N_c, N_m, and N_f denote the number of correctly detected frames of camera motion, number of missed frames of camera motion and number of falsely detected frames of camera motion, respectively. These two metrics are defined as follow:

$$Recall = \frac{\text{Number of correctly detected frames of camera motion } (=N_c)}{\text{Number of frames of all camera motion } (=N_c + N_m)} \times 100 \qquad (2.10)$$

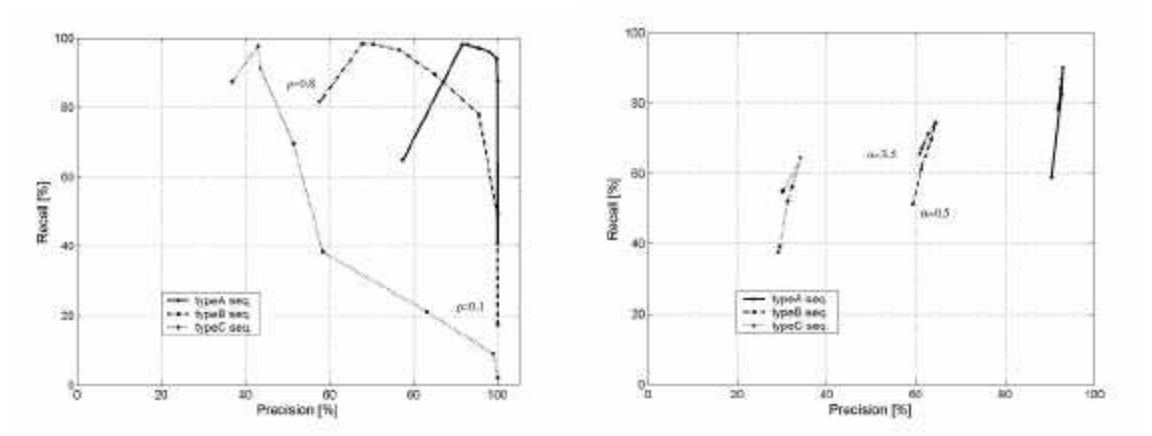
$$Precision = \frac{\text{Number of correctly detected frames of camera motion } (=N_c)}{\text{Number of detected frames of camera motion } (=N_c + N_f)} \times 100$$

Recall defines the percentage of correctly detected frames in relation to all the frames of camera motion in the data set. *Precision* defines the percentage of correctly detected frames in relation to all the frames of camera motion detected by the algorithm.

To compare the proposed approach with the conventional angle histogram based method, the method presented in [19] is applied to the same test sequences. From the optical flow field of each frame, an 8-direction angle histogram is computed. Then, the variance of the angle histogram is calculated to detect camera panning, tilting and zooming. Their detection scheme is based on the observation that the variance is low during a panning or tilting, while it is very high during a zooming. Since both dissolves and zooms exhibit high variance in angle histograms, inter-frame intensity histogram difference and the variance of the magnitude of flow vectors are also computed to distinguish dissolves from zooms. To detect panning, tilting and zooming, adaptive thresholding is applied to the extracted parameters. For each frame, the thresholds are determined adaptively as $T = \mu + \alpha\sigma$, where μ and σ are the mean and standard deviation of the local sliding window. α is an experimental parameter to be determined. The size of the sliding window is set to 15 previous frames as specified in [19]. The angle histogram method has been implemented and tested for a performance comparison.

Fig. 2.8 shows the plots of *Recall* and *Precision* rates for the three types of sequences. Fig. 2.8 (a) shows the performance of the proposed method for various threshold values. The

parameter r takes values in [0.1, 0.8] with 0.1 intervals and the threshold values are determined using Table 2.3 for each r. When r is small, *Precision* rates are very high while *Recall* rates are relatively very low. This means that the thresholds are set too strictly, so that the number of falsely detected frames of camera motion is very low and the number of missed frames of camera motion is very high. When r is large, the opposite results are obtained. The plot shows that both *Recall* and *Precision* rates are relatively high for r = 0.3~0.4. Fig. 2.8 (b) shows the performance of the angle histogram



(a) (b)

Figure 2.8. Plots of *Recall* and *Precision* rates. (a) Proposed method (b) Angle histogram method [19].

Table 2.5. Comparison of the performance for the test set 1 videos.

	Proposed method (r =0.4)				Angle histogram [19] (a =1.5)			
	Type A	Type B	Type C	Total	Type A	Type B	Type C	Total
N_c	1360	3055	829	5244	1283	2555	1130	4968
N_m	54	355	1329	1738	131	855	1028	2014
N_f	29	534	590	1153	97	1432	2533	4062
Rec.	96.2 %	89.6 %	38.4 %	75.1 %	90.7 %	74.9 %	52.4 %	71.2 %
Prec.	97.9 %	85.1 %	58.4 %	81.9 %	93.0 %	64.1 %	30.9 %	55.0 %

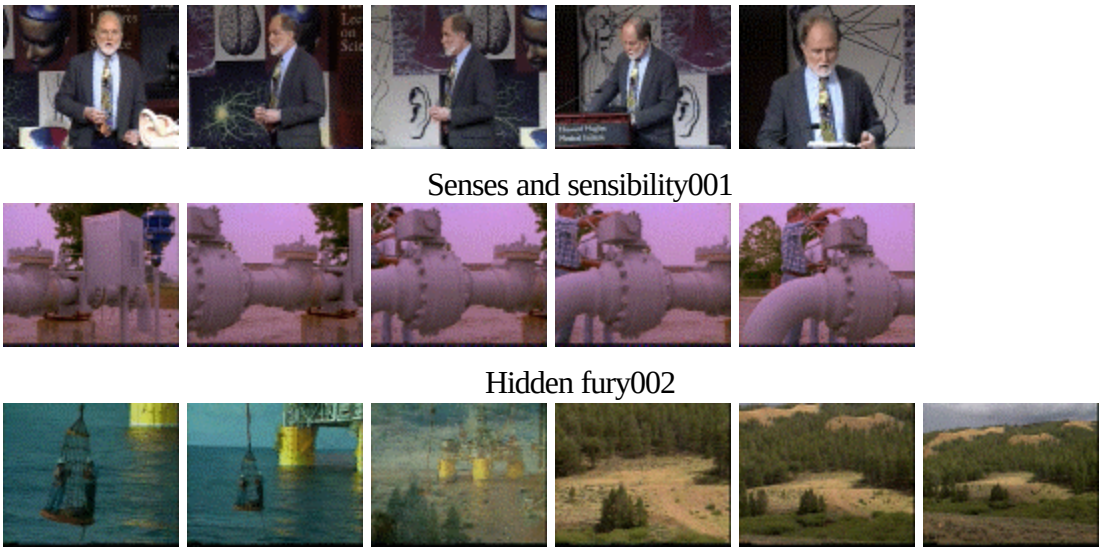
method. The parameter α, which is used for the adaptive thresholding, takes values in [0.5, 3.5] with 0.5 intervals. The plot shows that α = 1.0~2.0 gives relatively high *Recall* and *Precision* rates.

The performance of both methods is also compared at specific thresholding values as summarized in Table 2.5. Camera motions in type A sequences produce well-defined patterns in both 2-D motion cooccurrence matrices and angle histograms, due to the absence of large moving objects and noisy motion vectors. Consequently, both approaches detected temporal boundaries of camera motions in each sequence very well with high *Recall* and *Precision* rates. For sequences in type B, moving objects or smooth background did not produce significant detection error, since our method is designed to allow a certain degree of such disturbance. Falsely detected camera motion frames in the proposed method mainly came from camera jitters. The camera jittering is particularly noticeable after a high zoom-in. When the jittering occurred for a scene containing low textured areas, it was often falsely detected as a zoom. 'park008' sequence in Table 2.4 shows an example of such false detection. After the zoom-in, the shaking camera and multiple moving objects produce motion distribution similar to a zoom in the motion cooccurrence matrix. For type C sequences, both approaches obtained very bw *Recall* and *Precision* values. Table 2.5 shows that the proposed method outperforms the angle histogram based approach [19].

Test set 2

Test set 2 consists of 10 sequences extracted from the documentary videos [27]. Original videos are MPEG compressed streams. We decoded them into 352×240 YUV 4:2:0 format with a frame rate of 30 frames/sec. Table 2.6 shows the videos in test set 2. Sample frames of part of the test set are shown in Fig. 2.9. The performance of the proposed method is compared with the angle histogram method as summarized in Table 2.7.

Test set 2 videos contain 8 dissolve shots. Detection results in Table 2.6 show that both the difference entropy measure and the histogram difference measure can distinguish



Senses and sensibility001

Hidden fury002

Wrestling with uncertainty002

Figure 2.9. Sample frames of part of the sequences in Table 2.6

between zooms and dissolves. If zooming and panning (or tilting) exist at the same time, they are harder to detect than single camera motions because the assumption of a single dominant camera motion is violated in this case. In "wrestling with uncertainty002", the camera pans during the zoom-out. Since zoom is a more significant camera motion in this sequence, the camera motion is detected as a zoom. Table 2.7 shows the experimental results of both methods for the test set 2. For the proposed method, *Recall* = 77.7 % and *Precision* = 91.8 % were obtained, while *Recall* = 63.3 % and *Precision* = 76.7 % values were obtained for the angle histogram method.

To evaluate the computational speed of the proposed method, both methods were implemented on a 2.5 GHz personal computer with 512 MB RAM. The total time required for reading the input data, analyzing them for camera motion detection, and returning detection results for all sequences in the test database is shown in Table 2.8. The computation time in this table is the mean value of 10 measurements. In the proposed method, over 80 % of the time is spent on the computation of the optical flow field. In the angle histogram method, inter-frame intensity histogram difference and optical flow computation spend 50 % and 40 % of the time, respectively. Since the proposed method does not require additional features such as frame intensity histograms to distinguish

Table 2.6. Video sequences in test set 2

Seq. name	Length (frame)	Manual segmentation	Comments on content	Detection results (r =0.4)
Senses and sensibility, #1~#3	510	157-339: pan-left 358-445: zoom-in	154-342 : A man walking to the left	41-59 : zoom-out 192-211 : pan-left 246-316 : pan-left 402-430 : zoom-in 494-505 : zoom-out
	440	106-309: pan-left 318-394 : zoom-out	85-330 : A man walking to the left	6-23 : zoom-in 142-303 : pan-left 316-386 : zoom-out 397-408 : zoom-out
	400	54-220: pan-left	40-245 : A man walking to the left	77-187 : pan-left 191-214 : pan-left 215-228 : zoom-out 234-243 : zoom-out
Hidden Fury, #1~#2	320	11-294: tilt-up	High textured background	16-25: tilt-up 30-260 : tilt-up 264-283: tilt-up
	240	21-215: pan-left	120-239 : Two talking people	36-176 : pan-left
Wrestling with uncertainty, #1~#5	220	29-38: dissolve 41-187: zoom-out 189-201: dissolve	0-219: Low textured background	30-189: zoom-out
	550	30-139: zoom-out 140-160: dissolve 161-539: zoom-out	0-139 : A large moving net	13-35 : zoom-out 46-137: zoom-out 151-537: zoom-out
	330	20-49: dissolve 50-269: zoom-out 270-300: dissolve	50-269 : A low textured sky	No detected frames
	380	30-50: dissolve 58-331: zoom-out	135-379 : A machine with rotating arms	78-332: zoom-out
	450	31-53: dissolve 54-403: pan-right 404-425: dissolve	A smooth background over the whole sequence	51-351: pan-right

zooms and dissolves, its computation time over the whole sequences is 52 % faster than that of the angle histogram method.

Table 2.7. Comparison of the proposed method for the test set 2 videos with the angle histogram based method.

	Proposed method (r =0.4)	Angle histogram [19] (a =1.0)
N_c	2080	1696
N_m	598	982
N_f	184	514
Recall	77.7 %	63.3 %
Precision	91.8 %	76.7 %

Table 2.8. Comparison of the computation time for the sequences in the test database.

Total no. of sequences in the database	Total no. of frames	Proposed method	Angle histogram
60	20,034	1098.5 sec.	2283.6 sec.

2.5 Conclusions

This work addresses the temporal segmentation of camera motions from video sequences. Motivated by the application of a cooccurrence matrix to image texture analysis, the 2-D motion cooccurrence matrix has been introduced to detect camera motions from video sequences. We exploit the relevance of the 2-D motion cooccurrence matrix to camera motions and analyze the patterns in the cooccurrence matrix associated with specific camera motions.

We have tested the proposed method on test videos that contain various types of motion. The evaluation of the proposed approach demonstrates that it outperforms the angle histogram method [19] for the sequences where the motion due to camera movement is the dominant motion of the sequence. Large moving objects and low textured areas are the main reasons for missed detection. As the proposed method does not rely on the estimation of the parametric motion model that requires high computational complexity, it can effectively detect dominant camera motion. Our method is limited to detect single camera motion at each moment, therefore when a zoom and a pan occur at the same time, the more significant one will be detected. Detection of more various camera motions,

such as translation along vertical (or horizontal) axis or rotation around the optical axis, will be important to extract rich motion content from the video sequences. Future work will focus on the development of these issues.

Chapter 3

Robust Image Matching using Mutual Information and the Graph Search

We aim to integrate scene information, particularly motion due to camera movements, into a video coding framework for the purpose of reducing bit rates of the compressed video. The extraction of scene information consists of identifying camera motions in video sequences and estimating correspondences between the two boundary frames of the detected camera motion shot. In the previous chapter, we presented the camera motion detection algorithm that can detect the types and boundaries of camera motions using the global features in 2-D motion cooccurrence matrices. Since the start and the end frames of a camera motion shot usually have a large temporal distance, finding correspondences between the two frames is a very challenging task. In this chapter, we describe a new robust image matching method that can estimate correspondences between two images under scale changes, rotations, and illumination changes.

3.1 Introduction

Finding correspondences between two images of the same scene taken at a different time or position is one of the fundamental problems in computer vision. Image correspondences are needed in the estimation of depth from images, in the construction of a panoramic image from a collection of images, or in the estimation of motion for compression. To find correspondences between two images, sparse point features such as corners or edges must be first detected by feature detectors in each image. Then, each detected point is described by feature descriptors so that it can be matched to the points detected in the other image.

If two images of the same scene differ largely by a camera's viewpoint, focal length, orientation, or illumination, conventional corner detectors may not detect the same points in both images very well and correlation based similarity measures tend to fail to distinguish the detected points [32]. To solve this problem, many robust image matching techniques based on local invariant features have been proposed in literature. Local invariant features are local information that describe the content of image regions and are invariant to image distortions. Schmid and Mohr [44] used Harris corners as a feature detector and proposed Gaussian derivative based local descriptors. They showed that rotated images can be matched by using the proposed method. Dufournaud et al. [33] proposed a multi-scale framework that is invariant to rotation and scale changes. For two input images with different resolutions, a higher resolution image is smoothed at different scale levels to form the multi-scale representation. To estimate the unknown scale factor between the two input images, initial matching is conducted between the lower resolution input image and each smoothed higher resolution image. After obtaining the approximate scale factor between the two images, the higher resolution image is smoothed to the scale of a lower resolution image and feature points between two images are compared to establish correspondences between them. Their multi-scale representation of feature points enables the matching of two images that have significant scale changes. However, their distance metric requires training data and the matching results depend on the learned distance metric. Lowe [39] proposed scale invariant features named as the SIFT. The SIFT used the local extrema of difference-of-Gaussian in scale-space as feature points. This descriptor is computed from local image gradients around feature points. An experimental evaluation of several different descriptors has been reported by Mikolajczyk and Mohr [41]. In their evaluation, SIFT descriptors obtain the best matching results. However, SIFT descriptors are partially invariant to illumination changes. When the illumination of a scene changes globally or locally, the value of a pixel in an image can also change with time. Examples of source of global illumination changes are shadow, the sun rising or setting, and the turning on or off of a light source. Local illumination changes are typically caused by the motion of an object or camera with respect to the light source. Therefore, robustness to illumination variation has been an important issue in

image matching. To achieve illumination invariance, SIFT descriptors assumed that the corresponding pixel values between two images can be related by an additive factor and a multiplicative factor. Hence the descriptor normalization is used, which is only effective to cancel a brightness change (i.e., additive effect) or contrast change (i.e., multiplicative effect). In addition, SIFT is a local descriptor and the number of mismatches increases when two images have multiple similar regions [42].

In this chapter, we propose a novel approach to find correspondences between two images that employs a mutual information based local descriptor and a graph based global search. The proposed local descriptor is invariant to scale changes, rotations, and illumination changes. By combining the global search with local invariant descriptors, false matches due to the ambiguity of local image features can be reduced.

The remainder of this chapter is organized as follows: Section 3.2 describes the problem of feature matching between two images. Section 3.3 describes the proposed matching algorithm. Section 3.4 presents experimental results. Section 3.5 discusses our conclusion and future work.

3.2 Finding Correspondences Between Two Images

When finding corresponding points between two images of a scene, it is generally assumed that the two images overlap and corresponding image regions are similar. Under this assumption, the type of features (such as corners, edges, lines, etc.) and the similarity measure (such as correlation of image intensities, orientation along edges, lengths of lines) have to be determined [45].

Consider a typical matching scheme between two images, which uses the image corners as feature points and the correlation of pixel intensities as similarity measures. A corner is a point in which intensities of a local neighbourhood of the point change significantly in more than one direction. Corner detectors detect corners in both images. For each detected corner point in the first image, the corresponding one in the second image is

sought by comparing the correlation of the two corner points. The correlation measures the similarity between the image point neighbourhoods.

Fig. 3.1 shows an example of finding correspondences between two images with small translational displacements. The second image was obtained by horizontally translating the first image 10 pixels to the right. To extract feature points in both images, Harris corner detector [35] was used. Two parameters of the detector, the constant a in the measure of corner and the threshold value Th, were set to $a = 0.04$ and $Th = 1500000$. Local windows of 15×15 square around each corner points were used to measure correlations between corner points. We can observe that most of the points detected in the first image are also detected in the other image. For the image pair, 144 correct matches were detected. Fig. 3.2 shows the result of Harris corner matching method when the two images have a large scaling difference. These two images were obtained by using different focal lengths of a camera. The scale factor between the two images is 3. By comparing the two images, we can observe that the corner detector failed to detect the same points in both images. In addition, fixed-size matching windows did not cover the same image regions for the two corresponding points.

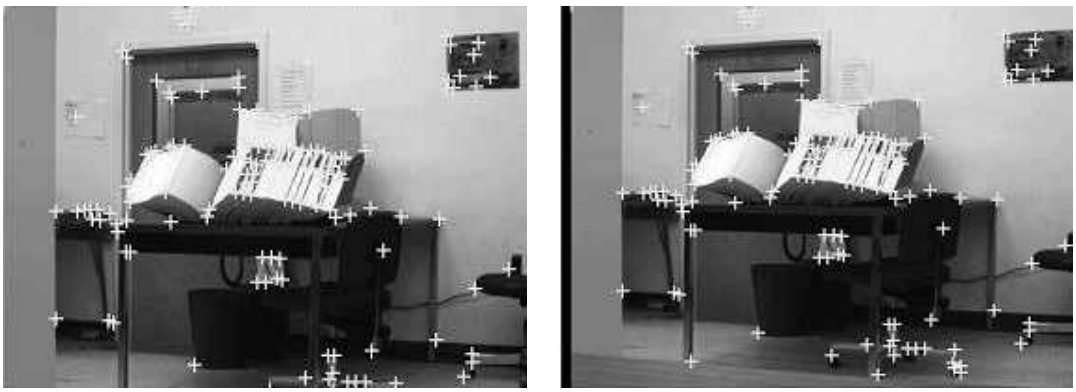


Figure 3.1. Matching of corner points between the two images with small translational displacements. White marks denote correct matching point pairs obtained by Harris detector and the correlation measurement.

