Diplomarbeit

Concealment of Video Transmission Packet Losses
Based on Advanced Motion Prediction

by

Claudius Volz

LITH-ISY-EX-3425-2003

June 2003
Concealment of Video Transmission Packet Losses
Based on Advanced Motion Prediction

by

Claudius Volz

LITH-ISY-EX-3425-2003

June 2003
Recent algorithms for video coding achieve a high-quality transmission at moderate bit rates. On the other hand, those coders are very sensitive to transmission errors. Many research projects focus on methods to conceal such errors in the decoded video sequence.

Motion compensated prediction is commonly used in video coding to achieve a high compression ratio. This thesis proposes an algorithm which uses the motion compensated prediction of a given video coder to predict a sequence of several complete frames, based on the last correctly decoded images, during a transmission interruption. The proposed algorithm is evaluated on a video coder which uses a dense motion field for motion compensation.

A drawback of predicting lost fields is the perceived discontinuity when the decoder switches back from the prediction to a normal mode of operation. Various approaches to reduce this discontinuity are investigated.

**Nyckelord**
error concealment, video coding, motion compensated prediction, backward coding, dense motion estimation
Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Diplomarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Claudius Volz
Abstract

Recent algorithms for video coding achieve a high-quality transmission at moderate bit rates. On the other hand, those coders are very sensitive to transmission errors. Many research projects focus on methods to conceal such errors in the decoded video sequence.

Motion compensated prediction is commonly used in video coding to achieve a high compression ratio. This thesis proposes an algorithm which uses the motion compensated prediction of a given video coder to predict a sequence of several complete frames, based on the last correctly decoded images, during a transmission interruption. The proposed algorithm is evaluated on a video coder which uses a dense motion field for motion compensation.

A drawback of predicting lost fields is the perceived discontinuity when the decoder switches back from the prediction to a normal mode of operation. Various approaches to reduce this discontinuity are investigated.
Acknowledgements

I wish to thank anyone who contributed to this thesis, especially:
Prof. Robert Forchheimer and Peter Johannson for the supervision of the thesis, their suggestions and their help in the administrative issues.
All others at the Image Coding Group who gave me a warm welcome when I arrived at Sweden.
Prof. Hänslcr and Maximilian Gauger for the supervision of the thesis at Darmstadt University of Technology.
This thesis was supported by the Erasmus student mobility programme. Many thanks to everyone who took part in the organisation of my stay. Special thanks to the International Student Coordinator and the Erasmus Student Network members. You are doing a great job.
This thesis is typeset in \LaTeX. The XEmacs editor was used. The layout is based on template files by Volker Ahlers and Stefan Skamphausen, which are available for download from \url{http://www.skamphausen.de}. 
Contents

List of Figures ix

List of Tables xi

1 Introduction 1
   1.1 Motivation ................................................. 1
   1.2 Research Context ....................................... 1
   1.3 Objectives and Approach ............................... 2
   1.4 Outline of the thesis ................................... 3

2 Background 5
   2.1 Interlaced Video ........................................ 5
   2.2 Forward vs. Backward Prediction ...................... 6
   2.3 Peak signal-to-noise ratio ............................ 7
   2.4 A Novel Video Coder .................................... 8
      2.4.1 Quadrature Filter Bank Representation ........... 9
      2.4.2 Calculation of Phase Differences ................. 10
      2.4.3 Calculation of Motion Estimates ................. 12
      2.4.4 Motion compensated prediction .................. 13
      2.4.5 Multiple Hierarchical Motion Estimation ....... 14

3 Motion Compensated Prediction for Error Concealment 17
   3.1 Motivation ............................................. 17
   3.2 Assumptions and Objectives ........................... 19
   3.3 Implementation Issues ................................ 20
   3.4 PSNR Performance ..................................... 21
   3.5 Visual Impression ..................................... 24
   3.6 Summary .............................................. 25

