Motion Estimation for Video Resolution Down-Conversion and Frame Type Conversion

A thesis submitted to
The Hong Kong University of Science and Technology
in partial fulfillment of the requirement for
the Degree of Master of Philosophy in
Electrical and Electronic Engineering

by
Wong Wai Chuen
Bachelor of Electrical and Electronic Engineering
The Hong Kong University of Science and Technology
Hong Kong, 1996

January, 1999
Motion Estimation for Video Resolution Down-Conversion and Frame Type Conversion

by
Wong Wai Chuen

Approved by:

Dr. Oscar C. AU
Thesis Supervisor

Prof. Ming L. LIOU
Thesis Examination Committee Chairman

Dr. Bing ZENG
Thesis Examination Committee Member

Prof. Philip C. H. CHAN
Head of Department

Department of Electrical and Electronic Engineering
The Hong Kong University of Science and Technology
January, 1999
# Table of Contents

List of Figures ........................................................................................................... iv

List of Tables ................................................................................................................ viii

Abstract ......................................................................................................................... ix

Acknowledgments ......................................................................................................... xi

Chapter 1 Introduction ................................................................................................. 1
  1.1 Motivation ............................................................................................................... 1

Chapter 2 Review of Block-based Motion Estimation ................................................... 4
  2.1 Introduction ............................................................................................................. 4
  2.2 Motion Estimation by Block Matching ................................................................. 4
  2.3 Computation Requirement of Full Search .......................................................... 5

Chapter 3 Fast Motion Estimation for Resolution Down-Conversion of Compressed Video .............................................................................................................. 7
  3.1 Introduction ............................................................................................................. 7
  3.2 Existing Algorithms ............................................................................................... 8
  3.3 Predictive Motion Estimation (PME) ................................................................... 10
  3.4 Modified Predictive Motion Estimation (MPME) .................................................. 12
  3.5 Motion Estimation by Spatial-Variant Filtering ................................................... 14
  3.6 Simulation Results ............................................................................................... 16
    3.6.1 Simulation results of PME ............................................................................. 16
    3.6.2 Simulation results of MPME ................................................................. 32
    3.6.3 Simulation results of ME-SVF ............................................................... 47
  3.7 Conclusion ............................................................................................................ 49
Chapter 4 Fast Motion Estimation for Frame Type Conversion

4.1 Introduction........................................................................................................... 50
4.2 Fast Motion Estimation for Frame Type Conversion ............................................ 51
  4.2.1 P-to-Bframe Conversion.................................................................................... 52
  4.2.2 P-to-P frame Conversion.................................................................................. 55
4.5 Simulation Results ................................................................................................. 56
  4.5.1 Simulation results of P to B-frame conversion.................................................. 56
  4.5.2 Simulation results of P to P-frame conversion................................................. 63
4.6 Conclusion ............................................................................................................ 69

Chapter 5 Lost Motion Vector Recovery for Error Concealment of Compressed Video

5.1 Introduction........................................................................................................... 70
5.2 Review of Lost Motion Vectors Recovery Algorithms ............................................ 71
  5.2.1 Boundary Matching Algorithm (BMA)............................................................. 71
  5.2.2 Overlapped Motion Compensation (OMC)...................................................... 72
  5.2.3 Decoder Motion Vector Estimation (DMVE)................................................... 73
5.3 Subblock Surrounding Pixels Search (SSPS) ......................................................... 74
5.4 SSPS with Intra-block Overlapping (SSPS-IO)..................................................... 75
5.5 Simulation Results ............................................................................................... 77
5.7 Conclusion ............................................................................................................ 87

Chapter 6 Conclusion.................................................................................................. 88

Reference .................................................................................................................... 90

List of Publications .................................................................................................... 92
List of Figures

3.1 Possible relationship among motion vectors of the original frame and the reduced frame ................................................................. 7

3.2 Possible relationships among motion vectors of the original frame and the reduced frame ........................................................................... 9

3.3 Local search points of PME-LS1 and PME-LS2 ........................................... 12

3.4 (a) 9-to-1 conversion and (b) 16-to-1 conversion ........................................... 14

3.5 PSNR of predicted frames using different algorithms for the sequence “Football” ....................................................................................... 19

3.6 The difference of PSNR of the predicted frames between PME-LS1 and AMVR for the sequence “Football” .................................................. 19

3.7 Original reduced frame #125 of “Football” .................................................... 20

3.8 Comparison for the frame #125 of “Football”; the predicted frame #125 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS1 (d) AMVR and (e) MEAN ........................................................................................................ 21

3.9 PSNR of predicted frames using different algorithms for the sequence “Salesman” ................................................................................. 23

3.10 The difference of PSNR of the predicted frames between PME-LS2 and AMVR for the sequence “Salesman” .................................................. 23

3.11 Original reduced frame #88 of “Salesman” .................................................... 24

3.12 Comparison for the frame #88 of “Salesman”; the predicted frame #88 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS2 (d) AMVR and (e) MEAN ........................................................................................................ 25

3.13 PSNR of predicted frames using different algorithms for the sequence “Table Tennis” ........................................................................... 26

3.14 The difference of PSNR of the predicted frames between PME-LS2 and AMVR for the sequence “Table Tennis” ........................................... 26

3.15 Original reduced frame #140 of “Table Tennis” ........................................... 27

3.16 Comparison for the frame #140 of “Table Tennis”; the predicted frame #140 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS2 (d) AMVR and (e) MEAN ........................................................................................................ 28

3.17 PSNR of predicted frames using different algorithms for the sequence “Miss America” ........................................................................... 29
3.18 The difference of PSNR of the predicted frames between PME-LS2 and AMVR for the sequence “Miss America” ......................................................... 29

3.19 Original reduced frame #53 of “Miss America” ........................................... 30

3.20 Comparison for the frame #53 of “Miss America”; the predicted frame #53 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS1 (d) AMVR and (e) MEAN. ......................................................... 31

3.21 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence “Football” ......................................................... 35

3.22 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence “Salesman” ......................................................... 36

3.23 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence “Table Tennis” ......................................................... 37

3.24 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence “Miss America” ......................................................... 38

3.25 Original reduced frame #96 of “Football” ......................................................... 39

3.26 Comparison for the frame #96 of “Football”; the predicted frame #96 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (e) MEAN. ......................................................... 40

3.27 Original reduced frame #92 of “Salesman” ......................................................... 41

3.28 Comparison for the frame #92 of “Salesman”; the predicted frame #92 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (e) MEAN. ......................................................... 42

3.29 Original reduced frame #24 of “Table Tennis” ......................................................... 43

3.30 Comparison for the frame #24 of “Table Tennis”; the predicted frame #24 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (e) MEAN. ......................................................... 44

3.31 Original reduced frame #96 of “Miss America” ......................................................... 45

3.32 Comparison for the frame #96 of “Miss America”; the predicted frame #96 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (e) MEAN. ......................................................... 46

3.33 PSNR difference between ME-SVF and AMVR for “Football” ......................... 48

3.34 PSNR difference between ME-SVF and AMVR for “Salesman” ......................... 48
4.1 Extraction of a fast playback preview video from a compressed video sequence with IBBP frame structure. ................................................................. 51

4.2 Available motion information .................................................................................. 52

4.3 The motion estimation of frame k & k+1 with: (a) P to P-frame forward prediction (b) B to P-frame backward prediction ........................................................................ 53

4.4 The motion estimation of frame k & k+1 with: (a) P to P-frame forward prediction (b) B to P-frame backward prediction ........................................................................ 54

4.5 The forward motion estimation of frame k-1, k & k+1 with: (a) original P-frame forward prediction (b) re-encoded P-frame forward prediction ........................................ 55

4.6 PSNR of predicted B-frames (a) using different algorithms; (b) PSNR difference of full search minus P2BS-LS and 3SS for the sequence “Football” ......................... 59

4.7 PSNR of predicted B-frames (a) using different algorithms; (b) PSNR difference of full search minus P2BS-LS and 3SS for the sequence “Table Tennis” .............. 60

4.8 PSNR of predicted B-frames (a) using different algorithms; (b) PSNR difference of full search minus P2BS-LS and 3SS for the sequence “Salesman” ....................... 61

4.9 PSNR of predicted B-frames (a) using different algorithms; (b) PSNR difference of full search minus P2BS-LS and 3SS for the sequence “Miss America” .............. 62

4.10 PSNR of predicted P-frames (a) using different algorithms; (b) PSNR difference of FS-14 minus P2PS-LS and 3SS for the sequence “Football” .......................... 65

4.11 PSNR of predicted P-frames (a) using different algorithms; (b) PSNR difference of FS-14 minus P2PS-LS and 3SS for the sequence “Table Tennis” .................... 66

4.12 PSNR of predicted P-frames (a) using different algorithms; (b) PSNR difference of FS-14 minus P2PS-LS and 3SS for the sequence “Salesman” ......................... 67

4.13 PSNR of predicted P-frames (a) using different algorithms; (b) PSNR difference of FS-14 minus P2PS-LS and 3SS for the sequence “Miss America” .................... 68

5.1 Selection of surrounding pixels .................................................................................. 73

5.2 Surrounding pixels region of Subblock Searching .................................................. 74

5.3 The 8 × 8 weighting matrices (a) W_{A1}, (b) W_{B1}, (c) W_{C1}, (d) W_{D1} ................. 76

5.4 PSNR of BMA, DMVE, SSPS-IO for the sequence of “Football” ............................. 78

5.5 Part of the frame 29 of “Football” error concealed by (a)DMVE and (b)SSO-IO 79

5.6 Comparison for frame #29 of “Football” (a) The original frame; the predicted frame #29 (left) and error image (right) by (b) original MV, (c) BMA, (d) DMVE, (e) SSPS, (f) SSPS-IO .......................................................... 81
5.7 PSNR of BMA, DMVE, SSPS-IO for the sequence of “Table Tennis” .............. 83

5.8 Comparison for frame #130 of “Table Tennis” (a) The original frame; the predicted frame #130 (left) and error image (right) by (b) original MV, (c) BMA, (d) DMVE, (e) SSPS, (f) SSPS-IO. ................................................................................. 84

5.9 PSNR of BMA, DMVE, SSPS-IO for the sequence of “Flower Garden” .......... 85

5.10 Comparison for frame #76 of “Flower Garden” (a) The original frame; the predicted frame #76 (left) and error image (right) by (b) original MV, (c) BMA, (d) DMVE, (e) SSPS, (f) SSPS-IO. ................................................................................. 86
List of Tables

3.1 Average PSNR (in dB) of the predicted frame using different algorithms .......... 17

3.2 Average PSNR (in dB) of the predicted frame using different algorithms .......... 17

3.3 Average PSNR (in dB) of the predicted frame using different algorithms for the test sequence (a) “Football”, (b) “Salesman”, (c) “Table Tennis” and (d) “Miss America” ................................................................. 33

3.4 Average PSNR (in dB) of the predicted frame using different algorithms for the test sequences .................................................................................. 47

4.1 Average PSNR (in dB) of the predicted frame using different algorithms for the backward prediction ........................................................................ 57

4.2 Average PSNR (in dB) of the predicted frame using different algorithms for the backward prediction ........................................................................ 58

4.3 Average PSNR (in dB) of the predicted frame using different algorithms for the P frames ................................................................................................. 63

4.4 Average PSNR (in dB) of the predicted frame using different algorithms for the P frames ................................................................................................. 63

5.1 Average PSNR (in dB) of different error concealment algorithms ................. 77
Abstract

In this thesis, we investigate three main problems: (1) fast motion estimation algorithms associated with resolution conversion of video, (2) fast motion estimation associated with frame type conversion and (3) motion vector recovery for error concealment of compressed video.