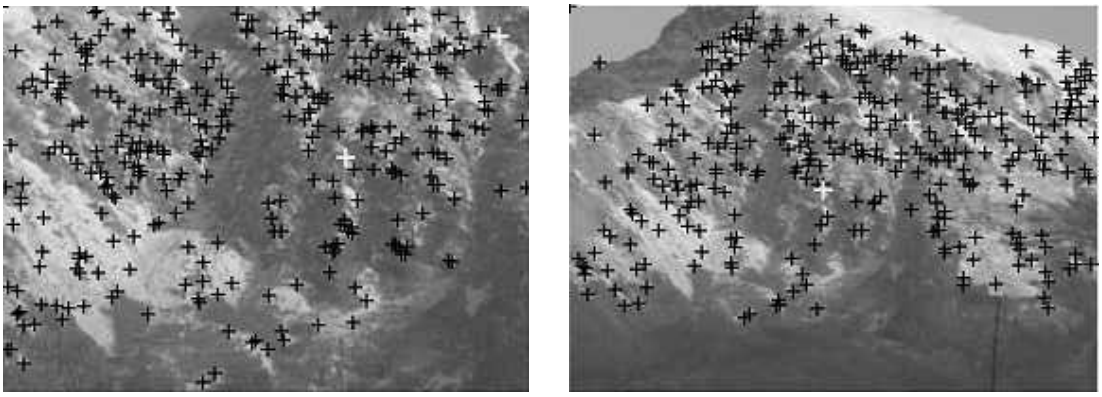


Figure 3.2. Matching of corner points between the two images with large scale difference (scale factor=3). 341 matching pairs (black ‘+’ marks) are detected and only 2 of them are correct (white ‘+’ marks).

3.3 Proposed Image Matching Method

To address the problem of point matching between two images under scale changes, rotations, and illumination changes, we propose a novel matching method. We introduce scale and rotation invariant mutual information as a local descriptor. Although mutual information is invariant to illumination changes [43], it is sensitive to image rotations or scale changes. To achieve scale and rotation invariance, the size and the orientation of the local measurement windows are adaptively determined by using the orientation and scale values of the feature point. The results of the local feature matching may contain some false matches, especially when an image has many similar regions. To further reduce these ambiguity matches, we combine the local feature descriptor with a global constraint through the graph search.

The proposed method consists of three steps: feature points detections, characterization of the feature points, and the search of matching points. To detect the feature points in the images, we use the Scale Invariant Feature Transform (SIFT) descriptor [39]. The SIFT can provide the location of feature points with additional information such as the scale factors of the point. After detecting feature points in both images, each point in one image is compared with all detected points in the other image by using the proposed scale-adaptive mutual information for initial matching. Then, we apply the graph search to the

initial matching candidates and find the largest set of corresponding point pairs. Each part of the proposed method is described as follows.

3.3.1 Feature point detection

The first step in feature point matching is feature point detection. We use the SIFT to detect feature points on both images. The basic idea of the SIFT feature detector is to extract feature points through a sequence of filtering that can localize stable points in the scale-space of an image. The scale-space of an image is a set of images represented at different levels of resolution. To extract the feature points of an image, by the SIFT, peak points are detected in scale-space images. Then, locations of keypoints and gradient orientations of keypoint neighbourhoods are determined. In the first step, the scale-space of an input image, $L(x, y, s)$, is defined as the convolution of a variable-scale Gaussian, $G(x, y, s)$ with an input image, $I(x, y)$:

$$L(x, y, s) = G(x, y, s) * I(x, y) \tag{3.1}$$

where * is the convolution operation in x and y, and $G(x, y, s)$ is the Gaussian function. Scale refers to the standard deviation of the Gaussian function. Then, difference-of-Gaussian (DoG) images, $D(x, y, s)$, are constructed from scale-space images:

$$\begin{aligned} D(x, y, s) &= (G(x, y, ks) - G(x, y, s)) * I(x, y) \\ &= L(x, y, ks) - L(x, y, s) \end{aligned} \tag{3.2}$$

where k determines the scale interval between two adjacent scale-space images. Lowe has experimentally determined $k = 2^{1/3}$ to achieve the highest detection rate of feature points [39]. Fig. 3.3 illustrates scale-space images and DoG images. In order to detect the local peaks (termed as “keypoints”) of the scale-space images, the local maximum and minimum points are searched in DoG images as shown in the right column of Fig.3.3. Each point in a DoG image is compared to its eight neighbours in the current image and nine neighbours in the above and below images. Current point is selected as a keypoint only if it is larger or smaller than all of its neighbours. Once initial keypoints are found,

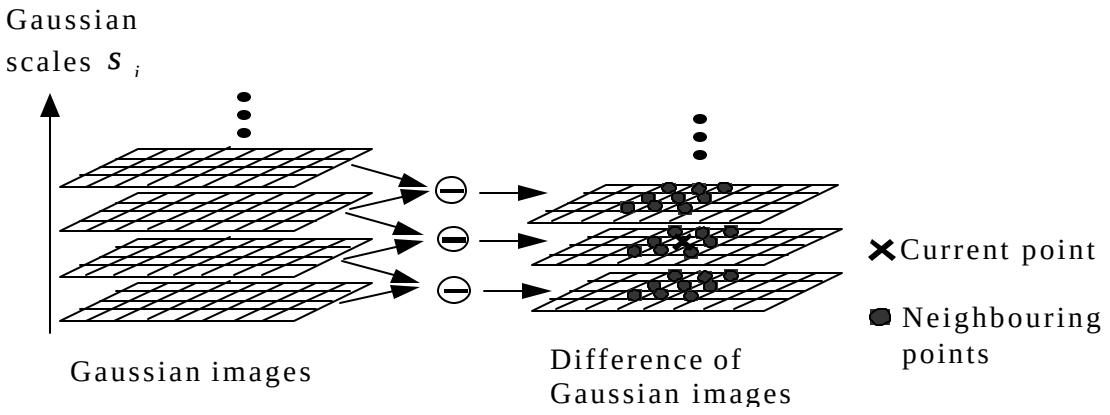


Figure 3.3. Scale-space representation of an image with the difference-of-Gaussian images.

sub-pixel/sub-scale accuracy of the keypoint is achieved by fitting a 3D quadratic function to the current sampled points.

After detecting keypoints of an image, the scale value of a keypoint is defined as the characteristic scale of the keypoint. Characteristic scales are important because they are known to reflect the size of the structure around the keypoint and the ratio of the two matching points' scales between two images is equal to the scale factor between the images [40]. For each keypoint, an orientation is assigned based on local image gradient directions. For a keypoint, an orientation histogram is formed from the gradient orientations of image points within a region around the keypoint. The highest peak and any other local peak within 80 % of the highest peak are assigned as orientations of the keypoint. Therefore, multiple keypoints can be created at the same location and scale but different orientations. The SIFT detector extracts hundreds or thousands of keypoints per image. The large number of feature points may be important for object recognition where small object detection is required. But the task of finding correspondences for image matching does not require such a large point set. To select feature points that are more likely to have correct matches, we have analyzed a set of images and obtained the distribution of correct matching ratio over characteristic scales. For this evaluation, we used 4 images of indoor/outdoor scenes as shown in Fig. 3.4. 'graffiti' and 'fruit' are from the INRIA dataset [37], 'poster' is from [47], and 'library' is from the image set taken by

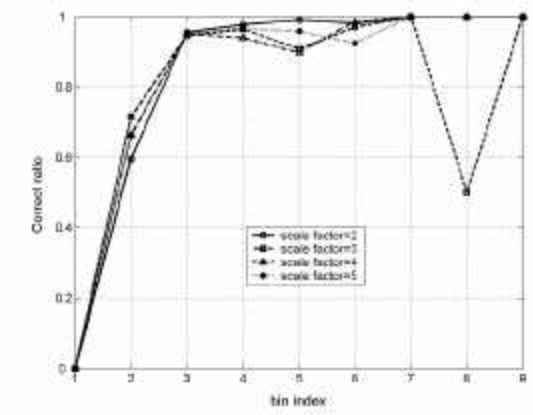
graffiti

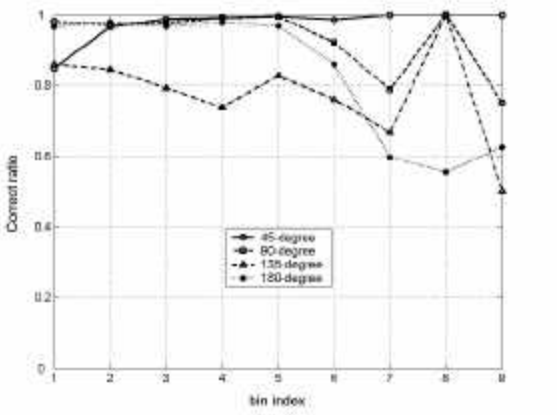
poster

fruit

library

Figure 3.4. Test images for the evaluation of the correct matching ratio.





(a) (b)

Figure 3.5. The average correct detection ratios of the four test images under scale changes or rotations.

Table 3.1. This table shows the bin range of characteristic scale values in Fig. 3.5.

Bin	1	2	3	4	5	6	7	8	9
Range	[0,1)	[1,2)	[2,3)	[3,4)	[4,5)	[5,10)	[10,15)	[15,20)	[20,40)

* $[a,b) = \{x \mid a \le x < b\}$

a digital camera. For each image, we applied a range of transformations that include rotation and scaling. These synthetic transformations enable us to predict precise correspondences and measure the correct matching ratio.

For a pair of images, one is input image and the other is transformed image, the standard SIFT algorithm [39] was applied to detect feature points and find correspondences between feature points. When the SIFT detected matching point was located within a 3-pixel distance from the ground-truth location, that matching pair was counted as a correct match. To measure the distribution of characteristic scale values of matching point pairs, the characteristic scale of each matching point in image I_1 has been assigned to one of the discrete bins as in Table 3.1. Then, for each bin, *Correct ratio* was computed as

Correct ratio = (# of correct matches)/(# of detected SIFT matches) (3.3)

Fig. 3.5 shows two graphs of the evaluation on test images. The first graph shows that the matching accuracy between scaled images decreases quickly for points having characteristic scales less than 3. From the second graph, the matching accuracy between rotated images does not seem to be related to the characteristic scales of the feature points.

3.3.2 Mutual information based feature descriptor

After extracting feature points from each image, the next step is to compute a descriptor for the local image region around the detected feature point. This descriptor should be as invariant as possible to changes such as scale, rotation, and illumination. To achieve illumination invariance, our descriptor is based on mutual information of local image

region. Mutual information has been used extensively as a similarity measure in medical image registration [43]. It has also been used for stereo matching under lighting changes [34, 38]. In conventional mutual information based matching [34, 38], a fixed-size window around an image pixel is used as a measurement window. For a window in a current image, the corresponding one is searched by comparing all local windows within a search range in the other image. The window showing maximum mutual information is selected as a corresponding window. These conventional approaches cannot be used directly for the scale and rotation invariant matching because the mutual information between the two local windows is sensitive to scale changes and rotations. To solve this problem, we propose an adaptive mutual information descriptor that selects the size and orientation of a measurement window based on the characteristic scale and orientation of the feature point.

Mutual information between two random variables depends on entropy and joint entropy of the two random variables. When a fixed-size measurement window is defined in each feature point, the image pixels within the window can be treated as discrete random variables. If we assume X and Y are pixel values of the two windows, the mutual information between X and Y is defined as

$$MI(X,Y) = H(X) + H(Y) - H(X,Y)$$
$$= -\sum_{x \in W1} P(x)\log_2(P(x)) - \sum_{y \in W2} P(y)\log_2(P(y)) + \sum_{x \in W1}\sum_{y \in W2} P(x,y)\log_2(P(x,y)) \quad (3.4)$$

where $H(X)$ and $H(Y)$ are the entropies of X and Y, $H(X,Y)$ is the joint entropy of X and Y, $P(x)$ and $P(y)$ are the probability densities of X and Y, $P(x,y)$ is the joint probability density of X and Y, W_1 and W_2 are the measurement windows centred at feature points in two images. The probability density function is approximately estimated from the normalized 2-D joint histogram of gray-level pixels in the measurement windows. It is given by $P(x,y) = h(x,y)/N$, where $h(x,y)$ denotes the occurrence of intensity pair (x,y) in the two images. N is the total number of pixels in the measurement window. The probability densities $P(x)$ and $P(y)$ can then be obtained by a summation over the row and column directions of the joint density, respectively.

To build a 2-D joint histogram between two local windows, we need to determine the number of bins. If it is set too large, the histogram will become very sparse one and the statistical meaning of the histogram will be reduced. If it is too small, the histogram will not provide enough discrimination power. We set the bin number as 16 in this work. Due to the joint entropy term within Eq.(3.4), mutual information measure is known to be sensitive to the size of overlapping parts of the two images. A normalized mutual information (*NMI*) which is less sensitive to changes in overlap is also defined [43].

$$NMI(X,Y) = \frac{H(X)+H(Y)}{H(X,Y)} \tag{3.5}$$

We use *NMI* as a measure of similarity between two local windows.

Let p be a feature point at (x, y) in the first image with characteristic scale s_p and orientation θ_p and q be a feature point at (x', y') in the second image with characteristic scale s_q and orientation θ_q. To achieve invariance to scale changes, the sizes of two measurement windows are set to $(2\cdot\langle k\cdot s_p\rangle+1)\times(2\cdot\langle k\cdot s_p\rangle+1)$ and $(2\cdot\langle k\cdot s_q\rangle+1)\times(2\cdot\langle k\cdot s_q\rangle+1)$ centred at each feature point. k is a constant that determines the size of the measurement window. The operator $\langle r\rangle$ returns an integer number nearest to r. We experimentally determined the value of k and set k = 4 in our work. To achieve invariance to rotations, window W_2 is rotated clockwise toward W_1 by $\Delta\theta = \theta_p - \theta_q$.

Fig. 3.6 shows the effect of scale and rotation invariant mutual information for the point matching. Fig. 3.6(a) shows a keypoint in the first image whose location is indicated by a '+' mark. The characteristic scale and orientation of the keypoint are 7.07 and 52.7 degrees. The boundary of the local window is shown as a square. For the second image, which is the rotated and scaled image, the SIFT feature detector extracts 1064 keypoints. For each keypoint in the second image, a fixed-size local window is defined. The size of the window is the same as that in the first image.

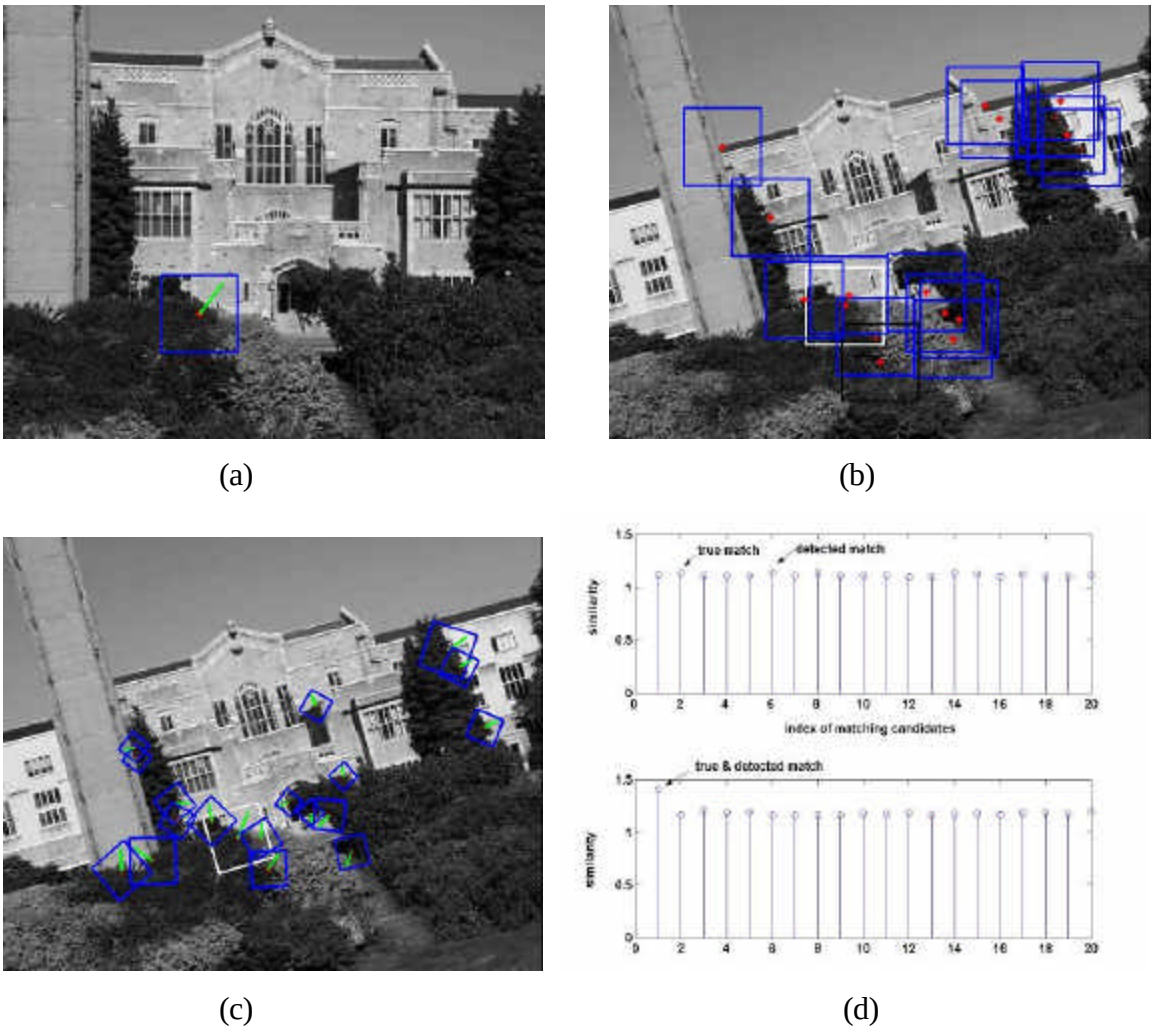


Figure 3.6. This figure shows the effect of invariant MI for the point matching. (a) A keypoint in the first image is marked by ‘+’. The boundary of the measure window and the orientation of the keypoint are displayed as a square and a line, respectively. (b) The boundaries of the 20 matching candidates in the second image are shown. The black square represents the matching window detected by the conventional MI measurement and the white square represents the true matching window. (c) Matching candidates that are detected by the invariant MI measurement are shown. (d) The top and bottom graphs show measured similarities of the 20 candidate points for the conventional MI and invariant MI measurements.

To find the corresponding point, similarities between keypoints are compared by measuring mutual information between local windows. Fig. 3.6(b) shows the 20 candidate matches detected in the second image. The true matching window and the detected matching window are displayed as white and black squares, respectively. Due to image

rotation and scaling, we can see that the content of the white square is different from that of the first image. Fig. 3.6(c) shows the 20 candidate points when the local windows around keypoints were adaptively selected and rotated. The white square in Fig. 3.6(c) and the square in Fig. 3.6(a) cover the same image areas, which results in correct matching. Fig. 3.6(d) shows similarities of the matching candidates. The top graph shows the measured similarities for the 20 candidate points when fixed-size windows are used. The detected matching point, which exhibits peak similarity, is different from the true matching point. The bottom graph shows the measurement results when invariant mutual information is used as a feature descriptor. When the true matching point exhibits the peak value, correct matching can be obtained.

3.3.3 Graph search

Establishing correspondences between image points using only local features may return many false matches. This often happens when both images contain many similar regions. In order to augment local features, a global descriptor is combined with a local descriptor [42]. To capture global properties of a feature point, they computed the 2-D histogram that accumulates the maximum curvatures of image pixels around the neighbourhood of the feature point. The global descriptor is rotation invariant, but it is not scale invariant. Horaud and Skordas [36] showed that the stereo correspondence problem could be solved by exploiting the relationships of local features. Straight lines were extracted from both images. In order to find matches between lines, an association graph was constructed by considering geometrical relationships of line properties (i.e. position, length, and orientation). The association graph represents the relationship between potential matches and line correspondences can be determined by finding the largest set of matches in the graph.