4 Cross-Fading to Remove Temporal Discontinuity 27
   4.1 Motivation ............................................. 27
   4.2 Experimental Results .................................. 27
   4.3 Using Wrong Residual Information for Prediction .... 30
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4 Automatic Abort of Cross-Fading</td>
<td>34</td>
</tr>
<tr>
<td>4.5 Summary</td>
<td>38</td>
</tr>
<tr>
<td>5 Block Matching Prediction to Remove Temporal Discontinuity</td>
<td>39</td>
</tr>
<tr>
<td>5.1 Idea</td>
<td>39</td>
</tr>
<tr>
<td>5.2 Identifying Object Displacements by Block Matching</td>
<td>42</td>
</tr>
<tr>
<td>5.2.1 Forward Estimation</td>
<td>42</td>
</tr>
<tr>
<td>5.2.2 Reverse Estimation</td>
<td>43</td>
</tr>
<tr>
<td>5.2.3 Combining Forward and Reverse Approach</td>
<td>43</td>
</tr>
<tr>
<td>5.2.4 Selection of Block Size and Search Region</td>
<td>45</td>
</tr>
<tr>
<td>5.3 Experimental results</td>
<td>46</td>
</tr>
<tr>
<td>5.4 Summary</td>
<td>49</td>
</tr>
<tr>
<td>6 Increased Stability Coding</td>
<td>51</td>
</tr>
<tr>
<td>6.1 Idea</td>
<td>51</td>
</tr>
<tr>
<td>6.2 Experiments</td>
<td>52</td>
</tr>
<tr>
<td>6.3 Summary</td>
<td>54</td>
</tr>
<tr>
<td>7 Summary</td>
<td>55</td>
</tr>
<tr>
<td>8 Suggestions for Further Research</td>
<td>57</td>
</tr>
<tr>
<td>References</td>
<td>59</td>
</tr>
</tbody>
</table>
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Interlaced sampling. Top field (+) and bottom field (•).</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>Video coder (left) and decoder (right) using forward prediction</td>
<td>6</td>
</tr>
<tr>
<td>2.3</td>
<td>Video coder (left) and decoder (right) using backward prediction</td>
<td>7</td>
</tr>
<tr>
<td>2.4</td>
<td>Outline of the motion estimation and motion compensated prediction of the discussed video coder</td>
<td>8</td>
</tr>
<tr>
<td>2.5</td>
<td>Filter net for computing the quadrature filter responses $q_k$</td>
<td>10</td>
</tr>
<tr>
<td>2.6</td>
<td>Magnitude of the quadrature filters $\hat{f}_1$ to $\hat{f}_9$</td>
<td>11</td>
</tr>
<tr>
<td>2.7</td>
<td>Multiple hierarchical motion estimation (MHME)</td>
<td>15</td>
</tr>
<tr>
<td>3.1</td>
<td>Block diagram of the backward video coder</td>
<td>17</td>
</tr>
<tr>
<td>3.2</td>
<td>Block diagram of the backward video decoder</td>
<td>18</td>
</tr>
<tr>
<td>3.3</td>
<td>Using the purely predicted field if residual information is not available</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>Example sequence that has been predicted without residual information (left side) and the associated correctly decoded fields (right side)</td>
<td>22</td>
</tr>
<tr>
<td>3.5</td>
<td>Difference between the predicted fields and the regularly decoded fields of two example sequences</td>
<td>23</td>
</tr>
<tr>
<td>4.1</td>
<td>Example for cross-fading from the predicted sequence to the correctly reconstructed sequence</td>
<td>29</td>
</tr>
<tr>
<td>4.2</td>
<td>Example for a situation where cross-fading does not work</td>
<td>31</td>
</tr>
<tr>
<td>4.3</td>
<td>Example sequence that has been predicted with residual information dropped for one field (left side) and the associated correctly decoded fields (right side)</td>
<td>32</td>
</tr>
<tr>
<td>4.4</td>
<td>Root mean square difference of the predicted field images after a transmission error to the correctly decoded fields of some example sequences</td>
<td>35</td>
</tr>
<tr>
<td>4.5</td>
<td>Mean absolute difference of the predicted field images after a transmission error to the correctly decoded fields of some example sequences</td>
<td>36</td>
</tr>
<tr>
<td>5.1</td>
<td>Shifting parts of the image to reduce discontinuities</td>
<td>40</td>
</tr>
<tr>
<td>5.2</td>
<td>Procedure at a transmission error</td>
<td>41</td>
</tr>
<tr>
<td>5.3</td>
<td>Forward estimation: Use blocks of A to predict image B</td>
<td>42</td>
</tr>
<tr>
<td>5.4</td>
<td>Reverse estimation: Use blocks of B to predict image A</td>
<td>43</td>
</tr>
<tr>
<td>5.5</td>
<td>Two example images and their block matching estimations</td>
<td>44</td>
</tr>
</tbody>
</table>
5.6 Example sequence: changing back from the erroneous to the error-free sequence using the suggested block matching approach . . . . . . . . . . 47