In many video applications, a compressed video sequence needs to be converted to a lower-resolution compressed video. In such applications, one typically needs to decompress the original sequence, down-sample each frame, and re-compress the down-sampled sequence. The challenge is that full re-compression has very high complexity due to the computational intensive motion estimation. In this thesis, we propose three fast motion estimation algorithms (PME, MPME and ME-SVF) for efficient re-compression using the motion vectors in the original compressed video. Our simulation results suggest the proposed fast algorithms can achieve a significantly higher quality than existing fast algorithms with low additional complexity. The proposed algorithms can also provide different trade-off between complexity and accuracy.

In some application such as transcoding video from IPPP format to IBPBP or IBBP format, there are needs to change the frame types of many frames. Again full re-compression can be very complex due to the computational intensive motion estimation. In this thesis, we propose two fast motion estimation algorithms (P2P, P2PS) for the P-frame to P-frame conversion, and two algorithms (P2B, P2BS) for
the P-frame to B-frame conversion. Our simulation results suggest that the proposed algorithms can give very high quality results with much lower complexity than full exhaustive search.

Moreover, delivery of digital video through different kinds of packetized network is very common. During the transmission, motion vectors might be lost or erroneous after going through some noisy channel, and the effect can propagate as the video frames are usually coded differentially. In this thesis, we propose two algorithms (SSPS and SSPS-IO) for the recovery of lost macroblocks and the performance of the proposed algorithms is found to be significantly better and more robust than the existing algorithms.
Acknowledgments

I would like to thank my supervisor, Dr. Oscar Au, for his invaluable guidance, inspiration, patience and encouragement for my studies and research.

I would also like to thank Prof. Ming L. Liou and Dr. Bing ZENG to serve on my thesis committee, and for reviewing my thesis.

I wish to thank my schoolmate, Peter Wong Hon Wah for his help throughout of my research.

Finally, special thanks are dedicated to my parents and Sharon for giving me full support and 100% freedom throughout my life.
Chapter 1

Introduction

1.1 Motivation

Nowadays, delivery of digital video through different kinds of network, such as internet, local area network, wireless network, cable TV network, is very common. Owing to the huge storage size of video and limited bandwidth, most stored video is only available in compressed format, encoded by different kind of video codecs. In order to ensure compatibility among video codecs from different manufacturers and applications and to simplify the development of new applications, intensive efforts have been undertaken in recent years to define digital video standards. These standards were the result of joint development efforts of video and audio compression as well as other system aspects required to support all the applications. The most popular video compression standards such as ISO MPEG-1/2/4, ITU-T H.261 or H.263 employ motion estimation and compensation to reduce temporal redundancy between successive frames in a video sequence in order to achieve high compression efficiency. However, the motion estimation process is a computation intensive operation and typically accounts for more than 60% of the workload of the video encoder [1],[10].

In applications like video browsing, previewing of a movie from Internet, picture-in-picture, transcoding for a lower bitrate video in wireless network and downconversion of HDTV video for low-resolution display, there is a need to reduce the
resolution of the compressed video sequence which is stored in the video server and encode the reduced video into a compatible bitstream for the decoder. In such situations, a straightforward way to scale down the compressed video sequence is to decompress the whole sequence, apply anti-aliasing lowpass filtering, down-sample the video and then re-encode the reduced sequence by conventional video encoders for efficient delivery.

On the other hand, it is well known that encoding video in IPPP format is not efficient in the sense of bitrate so it is more efficient to encode it with B-frames (interpolative frames), like IBP or IBBP format. In some occasions, some compressed videos in IPPP format need to be converted to IBP or IBBP format for efficient delivery and storage.

If brute force motion estimation is used in the recompression of these applications, it will be a heavy burden to the video server that has to generate the compressed video bitstream because of this computational intensive process. Therefore, fast motion estimation is highly desirable. In this thesis, we proposed different fast motion estimation algorithms that exploit the correlation of the motion vectors in the original compressed video and those of the resolution down converted video and the frame-type changed video.

When compressed video is transmitted through hostile channels, motion vectors and other information might be lost or in error. In such cases, the error must be detected for proper decoding. The decoded erroneous video is usually very disturbing and
need to be concealed. In this thesis, we investigate methods to “recover” motion vectors to generate missing blocks.

In chapter 2, an overview of block-based motion estimation will be given. In chapter 3, we propose three fast motion estimation algorithms (the PME, MPME and ME-SVF) for resolution down conversion of video. The three algorithms provide various trade-off between visual quality and complexity. In chapter 4, we propose two fast motion algorithms (P2B and P2BS) for the P-frame to B-frame conversion. We also propose two fast motion estimation algorithms (P2P and P2PS) for the P-frame to P-frame conversion. In chapter 5, we propose two motion vector recovering methods (SPSS, SPSS-IO) for the error concealment of damaged video.
Chapter 2

Review of Block-based Motion Estimation

2.1 Introduction

Motion estimation techniques can be categorized into two main classes: pel-recursion and block matching. The motion vectors can vary from pixel to pixel in Pel-recursion techniques, whereas block matching techniques are used primarily where a single motion vector is applied to a block of pixels usually in square shape (macroblock). The block matching techniques with different matching criteria are widely used because of the simplicity.

2.2 Motion Estimation by Block Matching

To find the best match motion vector for each macroblock, we need to perform a search inside a reference frame to find a reference block that has either maximum correlation or minimum distortion with the original macroblock in the source picture and the reference block in the reference picture. Some distortion measures include mean square error (MSE) and mean absolute difference (MAD) which tend to be more commonly used as the criteria than the correlation. But MSE and MAD are still computationally expensive. MAD is considered as the simplest and most commonly used measure of best match [2].

Let MAD(x,y) be the MAD between a $N \times N$ array of pixels of intensities $\{V_n(x+i, y+j)\}_{i=0,j=0}^{N-1,N-1}$ at macroblock position $(x,y)$ in source frame $n$, and a
corresponding \( N \times N \) array of pixels of intensity \( \{V_{m}(x+dx+i, y+dy+j)\}_{i=0, j=0}^{N-1,N-1} \) at macroblock position \((x + dx, y + dy)\) in reference picture \(m\):

\[
MAD(x, y) = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |V_{s}(x+i, y+j) - V_{m}(x+dx+i, y+dy+j)|
\]

The \( N \times N \) array in frame \(m\) is displaced horizontally by \(dx\) and vertically by \(dy\). By convention, \((x, y)\) refers to the upper left corner of the macroblock positive displacements \(dx\) and \(dy\) are to the right and to the bottom respectively. By omitting the normalization factor \(1/N^2\), the MAD becomes the sum of absolute difference (SAD), which is computationally more efficient than MAD and is equivalent to MAD in finding the best matched block.

A full search over every possible motion displacement is guaranteed to produce the best possible motion vector.

The best match found by motion estimation is then used as a predictor for the block in the source picture, and the displacement between the two blocks is used to define the motion vector associated with the source block. Only the motion vector and the residue block, which is defined as the difference between the source block and the predictor, are encoded in the bitstream. When the prediction is good, the needed bits are fewer than those needed to encode the source block.

### 2.3 Computation Requirement of Full Search

If a maximum displacement of \(W\) pixels/frame is allowed, then we will have \((2W+1)^2\) locations to search for the best match of the source block. The algorithm, which examines all these locations, is called the brute force exhaustive search (or full search
(FS)) and it can be seen that it is very computational intensive. For a frame of size \( P \times Q \) and a frame rate of \( T \) fps, the amount of computation is:

\[
T \cdot \left( \frac{P}{N} \cdot \frac{Q}{N} \right) \cdot (2W + 1)^2 (2N^2 - 1) \approx 8TPQW^2 \approx 1.09 \times 10^{10},
\]

(2)

for \( T=30 \), \( P=288 \), \( Q=360 \) and \( W=21 \). This can require up to 60\% of the computational power of the encoder. Even though there are currently several hardware devices that can implement this approach for video coding, they are mainly used for video production, and they are usually very expensive. Typically the full search algorithm cannot be implemented for other real time applications such as video conferencing or be considered for software implementation due to its huge computational requirement.

There have been several attempts in creating algorithms that can overcome the computational cost of FS, [3-9]. Still, even though these algorithms can make the search faster, they usually incur a significant loss in visual quality. Some examples include 3-step search [3], 2-D Log search [4], and New 3-step search [5].
Chapter 3

Fast Motion Estimation for Resolution Down-Conversion of Compressed Video

3.1 Introduction

In downscaling a video sequence from a frame size of $P \times Q$ to $P/2 \times Q/2$, a $16 \times 16$ macroblock in the reduced video corresponds to four $16\times16$ macroblocks in the original video. In Figure 3.1, it can be seen that the motion vector of the macroblock in the reduced video is closely related to the motion vectors of the four macroblocks in the original video. As the original motion information is available in the compressed bitstream and should be highly correlated to the motion of the reduced video, it is possible to estimate the new motion vectors from the original motion vectors in order to save much of the computational cost. On the other hand, most conventional fast motion estimation algorithms do not exploit this correlation. There are some existing algorithms for predicting the new motion vectors from the original motion information [12] but they do not perform very well.