To reduce ambiguities of local feature matching, we use the graph search [36]. Since the complexity of graph building and searching the graph is proportional to the number of nodes in the graph, it is very important to eliminate incorrect nodes as many as possible. In order to detect incorrect nodes, we established a set of rules for our matching method.

Let I_1 and I_2 be the two images. For each feature point in the image I_1, we search a number of initial matches in the other image I_2 using local feature matching. These initial matching pairs constitute the individual nodes of the association graph. The association graph shows the relationship between the potential correspondence pairs and enables the determination of the largest correspondence set. Let $G = (N,E)$ be the association graph, where $N=\{n_{ij},\ i \in [1,\ldots,N_1],\ ,j \in [1,\ldots,N_2]\}$ is the set of nodes and $E=\{e_{sk},\ ,\ s,k \in [1,\ldots,M],\ s \neq k\}$ is the set of edges. N_1 and N_2 are the total number of candidates in image 1 and image 2, n_{ij} is a node representing a match between two candidates, e_{sk} denotes an edge connecting two nodes, and M is the total number of nodes. A node score B_{node} is computed for each node, which takes the similarity (i.e., mutual information in this work) of that node.

After the nodes are built, they are linked by edges based on compatibility between nodes. If two nodes satisfy pre-defined conditions, these two nodes are considered as compatible. Three rules are used to examine node compatibilities. Consider two correspondence point pairs, $(p_1(i),\ p_2(j))$ and $(p_1(m),\ p_2(n))$, and their nodes n_{ij} and n_{mn}. Node n_{ij} is compatible with node n_{mn} if following three conditions are true:

Rule 1: $(i \neq j)$ and $(m \neq n)$
Rule 2: $\|p_1(i), p_1(m)\| \geq \|p_2(j), p_2(n)\|$
where $\|\bullet\|$ denotes the distance between two points.
Rule 3: $\text{Angle}(p_1(i), p_1(m)) \approx \text{Angle}(p_2(j), p_2(n))$

The first rule enforces one-to-one matching between feature points. Our second rule applies a distance constraint to the point pairs. Assume that image I_1 has a higher resolution than image I_2. Then, the spatial distance between two points in the image I_1 cannot be less than the distance between two corresponding points in the image I_2. The third rule reflects the topological constraint. The angle between two points $p_1(i)$ and $p_1(m)$ with respect to the dominant orientation of the neighborhood of the point $p_1(i)$ should not be changed for the corresponding two points in the image I_2 under scaling or rotation.

After building the association graph between the two images, correspondences between points are established by finding the largest set of matching points (i.e. maximal clique) in

the graph. When multiple maximal cliques, $\{C_i\}_{i=1}^{M}$, are found, the best one w(G) is determined by the following evaluation:

$$w(G) = \max_i(\text{Score}(C_i)) \tag{3.6}$$

$$= \max_i(\{\sum_{n=1}^{L} B_{n,i}\}_{i=1}^{M})$$

where L denotes the size of the maximal clique (i.e., the number of nodes) and M represents the total number of maximal cliques. In order to find the maximal cliques of an association graph, we used Mark Simmons' code [48].

3.4 Experimental Results

In this section, we describe the details of a test image set, an evaluation method and the experimental results. We used test images from the INRIA dataset [37] and ALOI image collection [46]. Images taken by a digital camera were also used. To evaluate the performance of the proposed method, we defined the correct matching ratio as in Eq. (3.7).

$$r = \frac{\text{total \# of correct matches}}{\text{total\# of detected matches}} \times 100 \tag{3.7}$$

Here, ' total # of detected matches' represents the number of matching point pairs detected by the algorithm (correct or false) and ' total # of correct matches' denotes the number of correct matches.

To obtain the number of correct matches in each image pair, we need to know the ground truth pixel correspondences between image pairs. For the INRIA dataset, transform matrices between all image pairs are given. Therefore, we use them to automatically identify matching points. If the transform matrix is not given, we define the transform matrix and synthetically create the transformed image pair. We use the transform matrix as follows.

Let T_{12} be the transform matrix that maps a point in image 1 to its corresponding point in image 2. Let p_1 and p_2 be the matching point pair detected by an algorithm. For the point p_1 in image 1, the ground truth matching point p_1' is obtained by $p_1' = T_{12}(p_1)$. We consider the detected matching pair to be correct if the distance between p_2 and p_1' is less than 3 pixels.

We compared the proposed method with the standard SIFT algorithm [39] using the evaluation metric in Eq. (3.7). In both methods, feature points were detected by the SIFT detector. The SIFT detector generates over 2000 keypoints for a typical 500×500 image. As the number of keypoints depends on the image size and content, some input images were converted to smaller sizes before the evaluation. To reduce the number of feature points and select points more likely to have correct matches, only points whose characteristic scales are larger than 3.5 were selected as feature points of image 1. Two types of descriptors were evaluated: the standard SIFT feature descriptor and the proposed invariant MI descriptor. For the proposed method, multiple candidates are selected in image 2 by choosing M most similar feature points in terms of invariant MI descriptors. We set $M = 3$ in our test. Then, final matches are determined by applying the graph search. For the SIFT method, the point pair with the highest similarity score is detected as a match if the ratio of the most similar and the second-most similar points is smaller than the pre-defined threshold [39].

Fig. 3.7, 3.8, and 3.9 show test results for illumination changes. In Fig. 3.7, the right column images were generated by increasing the brightness of the left column images. Using the keypoint detection and selection scheme in Sec. 3.3, 31 and 58 points were extracted from the left and right images, respectively. In Fig. 3.7(a), matching point pairs of the standard SIFT are displayed over the image pair. The SIFT detected 26 matches and all of them were correct. Fig. 3.7(b) shows the results of the proposed method. The proposed method detected 26 matches and all of them were correct.

Fig. 3.8 shows another matching results for brightness changes. Using the keypoint detection and selection scheme, 39 and 236 points were extracted from the left and right

images, respectively. Both the SIFT and proposed method detected 22 matches and all of them were correct. In Fig. 3.9, illumination changes were generated by varying the camera's aperture. 124 and 397 keypoints were extracted from the left and right images, respectively. The proposed method detected 26 matches and 22 were correct matches, while SIFT detected 33 matches and 13 of them were correct. Since the proposed method uses mutual information as a local descriptor, it consistently showed high correct matching ratios under illumination changes. The SIFT method is invariant to a brightness change and contrast change because the descriptor vector is normalized to unit length [39]. Therefore, it did not return false matches for brightness changes. However, the "cars" images contain intensity saturation due to the camera's over-exposure, which can not be effectively compensated by normalization. This illumination change resulted in many false matches.

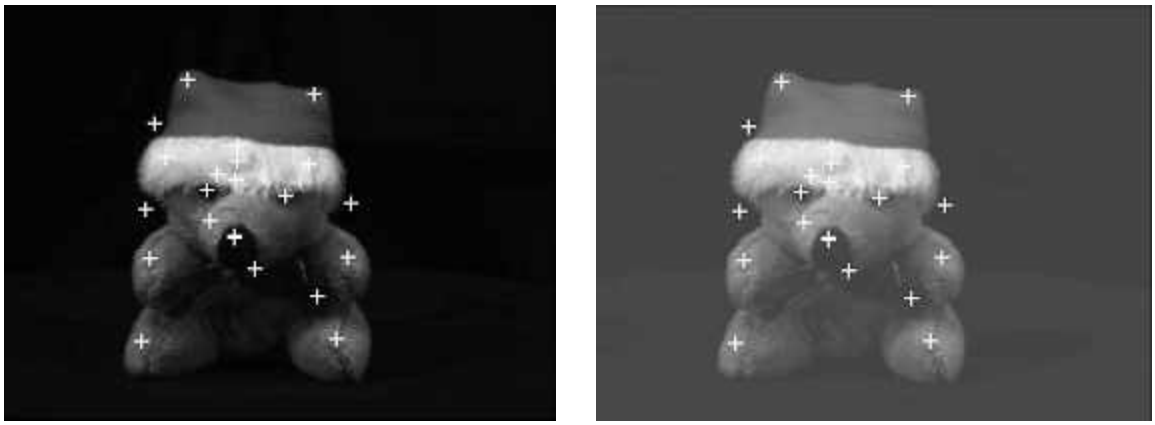
Fig. 3.10 shows matching results for images having many similar regions. The right column images were obtained by rotating the left column images. 52 and 523 points were extracted from the left and right images, respectively. The proposed method detected 33 matches and all of them were correct, while SIFT detected 32 matches and 27 of them were correct. Since the image pair has many areas that are locally similar (such as window structures), false matches increased in SIFT detection. The detection results show that the graph search efficiently removes the ambiguity of local matching. Fig. 3.11 shows another matching results for images having many similar regions. The right column images were obtained by reducing the left column images by a scale factor of 2. 67 and 264 keypoints were extracted from the left and right images. The proposed method detected 34 matches and all of them were correct, while SIFT detected 36 matches and 34 of them were correct.

Fig. 3.12 shows matching results for images under a scale change. The scale factor between the two images is 2.1. Using the keypoint detection and selection scheme, 66 and 323 points were extracted from the left and right images, respectively. The proposed method detected 54 matches and 52 were correct, while SIFT detected 58 matches and 56 of them were correct. Fig. 3.13 shows another matching results for images under a scale

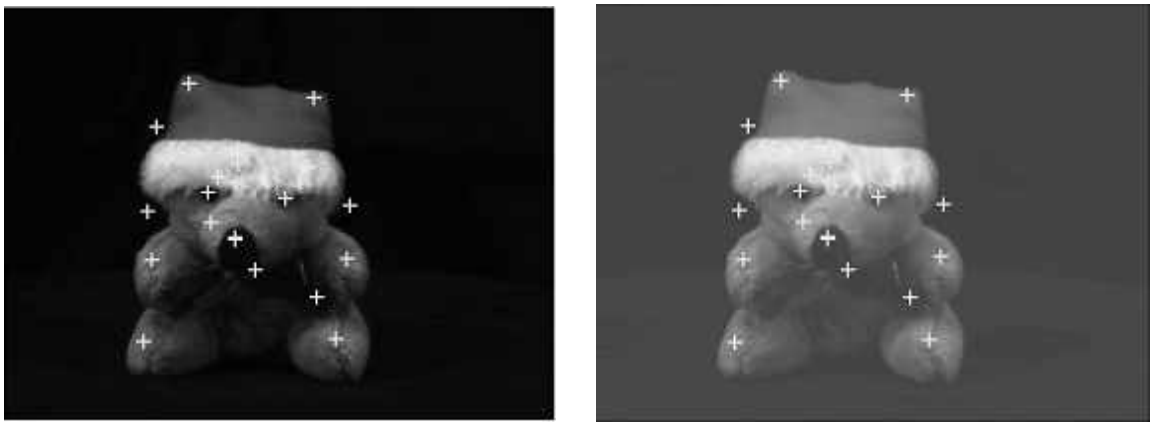
change. The scale factor between the two images is 3.9. Using the keypoint detection and selection scheme, 66 and 305 points were extracted from the left and right images, respectively. The proposed method detected 25 matches and all of them were correct, while SIFT detected 69 matches and 34 of them were correct.

Fig. 3.14 and 3.15 show matching results for rotated images. In Fig. 3.14, the rotation angle between the two images is 28 degrees. Using the keypoint detection and selection scheme, 91 and 1374 points were extracted from the left and right images, respectively. The proposed method detected 66 matches and 64 were correct, while SIFT detected 69 matches and 65 were correct. Fig. 3.15 shows the results for images with a 90 degree rotation. 91 and 1212 keypoints were extracted. The proposed method detected 37 matches and all of them were correct, while SIFT detected 61 matches and 58 of them were correct.

Table 3.2 summarized the detection results for the test images. The third and fourth columns of the table show the number of detected matches and the number of correct matches in each image pair. The last column shows the improvement of correct matching ratios compared to the standard SIFT method. From this table, we can see that the proposed method shows improved matching accuracy for images containing high illumination changes (saturation of intensities) or many similar regions. For image rotations, our method performs slightly better than the SIFT descriptors. For scale changes, the proposed method is able to match images up to the scale factor of 4. For larger scale factors, our method does not find correct matches, while the SIFT can detect correct matches up to the scale factor of 6.2.



(a)



(b)

Figure 3.7. Matching results for "1_i110" image from ALOI under an illumination change. Images in the right column were obtained by adding the constant value (=64) to the gray levels of the left column images. (a) SIFT detected 26 correct matches. (b) Proposed method detected the same number of matches.

(a)

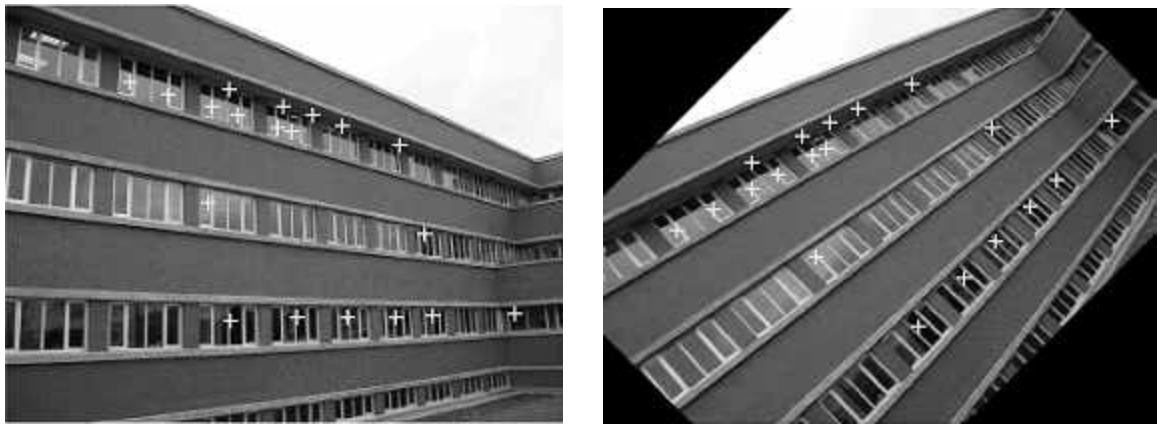
(b)

Figure 3.8. Matching results for "18_i110" image from ALOI under an illumination change. Images in the right column were obtained by adding the constant value (=128) to the gray level of the left column images. (a) SIFT detected 22 correct matches. (b) Proposed method detected the same number of matches.

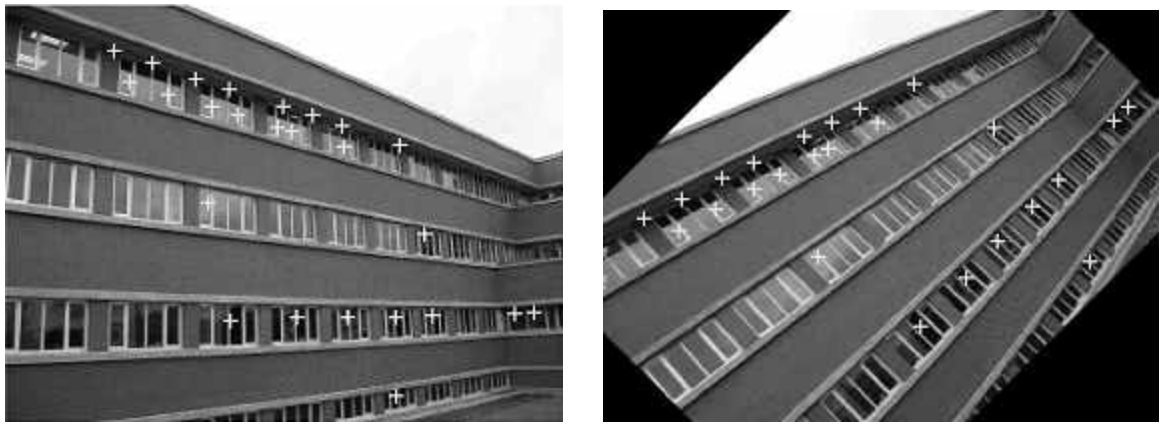
(a)

(b)

Figure 3.9. Matching results for "cars" images from INRIA under an illumination change. Images in the right column were obtained by varying the camera's aperture. (a) SIFT detected 13 correct matches. (b) Proposed method detected 20 correct matches.

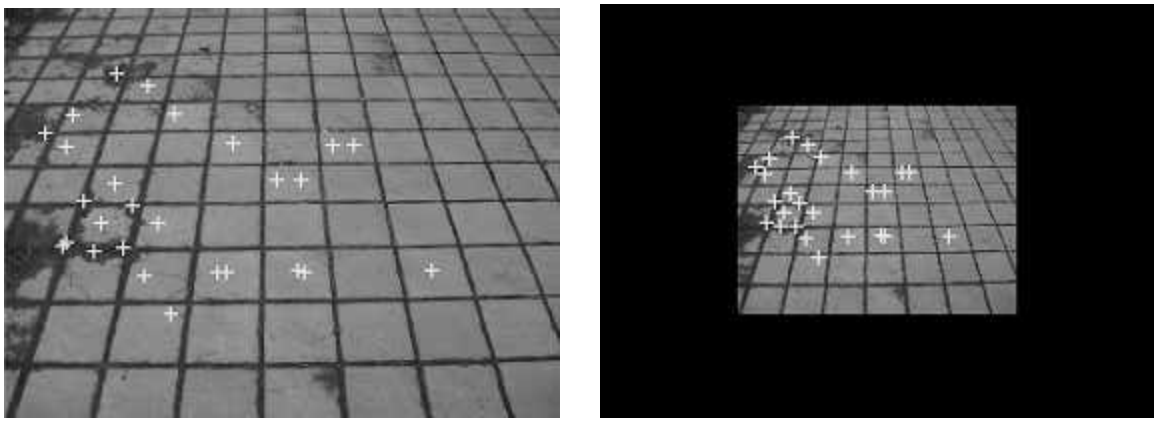


(a)

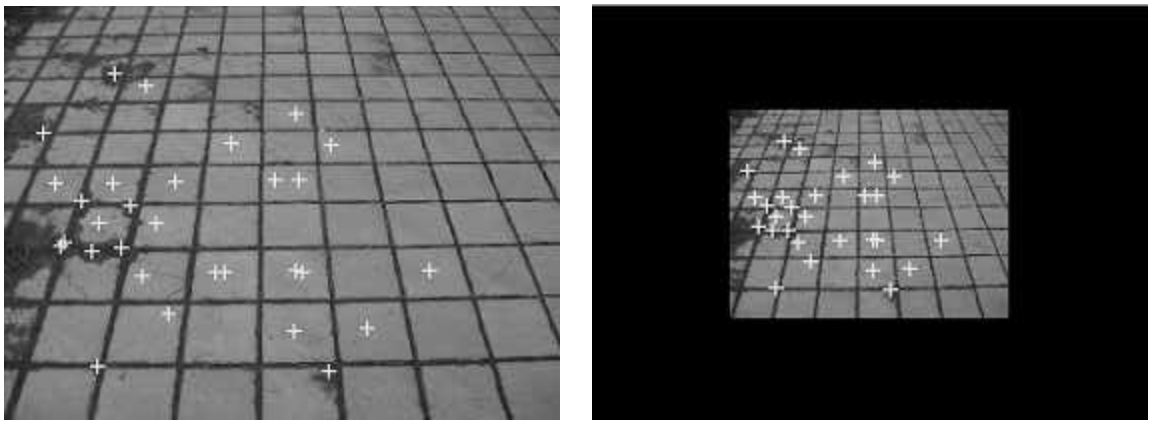


(b)

Figure 3.10. Matching results for “univ1” image taken by the digital camera. Images in the right column were obtained by rotating the left column images by 45 degrees. (a) SIFT detected 27 correct matches. (b) Proposed method detected 33 correct matches.



(a)



(b)

Figure 3.11. Matching results for “univ2” image taken by the digital camera. Images in the right column were obtained by reducing the left column images by a scale factor of 2. (a) SIFT detected 34 correct matches. (b) Proposed method detected 34 correct matches.