6.1 Simplified block diagram of the modified coder . . . . . . . . . . . . . . 52
6.2 Example sequence which shows how the decoder recovers from a transmission interruption . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
List of Tables

4.1 Evaluation of the perceived quality of a test sequence depending on the number of predicted fields and the cross-fade duration . . . . . . . . . . 28
4.2 Perceived quality of a test sequence depending on the number of predicted fields $k$ and the cross-fade duration $l$ where residual information is dropped only for the first predicted field . . . . . . . . . . . . . . . . 33
Chapter 1

Introduction

1.1 Motivation

Recently digital video transmission has become more and more important. It is used in a very wide range of applications. Digital video is about to replace analogue standards for example at TV broadcast, home video or video conferencing. Digital video also opens up new fields of application such as multimedia communications or telecollaboration.

There are two reasons for the success of digital video. On the one hand the available computing power of recent processors increases rapidly. On the other hand new sophisticated coding algorithms have been developed which use this computation power to achieve a high-quality transmission.

The main goal during the development of today’s most commonly used video coders has been a high compression ratio. Recent coders offer high perceived quality at low bit rates by removing redundant information from the video sequence. Information which can be reconstructed at the receiver is omitted, only the remaining part needs to be transmitted.

The drawback of this procedure is that the coded video will be very sensitive to transmission errors. Even very short transmission errors are likely to propagate spatially and temporally, a single error will affect a large area within the picture for many frames. Unfortunately, in today’s packet switched networks, transmission errors such as lost or delayed packets occur quite often. In this case the perceived quality is seriously decreased.

Many research projects focus on the development of error resilient coding algorithms or methods to conceal errors in the decoded video sequence.

1.2 Research Context

There is a lot of active research on error concealment for digital video. Wang and Zhu provide an extensive review of techniques in [WZ98]. A newer review of Wah et al. [WSL00] focusses on error concealment methods for real-time transmission over the
internet.

Wang and Zhu [WZ98] classify common error concealment techniques into three categories. The first category is called forward error concealment and refers to methods which modify the source coding or transport coding at the coder in order to reduce the impact of transmission errors. The second category achieves error concealment by post-processing, i.e. the decoder fulfills the task of error concealment without using additional information from the coder. The approach which is investigated within the scope of this thesis belongs to this category. Finally, there are interactive techniques, i.e. methods where information is exchanged between the decoder and the coder to conceal errors.

All algorithms of the category “error concealment by post-processing” use remaining redundancy in the coded video sequence to reconstruct lost information. Typically, continuity is assumed in the spatial, temporal or frequency domain. Neighbouring data in the chosen domain is then used for the reconstruction. A typical approach would be to replace the missing information with information that is extracted from surrounding picture areas or the previous or next frame.

Most post-processing approaches are extensions to standard coders. Those coders use block-based motion compensated prediction. Both the block motion and the prediction error for each block is transmitted. There are different reconstruction algorithms depending on which information is affected by a transmission error. Usually, those algorithms require that only a part of the image is lost (in case of spatial interpolation) or that only one field is affected (in case of temporal interpolation).

This thesis investigates a different approach to error concealment. The motion compensated prediction which is part of a given video coder is used to predict a sequence of several complete frames, based on the last two correctly decoded images, during a transmission interruption.

1.3 Objectives and Approach

The objectives of this thesis are as follows:

- Show that the motion estimation of a given video coder can be used to replace several consecutive fields of a video sequence. This coder uses a backward prediction scheme.\(^1\) In other applications, only the next field is predicted. The prediction is likely to degrade rapidly if more fields are predicted without additional information from the coder. Therefore it should be a success if between three and five fields can be predicted with tolerable error.