![Diagram](image)

Figure 3.1 Possible relationship among motion vectors of the original frame and the reduced frame
In this chapter, we propose a new motion estimation algorithm called predictive motion estimation (PME) to estimate the optimal motion vector for the reduced video sequence using the motion information of the original compressed video sequence. We will show that PME is much better than the existing algorithms.

3.2 Existing Algorithms

Let \( \{\text{MB}_{r,i}\}_{i=1}^{4} \) be four macroblocks of the original video with corresponding motion vectors \( \{\nu_{o,i}\}_{i=1}^{4} \). The four macroblocks are reduced to form one macroblock MB\(_r\) of the reduced video with the corresponding motion vector \( V_r \).

The simplest algorithm of getting the new motion vectors is to take the mean of the motion vectors of the four corresponding macroblocks in the original video and reduce it by one half and we call this MEAN which can be represented by the following equation:

\[
V_r = \frac{1}{2} \text{truncate} \left( \frac{\sum_{i=1}^{4} V_{o,i}}{4} \right)
\]

The MEAN is a good estimate if the four macroblocks have motion vectors pointing to similar directions with similar magnitude, as in the case of Figure 3.1. However, if the four motion vectors have different directions and/or magnitude, the resulting MEAN can be poor and meaningless as in the case of Figure 3.2. The simple arithmetic mean of the four different motion vectors is inadequate to describe all situations.
In [12], an algorithm called adaptive motion vector resampling (AMVR) was proposed to estimate the motion vector of the reduced video using a weighted mean of the original motion vectors. In the compressed video bitstream of MPEG-1, MPEG-2, H.261 and H.263, only the inter-coded macroblocks in the interframes (P or B frames) contain motion vectors information. AMVR applies only to these inter-coded blocks. AMVR uses a weighted mean of the four $V_o$ to estimate $V_r$,

$$V_r = \frac{1}{2} \sum_{i=1}^{4} V_{o,i} \cdot A_i = \frac{1}{2} \sum_{i=1}^{4} A_i$$

where $A_i$ denotes the activity measurement of residual macroblock $i$ (in the original video). In [12], the number of non-zero AC coefficients was used as $A_i$. The complexity of AMVR is also very low but the performance is not necessarily good. Actually, it gives a rather low peak signal-to-noise ratio (PSNR) when compared with full search in the reduced video.
3.3 Predictive Motion Estimation (PME)

Here we propose a new algorithm to predict the motion vector for the reduced video with significantly higher accuracy than AMVR. We will construct a weighted motion vector different from AMVR as a candidate. We will also consider the four original motion vector \( \mathbf{V}_{o,i} \) as four other candidates. All five candidates will be tested in search of the best motion vector.

By examining many simulated data, we observe that the motion vector of the MB, in the reduced video found by full search are, in many cases, very close to or even the same as one or more of the four original motion vectors reduced by one half. In these cases, selectively choosing one of the \( \mathbf{V}_{o,i} \) is more appropriate than taking the mean or weighted mean. In addition, all four original motion vectors appear to be equally likely. Thus, they should all be considered possible candidates. In order to choose the optimal motion vector from the candidates, we will go through the following steps for each reduced macroblock.

**Step 1:** If the 4 reference motion vector \( V_o \) \((V_{o,1}, V_{o,2}, V_{o,3}, V_{o,4})\) in the original video are the same, we will just take the common value and reduce it by half as the \( V_r \). Otherwise, we will go through the following step 2 to step 6 to get the \( V_r \).

**Step 2:** Compute 4 candidate vectors \( (V_{r,i}) \)

\[
V_{r,i} = \text{truncate}(V_{o,i})/2
\]

for \( i = 1, 2, ..., 4 \)

The \( V_{r,i} \) will be in half pixel precision.
Step 3: Compute the mean absolute difference ($MAD_i$) in the reduced video for each of the four candidates $\{V_{r,i}, i = 1, 2, \ldots, 4\}$. If one of the $MAD_i$ value is equal to zero, assign the corresponding $V_{r,i}$ as the new $V_r$ and stop searching. Otherwise, go to step 4.

Step 4: Compute a new candidate vector $V_{r,5}$ by

$$V_{r,5} = \frac{\sum_{i=1}^{4} \left[ V_{r,i} \cdot \left( \frac{1}{MAD_i} \right) \right]}{\sum_{i=1}^{4} \left( \frac{1}{MAD_i} \right)}$$

Step 5: If the $V_{r,5}$ is not equal to one of the first 4 candidates, compute $MAD_5$ using $V_{r,5}$ as the motion vector.

Step 6: The $V_{r,i}$ with the minimum $MAD$ value is chosen as the final $V_r$.

We call this algorithm predictive motion estimation (PME). For the MBs that are skipped blocks or intra blocks which do not provide any motion information, we set the $V_o$ of these blocks to be zero. The rationale for constructing the new candidate vector in step 4 is as follows. For any $V_{r,i}$ with a small MAD, it is a good candidate and should be given a higher weight. For any $V_{r,i}$ with large MAD, it is not a very good match, the weight should be smaller. Note that with our definition, it is impossible to compute $V_{r,5}$ without first computing the MAD for $\{V_{r,i}, i = 1, 2, 3, 4\}$.
For better performance, we can perform an optional additional step. We can search the 4 surrounding full or half pixel, which are shown in Figure 3.3, of the optimal $V^*$ found by PME. We call the PME with a local full pixel search the PME-LS1. The PME with a local half-pixel search will be called PME-LS2. Both algorithms have the following additional step.

**Step 7:** Compute the MAD for the surrounding 4 full pixels or half pixels (shaded region of Figure 3.3). The one with the minimum MAD will be the new motion vector $V^*$.

Note that 8 half pixel search is needed anyway if half pixel accuracy is needed in the full search or any other fast search algorithms. We considered searching the 8 surround points and found that it was not better than searching only the 4 shaded points in Figure 3.3.

![Figure 3.3 Local search points of PME-LS1 and PME-LS2](image)

### 3.4 Modified Predictive Motion Estimation (MPME)

In the previous sections, we only evaluated the performance of down-conversion of each side (length-wise and width-wise) by a factor of 2, i.e. $2 \times 2$ macroblocks to 1
macroblock (4-to-1) conversion, of different algorithms. In this section, we will evaluate the performance of PME in other down-conversion cases, e.g. down-conversion of each side by 3 and 4 (9-to-1 & 16-to-1). For downscaling from n to 1 using PME, if the n is large, the search points of PME will increase resulting in increased computation. In order to reduce the computations, we propose a modified predictive motion estimation (MPME) by selective reduction of search points.

For n-to-1 video down-conversion, each side of a frame is reduced by $\sqrt{n}$, where $\sqrt{n}$ is integer. Let $\{MB_{n,i}\}_{i=1}^{n}$ be n macroblocks of the original video with corresponding motion vectors $\{V_{n,i}\}_{i=1}^{n}$. The $n$ macroblocks are reduced to form one macroblock $MB_r$ of the reduced video with corresponding motion vector $V_r$.

Note that PME performs a search only when the candidate motion vectors disagree with each other. We observe that having two or more objects moving to different locations account for most of the disagreement and unfortunately the probability of this situation increases with $n$. The different object almost always appears at the corner blocks of these $n$ macroblocks. In light of these observations, we propose the modified predictive motion estimation (MPME) as PME with step 3 changed. In step 3 of MPME, we search only five candidate motion vectors: those of the four corner blocks and the mean of the center blocks, which are the shaded blocks in Figure 3.4. For the 9-to-1 conversion, there is only one center block. For the 16-to-1 conversion, there are only four center blocks. In general, for $n$-to-1 conversion, there are $(\sqrt{n-2})^2$ center blocks.
Figure 3.4 (a) 9-to-1 conversion and (b) 16-to-1 conversion.

For the 16-to-1 conversion, we only use the mean of the four motion vectors as one candidate for the four macroblocks in the center shown in Figure 3.4(b) instead of searching all four candidates separately. Computation is reduced and we only have 5 candidates for both cases.

### 3.5 Motion Estimation by Spatial-Variant Filtering

In previous sections, we proposed PME and MPME and the performance are good in terms of PSNR. However, they involve the block matching which will increase the computations. In this section, we are interested in improving the existing algorithms that do not perform any search. This includes MEAN and AMVR, but not PME. Without the searching, MEAN and AMVR are much faster than PME, but are significantly worse than PME. The proposed algorithm follows a similar trait, but yields better PSNR than MEAN and AMVR. The proposed algorithm, motion estimation by spatial-variant filtering or ME-SVF, predicts the motion vector for the $n$-to-1 reduced video by applying a spatial varying filter to the motion vector of the original high-resolution video.
In a way, ME-SVF improves on a "target", which can be the MEAN or AMVR. Let the "target" be MEAN. While the target may not be the best, it is a reasonable value. Among the \( n \) candidate motion vectors, if a vector is close to the target, it is probably quite reliable and thus should be given a larger weight. On the other hand, if a candidate vector is far away from the target, it is probably quite unreliable and thus should be given a lower weight. It is with this observation that the spatial filter is created.

The new motion vector \( V_r \) is obtained with the following spatial-variant filter equation

\[
V_r = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} V_{o,i} \cdot D_i \sum_{i=1}^{n} D_i
\]

which is essentially a weighted average of the \( n \) candidate motion vectors \( V_{o,i} \) for \( i = 1, 2, \ldots, n \), with spatially varying weights \( D_i \) for \( i = 1, 2, \ldots, n \). The weight \( D_i \) is the mapped value of the difference between the target and \( V_{o,i} \).

We observe that the MEAN can sometimes be a very good estimate if the \( n \) motion vectors are similar. In this case, the \( V_r \) from ME-SVF will be very similar to MEAN because all candidate motion vectors will have similar weights. The MEAN can be poor and meaningless if the \( n \) motion vectors have different directions and/or magnitude. In this case, the \( V_r \) from ME-SVF will give a larger weight to any candidate vector that is close to the target vector, and smaller weight to those far away from the target. This will selectively suppress the bad outliers, and give more weight to the more reasonable candidates. It will tend to give more weights to the candidate vectors that are clustered together.
We have tried some possible targets and maps. In our experiment, we found that MEAN is a reasonable good target and is easy to compute. Let this target motion vector be \( V_m \). An example of the weighting function used is

\[
D_i = \begin{cases} 
1 & \text{if } d_i < C_1, \\
1 - (d_i - C_1) / (C_2 - C_1) & \text{if } C_1 < d_i < C_2, \\
0 & \text{otherwise}.
\end{cases}
\]

where

\[
d_i = (V_{i,x} - V_{m,x})^2 + (V_{i,y} - V_{m,y})^2
\]

is the square Euclidean distance between \( V_i \) and \( V_m \), and \( C_1 \) and \( C_2 \) are predefined constant. We call this algorithm motion estimation by spatial variant filtering (ME-SVF).