(a)

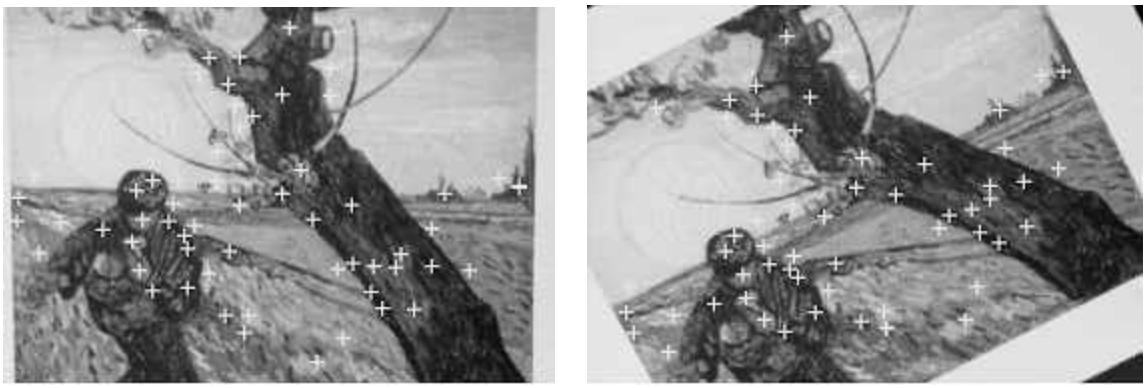
(b)

Figure 3.12. Matching results for “laptop” images from INRIA under a scale cha nge. The scale factor between the two images is 2.1. (a) SIFT detected 56 correct matches. (b) Proposed method detected 52 correct matches.

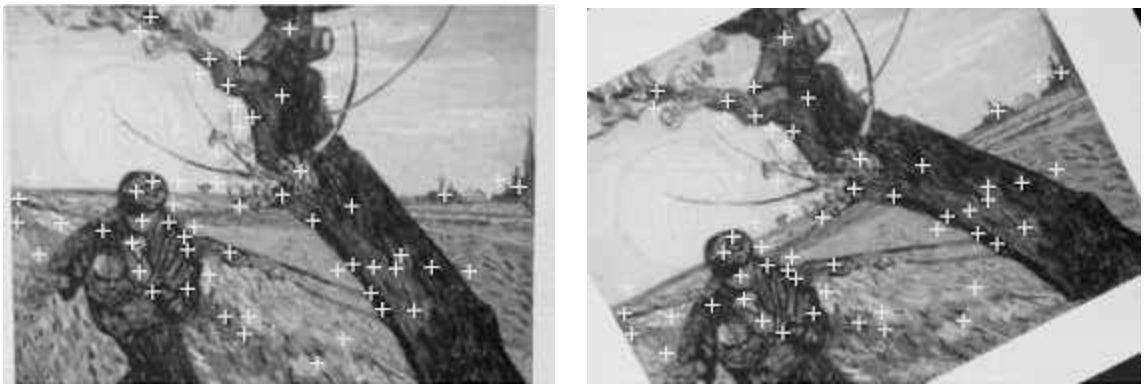
(a)

(b)

Figure 3.13. Another matching results for “laptop” images from INRIA under a scale change. The scale factor between the two images is 3.9. (a) SIFT detected 34 correct matches. (b) Proposed method detected 25 correct matches.

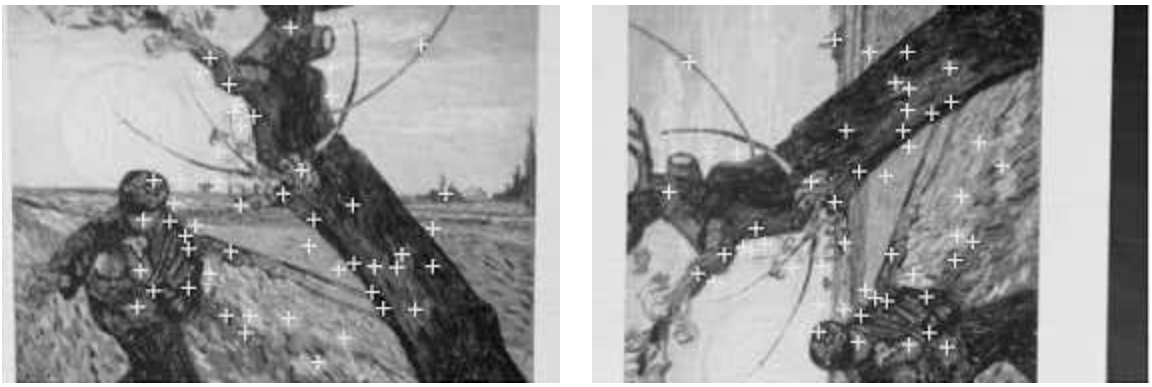


(a)

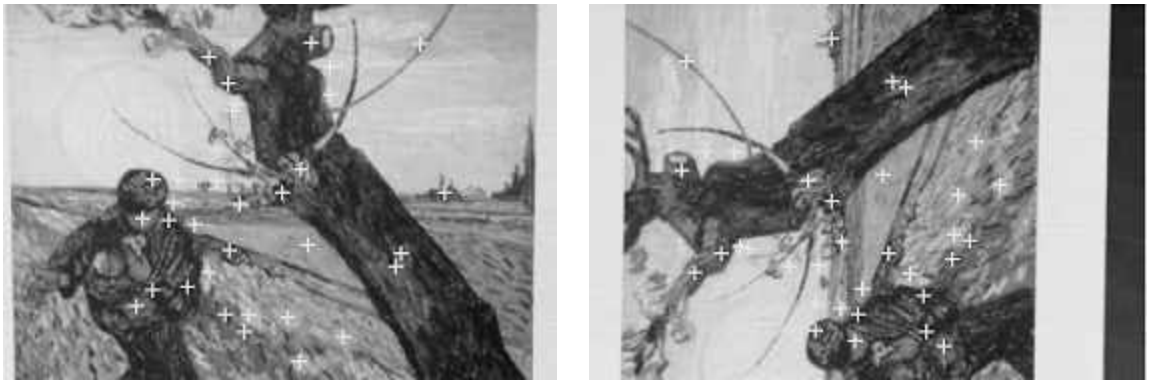


(b)

Figure 3.14. Matching results for “VanGogh” images from INRIA under a rotation change. The angle between the two images is 28 degrees. (a) SIFT detected 65 correct matches. (b) Proposed method detected 64 correct matches.



(a)



(b)

Figure 3.15. Matching results for “VanGogh” images from INRIA under a rotation change. The angle between the two images is 90 degrees. (a) SIFT detected 58 correct matches. (b) Proposed method detected 37 correct matches.

Table 3.2. Comparison of detection results

Images	size	Proposed method (correct /detect)	SIFT (correct /detect)	Δr (%)
1_i110	384×288	26/26	26/26	0.0
18_i110	384×288	22/22	22/22	0.0
cars	461×307	22/26	13/33	+ 45.2
univ1	320×240	33/33	27/32	+ 15.6
univ2	320×240	34/34	34/36	+ 5.5
laptop (#1,#9)	384×287	52/54	56/58	- 0.2
laptop (#1, #17)	384×287	25/25	34/69	+ 50.7
VanGogh(#2,#5)	384×261	64/66	65/69	+ 2.8
VanGogh(#2,#10)	384×261	37/37	58/61	+ 4.9

3.5 Conclusions

In this chapter, we presented a novel method for finding correspondences between two images in the presence of large scale changes, rotations, and illumination changes. We proposed a new local descriptor based on mutual information, and a robust matching strategy. To show the validity of the proposed approach, we have tested our approach on various real images and investigated the invariance of the proposed method. Our test results showed that the proposed method provides improved matching accuracy over the standard SIFT method for the images that contain local ambiguities or large illumination changes. For the images with scale changes or rotations, our method is comparable to the SIFT method. The proposed method has been able to match images which differ by a scale factor of 4.

The measurement of mutual information between local windows requires warping of all the local windows, which is the main reason for the high computational complexity of the proposed method. Efficient implementation of the proposed work will be our future work.

Chapter 4

Metadata for Video Coding

4.1 Introduction

The efficient representation of digital images and video signals has been one of the most important research topics in image processing and video communications. A wide range of applications in the visual communication fields such as video-on-demand, videoconferencing, and digital TV broadcasting have increased the interest in audio-visual coding technology and the need for international audio-visual compression standards. To meet this need, a number of international standardization activities started in the 1980s, which led to several video coding standards [51, 60]: H.263 and H.263+ with the target rate of up to 64 kbits/s; MPEG-1 for digital storage on CD-ROM at up to 1.5 Mbits/s; MPEG-2 for a wide range of audio-visual applications with target rates of 2 ~ 10 Mbits/s; MPEG-4 for more flexible and interactive coding of multimedia data from very low bits (below 64 kbits/s) to 4 Mbits/s.

With the development of digital audio-visual compression techniques, producing multimedia content has become easier and a large volume of audio-visual content is now available. If information about multimedia content becomes available, humans may be able to increase efficiently their ability to organize and search multimedia databases. The need to get information about multimedia content for efficient browsing, searching, and retrieval motivated the development of the multimedia metadata standards [52, 58].

The term metadata is widely used to describe some aspects of data. Metadata is defined as "structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource" [64]. Thus, metadata is often called

information about information [62]. In the library environment, metadata is referred to as a record of resources in a library. Searching library catalogs for particular materials on library shelves is a typical example of metadata use. Another example of the use of metadata is the information attached to MP3 audio files, called ID3 tags. In a regular MP3 audio file, we can easily find information such as title, artist, album, sampling rate or number of channels.

There are many metadata standards developed by various organizations [52]. These standards are focused on specific applications or domains such as digital libraries, digital broadcasting, or digital storage media; however, MPEG-7 focuses on a description of the content that is contained in generic audio-visual data. Previous MPEG standards have been very successful, therefore we can expect that future audio-visual material will be available with content information such as the MPEG-7 description. Although MPEG-7 has been mainly developed to facilitate efficient search, use or management of multimedia content, it can be used for other coding applications.

The remainder of this chapter is organized as follow: Section 4.2 describes an overview of the MPEG-7 standard including main elements of the standard and descriptions related to images/videos. Section 4.3 introduces existing approaches of MPEG-7 metadata based video coding. Section 4.4 introduces the delivery scheme of the MPEG-7 Systems layer and shows how the motion information in chapter 2 and chapter 3 can be encoded and transmitted within the MPEG-7 standard. Section 4.5 presents our conclusion and future work.

4.2 Multimedia Content Description: MPEG-7

The MPEG standards cover different areas. While the MPEG-1, MPEG-2 and MPEG-4 standards focus on the coding of audio-visual materials, MPEG-7 has been developed to describe the content of multimedia data. A framework that enables creation, delivery, and consumption of multimedia data over a variety of networks and terminals is being developed by the MPEG-21 standardization group [59]. In this section, we briefly present four main elements of MPEG-7 and then an overview of the Visual standard [61].

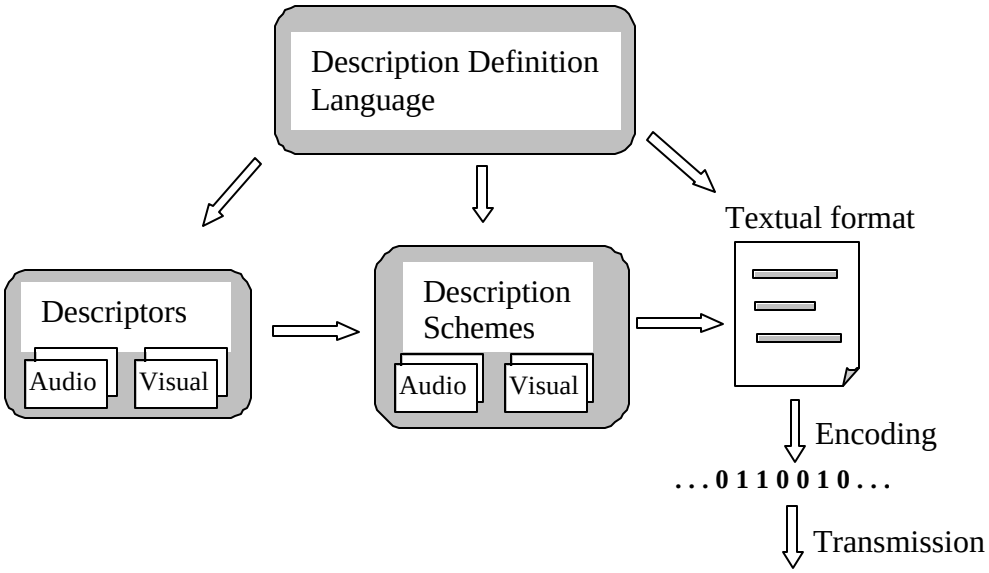


Figure 4.1. MPEG-7 main elements.

4.2.1 Elements of MPEG-7

The MPEG-7 standard consists of Description tools, Description Definition Language (DDL), and Systems, as are shown in Fig.4.1. Description tools create descriptions of multimedia content. They consist of Descriptors and Description Schemes. For computers, it may be difficult to recognize high-level content or event; such as a photo showing a sunset beach or a goal shot in a sports video. Therefore, descriptions based on low-level features are necessary for describing audio-visual content. A descriptor is a representation of a content feature; for example, a time code represents durations, dominant colors and color layouts represent color features. A descriptor also defines the syntax and semantics of the feature representation; for example, the syntax of the color layout descriptor specifies one or two bits to represent the number of coefficients to be used, three bits for the number of luminance coefficients, three bits for the number of chrominance coefficients, five or six bits for each luminance and chrominance component.

On the other hand, a description scheme is defined to describe the content of multimedia materials. Since a multimedia material can be characterized by multiple features, the description scheme specifies the relationship between descriptors and provides the structure and semantics of the relationships. For example, the description scheme of a

movie describes high-level content features related to creation, production, and usage, and also describe signal structures and features. Creation and production information describe the creation and production of the content, such as a title, a creator, a genre, a language and so on. Usage information describes information related to usage rights and usage records. Structural information describes a spatial or temporal portion of the content (e.g., image region, video frame etc.), Table of Content, and so on. Low level audio-visual features can be also used.

Description Definition Language (DDL) is a language that can create MPEG-7 descriptors and description schemes. The MPEG-7 DDL is basically Extensible Markup Language (XML) Schema language [53]. XML is a text based format to represent hierarchical data. A schema specifies the constraints that a valid MPEG-7 description must conform to. The DDL enables MPEG-7 users to create valid MPEG-7 descriptions in textual (XML) form and develop tools for processing descriptions.

The MPEG-7 Systems provides the tools to efficiently represent and transport descriptions. It specifies how to prepare MPEG-7 descriptions for efficient transport/storage, synchronization of content and descriptions, and development of conformant decoders [52]. For efficient representation of descriptions, MPEG-7 Systems defines a binary format which can compress XML documents [58]. The Systems layer defines the architecture of an MPEG-7 terminal that makes use of the MPEG-7 description.

4.2.2 The MPEG-7 Visual standard

The objective of the MPEG-7 Visual standard is to provide descriptions of images or video (called Visual Descriptors) that can help users to search, compare, or filter the content. Visual Descriptors describe audio-visual material on the basis of visual information. Color, texture, shape and motion features are 4 elements that are defined within visual descriptors. For example, a user of a search engine can present an image as a query and receive in return images that have similar colors or textures. Descriptor data

may be located within the associated audio-visual data stream or on the same storage system.

Color Descriptors

Color is one of the most widely used visual features. Color descriptors consist of a number of descriptors, including a color space, a dominant color, and a color layout. The Color space descriptor specifies the color space to be used in the description. Four types of color spaces are supported: monochrome, RGB, HSV, YCrCb, and HMMD. The dominant color descriptor provides how the colors in the image or image region are distributed. For each dominant color, value is expressed by a vector of color components and the percentage of pixels corresponding to the dominant color. The color layout represents the spatial distribution of color in the image or image region. It is known to allow image matching at small computational costs.

Texture Descriptors

Texture contains the structural information of surfaces. Examples of image texture are grass, a bed of flowers, or a pattern on a fabric. A homogeneous texture descriptor describes directionality, coarseness and regularity of texture patterns, and it can quantitatively characterize a texture. In order to describe non-homogeneous texture areas, the edge histogram descriptor has been defined, which captures spatial distribution of edges within an image area.

Shape Descriptors

The shape of an object is a powerful clue for similarity matching. Region shape, contour shape, and 3-D shape have been defined.

Motion Descriptors

Motion is an important feature in video sequences and motion descriptors have been defined in order to capture motion characteristics in video sequences. There are four kinds of Motion Descriptor: Camera motion, motion trajectory, parametric motion, and motion activity. The camera motion descriptor describes the movement of a camera in a scene.

For each sub-shot, in which all frames are characterized by a particular type of camera motion, the camera motion descriptor describes the start time, the duration, the speed of the induced image motion, and the Focus-of-Expansion (FOE) or Focus-of-Contraction (FOC). These parameters can be estimated from image sequences. The motion trajectory descriptor describes the motion of each moving object in a video sequence. The motion of an object is specified by the displacement of the object over time. The descriptor is a list of keypoints, (x, y, t), where x and y are 2-D coordinates of the object position at time t. The parametric motion descriptor describes the motion of objects in video sequences as a 2-D parametric model. Different types of models, such as affine, perspective, and quadratic models are defined to describe the motion of foreground or background objects. The motion activity describes whether a scene is perceived by a viewer as being slow, fast, or action paced. The descriptor describes intensity, direction, spatial distribution, and temporal distribution of activities in video sequences.

4.3 Examples of Metadata-based Coding Applications

4.3.1 Transcoding of audio-visual content

A large amount of compressed audio-visual contents are transmitted over wired, wireless and optical networks. Furthermore, a variety of terminals that have different processing powers are connected to these networks. To support different network characteristics and client terminal capabilities, the same content may be stored in several formats and rates at the content provider's server. To avoid storing the same content in several different formats, a process of transcoding the bit stream from one format to another is required. Due to the fact that many audio and video applications operate in a real-time environment, the complexity of a transcoder is an important issue [13].

To facilitate transcoding in terms of complexity reduction, transcoding hints have been defined by the MPEG-7 standard [15]. Transcoding hints can be used for complexity reduction as well as for quality improvement in the transcoding process. Transcoding hints can be generated in advance and stored together with the compressed audio-visual content, as shown in Fig. 4.2.

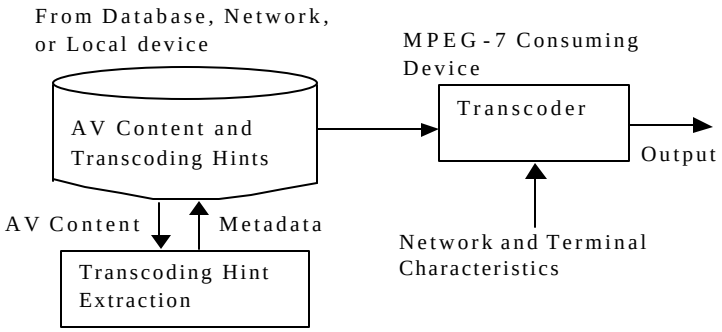


Figure 4.2. Generic illustration of a transcoding application ([13])

Among the transcoding hints, for example, the difficulty hint describes the encoding complexity of each shot within a video sequence. It is extracted by encoding a whole sequence at the smallest quantization step and then calculating the normalized bit rates for each shot. This difficulty hint is a weight, which indicates the relative encoding complexity of each shot. The difficulty hint can be used for the conversion from a Constant Bit Rate (CBR) stream to a Variable Bit Rate (VBR) stream by assigning more bits to a difficult scene and assigning fewer bits to a relatively easy scene.

The motion uncompensability hint describes the amount of newly appeared content per every single frame within a shot and it can be used to help coding mode decisions during the transcoding process. When this hint is near 0.0, the current frame is highly related to the next frame. Therefore, the next frame can be predicted by the Predictional or Bi-predictional mode. When the motion uncompensability hint is near 1.0, little relation exists between the successive frames and the next frame may need to be coded by the Intra mode.