### 3.6 Simulation Results

We tested our algorithms on several MPEG I video sequence ("Football", "Table Tennis", Miss America" and "Salesman") with SIF resolution (352×240) and a GOP of 15 frames with IPPP frame structure in each GOP. In order to have an objective comparison of the performance for different algorithms we calculate the peak signal to noise ratio (PSNR) between the motion compensated frames of the reduced video using different algorithms and the original reduced frames.

#### 3.6.1 Simulation Results of PME

In the simulation, each side of the frame is downscaled by 2 and the motion vectors are estimated by using half of the average value of four original motion vectors (MEAN), adaptive motion vector resampling (AMVR), proposed predictive motion estimation (PME) with and without local search (PME-LS) and full search. The full
search we used is with half pixel accuracy and search window is ±7. In fact that the search window of full search is ±7 in both the original motion vectors in high-resolution video and low-resolution video whereas the maximum motion vectors values will be one half of the full search for other algorithms. The average PSNR (dB) values of the predicted frames using different algorithms are shown in Table 3.1 and Table 3.2.

<table>
<thead>
<tr>
<th></th>
<th>Football</th>
<th>Table Tennis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR (dB)</td>
<td>Avg. Search pt</td>
</tr>
<tr>
<td>MEAN (dB)</td>
<td>24.05</td>
<td>*</td>
</tr>
<tr>
<td>AMVR</td>
<td>24.23</td>
<td>*</td>
</tr>
<tr>
<td>PV4</td>
<td>24.78</td>
<td>2.2</td>
</tr>
<tr>
<td>PME</td>
<td>24.92</td>
<td>2.6</td>
</tr>
<tr>
<td>PME-LS1</td>
<td>25.16</td>
<td>5.1</td>
</tr>
<tr>
<td>PME-LS2</td>
<td>25.12</td>
<td>5.1</td>
</tr>
<tr>
<td>Full Search</td>
<td>25.87</td>
<td>225+8</td>
</tr>
</tbody>
</table>

Table 3.1 Average PSNR (in dB) of the predicted frame using different algorithms

<table>
<thead>
<tr>
<th></th>
<th>Salesman</th>
<th>Miss USA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR (dB)</td>
<td>Avg. Search pt</td>
</tr>
<tr>
<td>MEAN (dB)</td>
<td>37.99</td>
<td>*</td>
</tr>
<tr>
<td>AMVR</td>
<td>37.73</td>
<td>*</td>
</tr>
<tr>
<td>PV4</td>
<td>38.82</td>
<td>0.65</td>
</tr>
<tr>
<td>PME</td>
<td>38.93</td>
<td>0.67</td>
</tr>
<tr>
<td>PME-LS1</td>
<td>38.94</td>
<td>1.7</td>
</tr>
<tr>
<td>PME-LS2</td>
<td>39.02</td>
<td>1.7</td>
</tr>
<tr>
<td>Full Search</td>
<td>39.02</td>
<td>225+8</td>
</tr>
</tbody>
</table>

Table 3.2 Average PSNR (in dB) of the predicted frame using different algorithms

From Table 3.1 & Table 3.2, it seems that PME-LS1 has slightly better performance than PME-LS2 in the fast moving sequence “Football”. However, for other sequences, PME-LS2 is significantly better. The average PSNR of PME-LS1 and PME-LS2 for the sequence “Football” and “Table Tennis” respectively are about 0.92 higher than AMVR. For the sequence “Salesman” and “Miss America”, the gain over
AMVR are about 1.3 and 1.1 respectively. For all of the test sequences apart from “Football”, the performance of PME-LS2 is very close to the full search.

If we use the PME without step 3 and step 4, i.e. without the $V_r,5$ (we call this PV4 in Table 3.1), the PSNR will be about 0.1dB lower than PME and with only a negligible decrease in average search points. This shows that the fifth candidate $V_r,5$ can improve the best match searching with insignificant increase in complexity.

In PME, we calculate the MAD for at most 5 points for each macroblock and do not need to calculate for those blocks with four identical motion vectors. Moreover, the chance that two or three motion vectors are identical is very high and we only need to calculate the MAD for fewer than five points. In the case of the fast moving scenes in the “Football” sequence, the average search points for each motion vector is about 2.6 that is much smaller than full search of 225 search points. In the case of head-and-shoulder scene like salesman, the average search point is about 0.7. We can achieve an even better quality with the additional local search in PME-LS1 or -LS2 at the price of more search points.

The PSNR of predicted frames using different algorithms for the sequence “Football” is shown in Figure 3.5 and the difference of PSNR of the predicted frames between PME-LS1 and AMVR is shown in Figure 3.6. In Figure 3.5, it shows that the PSNR of using different algorithm are very similar, as the sequence “Football” has not much motion before frame #70. The average PSNR after frame #70 of PME-LS1 is about 1.3 dB higher than AMVR and the maximum gain is about 3.6 dB.
Figure 3.5 PSNR of predicted frames using different algorithms for the sequence "Football"

Figure 3.6 The difference of PSNR of the predicted frames between PME-LS1 and AMVR for the sequence "Football"
The original reduced frame #125 of "Football" is shown in Figure 3.7. From the Figure 3.8, we can compare different algorithms by the predicted frames and error frames. From the error frames, we can clearly see that the distortion of PME and PME-LS1 (Figure 3.8(b)&(c)) are very close to that in full search (Figure 3.8(a)), whereas, the distortion in AMVR (Figure 3.8(d)), and MEAN are much higher in the center portion of the frame.

Figure 3.7 Original reduced frame #125 of "Football"
Figure 3.8 Comparison for the frame #125 of "Football": the predicted frame #125 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS1 (d) AMVR and (e) MEAN.
The PSNR of predicted frames using different algorithms for the sequence “Salesman”, “Table Tennis” and “Miss America” are shown in Figure 3.9, Figure 3.13 and Figure 3.17 respectively. Moreover, the difference of PSNR of the predicted frames between PME-LS2 and AMVR of these three test sequences are shown in Figure 3.10, Figure 3.14 and Figure 3.18. In Figure 3.9, it shows that the PSNR of using PME-LS2 is almost the same as full search and it is always higher than other algorithms.

The original reduced frame #88 of “Salesman”, #140 of “Table Tennis” and #53 of “Miss America” are shown in Figure 3.11, Figure 3.15 and Figure 3.19 respectively. From the Figure 3.12, Figure 3.16 and Figure 3.20, we can compare different algorithms by the predicted frames and error frames for these three test sequences. Similar to the case of test sequence “Football”, the distortion shown in the error frame by using PME or PME-LS2 are very close to full search and is lower than that of AMVR.
Figure 3.9 PSNR of predicted frames using different algorithms for the sequence "Salesman"

Figure 3.10 The difference of PSNR of the predicted frames between PME-LS2 and AMVR for the sequence "Salesman"
Figure 3.11 Original reduced frame #88 of “Salesman”
Figure 3.12 Comparison for the frame #88 of "Salesman"; the predicted frame #88 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS2 (d) AMVR and (e) MEAN.
Figure 3.13 PSNR of predicted frames using different algorithms for the sequence “Table Tennis”

Figure 3.14 The difference of PSNR of the predicted frames between PME-LS2 and AMVR for the sequence “Table Tennis”
Figure 3.15 Original reduced frame #140 of "Table Tennis"
Figure 3.16 Comparison for the frame #140 of "Table Tennis"; the predicted frame #140 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS2 (d) AMVR and (e) MEAN.
Figure 3.17 PSNR of predicted frames using different algorithms for the sequence “Miss America”

Figure 3.18 The difference of PSNR of the predicted frames between PME-LS2 and AMVR for the sequence “Miss America”
Figure 3.19 Original reduced frame #53 of “Miss America”

(a)

(b)
Figure 3.20 Comparison for the frame #53 of "Miss America"; the predicted frame #53 (left) and error image (right) by (a) full search, (b) PME, (c) PME-LS1 (d) AMVR and (e) MEAN.
3.6.2 Simulation results of MPME

In MPME, we are working on the $n$-to-1 down conversion where $n$ equal to 9 or 16. The motion vectors are estimated by using the average value of the four original motion vectors and divided by $\sqrt{n}$ (MEAN), adaptive motion vector resampling (AMVR), predictive motion estimation (PME) with and without local search (PME-LS2), modified PME (MPME), modified PME-LS2 (MPME-LS) and full search. The full search we used is with half pixel accuracy and search window is $\pm 7$ for $n = 9$ and $\pm 5$ for $n = 16$.

The average PSNR and average search points per macroblock of the predicted frame using different algorithms for all test sequences are shown in Table 3.3.

<table>
<thead>
<tr>
<th>Football</th>
<th>9-to-1</th>
<th>16-to-1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR (dB)</td>
<td>Avg. Search pt</td>
</tr>
<tr>
<td>MEAN</td>
<td>25.21</td>
<td>*</td>
</tr>
<tr>
<td>AMVR</td>
<td>25.29</td>
<td>*</td>
</tr>
<tr>
<td>PME</td>
<td>26.24</td>
<td>4.1</td>
</tr>
<tr>
<td>PME-LS2</td>
<td>26.39</td>
<td>6.9</td>
</tr>
<tr>
<td>MPME</td>
<td>26.16</td>
<td>2.9</td>
</tr>
<tr>
<td>MPME-LS</td>
<td>26.35</td>
<td>5.5</td>
</tr>
<tr>
<td>Full Search</td>
<td>26.47</td>
<td>225+8</td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th>Salesman</th>
<th>9-to-1</th>
<th>16-to-1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR (dB)</td>
<td>Avg. Search pt</td>
</tr>
<tr>
<td>MEAN</td>
<td>39.29</td>
<td>*</td>
</tr>
<tr>
<td>AMVR</td>
<td>38.77</td>
<td>*</td>
</tr>
<tr>
<td>PME</td>
<td>40.12</td>
<td>1.2</td>
</tr>
<tr>
<td>PME-LS2</td>
<td>40.15</td>
<td>2.8</td>
</tr>
<tr>
<td>MPME</td>
<td>40.11</td>
<td>0.8</td>
</tr>
<tr>
<td>MPME-LS</td>
<td>40.15</td>
<td>2.1</td>
</tr>
<tr>
<td>Full Search</td>
<td>40.15</td>
<td>225+8</td>
</tr>
</tbody>
</table>

(b)
Table 3.3 Average PSNR (in dB) of the predicted frame using different algorithms for the test sequence (a) “Football”, (b) “Salesman”, (c) “Table Tennis” and (d) “Miss America”

For the case of 9-to-1 conversion, we calculate the MAD for at most 6 points for each macroblock in MPME instead of 11 points in PME and do not need to calculate for those blocks with identical candidate motion vectors. In the case of the fast moving sequence like “Football”, the average search points for each motion vector is about 2.9 for MPME and 4.2 for PME that is about 1.45 times speed up with 0.1dB lost in PSNR. In the case of the head-and-shoulder sequence like salesman, the average search point is about 0.88 for MPME and 1.18 for PME that is 1.34 times speed up with a negligible lost in PSNR. We can achieve a even better quality with the additional half-pixel local search (MPME-LS) at the price of more search points. The
PSNR of MPME-LS is about 0.04dB and 0.1dB lower than PME-LS2 and full search respectively in the both “Football” and “Table Tennis” sequences. For the sequences “Salesman” and “Miss America”, the PSNR of MPME-LS are nearly the same as PME-LS2 and full search.

The simulation results of all test sequence are plotted in Figure 3.21 to Figure 3.24. Graph (a) show the PSNR of each frame, (b) show the PSNR of full search minus AMVR, MPME and MPME-LS and (c) show the PSNR of PME-LS2 minus MPME and MPME-LS. From the graphs, we can see that the performance of MPME is much better than AMVR and that of MPME-LS is very close to PME-LS2 and full search for all test sequence.

The original reduced frame #96 of “Football”, #92 of “Salesman”, #24 of “Table Tennis” and #96 of “Miss America” with the comparison of different algorithms by the predicted frames and error frames for the corresponding frames are shown in Figure 3.25 to Figure 3.32. From the error frames, we can clearly see that the distortion of MPME and MPME-LS are very close to that in PME and PME-LS2 and also full search, however, AMVR and MEAN produce much higher distortion.
Figure 3.21 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence "Football"
Figure 3.22 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence "Salesman"
Figure 3.23 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence “Table Tennis”
Figure 3.24 (a) PSNR of predicted frames using different algorithms, (b) PSNR difference AMVR, MPME and MPME-LS VS Full Search, (c) PSNR difference of MPME and MPME-LS VS PME-LS2 for the sequence “Miss America”
Figure 3.25 Original reduced frame #96 of "Football"
Figure 3.26 Comparison for the frame #96 of “Football”; the predicted frame #96 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (e) MEAN.
Figure 3.27 Original reduced frame #92 of "Salesman"
Figure 3.28 Comparison for the frame #92 of “Salesman”, the predicted frame #92 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (e) MEAN.
Figure 3.30 Comparison for the frame #24 of "Table Tennis": the predicted frame #24 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (g) MEAN.
Figure 3.31 Original reduced frame #96 of "Miss America"
Figure 3.32 Comparison for the frame #96 of "Miss America"; the predicted frame #96 (left) and error image (right) by (a) full search, (b) MPME, (c) MPME-LS (d) PME, (e) PME-LS2, (f) AMVR and (e) MEAN.
3.6.3 Simulation results of ME-SVF

We applied ME-SVF to the 9-to-1 and 16-to-1 down-conversion. The motion vectors are estimated by using the MEAN, AMVR, proposed ME-SVF and full search then calculate the PSNR between the predicted frames of the reduced video using different algorithms and the original reduced frames. Note that the MEAN, AMVR and full search are the same as those in the simulations of MPME.

Table 3.4 shows the mean PSNR values of different algorithms, the difference in mean PSNR values between full search and different algorithms for the test sequences. For the 9-to-1 conversion of “Football”, ME-SVF is 0.26dB higher in PSNR than AMVR, whereas for “Salesman”, ME-SVF is 1.31dB higher than AMVR. In the 16-to-1 conversion, the PSNR of ME-SVF gain over AMVR around 0.1dB for four test sequences.

<table>
<thead>
<tr>
<th></th>
<th>Football</th>
<th>Salesman</th>
<th>Table Tennis</th>
<th>Miss America</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>9-to-1</td>
<td>16-to-1</td>
<td>9-to-1</td>
<td>16-to-1</td>
</tr>
<tr>
<td>MEAN</td>
<td>25.21</td>
<td>25.32</td>
<td>39.29</td>
<td>40.17</td>
</tr>
<tr>
<td>AMVR</td>
<td>25.29</td>
<td>25.40</td>
<td>38.77</td>
<td>40.13</td>
</tr>
<tr>
<td>ME-SVF</td>
<td>25.55</td>
<td>25.51</td>
<td>40.08</td>
<td>40.21</td>
</tr>
<tr>
<td>Full Search</td>
<td>26.47</td>
<td>26.10</td>
<td>40.15</td>
<td>40.23</td>
</tr>
</tbody>
</table>

Table 3.4 Average PSNR (in dB) of the predicted frame using different algorithms for the test sequences

Figure 3.33 and Figure 3.34 show the PSNR difference between ME-SVF and AMVR for the “Football” and “Salesman” sequences. In Figure 3.33, we notice that the performance of ME-SVF is somehow similar to AMVR before the frame #70, However, ME-SVF is obviously better than AMVR after frame #70, as the sequence
"Football" has not much motion before frame #70. From Figure 3.34, ME-SVF shows a much better performance over AMVR and the maximum gain is over 6dB.

Figure 3.33 PSNR difference between ME-SVF and AMVR for "Football"

Figure 3.34 PSNR difference between ME-SVF and AMVR for "Salesman"
3.7 Conclusion

In this chapter, we propose different fast algorithms to estimate the best motion vector for reduced resolution video sequences using the motion information of the original compressed video sequence. We propose predictive motion estimation (PME), modified predictive motion estimation (MPME) and motion estimation by spatial-variant filtering (ME-SVF).

PME target on the 4-to-1 conversion and the computational need of the proposed algorithm is minimal and the prediction is much better than AMVR and MEAN in terms of PSNR.

MPME is used to reduce the complexity of PME in the 9-to-1 and 16-to-1 conversion. The computational need of the modified algorithm is reduced with only very minor degradation when comparing with PME in terms of PSNR.

ME-SVF improves the existing algorithms that do not perform any search. This includes MEAN and AMVR, but not PME. Without the searching, MEAN and AMVR are much faster than PME, but are significantly worse than PME. The proposed algorithm follows a similar trait, but yields better PSNR and more robust to different scene than MEAN and AMVR.
Chapter 4

Fast Motion Estimation for Frame Type Conversion

4.1 Introduction

It is well known that encoding video in IPPP format is not efficient in the sense of bit rate. It is more efficient to encode it with B-frames (interpolative frames), such as in IBP or IBBP format. In some occasions, some compressed videos in IPPP format need to be converted to IBP or IBBP format for efficient delivery.

One example is any ITU-T H.261 video that needs to be converted to MPEG-I / II video and the original uncompressed sequence is not available. The H.261 video is in IPPP format and one needs to transcode it to IBP or IBBP format for more efficient compression. However, full re-compression involves motion estimation, which is a computation intensive process.

Another example is that one often encounters large compressed video files (e.g. MPEG file) on the internet or Web TV but hesitates whether to spend a significant amount of time and money to download such large files. A fast playback preview version of the video can give an overview of the video for the user and the file size can be reduced significantly. This preview video should still be compressed with the same standard method as the original compressed video such that the same video decoder can be used to view it. One simple way of fast playback is to discard some frames and then re-compress it again but it may cause the change in the frame type
structure. For a MPEG bitstream, it has I-frame (intra-frame), P-frame (predictive-frame) and B-frame (interpolated-frame). As shown in Figure 4.1, all P- and B-frames are predicted from I- and P-frames (anchor frames). So the simplest way of getting a fast playback MPEG bitstream is to discard the B-frames because they are not referenced by any other frames. The frame type structure is changed from IBBP to IPPP format. For efficient coding, this IPPP format can be further converted to IBP or IBBP format. And again, full recompression of the IBBP format video is a computation intensive process due to the motion estimation.

![Figure 4.1 Extraction of a fast playback preview video from a compressed video sequence with IBBP frame structure.](image)

In this chapter, we propose some fast motion estimation algorithms to reduce the computation requirement of the conversion from IPPP format to IBP format.

### 4.2 Fast Motion Estimation for Frame Type Conversion

Figure 4.2 shows an example in which the video in IPPP format is converted to the IBP format. In this process, frame $k$ is converted from P-frame to B-frame and frame $k+1$ is a P-frame predicted from the frame $k-1$. This involves two frame type conversion: (i) P-to-B frame conversion and, (ii) P-to-P frame conversion. To reduce the computation requirement of motion estimation of these two conversions, we can apply existing fast searching algorithms, e.g. 3-Step-Search (3SS) [3]. However, as
the original forward motion vectors are available, we propose four algorithms P2B, P2BS, P2P, P2PS to estimate the motion vectors by exploiting the correlation with the original motion information.

![Diagram](image)

Figure 4.2 Available motion information

### 4.2.1 P-to-B Frame Conversion

For the P-to-B frame conversion, we observe that in many cases motion vectors of the backward prediction are highly related to the motion vectors of the original forward prediction between two P-frames. This is probably because the forward and backward prediction are performed on the same pair of frames, and the resulting object motions should be opposite in direction but with same magnitude. Figure 4.3(b) shows that the backward motion vector of the macroblock at the position (x,y) in frame k (\(MB_k(x,y)\)) should be close to the negative values of forward motion vector of \(MB_{k+1}(x,y)\) shown in Figure 4.3(a).
Let $v_{k,B}(x,y)$ be the backward motion vector at location $(x,y)$ found by full search for frame $k$ (a B-frame in the IBP format) and let $v_{k+1,P}(x,y)$ be the original forward motion vector at location $(x,y)$ of frame $k+1$ (a P-frame in the IPPP format). Let $v'_{k,B}(x,y)$ be the predicted version of $v_{k,B}(x,y)$ to be computed.