4.3.2 Color Descriptors for the multi-frame prediction mode

Standard video coders such as H.263, MPEG-2 and MPEG-4 achieve high compression efficiency by exploiting temporal redundancy in normal video sequences. To encode a current frame, past or future frames are used as references, and motion compensation is conducted in the encoder and decoder. Recent studies have shown that coding efficiency

can be increased when more than one reference frame is used [65]. Sometimes the best possible reference for a current block may be several frames away from the current frame due to occlusion or illumination changes, etc. Therefore, it is natural to search multiple frames for the best reference block; however, the computational complexity of the motion estimation step also significantly increases [57].

To reduce the complexity of searching multiple frames, a selection scheme using the MPEG-7 color descriptor has been proposed [57]. If a group of candidate reference frames can be selected by using simple similarity measures, the computational complexity can be reduced because the search space is limited to the selected frames. In [57], the MPEG-7 color layout descriptor is used as a measure of frame similarities. This descriptor is designed to specify the spatial distribution of color in an image. An input image is divided into 64 8×8 blocks and each block is represented by its average color. The Discrete Cosine Transform (DCT) is applied to this 8×8 representation and a few low frequency coefficients are selected to form the color layout descriptor of an image. Test results on various video sequences show that bit rates can be reduced by up to 12 % in comparison to the standard H.264/AVC coding scheme.

4.3.3 Adaptive video transmission using Motion Descriptors

The motion activity of a video sequence is considered in the transport layer to transmit videos over unreliable packet networks. In [14], adaptive transmission schemes consider not only the channel conditions but also the visual content. To measure the degree of motion of a video sequence, the intensity of motion activity is extracted from the motion vectors. The MPEG-7 motion activity descriptor represents the intensity of motion by an integer in the range 1~5, where a high value indicates high motion intensity and a low value indicates low motion intensity [56]. After extracting the motion intensity of a video sequence, packets of Bi-predictional frames are dropped according to the channel conditions and motion intensity of the sequence. When a channel's packet loss rate is low, the congestion control algorithm selectively drops Bi-predictional frames that have low motion intensity. When the packet loss rate is high, Bi-predictional frames that have high motion intensity values are dropped. Since sequences of low motion intensity have

smaller bit rates than high motion intensity sequences and since lost frames can be recovered by error concealment, the proposed scheme improves the quality of video over conventional packet dropping methods.

4.4 Coding and Delivery of Metadata

In our metadata-based video coding scheme, metadata about the input video sequence will be used by the encoder and decoder. Consequently, we need to transmit metadata with the video stream to the decoder side. In this section, we first introduce how the MPEG-7 standard defines the delivery of the descriptions about multimedia content in the MPEG-7 Systems layer, and then show how our camera motion related metadata can be coded by the MPEG-7 motion descriptors.

4.4.1 MPEG-7 description delivery

As mentioned in section 4.2, MPEG-7 descriptions about multimedia content are represented in textual formats or binary formats. MPEG-7 description may be delivered independently or with the associated content. Fig. 4.3 shows an illustration of MPEG-7 metadata delivery. Given some audio-visual material to be described, descriptive data should be created in the form of MPEG-7 XML document. Since the techniques for description extraction are out of the scope of the MPEG-7 standard, proper programs or machines should be used to analyze the multimedia materials and create content descriptions. Annotation tools such as *VideoAnnEx* [66] can be used to assist users to add scene descriptions to each segmented video shot and output the descriptions as MPEG-7 XML files. The MPEG-7 reference software [67] provides a set of extraction tools that extract low level features and create the XML description of the input media data. XML descriptions are formatted as a description stream, either text format or binary format, by the description encoder. This description stream consists of a sequence of Access Units, which allow a description to be delivered in a single chunk or in small pieces. Description streams can be delivered over various transmission/storage media; for example, they can be multiplexed into MPEG-2 Transport streams under digital broadcasting environments or Internet Protocol for Internet transmission. The transport of MPEG-7 data on MPEG-2 transport streams shall be done according to the MPEG-2

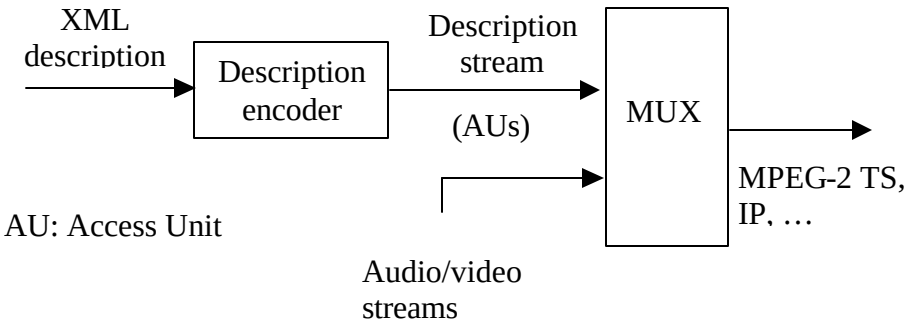


Figure 4.3. Delivery of MPEG-7 descriptions

Systems specification [68]. The transport of MPEG-7 data with MPEG-4 coded audio-visual data can be done by considering MPEG-7 data as MPEG-4 elementary streams [61].

4.4.2 Coding the metadata of the proposed approach

We use two types of content information in our metadata-based coding approach. One is camera motion information in a video sequence and the other is global motion between two boundary images in the detected camera motion shot. In MPEG-7, The Camera Motion Descriptor specifies the camera motion description that is associated with a video sequence. The Camera Motion Descriptor is defined as shown in Table 4.1 [54, 58]. The basic camera operations supported by this descriptor are illustrated in Fig. 4.4. In Table 4.1, the first two parameters, num_segments and descrip_mode_flag, are defined for a given video sequence. As different camera motion types can occur either simultaneously or in succession within a given video sequence, 'description_mode_flag' is used to indicate whether different camera motion types are combined or not [55]. For example, both camera panning and zooming can overlap during a certain period of time. This type of combined motion is called 'MixtureMode' in the descriptor. In 'NonMixtureMode', on the other hand, camera panning can be followed by zooming after the stop in panning motion. The remaining parameters specify the description of each camera motion segment within the sequence.

Table 4.1. Semantics of Camera Motion Descriptor

Name	Definition	Data length
num_segments	Number of camera motion segments	16 bits
description_mode_flag	Flag showing whether the camera motion types are combined or not. ‘0’ for NonMixtureMode, ‘1’ for MixtureMode.	1 bit
start_time	Start time of the camera motion segment (in seconds).	16 bits
duration	Duration of the camera motion segment (in seconds).	16 bits
motion_type	15 motion types are defined: TrackLeft, TrackRight, BoomUp, BoomDown, DollyFwd, DollyBwd, PanLeft, PanRight, TiltUp, TiltDown, RollClockwise, RollAnticountclock, ZoomIn, ZoomOut, Fixed	For ‘MixtureMode’: 16 bits For ‘NonMixtureMode’: 5 bits
fractional_presence []	Temporal fraction of the specified camera motion type within a given video sequence duration.	7 bits for each motion type
amount_of_motion []	Spatial fraction of the image area that is covered due to the specified camera motion type.	11 bits for each motion type

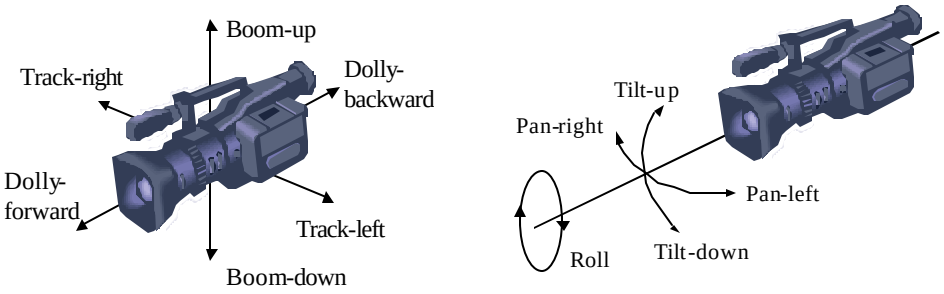


Figure 4.4. Basic camera operations in MPEG-7 Camera Motion Descriptor [15]

Based on the structure of Camera Motion Descriptor in Table 4.1, we can estimate the size of coded metadata for a specific camera motion. Since our camera motion detection method in chapter 2 is designed to detect non-overlapping camera motions, we can calculate the size of coded metadata as follows:

$$\begin{aligned} \text{Bits\_metadata} &= n(\text{num\_segments}) + n(\text{description\_mode\_flag}) + n(\text{start\_time}) + \\ &\quad n(\text{duration})n(\text{motion\_type}) + n(\text{fractional\_presence}[i]) + \\ &\quad n(\text{amount\_of\_motion}[i]) \\ &= 16 + 1 + 16 + 16 + 5 + 7 + 11 \\ &= 72 \text{ (bits)} \end{aligned} \tag{4.1}$$

Here, $n(\cdot)$ represents the number of bits for the corresponding parameter, and i represents the index of a specific camera motion type.

Global motion between two images is estimated for the purpose of relating one image to the other image. In chapter 3, we propose a robust matching method that finds the correspondences between two images. Given a set of corresponding point pairs, global motion parameters between two images can be estimated by robust estimation methods [63]. The MPEG-7 Parametric Motion Descriptor is designed to represent the motion of a region or image by a parametric motion model. The descriptor supports 5 motion models and is defined below in Table 4.2 [54, 58].

Table 4.2. Semantics of Parametric Motion Descriptor

Name	Definition	Data type
motion_model	5 motion models are defined: translational, rotationOrScaling, affine, perspective, quadratic	3 bits
coord_flag	Flag showing whether image coordinates are used or not. ‘1’ for image coordinates, ‘0’ for world coordinates.	1 bit
coord_ref	Reference coordinates (Required when world coordinates are used)	8 bits
x_ori	x coordinate of the origin (Required when world coordinates are used)	32 bits
y_ori	y coordinate of the origin (Required when world coordinates are used)	32 bits
start_time	Time of the first frame (in seconds).	16 bits
duration	Duration of the time interval between two frames where the motion model is applied.	16 bits
parameters[]	2~12 parameters depending on the motion model.	32 bits for each parameter

As we use the 6-parameter affine model as a parametric motion model and image coordinates as reference coordinates, the size of metadata can be calculated as follows:

$$\begin{aligned} \text{Bits\_metadata} &= n(\text{motion\_model}) + n(\text{coord\_flag}) + n(\text{start\_time}) + n(\text{duration}) + \\ &\quad n(\text{parameter[1]}\sim\text{parameter[6]}) \\ &= 3 + 1 + 16 + 16 + 6\times 32 \\ &= 228 \text{ (bits)} \end{aligned} \tag{4.2}$$

4.5 Conclusions

This chapter provided an overview of the MPEG-7 standard and its main components. MPEG-7 consists of standardized tools that enable a detailed description of audio-visual content. Although this standard has primarily been developed to facilitate efficient search, use, or management of multimedia content, it can be used for other coding applications. Thus, we introduced recent approaches that show how metadata can be utilized for complexity reduction, efficient motion prediction, and adaptive video transmission applications. We also discussed encoding and transmission of metadata. Moreover, we showed that the motion modeling parameters extracted in chapter 2 and 3 could be compactly represented by using MPEG-7 motion descriptors.

In a practical MPEG-7 compliant system, the extracted modeling parameters should be coded in MPEG-7 motion descriptors and delivered through existing transport schemes such as MPEG-2 Systems or Internet Protocol. The implementation and test of the MPEG-7 compliant delivery will be our future work.

Chapter 5

Enhanced Video Coding based on Metadata

5.1 Introduction

The progress of digital image and video compression technologies has enabled the efficient representation of image and video signals for storage or transmission. Image compression makes use of the spatial correlation among neighboring pixels in the image and achieves high compression. Video compression exploits the temporal correlation between succeeding frames as well as the spatial correlation within the same frame. Most video coding standards such as MPEG-1, MPEG-2, MPEG-4, H.263, and H.264 aim at removing the redundancy in spatial and temporal domains without significant loss of quality. At the encoder side, the current frame is predicted from the previously decoded frames. The motion information and residual data between the current and predicted frames are sent to the decoder for reconstruction. Using the transmitted motion and residual data, the decoder reconstructs the input video. These video coding standards utilize the redundancy of the video signal without knowledge of the scene content.

To achieve improved coding efficiency, content information of the scene can be used. Model-based coding methods use 3-D models and model parameters to specify the shape, texture, and motion of the objects in the scene. This method has been studied in the field of videoconferencing where a 3-D model of the human face is used. In [7], the model-based codec has been incorporated into the H.263 video codec. The model frame is generated from the 3-D head model and used as a second reference frame for block-based motion compensation. For the blocks that are well approximated by the model frame, only model parameters are transmitted at low bit rates. However, the model-based method requires an *a priori* known 3-D model at the encoder and decoder, which is not available

for general video sequences. In [76], a 3-D depth model of a scene has been estimated from the input video sequence for the purpose of efficient video coding. An input video is divided into sub-sequences (a sub-sequence is called a group of pictures). For each sub-sequence, a 3-D depth model is extracted using the first and last frames of the sub-sequence. Using the 3-D model and two key frames (i.e. the first and last frames), all intermediate frames of the sub-sequence can be reconstructed at the decoder.

Region-based coding method describes a video frame by a set of regions, which are more associated with physical regions in the scene. This approach allows the coding of each region with different quality by distributing more bits to the region of interest [8, 9]. For instance, in typical videoconference scenes, more bits can be assigned to the foreground object than the background.

Along with the digital video compression technologies, the availability of a large amount of audio-visual content has driven the need for tools that can be used to index, search, and manage audio-visual data. This motivation has resulted in the metadata standards such as MPEG-7 [52]. It can be expected that multimedia content will be available with the metadata that describes the content [11, 12]. Although these metadata standards aim to provide detailed descriptions of audio-visual data for content management and exchange, the knowledge of the scene content can be used to improve coding efficiency and strategy. In [71], the authors model faded transition shots by a linear scaling of intensities and show that over 50 % bit-rate savings can be achieved inside the transition periods of the test sequences. In [12], metadata such as texture and motion descriptors are used to pre-select multiple reference frames for long-term temporal prediction mode or to provide motion activity of the frames for improved rate control.

In this section, we propose a novel coding method that incorporates the MPEG-7 motion descriptors into the standard video coding framework. In contrast to the model-based coding, the proposed system does not require *a priori* models of objects. Instead, the system estimates modeling parameters from input images and uses them for the reconstruction of input images. The basic assumption is that the motion between two

shot-boundary images can be represented by a geometric transformation and the in-between images can be approximated by using the two boundary frames and the geometric transformation. By removing the frames in specific shots and representing them with compact modeling parameters, we show that the content information can help to reduce the bit rate of the compressed bit stream compared to the standard video coding scheme. In this work, we focus on the modeling and reconstruction of camera zoom shots in video sequences.

The rest of this chapter is organized as follows: In 5.2, we describe the proposed coding system; in 5.3, experimental results are presented; discussion and conclusion are included in 5.4.

5.2 Proposed Coding System

In this work, our goal is the bit-rate reduction of compressed video sequences while maintaining the visual quality of the decoded sequences. To achieve this goal, we exploit the scene content information in terms of efficient background coding. In our proposed coding system, as shown in Fig. 5.1, the scene content information has been incorporated into the video codec. At the encoder side, a module called the camera motion detector detects camera motions and locates their shot boundaries from the input video sequence. For each detected camera motion shot, two boundary frames are encoded and transmitted to the decoder. In addition, the image-matching module finds correspondences between two boundary fames and outputs a geometric transformation matrix. The intermediate frames between the two boundary frames are not transmitted to the decoder. To reconstruct the intermediate frames, we also send metadata consisting of the camera motion information and geometric transformation matrices. All other frames in the input sequence are coded as normal coded frames. Using the transmitted boundary frames and the modeling parameters, the decoder reconstructs the skipped intermediate frames by a geometric transformation of the two boundary frames.

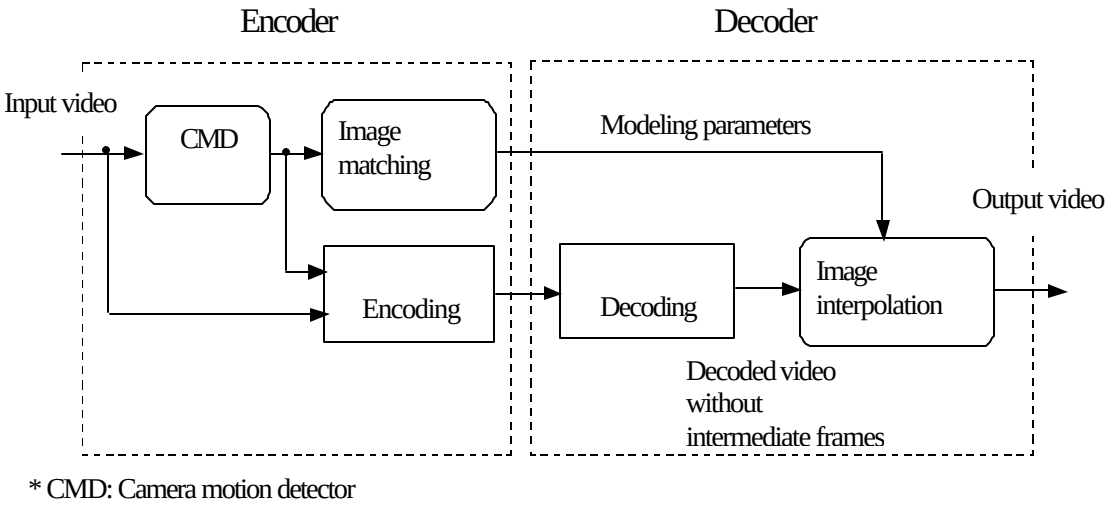


Figure 5.1. Block diagram of the proposed coding system

5.2.1. Camera motion detection and analysis

At the encoder, the camera motion detector analyzes the input video and detects the types and boundaries of the camera motions within the input sequence. Our implementation of the camera motion detector is based on the second-order statistics of motion fields between consecutive images. For two consecutive video frames, the camera motion detector computes the motion field between the two frames and constructs the 2-D motion cooccurrence matrix from the motion field. For each motion cooccurrence matrix, global feature parameters are extracted and different types of camera motions are detected by thresholding the feature parameters. The output of the camera motion detector, for each detected camera motion, consists of starting frame's number, ending frame's number, and the type of camera motion. Details are described in chapter 2.

After detecting the camera motions in the input sequence, the global motion between two boundary frames of the detected camera motion shot has to be estimated. Since typical camera motions last over hundreds of frames, the corresponding regions in the boundary images may contain illumination and scaling changes as well as large spatial displacements. Therefore, conventional feature matching that consists of corner detection and correlation measurement tends to fail to find correspondences between two images

[32]. In this work, we use a robust image matching to find correspondences between the two boundary images. The image-matching module consists of several steps. First, scale invariant feature points are detected in both images. Then, point correspondences between two images are obtained by using a mutual information based similarity measurement and the graph search. Details of finding correspondences between feature points are described in chapter 3. Finally, to estimate the global motion between two boundary frames, a 3×3 geometric transformation matrix that is estimated from the corresponding point pairs. The 3×3 transformation matrix defines a mapping between image points in the boundary frames. As the obtained correspondences may contain incorrect matches, we use RANSAC algorithm [73] that can estimate a geometric transformation from a set of observed data containing outliers. Outliers are data points that do not fit the estimated model. Given a set of corresponding points, the procedure of estimating the geometric transformation matrix using RANSAC can be summarized as follows:

$N_{iteration} = 0$; $H_{best} = 3\times3$ null matrix;

i) Randomly select three point pairs from the data and generate a current transformation from these three point pairs.

ii) Find all point pairs that fit the current transformation matrix.

iii) Estimate a new transformation matrix using the found inliers.

iv) Compare the new transformation matrix with H_{best} and updates H_{best}.

v) Increase $N_{iteration}$.

iv) Repeat i)~v) until $N_{iteration}$ reaches the pre-defined number.

vi) Output H_{best}.

5.2.2. Coding and decoding

Table 5.1 describes the encoding and decoding procedures of the proposed coding scheme. After detecting camera motions in the input sequence, the encoding module encodes input video frames to generate a compressed video bit stream. When the current frame is the start frame of the detected camera motion shot, it is coded as Intra-frame mode. All the following intermediate frames are coded as skipped frames until the end frame of the camera motion shot. For skipped frames, only picture header information is transmitted to

the decoder, which enables high bit-rate reductions. For the end frame of the camera motion shot, Intra-frame mode is used. The extracted modeling parameters, i.e. shot boundary information and the transformation matrix, are transmitted to the decoder as side information.

At the decoder, the decoding module decodes the transmitted bit stream and outputs a decoded video. If camera motions are present in the input video, the decoded video may contain skipped frames. Therefore, the image-interpolation module is called and the skipped frames are reconstructed by using the two decoded boundary frames and modeling parameters. The procedure of image interpolation will be described in the next section.

Table 5.1. Encoding and decoding procedures of the proposed scheme

```
/* Encoding procedure */
 1: Read one input frame;
 2: If the current frame is the boundary frame of a camera motion shot,
 3:       encode the current frame by Intra-frame mode;
 4: else if the current frame is the intermediate frame of the camera motion shot,
 5:       set the current frame as skipped mode;
 6: else
 7:        do normal coding;
 8: Repeat 1~7 until the end of the sequence;

/* Decoding procedure */
 1: Find the start code of a new frame;
 2: If the current frame is a skipped frame mode,
 3:       fill the current frame with the previously decoded frame;
 4: else
 5:       do normal decoding;
 6: Repeat 1~5 until the end of the bit stream;
 7:
 8: Read modeling parameters;
 9: Reconstruct the intermediate frames by image interpolation;
10: Replace the skipped frames with the generated frames;
11: Repeat 8~10 for the next camera motion;
```

5.2.3. Reconstruction of the camera motion shot

To reconstruct the intermediate frames from the two boundary frames and their transformation matrix, we need to know the transformation matrices that relate the two boundary frames to the intermediate frame. If we assume that the speed of the camera movement is constant over time, then the transformation matrix of an arbitrary intermediate frame can be estimated by linearly interpolating the given transformation matrices of the two boundary frames.

Suppose that we have a transformation matrix, $\boldsymbol{H}$, which maps points in the image $\boldsymbol{I_S}$ into points in the image $\boldsymbol{I_E}$ as shown in Fig. 5.2. For an image $\boldsymbol{I_i}$ between $\boldsymbol{I_S}$ and $\boldsymbol{I_E}$, we want to estimate an intermediate transformation matrix, $\boldsymbol{H_i}$, which maps the image $\boldsymbol{I_S}$ into the image $\boldsymbol{I_i}$. When the scene projected into the two images is planar, spatial mapping between the two images can be approximately represented by an affine transformation [74]. Let $\boldsymbol{H}$ be a general 3×3 affine matrix which consists of translation, rotation, shear, and scaling in homogeneous coordinates. Homogeneous coordinates allow affine transformations to be represented by a matrix. Furthermore, a series of affine transformations can be combined by matrix multiplications. As a first step of the matrix decomposition, a translation component $\boldsymbol{T}$ can be extracted as the left factor of the remaining matrix $\boldsymbol{M}$ as in Eq. (5.1).

$$\boldsymbol{H} = \boldsymbol{T} \cdot \boldsymbol{M} \tag{5.1}$$

where $\mathbf{H} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}$, $\mathbf{T} = \begin{bmatrix} 1 & 0 & a_{13} \\ 0 & 1 & a_{23} \\ 0 & 0 & 1 \end{bmatrix}$, $\mathbf{M} = \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

Since the matrix $\boldsymbol{M}$ contains a rotation component, direct linear interpolation of $\boldsymbol{M}$ will introduce the distortion in the interpolated image [70]. To avoid the distortion, we use polar decomposition [70], which decomposes the matrix $\boldsymbol{M}$ into two components, a matrix $\boldsymbol{Q}$ that contains a rotation component and a matrix $\boldsymbol{S}$ that contains scaling and shear components as shown in Eq. (5.2).

$$\boldsymbol{H} = \boldsymbol{T} \cdot \boldsymbol{Q} \cdot \boldsymbol{S} \tag{5.2}$$

where

$$\boldsymbol{Q} = \begin{bmatrix} \cos(q) & -\sin(q) & 0 \\ \sin(q) & \cos(q) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{S} = \begin{cases} \begin{bmatrix} s_1 & s_2 & 0 \\ 0 & s_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \text{when a shear along the x axis is present} \\ \text{or} & \\ \begin{bmatrix} s_1 & 0 & 0 \\ s_2 & s_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \text{when a shear along the y axis is present} \end{cases}.$$

The term q in the matrix $\boldsymbol{Q}$ represents the counter-clockwise rotation angle about the origin and the term s_2 in the $\boldsymbol{S}$ matrix represents the proportionality constant. Since the $\boldsymbol{S}$ matrix can be further decomposed into two components, a scaling matrix $\boldsymbol{C}$ and a shear matrix $\boldsymbol{A}$, it can be expressed as

$$\boldsymbol{S} = \boldsymbol{C} \cdot \boldsymbol{A} \tag{5.3}$$

where $\boldsymbol{C} = \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\boldsymbol{A} = \begin{bmatrix} 1 & s_2/s_1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ or $\begin{bmatrix} 1 & 0 & 0 \\ s_2/s_3 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ for a horizontal or vertical shear.

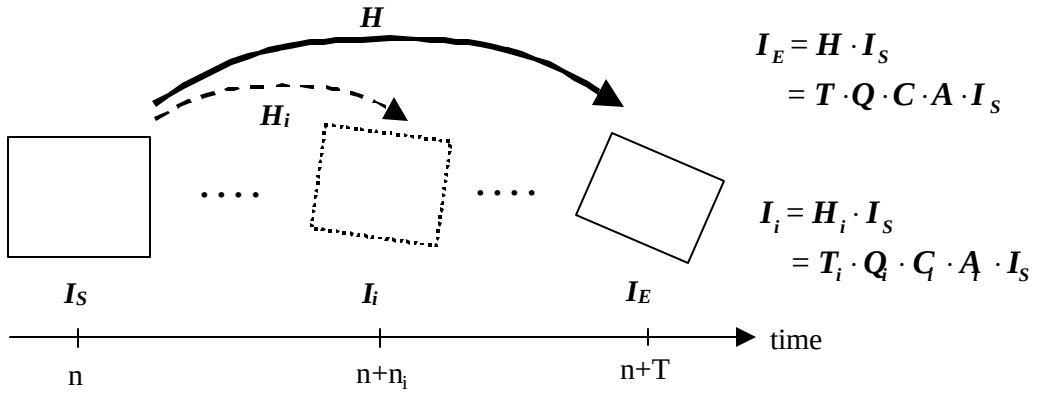


Figure 5.2. Computation of the transformation matrix between images $\boldsymbol{I_S}$ and $\boldsymbol{I_i}$. Let $t = n_i/T$ be a temporal distance ratio between images $\boldsymbol{I}_i$ and $\boldsymbol{I_S}$, and $\boldsymbol{I}$ be a 3×3 identity matrix. Then, the translational component $\boldsymbol{T}_i$ can be obtained by $\boldsymbol{T}_i = (1-t) \cdot \boldsymbol{I} + t \cdot \boldsymbol{T}$. Other components at time = $n+n_i$ can be obtained similarly.

Now we can calculate any intermediate matrices by weighted interpolation of the given transformation matrices. For example, suppose we have a rotation component $\boldsymbol{Q}$ of a transformation matrix $\boldsymbol{H}$. Then, $\boldsymbol{Q}_i$ for an intermediate frame i can be obtained by linear interpolation between an identity matrix $\boldsymbol{I}$ and $\boldsymbol{Q}$.

$$\boldsymbol{Q}_i = (1-t)\cdot \boldsymbol{I} + t\cdot \boldsymbol{Q} \tag{5.4}$$

where $t = n_i/T$ denotes the temporal distance ratio between the first boundary frame and the intermediate frame i.

Similar to the above forward mapping steps, for any intermediate frames, we can calculate backward transformation matrices. We warp the two boundary frames and blend the two warped images to reconstruct an intermediate frame.

5.3 Experimental Results

In this section, we describe details of test sequences, test conditions and experimental results of the proposed coding scheme. To evaluate the performance of the proposed coding method, we have used five video sequences: one was captured by a digital camcorder and others were converted from the documentary videos. The length of each sequence, image size, and shot boundaries identified by manual segmentation are summarized in Table 5.2. The video sequences are in YUV format and the frame rate is 30 frames/sec.

Table 5.2. Test video sequences

Seq. name	Frame size	Length (frame)	Manual segmentation	Detected boundaries (length)
gallery	352 × 288	160	31 – 148: zoom-out	29 – 147 (119)
docu1	352 × 240	335	90 – 308: zoom-out	92 – 283 (192)
travel1	352 × 240	220	23 – 158: zoom-in	53 – 150 (98)
museum1	352 × 240	150	25 – 97: zoom-in	27 – 89 (63)
museum2	352 × 240	140	9 – 116: zoom-in	11 – 105 (95)

For each detected zoom shot, the image-matching module finds corresponding points between two boundary frames and estimates the 3×3 geometric transformation matrices from the point pairs by using the robust estimation technique. We used the RANSAC algorithm [73] in this test. Fig. 5.3 shows the results of the image matching for the first zoom-out shot, 'gallery'. Detected boundary frames are shown in Fig. 5.3 (a) and (b). Fig. 5.3 (c) shows the estimated transformation matrix **H**, which is a forward transformation matrix that maps the start boundary frame (i.e. frame 29) into the end boundary frame (i.e. frame 147). The white rectangle in Fig. 5.3 (c) represents the boundary of the frame 29 after being warped by the matrix **H**. Fig. 5.4 shows the results of the image matching for the second zoom shot, 'docu1'.

We have integrated the camera motion detector, the image-matching module, and the image interpolator with the standard MPEG-4 video codec [69]. To evaluate the performance of the proposed method, we compared our method with the MPEG-4 coding system in terms of bit rates, computation complexity, objective quality and subjective quality. In our experiment, the standard MPEG-4 codec encodes the first frame of the input sequence by Intra (I) mode and all the following frames by Predicted (P) mode. This standard coding mode will be designated "MP4". For the efficient coding of the global motion in a video scene, MPEG-4 supports Global Motion Compensation (GMC) mode. When GMC is used, global motion parameters between the current frame and reference frame are estimated. For each macroblock, the current macroblock is predicted by both global motion compensation mode and local motion compensation mode. Then, the encoder selects either GMC or local motion compensation, whichever gives fewer prediction errors. This coding mode will be designated "MP4 GMC". When our proposed method is applied to the same sequence, we encode the first and the last frames of the detected zoom shots by Intra mode. All the intermediate frames are coded by Frame-Skip mode. Our coding scheme will be designated "Proposed".

The amount of the coded bits in the proposed coding mode includes side information. In this implementation, we assume that each element of the 9-parameter transformation

matrix, boundary frame number, and camera motion type is expressed by a 32-bit float number, a 16-bit integer number, and a 3-bit number, respectively. The size of the side information for each zoom shot becomes

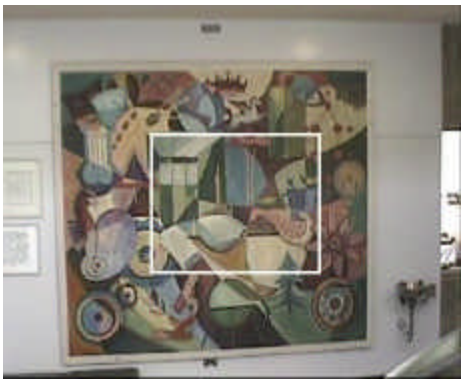
$$\text{Side information} = 9\times 32 + 2\times 16 + 3 = 323 \text{ (bits)} \qquad (5.5)$$

In a practical MPEG-7 compliant system, one would create the content information in MPEG-7 motion descriptor formats and multiplex the descriptors with the coded video streams for the transmission as described in chapter 4.

Fig.5.5 shows the sizes of the compressed bit streams of three coding schemes. For each coding scheme, three quantization parameters (QP=8, 16, and 24) have been tested. The top and bottom graphs show the total bit rates for 'gallery' and 'docu1' sequences, respectively. For each detected camera zoom shot, only two boundary frames and motion modeling parameters are sent to the decoder. Therefore, we are able to reduce the bit rate significantly. As Fig. 5.5 illustrates, the size of the compressed sequences are much smaller than those of the compressed sequences of MPEG-4 and MPEG-4 with GMC. The size of the compressed bit streams and reduced bit rate for all test sequences are summarized in Table 5.3 and Table 5.4. The results in these tables show that the ratio of bit-rate saving of 52 % to 83 % are obtained for the test videos and the ratio is closely related to the proportion of zoom shots to the total length of a sequence.

To evaluate the subjective quality of the proposed method, reconstructed images are compared with the original ones. In Fig. 5.6, the original images, the MPEG-4 decoded images, and the images reconstructed by the proposed method are shown. Top and bottom rows are the results for frame 40 and 90 of 'gallery' sequence. For visibility, parts of the images are shown. During the reconstruction of the skipped intermediate frames within the detected zoom shots, the speed of camera motions are assumed to be constant. In real videos, the speed of camera motions is not constant. Hence, the inter-frame motion derived by the linear interpolation of the motion between boundary frames may differ from the real motion. This motion prediction error results in geometric distortion as

(a) Start boundary image



(b) End boundary image

$$X_2 = \mathbf{H} \bullet X_1$$

$$\text{where } \mathbf{H} = \begin{bmatrix} 0.3663 & -0.0034 & 115.6876 \\ -0.0014 & 0.3639 & 100.8673 \\ 0.0000 & 0.0000 & 1.0 \end{bmatrix}$$

X_1: coordinates in the start image,

X_2: coordinates in the end image.

Figure 5.3. Results of the image matching for ‘ gallery’ sequence.

(a) Start boundary image

(b) End boundary image

$$X_2 = \mathbf{H} \bullet X_1$$

$$\text{where } \mathbf{H} = \begin{bmatrix} 0.4238 & -0.0041 & 183.8872 \\ -0.0068 & 0.4234 & 6.0892 \\ 0.0000 & 0.0000 & 1.0 \end{bmatrix}$$

X_1: coordinates in the start image,

X_2: coordinates in the end image.

Figure 5.4. Results of the image matching for ‘ docu1’ sequence.

shown in Fig. 5.6(c). Geometric distortion means scaling changes, translational changes etc. between original images and decoded images. The visual quality of the reconstructed frames is comparable to those of the MPEG-4 coding.

To evaluate the objective quality, the Peak Signal-to-Noise Ratio (PSNR) values of the reconstructed sequences are measured. For each frame of the reconstructed sequence, luminance PSNR value is measured with respect to the original input sequence. By averaging the PSNR values of the sequence, the average PSNR value is obtained at a given quantization parameter. Fig. 5.8 shows the average PSNR values of the decoded sequences. The top and bottom graphs show the plots for ‘gallery’ and ‘docu1’ sequences, respectively. For these two sequences, the average PSNR values of the proposed method are 6.7 dB ~ 10.3 dB lower than that of the standard MPEG-4 decoded sequences. Note that small geometric distortion can greatly decrease PSNR values. Geometric distortion on the images of the proposed method is mainly produced by the motion-prediction error between the motion model and the real camera motion. The average PSNR values for all test sequences are summarized in Table 5.5 and Table 5.6.

We evaluate the computational complexity of the proposed method by measuring the encoding/decoding time. Table 5.7 shows the results of the computation-time measurement. We run the codec 10 times and measure the average values. According to the table, the proposed method requires more computation time than the MPEG-4 codec in both encoding and decoding. The main reason for the increased computation time can be contributed to the image-matching module. Current implementation of the image-matching module compares a feature point in one boundary image with all feature points in the other boundary image to find initial matching candidates. In the decoder, image warping and interpolation of warped images produce an intermediate frame.

For encoding, the proposed method takes 3.3 ~ 6.9 times the encoding time of the MPEG-4 coding. It takes 1.3 ~ 2.8 times the encoding time of the MPEG-4 with GMC. For decoding, the proposed method takes 1.4 ~ 1.9 times the decoding time of the MPEG-4 coding, and 0.62 ~ 0.97 times the decoding time of the MPEG-4 with GMC.

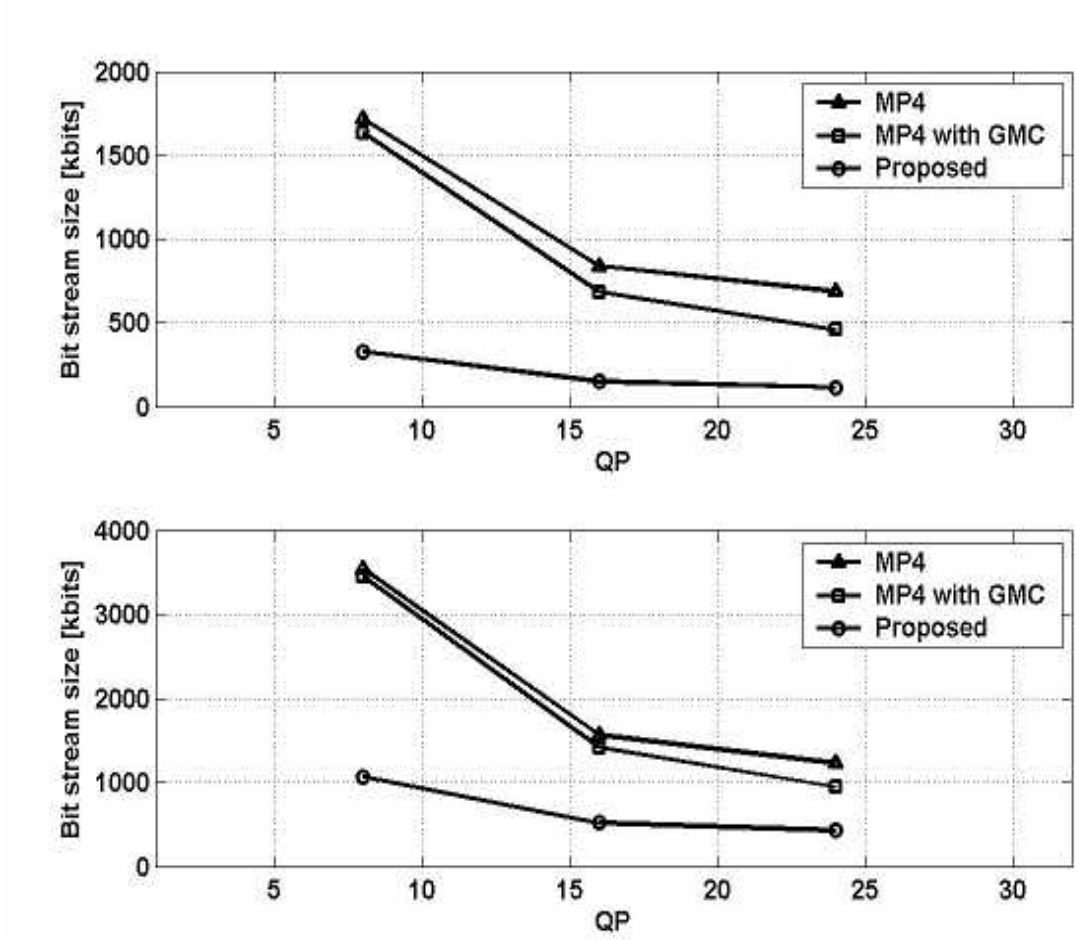


Figure 5.5. Comparison of the bit stream size for three coding schemes. The top graph shows the results of ‘ gallery’ sequence for three different quantization parameters, QP=8, 16, and 24. The bottom graph shows the results of ‘ docu1’ sequence.