We propose to use the negative value of $v_{k+1,P}(x,y)$ as $v'_{k,B}(x,y)$, i.e.,

$$v'_{k,B}(x,y) = -v_{k+1,P}(x,y) \quad \text{for all } x, y. \quad (1)$$

However, in the case of the conversion of the frame $k$ where frame $k+1$ is an I-frame ($k+1 = \text{GOP}$) in the IPPP format, the $v_{k+1,P}(x,y)$ is not available. In this situation, we use the negative value of $v_{k,P}(x,y)$ as the $v'_{k,B}(x,y)$ which is a poorer prediction than $-v_{k+1,P}(x,y)$. So the equation (1) becomes

$$v'_{k,B}(x,y) = \begin{cases} -v_{k+1,P}(x,y) & \text{for } k+1 \neq \text{GOP}, \\ -v_{k,P}(x,y) & \text{otherwise.} \end{cases} \quad (2)$$

and we called this algorithm P2B.

We will show in the simulation results that P2B a reasonably good prediction of the $v_{k,B}(x,y)$. However, the performance of P2B can be poor in terms of PSNR when
compared with Full Search (FS). Figure 4.4 shows one of the situations that P2B can fail to estimate the $v_{k,B}(x,y)$ accurately. To overcome the shortcomings of P2B, we propose a modification to it.

Figure 4.4 The motion estimation of frame $k$ & $k+1$ with: (a) P-to-P frame forward prediction (b) B-to-P frame backward prediction

In Figure 4.4, the $v_{k,B}(x,y)$ is not related to the $v_{k+1,P}(x,y)$ but is actually more related to the motion vector $v_{k+1,P}(x+16,y)$ of the neighboring macroblock, $MB_{k+1}(x+16,y)$, that is next to the $MB_{k+1}(x,y)$. This suggests that the original motion vectors of the neighboring macroblocks can possibly be good prediction. Here we propose to use the $v'_{k,B}(x,y)$ found by P2B and the $v'_{k,B}$ from the eight neighboring macroblocks, which are $v'_{k,B}(x-16,y-16)$, $v'_{k,B}(x,y-16)$, $v'_{k,B}(x+16,y-16)$, $v'_{k,B}(x-16,y)$, $v'_{k,B}(x+16,y)$, $v'_{k,B}(x-16,y+16)$, $v'_{k,B}(x,y+16)$ and $v'_{k,B}(x+16,y+16)$. We compute the MAD of these nine candidate motion vectors and find the best one. We called this modification the P2B with search (P2BS).

In order to get a better prediction, an additional half-pixel search can be performed and we called this P2BS-LS (P2BS with local search). The half-pixel search involves computing the MAD of the surrounding 8 locations which are half-pixel away.
4.2.2 P-to-P frame Conversion

Notice that the motion vectors $v_{k+1,P}(x,y)$ of the original frame $k+1$ (a P-frame in the IPPP format) is predicted with respect to frame $k$ and the motion vectors $v'_{k+1,P}(x,y)$ of the re-encoded P-frame $k+1$ is predicted with respect to the frame $k-1$. We found that it is possible to trace the motion vector $v'_{k+1,P}(x,y)$ by using $v_{k,P}(x,y)$ and $v_{k+1,P}(x,y)$ or the neighboring $v_{k,P}$. Figure 4.5 shows an example of the object motion tracing. We can see that $v'_{k+1,P}(x,y)$ is close to $v_{k,P}(x',y') + v_{k+1,P}(x,y)$ for some $(x',y')$. When tracing the object, the predicted block for frame $k+1$ will usually overlap with four macroblocks in frame $k$. In this example, as the overlapping region of the predicted block and the macroblock $MB_{k,P}(x,y-16)$ is the largest, the object in $MB_{k+1,P}(x,y)$ is more likely matched to $MB_{k,P}(x,y-16)$ than $MB_{k,P}(x,y)$. It is more reasonable to use the $v_{k,P}(x,y-16) + v_{k+1,P}(x,y)$ as the $v'_{k,P+1}(x,y)$.

![Diagram](image)

Figure 4.5 The forward motion estimation of frame k-1, k & k+1 with: (a) original P-frame forward prediction (b) re-encoded P-frame forward prediction
Let \((x', y')\) be the position of the macroblock in frame \(k\) that gives the largest overlapping region with the predicted block associated with \(v_{k+1, P}(x, y)\). We propose to use \(v'_{k, P}(x, y) = v_{k, P}(x', y') + v_{k, P+1}(x, y)\) and we called this algorithm P2P. From our simulation results, P2P can be poor in terms of PSNR when compared with FS. To improve the performance, we further propose a modification of P2P. Similar to the candidate search in P2BS, we propose to use the original motion vectors \(v_{k, P}\) of the four macroblocks in the frame \(k\) which overlaps with the predicted block associated with \(v_{k+1, P}(x, y)\). With the four candidate vectors, we compute the MAD and choose the one with minimum MAD. We called this algorithm P2P with search (P2PS). In order to get a better prediction, an additional half-pixel search can be performed and we called it P2PS-LS.

### 4.5 Simulation Results

We tested our algorithms by converting several MPEG I video sequence ("Football", "Table Tennis", Miss America" and "Salesman") with SIF resolution (352×240) and a GOP of 30 frames with IPPP frame structure in each GOP into IBP frame structure. In order to have an objective comparison of the performance for different algorithms we calculate the peak signal to noise ratio (PSNR) between the motion compensated frames of the video using different algorithms and the original frames.

#### 4.5.1 Simulation Results of P-to-B Frame Conversion

In our simulation, we construct the motion compensated B-frames by using only the backward motion vectors that are estimated by using 3-Step-Search (3SS), the proposed P2B, P2BS, P2BS-LS and full search. The full search we used is with half pixel accuracy and a search window from -7 to +7 (FS-7). The average PSNR (dB)
values and average search points per macroblock of the predicted B-frames using different algorithms are shown in Table 4.1 and Table 4.2.

Table 4.1 and Table 4.2 show that the average PSNR of P2B, with no search, is already higher than 3SS except possibly in “Football”. It shows that P2B can give an acceptable estimate of the motion vectors. However, the performance is poor when compared with full search. With a candidate search, the P2BS achieves a PSNR larger that of P2B by 1dB, 1.5dB, 0.5dB and 0.5dB for the sequences “Football”, “Table Tennis”, “Salesman” and “Miss America” respectively. The average search points per macroblock are about 2 to 5 points, which is much smaller than the 25 search points of 3SS. This suggests that the candidate search can really improve the quality of the motion vectors of P2B, with only a very small computation requirement. The optimal local search can further increase the PSNR by 0.3dB, 0.2dB, 0.12dB and 0.75 dB in the four sequences. Notice that the PSNR of P2BS-LS is very close to that of FS-7, which has half-pixel accuracy.

<table>
<thead>
<tr>
<th></th>
<th>Football</th>
<th>Table Tennis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR (dB)</td>
<td>Avg. Search pt</td>
</tr>
<tr>
<td>3 SS</td>
<td>24.39</td>
<td>25</td>
</tr>
<tr>
<td>P2B</td>
<td>23.96</td>
<td></td>
</tr>
<tr>
<td>P2BS</td>
<td>24.92</td>
<td>5.44</td>
</tr>
<tr>
<td>P2BS-LS</td>
<td>25.20</td>
<td>12.86</td>
</tr>
<tr>
<td>FS-7</td>
<td>25.50</td>
<td>225+8</td>
</tr>
</tbody>
</table>

Table 4.1 Average PSNR (in dB) of the predicted frame using different algorithms for the backward prediction
<table>
<thead>
<tr>
<th></th>
<th>Salesman</th>
<th></th>
<th>Miss America</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR (dB)</td>
<td>Avg. Search pt</td>
<td>PSNR (dB)</td>
<td>Avg. Search pt</td>
</tr>
<tr>
<td>3 SS</td>
<td>35.39</td>
<td>25</td>
<td>38.54</td>
<td>25</td>
</tr>
<tr>
<td>P2B</td>
<td>35.57</td>
<td>–</td>
<td>39.03</td>
<td>–</td>
</tr>
<tr>
<td>P2BS</td>
<td>36.00</td>
<td>2.55</td>
<td>39.56</td>
<td>2.34</td>
</tr>
<tr>
<td>P2BS-LS</td>
<td>36.12</td>
<td>8.93</td>
<td>40.31</td>
<td>9.76</td>
</tr>
<tr>
<td>FS-7</td>
<td>36.17</td>
<td>225+8</td>
<td>40.34</td>
<td>225+8</td>
</tr>
</tbody>
</table>

Table 4.2 Average PSNR (in dB) of the predicted frame using different algorithms for the backward prediction.

The simulation results of all test sequences are plotted in Figure 4.6 to Figure 4.9. The part (a) of each figure shows the PSNR of each frame using different algorithms and part (b) shows the PSNR of full search minus that of P2BS-LS and 3SS. From the graphs, we can see that the performance of P2BS-LS is much better than 3SS and can be very close to full search for all test sequences.
Figure 4.6 (a) PSNR of predicted B-frames using different algorithms; (b) PSNR of full search minus that of P2BS-LS and 3SS for “Football”
Figure 4.7 (a) PSNR of predicted B-frames using different algorithms; (b) PSNR of full search minus that of P2BS-LS and 3SS for "Table Tennis"
Figure 4.8 (a) PSNR of predicted B-frames using different algorithms; (b) PSNR of full search minus that of P2BS-LS and 3SS for "Salesman"
Figure 4.9 (a) PSNR of predicted B-frames using different algorithms; (b) PSNR of full search minus that of P2BS-LS and 3SS for "Miss America"
4.5.2 Simulation Results of P-to-P Frame Conversion

In our simulation, we construct the motion compensated P-frames by the motion vectors that are estimated using 3-Step-Search with 4 steps (3SS-14) with a search window from $-14$ to $+14$, the proposed P2P, P2PS, P2PS-LS and full search. The full search we used is with half pixel accuracy and the search window is from $-7$ to $+7$ (FS-7) and $-14$ to $+14$ (FS-14). The average PSNR (dB) values and average search points per macroblock of the predicted P-frames using different algorithms are shown in Table 4.3 and Table 4.4.

<table>
<thead>
<tr>
<th>Football</th>
<th>Football</th>
<th>Table Tennis</th>
<th>PSNR (dB)</th>
<th>Avg. Search pt</th>
<th>PSNR (dB)</th>
<th>Avg. Search pt</th>
</tr>
</thead>
<tbody>
<tr>
<td>3SS-14</td>
<td>23.02</td>
<td>23.53</td>
<td>Avg. Search pt</td>
<td>33</td>
<td>Avg. Search pt</td>
<td>33</td>
</tr>
<tr>
<td>P2P</td>
<td>22.45</td>
<td>25.59</td>
<td>2.14</td>
<td>1.35</td>
<td>8.43</td>
<td>7.54</td>
</tr>
<tr>
<td>P2PS</td>
<td>23.12</td>
<td>25.95</td>
<td>225+8</td>
<td>225+8</td>
<td>26.71</td>
<td>26.61</td>
</tr>
<tr>
<td>P2PS-LS</td>
<td>23.55</td>
<td>26.71</td>
<td>841+8</td>
<td>841+8</td>
<td>27.07</td>
<td>27.07</td>
</tr>
</tbody>
</table>

Table 4.3 Average PSNR (in dB) of the predicted frame using different algorithms for the P frames

<table>
<thead>
<tr>
<th>Salesman</th>
<th>Salesman</th>
<th>Miss America</th>
<th>PSNR (dB)</th>
<th>Avg. Search pt</th>
<th>PSNR (dB)</th>
<th>Avg. Search pt</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 SS-14</td>
<td>36.75</td>
<td>37.78</td>
<td>Avg. Search pt</td>
<td>33</td>
<td>Avg. Search pt</td>
<td>33</td>
</tr>
<tr>
<td>P2P</td>
<td>35.88</td>
<td>37.95</td>
<td>0.65</td>
<td>0.81</td>
<td>8.06</td>
<td>8.24</td>
</tr>
<tr>
<td>P2PS</td>
<td>36.68</td>
<td>38.18</td>
<td>225+8</td>
<td>225+8</td>
<td>39.25</td>
<td>39.55</td>
</tr>
<tr>
<td>P2PS-LS</td>
<td>37.47</td>
<td>39.25</td>
<td>841+8</td>
<td>841+8</td>
<td>39.58</td>
<td>39.58</td>
</tr>
</tbody>
</table>

Table 4.4 Average PSNR (in dB) of the predicted frame using different algorithms for the P frames

Table 4.3 and Table 4.4 show that the average PSNR of P2P is higher than that of 3SS in “Table Tennis” and “Miss America”, but worse in “Football” and “Salesman”. The search in P2PS improves the PSNR by 0.7dB, 0.4dB 0.8dB and

63
0.2dB in the four sequences with a very small average search point of 2.14, 1.35, 0.65 and 0.81 respectively. But the local search in P2PS-LS can significantly improve the PSNR by 0.8dB in the cases of "Table Tennis" and "Salesman", making the final PSNR rather close to that of FS-14. Actually, the PSNR of P2PS-LS is already larger than FS-7. And average search point of P2PS-LS is very small compared with FS-7 and FS-14.

The simulation results of all test sequence are plotted in Figure 4.10 to Figure 4.13. The part (a) of each figure shows the PSNR of each frame using different algorithms and part (b) shows the PSNR of full search (FS-14) minus that of P2BS-LS and 3SS. From the graphs, we can see that the performance of P2BS-LS is much better than 3SS and can be very close to full search for all test sequences. From the graphs, we can see that the performance of P2PS-LS is much better than 3SS-14 and very close FS-14 apart from the later part of sequence "Football".
Figure 4.10 (a) PSNR of predicted P-frames using different algorithms; (b) PSNR of FS-14 minus that of P2PS-LS and 3SS for "Football"
Figure 4.11 (a) PSNR of predicted P-frames using different algorithms; (b) PSNR of FS-14 minus that of P2PS-LS and 3SS for "Table Tennis"
Figure 4.12 (a) PSNR of predicted P-frames using different algorithms; (b) PSNR of FS-14 minus that of P2PS-LS and 3SS for "Salesman"
Figure 4.13 PSNR of predicted P-frames (a) using different algorithms; (b) PSNR difference of FS-14 minus P2PS-LS and 3SS for the sequence "Miss America"
4.6 Conclusion

In this chapter, we proposed two algorithms the P2B and P2BS for the estimation of backward motion vectors for the P-to-B frame conversion. We also proposed two algorithms the P2P and P2PS for the estimation of motion vectors for the P-to-P frame conversion. The performance of the proposed algorithms is much better than the 3-Step-Search with much lower computation load. The proposed algorithms can achieve various quality and complexity tradeoff. In particular, the P2BS-LS and P2PS-LS have close to optimal performance with very small computation requirement.
Chapter 5

Lost Motion Vector Recovery for Error Concealment of Compressed Video

5.1 Introduction

In previous chapters, we focus on the motion estimation of extracting the preview version from compressed video for transmission on different kinds of networks. In this chapter, we look at a related but different problem. Many existing networks cannot provide guaranteed quality of service. Even though channel coding is usually employed to protect the bit stream, some error remains when channel conditions are bad leading to loss of data such as DCT coefficient, motion vector and etc. If one or more motion vectors are lost or received with errors after going through some noisy channel, it might cause the loss of one or more macroblocks that can corrupt not only the current frame, but also the succeeding frames. The effects of errors can be further magnified by the fact that motion vectors are usually coded differentially. This problem can be prevented or corrected by channel coding, e.g. Automatic Retransmission Request (ARQ) and forward error control coding techniques. However, the delay of re-sending the error packets is not acceptable for many applications and it will require considerable overhead for error detection and/or correction. Thus, it is important to have some error concealment algorithms in the decoder, which conceal the bit error effects by exploiting the high temporal and spatial correlation in the video and the limitations of human visual sensitivity without any overhead in the bitstream [13].
There are many existing error concealment algorithms that are done by exploiting the spatial [14, 15] or temporal [16-19, 21] correlation or both of them [20].

Those spatial reconstruction techniques [14,15] usually assume the existence of statistical correlation between neighboring blocks and thus work well when all the neighboring blocks belong to a homogeneous region. However, these algorithms produce inaccurate estimates if the neighboring blocks are coming from different regions.

Most of the temporal reconstruction techniques are efficient and with lower complexity than spatial approaches. The use of an intraframe median or a past motion vector was proposed for the recovery of lost motion vectors [17]. In [18], boundary matching algorithm (BMA) was proposed and it yields noticeably better results than previous work. A post-processing algorithm applying overlapped motion compensation (OMC) to enhance the performance of BMA was proposed in [20] and it showed a noticeable improvement. The decoder motion vector estimation (DMVE) algorithm [21] even show a better performance than the BMA.

5.2 Review of Lost Motion Vectors Recovery Algorithms

5.2.1 Boundary Matching Algorithm (BMA)

The BMA was proposed to find the best-match motion vector from a set of candidate vectors. The matching criterion was minimum gray-level variation across the boundaries. The total gray-level boundary variation $C$ is defined as

$$ C = C_{\text{top}} + C_{\text{left}} + C_{\text{bottom}} $$

where
\[
C_{\text{top}} = \sum_{x=x_0}^{x_0+N-1} \left( \hat{f}_k(x, y_0, n) - f_k(x, y_0-1, n) \right)^2
\]
\[
C_{\text{left}} = \sum_{y=y_0}^{y_0+N-1} \left( \hat{f}_k(x_0, y, n) - f_k(x_0-1, y, n) \right)^2
\]
\[
C_{\text{bottom}} = \sum_{x=x_0}^{x_0+N-1} \left( \hat{f}_k(x, y_0 + N - 1, n) - f_k(y, y_0 + N, n) \right)^2
\]

and \( f_k(x,y,n) \) is the \((xy)th\) pixel of the \(n^{th}\) reconstructed frame and \( \hat{f}_k(x,y) \) is the \((xy)th\) pixel of estimated image block using an estimated motion vector. The macroblock size is \(N \times N\) and the upper left coordinate of the block is \((x_0,y_0)\).

The BMA choose the motion vector \(m_v\), from the candidate vectors that produces the smallest value of \(C\). The set of candidate motion vectors include the motion vector (MV) of the same block in the previous frame, the MVs of the three available neighboring blocks, the median of the three neighboring MVs, the average of the three neighboring MVs and the zero motion vector. However, BMA does not work properly when there are slanted edges or abrupt gray level changes in the image.

5.2.2 Overlapped Motion Compensation (OMC)

A post-processing OMC enhancing the performance of BMA is proposed in [20]. The block corresponding to the recovered motion vector \(m_v\), is divided into four \(8 \times 8\) subblocks. Each subblock is synthesized by the weighted average of three predicted subblocks: one according to the recovered vector \(m_v\), the second and third according to the motion vectors of the horizontally and vertically neighboring \(8 \times 8\) block respectively.
5.2.3 Decoder Motion Vector Estimation (DMVE)

DMVE estimate the lost motion vectors. It uses the surrounding pixels as the pixels in the shaded region surrounding the missing block in the center in Figure 5.1. The width of the surrounding region \((w_r)\) is a design parameter. In Figure 5.1, \(M_R\) indicates the available reconstructed neighboring blocks and \(M_C\) indicate the missing block. If a predicted block \(M_P\) from the search area is matched to the missing block \(M_C\), the surrounding regions of \(M_P\) and \(M_C\) should also be matched. As a result, it chooses the distortion measure \(D\) to be the mean square error (MSE) between the surrounding region of the missing macroblock \(M_C\) (in current reconstructed frame) and the surrounding region of the each predicted block \(M_P\) (in previous reconstructed frame) corresponding to the candidate motion vectors. To reduce the computation, the same set of candidates motion vectors in BMA is used for searching. The motion vector with the minimum value of \(D\) is chosen as the recovery motion vector \(mv_r\).

\[
\begin{array}{c|c|c}
\hline
M_{a}(x-1,y-1) & M_{a}(x,y-1) & M_{a}(x+1,y-1) \\
\hline
M_{b}(x-1,y) & M_{c}(x,y) \\
\hline
M_{a}(x-1,y+1) & M_{a}(x,y+1) & M_{a}(x+1,y+1) \\
\hline
\end{array}
\]

Figure 5.1 Selection of surrounding pixels
5.3 Subblock Surrounding Pixels Search (SSPS)

In the previous sections, all the algorithms use the typical macroblock size $16 \times 16$ for the temporal reconstruction of the lost macroblocks. However, this is quite a large block size. It is possible to have difference objects within the same macroblock moving in complex directions is high and the prediction of the missing motion compensated macroblock will become unreliable. As we are working on the post-processing parts, we do not need to follow the standard directly. We modify the DMVE by dividing the missing macroblock into four $8 \times 8$ subblocks $SB_i$ and the surrounding pixels into four regions $R_i$ which are shown in Figure 5.2.