Table 5.3. Comparison of the coded bits of the test sequences (QP=8)

Seq. name	MPEG-4	MPEG-4 with GMC	Proposed method	Saved bits* (%)
gallery	1,718,744	1,634,160	329,411	80.8
docu1	3,540,288	3,455,264	1,070,800	69.8
travel	1,865,416	1,536,376	899,299	51.8
museum1	1,009,320	958,432	469,123	53.5
museum2	1,615,824	1,570,720	386,291	76.1

* Saved bits = (Bits of MPEG-4 – Bits of Proposed method)/(Bits of MPEG-4)

Table 5.4. Comparison of the coded bits of the test sequences (QP=24)

Seq. name	MPEG-4	MPEG-4 with GMC	Proposed method	Saved bits* (%)
gallery	688,896	461,704	114,827	83.3
docu1	1,229,024	948,224	430,112	65.0
travel	426,384	281,144	185,267	56.6
museum1	362,392	225,832	134,723	62.8
museum2	428,728	337,336	122,859	71.3

Table 5.5. Comparison of the PSNR values of the decoded sequences (QP=8)

Seq. name	MPEG-4	MPEG-4 with GMC	Proposed method	Decrease
gallery	37.68	37.70	27.48	10.2
docu1	38.85	36.89	28.56	10.3
travel	33.63	34.05	27.80	5.83
museum1	34.37	34.38	28.70	5.67
museum2	34.56	34.54	26.05	8.51

Table 5.6. Comparison of the PSNR values of the decoded sequences (QP=24)

Seq. name	MPEG-4	MPEG-4 with GMC	Proposed method	Decrease
gallery	33.36	32.91	26.62	6.74
docu1	31.46	31.34	26.59	4.87
travel	28.91	29.12	25.18	3.73
museum1	29.3	28.95	25.64	3.66
museum2	29.3	28.97	24.12	5.18

(a) Original image (b) MPEG-4 (c) Proposed method

Figure 5.6. Comparison of the original/decoded images from ‘gallery’ sequence. Top images are part of the frame #40. Bottom images are part of the frame #90. QP=8.

(a) Original image (b) MPEG-4 (c) Proposed method

Figure 5.7. Comparison of the original/decoded images from ‘docu1’ sequence. Top images are part of the frame #112. Bottom images are part of the frame #188. QP=24.

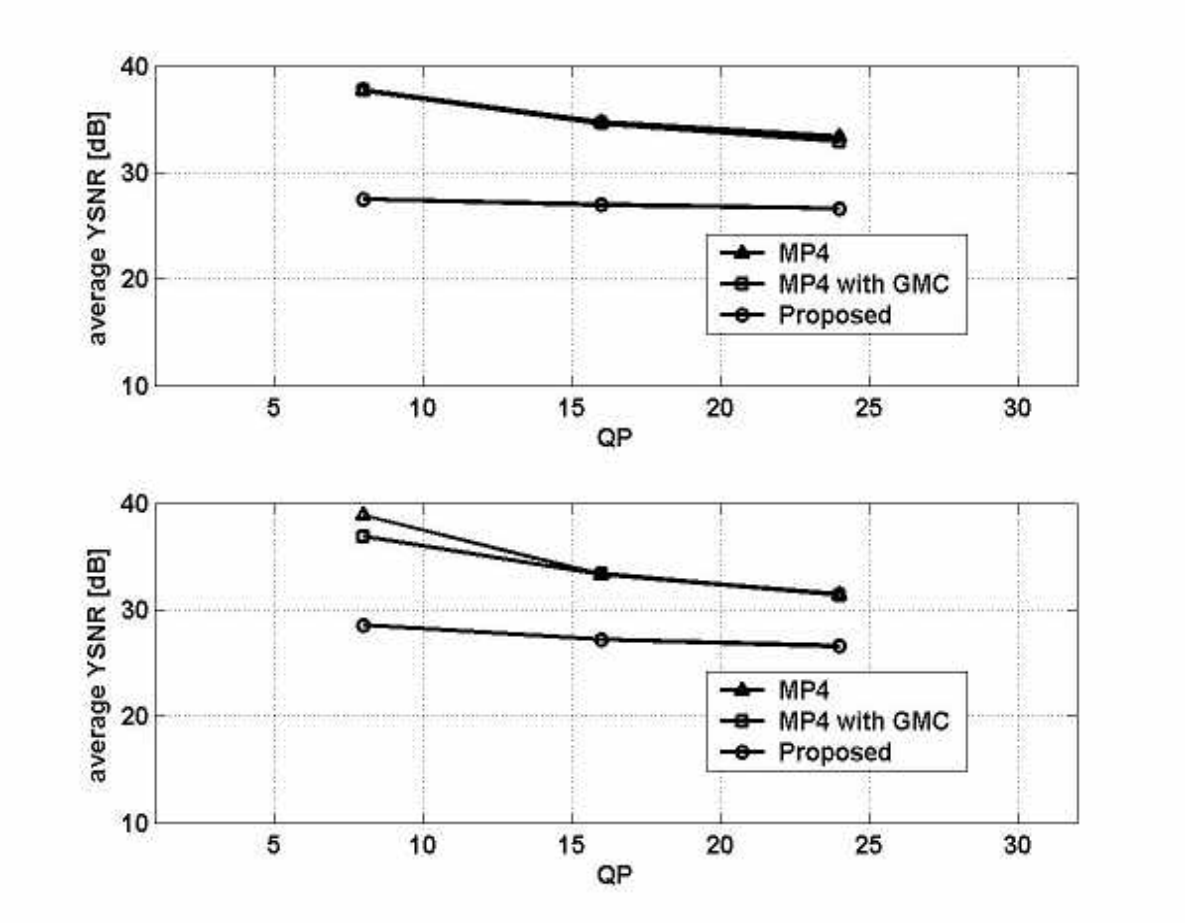


Figure 5.8. Comparison of the PSNR values for three coding schemes. The top graph shows the results of ' gallery' sequence for three different quantization parameters, QP=8, 16, and 24. The bottom graph shows the results of ' docu1' sequence.

Table 5.7. Comparison of the computation time of the test sequences (Unit: sec)

Sequence name	MPEG-4		MPEG-4 with GMC		Proposed method	
	Enc.	Dec.	Enc.	Dec.	Enc.	Dec.
gallery	148.2	28.6	430.0	59.7	767.3	55.9
docu1	272.1	51.8	674.5	108.1	1228.9	83.2
travel1	186.6	32.8	480.6	52.4	610.1	50.9
museum1	115.0	21.7	259.2	49.2	717.6	30.1
museum2	108.7	20.0	305.7	45.9	741.4	36.9

5.4 Conclusions

We have presented an approach that exploits the video metadata and achieves high bit-rate savings in current standard video-coding scheme. We have considered the problem of detecting a predictable scene taken by a zooming camera and producing images of the scene by using two images. Under the assumption that the speed of a zoom is constant over time and the scene only contains backgrounds, the intermediate frames in the detected camera motion scene are discarded at the encoder. By warping the decoded boundary frames and blending the predicted images at the decoder, our method ensures a natural-looking reconstruction of camera motion. Experimental results show that the proposed technique can save 52 % ~ 83 % of coded bits compared to the MPEG-4 coded bits with the trade-off of increased computational time.

The proposed method can be extended to more general zoom sequences that have moving foreground objects. Foreground and background objects can be separated from images by using image segmentation techniques [75]. Then, the background objects can be coded and decoded using the proposed method. The foreground objects can be coded using the standard MPEG-4 object coding mode. By inserting more key-frames, more dynamic zoom shots or irregular zoom motion can be reproduced by the proposed method.

Chapter 6

Conclusions and Future Work

The goal of this thesis was to develop video compression schemes that could significantly reduce bit-rates of the compressed video while maintaining the quality of the decoded video perceptually close to that of the standard video coding. To achieve this goal, we proposed a visual metadata-based video coding scheme. Visual metadata was used to locate specific video segments in the input video. It was also used to represent modeling parameters of the detected video segment. The proposed coding scheme was applied to video sequences containing camera zooms and the performance was analyzed and compared with that of the state-of-the-art coding method.

To address the temporal segmentation of camera motions from video sequences, a new camera motion detector that directly analyzes the 2-D distribution of inter-frame motion fields was developed. A modified cooccurrence matrix was proposed for the compact representation of global motion patterns in images. In contrast to 1-D histogram-based approaches, the analysis of cooccurrence matrices allows the detection of camera zooms with no additional extraction of image features. Experimental results showed that the proposed method achieved higher detection rates than the angle-histogram based method when camera motion was the dominant motion of the video sequence.

To estimate global motion between two boundary frames of a camera zoom, a robust image-matching method was developed. Since two boundary frames of a zoom shot may contain illumination changes and large scaling changes, an illumination and scaling invariant matching method was introduced. Rotation invariance was also implemented for more general matching purposes. Our experimental results showed that the proposed method is particularly effective to find matching points between images that have many similar regions or large illumination changes.

We integrated the camera motion detector, the image-matching module, and the image interpolator with the standard MPEG-4 video codec and evaluated the performance of the proposed coding scheme. Experimental results on the test videos showed that the proposed method saved 52 % ~ 83 % of bits compared to normal MPEG-4 coded videos. The ratio of saved bits is related to the proportion of a zoom shot to the total length of a sequence.

Additional research can be undertaken as future work. Some of these works include the following:

- For the proposed camera motion detector in chapter 2, a possible extension to this method is the detection of arbitrary camera motions such as horizontal or vertical tracking, rotation around the optical axis etc. For instance, horizontal tracking would be distinguished from panning by additional analysis of motion-magnitude distribution.
- For the image-matching algorithm in chapter 3, computation of mutual information between feature points is a very time-consuming process. Approximate estimation of mutual information, restricted search range etc. would lead to a faster initial matching.
- Current implementation of camera-zoom prediction from two views is very restrictive in representing dynamic camera zooms. By increasing the number of views in a zoom scene, the rendered camera motion would be closer to the original one. The determination of the number of views can be considered as an optimization problem constrained by rate and distortion.

Bibliography

[1] A. Netravali and B. Haskell, *Digital Pictures*, Plenum Press, 1988.

[2] R. Gray, "Vector Quantization", *IEEE ASSP magazine*, vol.1, pp.4-29, 1984.

[3] R. Clarke, "Image and video compression: A survey", *Jr. of Imaging Systems and Technology*, vol.10, no.1, pp.20-32, 1999.

[4] J. Barbosa and V. Barrosso, "Multiple and simultaneous image compression for video sequences", *Proc. ConfTele*, 2001.

[5] S. Ericsson, "Fixed and adaptive predictors for hybrid predictive/transform coding", *IEEE Tr. Communications*, vol.33, no.12, pp.1291-1302, 1985.

[6] T. Sikora, "Trends and perspectives in image and video coding", *Proceeding of the IEEE*, vol.93, no.1, pp.6-17, 2005.

[7] P. Eisert et al., "Model-aided coding: A new approach to incorporate facial animation into motion-compensated video coding", *IEEE Tr. Circuits and Systems for Video Technology*, vol.10, no.3, pp.344-358, 2000.

[8] H. Song and C. Kuo, "Rate control of a region-based H.263 video codec under time-varying channels", *Proc. Multimedia Signal Processing*, pp.339-344, 1999.

[9] H. Zhang and F. Bossen, "Region-based coding of motion fields for low-bit rate video compression", *Proc. International Conf. Image Processing*, vol.2, pp.1117-1120, 2004.

[10] ISO/IEC 14496-2:2001, *Coding of Audio-Visual Objects – Part 2: Visual*, 2001.

[11] L. Torres and E. Delp, "New trends in image and video compression", *Proc. of EUPSICO*, pp.651-657, 2000.

[12] J. Hidalgo and P. Salembier, "Metadata-based coding tools for hybrid video codecs", *Proc. Picture Coding Symposium*, pp.473-477, 2003.

[13] P. Beek et al., "Metadata-driven multimedia access", *IEEE Signal Processing Magazine*, pp.40-52, March 2003.

[14] O. Lotfallah, M. Reisslein, and S. Panchanathan, "Adaptive video transmission schemes using MPEG-7 motion intensity descriptor", *IEEE Tr. Circuits and Systems on Video Technology*, vol.16, no.8, pp.929-946, 2006.

[15] "MPEG-7 Overview", ISO/IEC JTC1/SC29/WG11 N6828, Oct. 2004.

[16] H. Wang et al., "Survey of compressed-domain features used in audio-visual indexing and analysis", *J. of Visual Communication & Image Representations*, no. 14, pp.150-180, 2003.

[17] Y. Tan et al., "Rapid estimation of camera motion from compressed video with application to video annotation", *IEEE Tr. on Circuits and Systems for Video Technology*, vol.10, no. 1, pp.133-146, 2000.

[18] F. Dufaux and J. Konrad, "Efficient, robust and fast global motion estimation for video coding", *IEEE Tr. on Image Processing*, vol.9, no.3, pp.497-501, Mar. 2000.

[19] C. Doulaverakis et al., "Adaptive method for motion characterization and segmentation of MPEG compressed frame sequences", ICIAR 2004, *Lecture Notes in Computer Science* 3211, pp.310-317, 2004.

[20] S. Lee and M. Hayes, "Real-time camera motion classification for content-based indexing and retrieval using templates", *IEEE International Conference on Acoustics*, Speech, and Signal Processing, pp.3664-3667, 2002.

[21] R. Haralick, M. Shanmugam and I. Dinstein, "Textural features for image classification," *IEEE Tr. Systems, Man, and Cybernetics*, vol.SMC-3, pp.610-621, 1973.

[22] R. Nelson and R.Polana, "Qualitative recognition of motion using temporal texture", *CVGIP: Image Understanding*, 56(1):78-89, July 1992.

[23] P. Bouthemy and R. Fablet, "Motion characterization from temporal cooccurrences of local motion-based measures for video indexing", *IEEE International Conference on Pattern Recognition*, vol. 1, pp.905-908, 1998.

[24] J. Barron et al., "Performance of optical flow techniques", *International Jr. of Computer Vision*, 12, pp.43-77, 1994.

[25] H. Chang and S. Lai , "Robust camera motion estimation and classification for video analysis", *SPIE Visual Communications and Image Processing*, pp.912-923, 2004.

[26] J. Kim et al., "Efficient camera motion characterization for MPEG video indexing", *IEEE International Conference on Multimedia Expo*, pp.1171-1174, 2000.

[27] http://www.open-video.org

[28] V. Kobla, D. Doermann and K. Lin, "Archiving, indexing, and retrieval of video in the compressed domain", *SPIE Multimedia Storage and Archiving Systems*, pp.78-89, 1996.

[29] A. Dumitras and B. Haskell, "A look-ahead method for pan and zoom detection in video sequences using block-based motion vectors in polar coordinates", *IEEE International Symposium on Circuits and Systems*, vol.III, pp.853-856, 2004.

[30] Y. Ma and H. Zhang, "Motion texture: a new motion based video representation", *IEEE International Conference on Pattern Recognition*, vol.2, pp.548-551, 2002.

[31] M. Sonka et al., *Image Processing, Analysis, and Machine Vision*, 2nd Ed., PWS Publishing, 1998.

[32] M. Brown and D. Lowe, "Invariant features form interest point groups", *Proc. British Machine Vision Conference*, pp.253-262, 2002.

[33] Y. Dufournaud et al, "Image matching with scale adjustment", *Computer Vision and Image Understanding* 93 (2004), pp.175-194, 2004.

[34] G. Egnal, "Mutual information as a stereo correspondence measure", Technical report, Dept. of computer and information science, University of Pennsylvania, 2000.

[35] C. Harris and M. Stephens, "A combined corner and edge detector", *Proc. Alvey Vision Conf.*, pp.189-192, 1988.

[36] R. Horaud and T. Skordas, "Stereo correspondence through feature grouping and maximal cliques", *IEEE Tr. Pattern Anal. Mach. Intell.*, vol. 11, no.11, pp.1168 – 1180, 1989.

[37] http://lear.inrialpes.fr/people/Mikolajczyk/Database

[38] M. Lourakis et al., "A graph-based approach to corner matching using mutual information as a local similarity measure", *Proc. International Conference on Pattern Recognition*, 2004.

[39] D. Lowe, "Distinctive image features from scale-invariant keypoints", *International Jr. of Computer Vision*, 60(2), pp.91-110, 2004.

[40] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points", *Proc. International Conf. on Computer Vision*, pp.1150-1157, 1999.

[41] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors", *Proc. Computer Vision and Pattern Recognition*, pp.257-264, 2003.

[42] E. Mortensen, H. Deng and L. Shapiro, "A SIFT descriptor with global context", *Proc. Computer Vision and Pattern Recognition*, vol.1, pp.184-190, 2005.

[43] J. Pluim et al., "Mutual-information-based registration of medical images: A survey", *IEEE Tr. Medical Imaging*, vol.22, no.8, pp.986-1004, 2003.

[44] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval", *IEEE Tr. Pattern Anal. Mach. Intell.*, 19 (5) pp.530-534, 1997.

[45] M. Sonka, V. Hlavac and R. Boyle, *Image Processing, Analysis, and Machine Vision*, 2nd Ed., PWS Publishing, 1999.

[46] http://staff.science.uva.nl/~aloi/

[47] http://bj.middlebury.edu/~schar/stereo/data/

[48] Maxclique, http://www.colostate.edu/Depts/Biology/Research/simmons-carr-oneill-2004.htm

[49] H. Jeon, A. Basso, and P. Driessen, "Camera motion detection in video sequences using motion cooccurrences", PCM 2005, *Lecture Notes in Computer Science* 3767, pp.524-534, 2005.

[50] H. Jeon, A. Basso, and P. Driessen, "A global correspondence for scale invariant matching using mutual information and the graph search", *Proc. IEEE International Conf. Multimedia and Expo*, pp.1745-1748, 2006.

[51] V. Bhaskaran and K. Konstantinides, *Image and video compression standards: Algorithms and architectures*, 2nd Ed., Kluwer Academic Publishers, 1997.

[52] S. –F. Chang, T. Sikora, and A. Puri, "Overview of the MPEG-7 Standard", *IEEE Tr. Circuits and Systems for Video Technology*, vol.11, no.6, pp.688-695, 2001.

[53] Extensible Markup Language (XML) Version 1.0, http://www.w3.org/TR/REC-xml

[54] ISO/IEC FDIS 15938-6:2003, "Multimedia Content Description Interface – Part 6: Reference Software".

[55] S. Jeannin et al., "Motion descriptors for content-based video representation", *Signal Processing: Image Communications* 16 (2000) pp.59-85.

[56] S. Jeannin and A. Divakaran, "MPEG-7 visual motion descriptors", *IEEE Tr. Circuits and Systems on Video Technology*, vol.11, no.6, pp.720-724, 2001.

[57] J. Hidalgo and P. Salembier, "On the use of indexing metadata to improve the efficiency of video compression", *IEEE Tr. Circuits and Systems on Video Technology*, vol.16, no.3, pp.410-419, 2006.

[58] B. Manjuhath, P. Salembier, and T. Sikora, *Introduction to MPEG-7*, John Wiley & Sons, 2002.

[59] "MPEG-21 Overview", ISO/IEC JTC1/SC29/WG11 N5231, Oct., 2002.

[60] T. Sikora, "MPEG digital video-coding standards", *IEEE Signal Processing Magazine*, pp.82 – 100, Sep. 1997.

[61] T. Sikora, "The MPEG-7 Visual standard for content description – An overview", *IEEE Tr. Circuits and Systems for Video Technology*, vol.11, no.6, pp.696-702, 2001.

[62] A. Steinaker et al., "Metadata standards for web-based resources", *IEEE Multimedia*, vol.8, no.1, pp.70-76, 2001.

[63] C. Stewart, "Robust parameter estimation in computer vision", *SIAM Review*, vol.41, no.3, pp.513-537, 1999.

[64] "Understanding Metadata", http://www.niso.org/standards/resources.

[65] T. Wiegand, X. Zhang, and B. Girod, "Long-term memory motion compensated prediction", *IEEE Tr. Circuits and Systems on Video Technology*, vol.9, no.1, pp.70-84, 1999.

[66] IBM VideoAnnEx, http://www.research.ibm.com/VideoAnnEx/

[67] MPEG-7 Reference Software,
http://www.lis.e-technik.tu-muenchen.de/research/bv/topics/mmdb/mpeg7.html

[68] ISO/IEC 13818-1:2000/FPDAM 1, ver.1.0

[69] MPEG-4 Video Verification Model version 16.0, ISO/IEC JTC1/SC29/WG11 N3312, March 2000.

[70] K. Shoemake and T. Duff, "Matrix animation and polar decomposition", *Proc. Graphics Interface*, pp.258-264, 1992.

[71] D. Tian et al., "Coding of faded scene transitions", *Proc. International Conference on Image Processing*, vol.2, pp.505-508, 2002.

[72] H. Jeon, A. Basso, and P. Driessen, "Enhanced video coding based on camera motion and image synthesis", *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, vol.1, pp.241-244, 2003.

[73] M. Fischler and R. Bollers, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM* 24(6), pp.381-395, 1981.

[74] D. Forsyth and J. Ponce, *Computer Vision – A modern approach*, ch.2, Prentice Hall, 2003.

[75] D. Zhang and G. Lu, "Segmentation of moving objects in image sequence: A review", *Circuits System Signal Processing*, vol.20, no.2, pp.143-183, 2001.

[76] F. Galpin et al., "3D models coding and morphing for efficient video compression", *Proc. Computer Vision and Pattern Recognition*, vol.1, pp.334-341, 2004.

[77] J. Wang and E. Adelson, "Representing moving images with layers", *IEEE Tr. Image Processing*, vol.3, no.5, pp.625-638, 1994.

[78] R. Babu, K. Ramakrishnan and S. Srinivasan, "Video object segmentation: a compressed domain approach**",** *IEEE Tr. Circuits and Systems for Video Technology*, vol.14, no.4, pp.462-474, 2004.

Appendix A

Derivation of the thresholds in Table 2.3 in Sec. 2.3.4

Let the parameter r represent the percentage of motion vectors taken by a moving object or unreliable motion vectors within a flow field. By assuming that a moving object generates a group of motion vectors with a single direction and unreliable motion vectors have randomly distributed orientations, the normalized elements of a cooccurrence matrix can be expressed as

$$p_{MO}(i,j) = \begin{cases} r & \text{if } i = j = m \\ 0 & \text{otherwise} \end{cases} \quad \text{for a moving object} \tag{A.1}$$

$$p_{UM}(i,j) = \frac{r}{N_q^2} \quad \text{for unreliable motion vectors}$$

The normalized elements of a cooccurrence matrix for a pan/tilt and zoom can be expressed as

$$p_{pantilt}(i,j) = \begin{cases} 1-r & \text{if } i = j = k \\ 0 & \text{otherwise} \end{cases} \tag{A.2}$$

$$p_{zoom}(i,j) = \begin{cases} \dfrac{1-r}{N_q} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

A) The threshold for the *Average* feature

The *Average* in Eq.(2.5) is supposed to be its maximum value, 1, when all points in a picture have motion. Since the *Average* value decreases directly proportional to the size of the static area during a camera motion, we set the threshold $T_{average}$ as

$$T_{average} = 1 - r \tag{A.3}$$

where r denotes the ratio of the static vectors in a flow field.

B) The threshold for the *ASM* feature

By substituting the *p(i,j)* in Eq.(2.6) with Eq. (A.1), we obtain

$$ASM\big|_{pantilt} = \begin{cases} [p(k,k)^2]_{pantilt} + [p_{MO}(m,m)^2]_{movingobj.} \\ = (1-r)^2 + r^2 \quad \text{for a moving object} & \text{(A.4-a)} \\ [p(k,k)^2]_{pantilt} + [\sum_{i,j}^{N_q} p_{UM}(i,j)^2]_{unreliablevectors} \\ = (1-r)^2 + r^2 / N_q^2 \quad \text{for unreliable motion vectors} & \text{(A.4-b)} \end{cases}$$

The value in Eq.(A.4-b) is always lower than that of Eq.(A.4-a), which means that unreliable motion vectors more severely decrease the *ASM* feature values than a moving object does. Thus, we set the threshold T_{asm} as

$$T_{asm} = (1-r)^2 + r^2 / N_q^2 \tag{A.5}$$

C) The threshold for the *Entropy* feature

By substituting the *p(i,j)* in Eq.(2.7) with Eq. (A.1), we obtain

$$Entropy\big|_{zoom} = \begin{cases} -\sum_{i=1}^{N_q} p_{zoom}(i,i)\log(p_{zoom}(i,i)) - p_{MO}(m,m)\log(p_{MO}(m,m)) \\ = -(1-r)\cdot\log(\frac{1-r}{N_q}) - r\cdot\log(r) \qquad \text{for a moving object} \quad \text{(A.6-a)} \\ -\sum_{i=1}^{N_q} p_{zoom}(i,i)\log(p_{zoom}(i,i)) - \sum_{i=1,j=1}^{N_q} p_{UM}(i,j)\log(p_{UM}(i,j)) \\ = -(1-r)\cdot\log(\frac{1-r}{N_q}) - r\cdot\log(\frac{r}{N_q^2}) \qquad \text{for unreliable vectors (A.6-b)} \end{cases}$$

As Eq.(A.6-a) is always lower than Eq.(A.6-b), the threshold T_{ent} is set as

$$T_{ent} = -(1-r)\cdot\log(\frac{1-r}{N_q}) - r\cdot\log(r) \qquad \text{(A.7)}$$

D) The threshold for the *Difference Entropy*

During a zoom, the *Difference Entropy* value in Eq.(2.8) is supposed to be very low. When unreliable motion vectors are present, this feature value increases. Note that the presence of a large moving object does not affect the *Difference Entropy* value. Then, the *Difference Entropy* of the zoom image containing unreliable motions can be expressed as below.

$$Difference\ Entropy\big|_{zoom} = -[\sum_{i=0}^{N_q-1} p_{x-y}(i)\log(p_{x-y}(i))]_{zoom} - [\sum_{i=0}^{N_q-1} p_{x-y}(i)\log(p_{x-y}(i))]_{unreliablevectors} \qquad \text{(A.8)}$$

For the first term in Eq.(A.8), $p_{x-y}(i)$ is computed by substituting $p(i,j)$ in Eq.(2.8) with $p_{zoom}(i,j)$ in Eq.(A.2). Similarly, $p_{x-y}(i)$ in the second term in Eq.(A.8) is computed by using $p_{UM}(i,j)$ in Eq.(A.1). Now, Eq.(A.8) becomes

$$Difference\ Entropy\big|_{zoom} = -(1-r)\log(1-r) \qquad \text{(A.9)}$$

$$-\{r/N_q \cdot \log(r/N_q) + r \cdot [\sum_{i=1}^{N_q-1} 2\cdot(N_q - i)/N_q^2 \cdot \log(2\cdot r \cdot (N_q - i)/N_q^2)]\}$$

The threshold T_{diff_ent} is set as

$$T_{diff\_ent} = -(1-r)\log(1-r)$$

$$-\{r/N_q \cdot \log(N_q) + r \cdot [\sum_{i=1}^{N_q-1} 2\cdot(N_q - i)/N_q^2 \cdot \log(2\cdot r \cdot (N_q - i)/N_q^2)]\} \quad \text{(A.10)}$$

Appendix B

Segmentation results and sample frames of the test video set

This section provides segmentation results and sample frames of the full video set used in chapter 2.

Table B.1 Test set 1 video sequences

Seq. name (Type)	Length (frames)	Manual segmentation	Comment on content	Detection results
beach007(A)	235	47-190 : pan-left	Pan-left of a stationary scene	46-190 : pan-left
busstop003(A)	200	23-189 : pan-left	Pan-left of a bus terminal	45-189 : pan-left
gallery009(A)	256	38-209 : tilt-up	Tilt-up of an artwork	48-209 : tilt-up
garden002 (A)	195	46-134 : zoom-in	Zoom-in of a stationary scene	46-135 : zoom-in
ground001(A)	375	119-240 : pan-left	Pan-left of a lawn	118-241 : pan-left
library002(A)	169	66-151 : zoom-in	Zoom-in of a stationary scene	66-156 : zoom-in
mall006(A)	325	92-261 : pan-right	Two walking people during the pan	92-239 : pan-right
park004 (A)	385	37-121 : zoom-out 153-207 : dissolve 253-360 : pan-right	A smooth background at the end of the zoom	37-121 : zoom-out 253-360 : pan-right
road003(A)	233	80-217 : zoom-in	Zoom-in of a building and trees	80-217 : zoom-in 221-230 : zoom-in
totempole001(A)	201	46-178 : zoom-in	Zoom-in of two totem poles	46-187 : zoom-in
beach004(B)	255	27-204 : pan-right	A smooth background (frames 0-254)	27-200 : pan-right
boat002(B)	303	No camera motion	A passing boat with lights (frames 0-302)	-

building003 (B)	210	62-161 : zoom-out	A passing man (frames 40-111)	38-89 : zoom-out
busstop004 (B)	137	17-133 : pan-right	A passing van (frames 15-137)	8-18 : zoom-out 19-132 : pan-right
busstop005(B)	249	65-191: zoom-in	Swaying trees (frames 195-249)	65-215 : zoom-in 220-232 : zoom-out
fountain001 (B)	416	18-142 : pan-right 156-184 : dissolve 189-400 : pan-left	A passing car (frames 332-398)	18-64, 68-142 : pan-right 197-401 : pan-left
fountain004 (B)	391	41-126 : zoom-in 142-174 : dissolve 175-390 : no camera motion	Passing cars (frames 182-370)	6-16 : zoom-in 41-137 : zoom-in
gallery001(B)	355	17-121 : pan-left 205-286 : zoom-in	Camera jittering after the end of zoom-in.	28-37, 41-122 : pan-left 205-293 : zoom-in 303-316, 321-331, 335-346: zoom-in
gallery003 (B)	350	27-145 : tilt-up 201-309 : zoom-in	A large white wall (frames 0-349) Camera jittering after the end of zoom-in.	41-135 : tilt-up 201-310 : zoom-in 314-325, 336-347 : zoom-out
hall002 (B)	193	44-184 : pan-right	Swaying branches (frames 0-192)	49-182 : pan-right
harbour001(B)	400	74-327 : pan-left	A smooth background (frames 0-399)	30-54, 65-75 : zoom-out 76-201, 207-318 : pan-left 319-338, 375-393 : zoom-out
harbour002(B)	275	30-253 : pan-right	A smooth background (frames 0-274)	42-234 : pan-right 29-41, 235-259 : zoom-out
harbour003(B)	175	28-106 : zoom-out	A smooth background (frames 40-175)	27-114 : zoom-out 118-134 : zoom-out

mall008 (B)	425	77-168 : zoom-in 182-226 : dissolve 245-398 : pan-left Contains walking women	A walking person (frames 0-158, 345-399)	73-169 : zoom-in 246-399 : pan-left
office003 (B)	181	53-130 : pan-left	A passing toy train (frame 35-130)	55-129 : pan-left
office008 (B)	194	30-114 : pan-left	A passing toy train	80-114 : pan-left 115-134 : zoom-in
office009 (B)	263	24-174 : zoom-in	A passing toy train	37-173 : zoom-in 245-257 : zoom-out
park008 (B)	137	26-86 : zoom-in	A group of walking people (frames 0-136)	25-119 : zoom-in 123-132 : zoom-in
ringrd001(B)	295	77-190 : zoom-out	Passing cars (frames 57-213)	76-190 : zoom-out
road009(B)	420	No camera motion	A passing truck (frames 139-164)	-
sub006 (B)	345	64-226 : zoom-in	Passing cars (frames 65-242)	64-226 : zoom-in 255-265, 269-296 : zoom-out 306-341 : zoom-in
sw001(B)	385	31-340 : pan-left	Large dark background and camera jittering (frames 0-384)	46-120, 124-148, 152-187, 207-229, 234-246, 261-306, 313-328 : pan-left
sw003(B)	290	74-216 : pan-left	Large dark background and camera jittering (frames 0-289)	2-12 : zoom-in 84-97, 102-201 : pan-left
building002 (C)	245	16-213 : tilt-up	An approaching pedestrian (frames 81-150) Large white wall (frames 40-244)	20-32 : zoom-in 33-84, 149-158, 164-188 : tilt-up
fountain007 (C)	265	88-176 : zoom-out	Swaying trees (frames 120-264)	97-180: zoom-out

				191-227 : zoom-in
fountain010 (C)	490	No camera motion	Fast flowing water	-
fountain011 (C)	520	161-346 : zoom-out	Fast flowing water during the zoom	No detected frames
fountain013 (C)	660	166-323 : zoom-out 375-487 : pan-left	Falling water during the zoom	255-324 : zoom-out 368-378 : zoom-in 379-489 : pan-left
fountain014 (C)	400	152-277 : tilt-up	Falling water during the tilt-up	169-277 : tilt-up
office010 (C)	184	18-120 : zoom-in	A passing toy train (frames 69-88)	18-176 : zoom-in
office016 (C)	157	3-150 : zoom-out	A smooth background (frames 0-156)	No detected frames
park013 (C)	270	67-230 : pan-left.	A smooth background (frames 0-269)	No detected frames
park005 (C)	514	88-203 : zoom-in 268-506 : pan-left	A smooth background (frames 0-269) A sky taking upper half of the scene (frames 268-506)	85-204 : zoom-in 210-237 : zoom-in 241-252 : zoom-out
road010(C)	285	No camera motion	A passing bus (frames 90-180)	95-105, 152-175 : zoom-out
road011(C)	380	No camera motion	A passing bus (frames 130-260)	155-191, 203-245 : zoom-out
road012(C)	210	No camera motion	A passing bus overlapping with ripples (frames 35-90)	-
road014(C)	395	No camera motion	A large truck passing away (frames 70-330)	71-153, 214-223, 229-276 : zoom-out 292-309 : zoom-in
road017(C)	785	No camera motion	A large truck	190-203 : zoom-

			passing away (frames 150-360)	out 210-259 : pan-left
sw002(C)	450	113-247 : pan-right 302-409 : tilt-down	Dark background and camera jittering during panning and tilting	138-186 : pan-right 370-379 : tilt-down 31-50, 61-75, 79-91 : zoom-in 95-130 : zoom-out
window001 (C)	320	28-302 : zoom-in	Large smooth areas (wall, windows)	170-185 : zoom-in

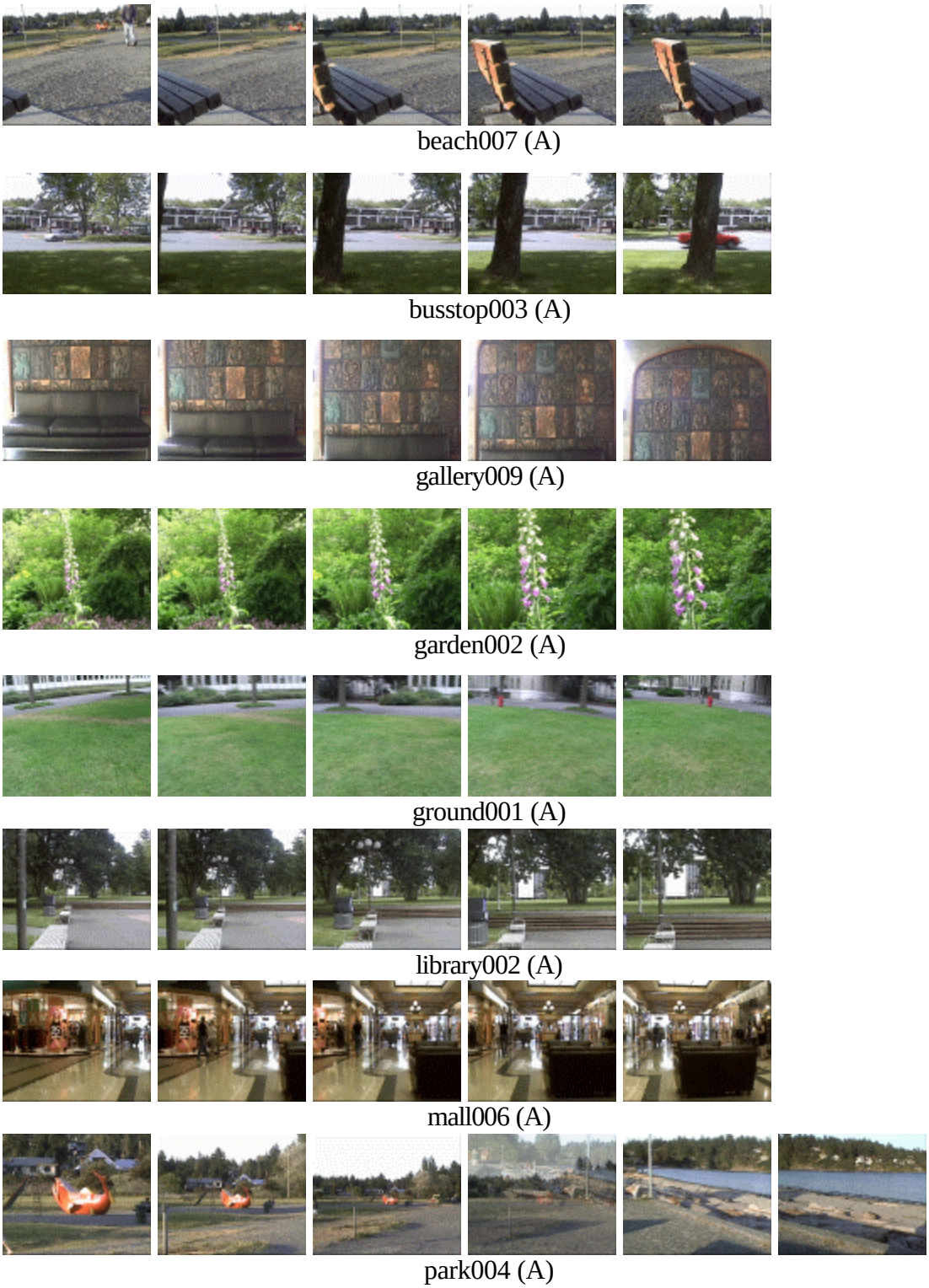


Figure B.1 Sample frames of each test sequence in test set 1.

Figure B.1 (continued)

fountain004 (B)

gallery001 (B)

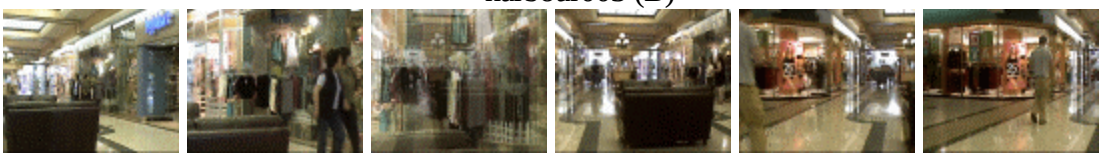
gallery003 (B)

hall002 (B)

harbour001 (B)

harbour002 (B)

harbour003 (B)



mall008 (B)

Figure B.1 (continued)

Figure B.1 (Continued)

sw003 (B)

building002 (C)

fountain007 (C)

fountain010 (C)

fountain011 (C)

fountain013 (C)

fountain014 (C)

office010 (C)

Figure B.1 (Continued)

Figure B.1 (Continued)

sw002 (C)

window001 (C)

Figure B.1 (Continued)

Senses and sensibility001

Senses and sensibility002

Senses and sensibility003

Hidden fury001

Hidden fury002

Figure B.2 Sample frames of each test sequence in test set 2.

Wrestling with uncertainty001

Wrestling with uncertainty002

Wrestling with uncertainty003

Figure B.2 (Continued)

VITA

Surname: Jeon Given Name: Hyun-Ho

Place of Birth: Pusan, Korea

Educational Institutions Attended:

Korea Advanced Institute of Science and Technology, Korea	1988 – 1990
Pusan National University, Korea	1984 – 1988

Degree Awarded:

M. Sc. Korea Advanced Institute of Science and Technology, Korea	1990
B. Eng. Pusan National University, Korea	1988

Awards:

NSERC IPS scholarship	2001 – 2002
University of Victoria President's Research Scholarship	2001 – 2002