![Figure 5.2 Surrounding pixels region of Subblock Searching](image)

We define the best match motion vectors $mv_{r,i}$ using the same criteria and same candidates motion vectors as DMVE but with the surrounding regions $R_i$ for each $SB_i$. We call it Subblock Surrounding Pixels Search (SSPS). However, the R2 and R4 are not reliable enough because of small number of pixels in these regions. As a result, combine the $R_2$ and $R_4$ into one region $R_2'$, i.e. $SB_2$ and $SB_4$ will always use the same motion vector and we call this modified algorithm the SSPS2. The simulation results show that the SSPS is better than DMVE in PSNR with only a negligible increase in
complexity. The computation of searching the best match for four \( mv_{r,i} \) is the same as searching the best match for one \( mv_r \) in DMVE because adding up four \( R_i \) are exactly the same region of the surrounding of DMVE. There is only a minor increase in complexity for the motion compensation parts because we need to compensate four 8 × 8 blocks instead of one 16 × 16 block.

5.4 SSPS with Intra-block Overlapping (SSPS-IO)

The SSPS is better than DMVE in terms of PSNR; however, blocking artifacts will arise inside the 16 × 16 macroblocks because the each of the macroblock are formed by four 8 × 8 motion compensated subblocks, which are taken from different locations. To overcome this problem, we synthesize the missing macroblocks \( M_C \) by apply overlapping algorithm which is similar to the overlapping motion compensation used in H.263. We use a different weighting function from the OMC and all four motion vectors of the intra subblocks. We name it as 'SSPS with Intra-block Overlapping (SSPS-IO). Assume that SSPS has already been performed such that each of the \( SB_i \) has the associated motion vector \( mv_{r,i} \). Each macroblock is synthesized by the weighted average of four predicted macroblocks \( M_{P,i} \) according to the \( mv_{r,i} \). Let the synthesized macroblock be \( M_S \), and the weighting masks \( W_i \). The \( W_1 \) is shown in Figure 5.3. The width \( (w_o) \) of the overlapping regions is a design parameter ranging from 1 to 8 pixels. The weighting mask \( W_2 \) is the horizontally flipped version of \( W_1 \) whereas \( W_3 \) and \( W_4 \) are the vertically flipped version of \( W_3 \) and \( W_4 \) respectively. The \( M_S \) will be formed by:

\[
M_S = \frac{1}{8} \sum_{i=1}^{4} W_i \times M_{P,i}
\]
This overlapping function can decrease the visual discontinuity at the boundaries of the subblocks.

Figure 5.3 The 16 × 16 weighting matrix $W_1$
5.5 Simulation Results

We tested the proposed algorithm and other existing algorithms on the reconstructed video sequence “Football”, “Table Tennis” and “Flower Garden” which were MPEG-1 encoded with IPPP structure and a GOP size of 15 frames with a frame size of 352×240. For thorough evaluation, we applied the testing algorithms to every macroblock independently assuming that the macroblock is missing and the surrounding blocks are not. We conceal each macroblock and calculate the total sum of error of all motion compensated macroblock compared with the original macroblock of the non-coded sequence and the average PSNR of different algorithms is shown in Table 5.1. The plot of PSNR for Boundary Matching Algorithm (BMA), Decoder Motion Vector Estimation (DMVE) and the proposed Subblock Surrounding Pixels Search with Intra-block Overlapping (SSPS-IO) applied to the sequence “Football” is shown in Figure 5.4.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Football (200 frames)</th>
<th>Table Tennis (300 frames)</th>
<th>Flower Garden (150 frames)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average MV</td>
<td>22.09</td>
<td>24.17</td>
<td>22.77</td>
</tr>
<tr>
<td>BMA</td>
<td>23.08</td>
<td>23.96</td>
<td>20.01</td>
</tr>
<tr>
<td>BMA+OMC</td>
<td>23.63</td>
<td>25.07</td>
<td>21.54</td>
</tr>
<tr>
<td>DMVE</td>
<td>23.36</td>
<td>26.70</td>
<td>23.71</td>
</tr>
<tr>
<td>DMVE+OMC</td>
<td>23.79</td>
<td>26.99</td>
<td>24.04</td>
</tr>
<tr>
<td>SSPS</td>
<td>24.20</td>
<td>27.19</td>
<td>23.90</td>
</tr>
<tr>
<td>SSPS-IO</td>
<td>24.56</td>
<td>27.49</td>
<td>24.21</td>
</tr>
<tr>
<td>Original MV (dB)</td>
<td>25.14</td>
<td>27.78</td>
<td>23.91</td>
</tr>
</tbody>
</table>

Table 5.1 Average PSNR (in dB) of different error concealment algorithms

From Table 5.1, the average PSNR of proposed SSPS is higher than all other existing algorithms for all the three testing sequences with only a negligible increase in
computation when comparing with DMVE. Moreover, it is also higher than applying the enhancement process OMC to DMVE in the cases of “Football” and “Table Tennis”. The performance can be further enhanced by applying the intra-block overlapping and the PSNR of SSPS-IO is about 0.3dB higher than SSPS.

![Graph showing PSNR of BMA, DMVE, SSPS-IO for the sequence of “Football”](image)

Figure 5.4 PSNR of BMA, DMVE, SSPS-IO for the sequence of “Football”

For the “Football” sequence, the average PSNR of SSPS-IO is 1.2dB higher than DMVE. Moreover, from Figure 5.4, it shows that the PSNR of SSPS-IO is always higher than DMVE and BMA and the maximum gain over DMVE and BMA are about 2dB and 3.3dB respectively.
Figure 5.5 Part of frame 29 of ‘Football’ error concealed by (a) DMVE, (b) SSPS-IO
In Figure 5.5 (a) and (b), we show a zoom-in portion of the error concealed frame #29 by DMVE and proposed SSPS-IO respectively. The one produced by DMVE is much more blocky and shows more discontinuity of the football players’ body than that of SSPS-IO. Figure 5.6 gives a comparison of the performance of different. For each algorithm, we show a pair of frames, which contain the error concealed frame #29 of “Football” and the corresponding error frame when compared with the original frame. From the figures, we can see that the error concealed frame of DMVE looks a little better than that of BMA; however, when compared with the predicted frame using the original MV, the error is much higher. The visual quality by using SSPS and SSPS-IO are better than DMVE and BMA and the distortion shown in the error frames are lower than that of DMVE and BMA.
Figure 5.6 Comparison for frame #29 of "Football" (a) The original frame; the predicted frame #29 (left) and error image (right) by (b) original MV, (c) BMA, (d) DMVE, (e) SSFS, (f) SSFS-IO.
For the “Table Tennis” and “Flower Garden” sequences, the average PSNR of SSPS-IO are 0.79dB and 0.5 dB higher than DMVE respectively. In Figure 5.7 and Figure 5.9, the BMA fails totally to conceal the error. Similar to the behavior in “Football”, the PSNR of SSPS-IO is always higher than DMVE in these two test sequences and the maximum gain over DMVE are about 2.3dB and 1dB respectively.

In Figure 5.8 and Figure 5.10, we show pairs of frames, which contain an error concealed frame and the corresponding error frame compared with the original frame #130 of “Table Tennis” and #76 of “Flower Garden”. Again, we can notice that the proposed SSPS and SSPS-IO are significantly better than DMVE and BMA both in terms PSNR and visual quality.
Figure 5.7 PSNR of BMA, DMVE, SSPS-IO for the sequence of "Table Tennis"
Figure 5.8 Comparison for frame #130 of “Table Tennis” (a) The original frame; the predicted frame #130 (left) and error image (right) by (b) original MV, (c) BMA, (d) DMVE, (e) SSPS, (f) SSPS-I0.
Figure 5.9 PSNR of BMA, DMVE, SSPS-IO for the sequence of "Flower Garden"
Figure 5.10 Comparison for frame #76 of “Flower Garden.” (a) The original frame; the predicted frame #76 (left) and error image (right) by (b) original MV, (c) BMA, (d) DMVE, (e) SSPS, (f) SSPS-JO.
5.7 Conclusion

In this chapter, we propose two algorithms Subblock Surrounding Pixels Search (SSPS) and the SSPS with Intrablock Overlapping (SSPS-IO) for the recovery of lost macroblocks. The performance of the proposed algorithms is much better than the DVMA, BMA and also the BMA with enhancement post-processing OMC. Moreover, the proposed algorithm is consistently good in various kinds of video scenes.
Chapter 6

Conclusion

In this thesis, we studied some methods of fast motion estimation for video resolution down-conversion and frame type conversion and some algorithms for error concealment of lost motion vectors.

In resolution down-conversion, we proposed the Predictive Motion Estimation (PME) and Modified Predictive Motion Estimation (MPME) and Motion Estimation by Spatial-Variant Filtering (ME-SVF) to find the best motion vectors. Simulation results show that the performance of PME and MPME are better than existing algorithms in terms of PSNR; however, the complexity is still high. The complexity of ME-SVF is much lower than PME and MPME but with degradation in PSNR. However, the performance of ME-SVF is better than other existing low-complexity algorithms in terms of both PSNR.

In frame type conversion, we proposed the P2B, P2BS and P2BS-LS for the estimation of backward motion vectors for the P-to-B frame conversion; the P2P, P2PS and P2PS-LS for the estimation of motion vectors for the P-to-P frame conversion. The performance of the proposed algorithms are much better than the 3-Step-Search and very close to full search with much lower computation load.
In error concealment of lost motion vectors, we proposed the Subblock Surrounding Pixels Search (SSPS) algorithm for the recovery of lost macroblocks. The visual quality of the restored frames using the proposed algorithm are much better than the existing algorithms DVMA and BMA and it is robust to various kinds of video scenes.
Reference


List of Publications