- The prediction differs from the corresponding correct fields. This difference causes a visible discontinuity in the resulting sequence. An algorithm which reduces this discontinuity is investigated.

\(^1\)The concept of forward prediction and backward prediction is described in section 2.2.
discontinuity is to be developed and investigated in experiments. This algorithm should work with a large number of input sequences.

The thesis is based on a video coder that contains a sophisticated motion prediction. This coder has been developed during a PhD thesis [And03]. The MATLAB program code of the coder was available.

A first task within the scope of this thesis was to build a test environment that consists of the coder, a decoder and a means to interrupt the transmitted data at a given point of time. This test environment was used to evaluate the suggested error concealment algorithms. MATLAB has been used for the implementation of the test environment, additional C routines were available from [And03] for some time-critical tasks. The calculations were far away from real-time on a common 1 GHz PC, therefore the generated video sequences were stored on disk for later evaluation.

This diploma thesis has been carried out at the Image Coding Group within the Department of Electrical Engineering of Linköpings University, under the supervision of Peter Johansson and Prof. Robert Forchheimer. It has been published and supervised by Maximilian Gauger and Prof. Dr.-Ing. Hänsler at the Signal Processing Group within the Institute for Communication Technology at Darmstadt Technical University.

1.4 Outline of the thesis

Chapter 2 gives an overview of the video coder that has been used. This coder contains a new form of motion estimation which is based on a dense motion field. This dense motion field is used in a backward coding scheme.

Chapter 3 illustrates how the motion compensated prediction can be used to predict lost fields. The peak signal-to-noise ratio and the visual impression of some example sequences are considered.

A drawback of using the prediction to replace lost fields is the perceived discontinuity when the decoder switches back from the prediction to a normal mode of operation. An algorithm to reduce this temporal discontinuity based on cross-fading over several fields is discussed in chapter 4.

A more sophisticated approach which uses block matching to reduce the perceived discontinuity is suggested in chapter 5. As the discontinuity is caused mainly by displacements between the predicted image and the correct image, the proposed algorithm tries to establish a smooth transition by shifting parts of the image appropriately.

Chapter 6 presents an approach which is different to the preceding chapters. Now the coder is modified in a way that includes additional energy in the residual image that is transmitted. This causes the decoder to recover from transmission errors within several fields. No dedicated error concealment mechanism is needed in this case.
The thesis concludes with a summary of the main thoughts in chapter 7 and suggestions for further research in chapter 8.
Chapter 2

Background

This chapter gives background information on a number of concepts that are used in the thesis. It also introduces the video coder on which the thesis work is based.

2.1 Interlaced Video

Most common video applications use interlaced sampling. Figure 2.1 illustrates the concept of interlaced video. Each frame of a video sequence is split into two field images. All odd numbered rows are allocated to the so-called top field, while all even numbered rows form the bottom field. The odd and the even field of each frame are transmitted consecutively. The sampling rate (in images per second) of the resulting image sequence is doubled, but the vertical resolution of a single field image is reduced.

Interlaced video had been introduced with analogue television. A refresh rate of 50 Hz is necessary to avoid flicker at usual CRT displays. By interlaced sampling, this refresh rate could be achieved while meeting the analogue bandwidth limits, too. Today, interlaced video is still the standard for most video systems.

The opposite of interlaced sampling is commonly called progressive sampling.
2.2 Forward vs. Backward Prediction

The concept of motion compensated prediction is widely used in video coding. It is based on the fact that the change in motion between adjacent frames of a video sequence is usually very small. Therefore, the motion of recently decoded frames can be used to predict the next frame. The remaining prediction error will be small, i.e. little information needs to be transmitted. Thus high compression ratios can be achieved.

We distinguish two types of motion estimation, which are called forward prediction and backward prediction. In the following chapters of this thesis, we will often refer to those two concepts. The following examples consider a progressive coder to be more clear.

A forward prediction coder estimates the motion between the current reference frame $I(n)$ and the previously decoded frame $I_{\text{dec}}(n-1)$. The resulting motion field is used to generate the motion compensated prediction $\hat{I}(n)$ for the current frame. The remaining error is coded and transmitted over the channel. The estimated motion is transmitted, too.

The decoder recalculates the motion compensated prediction $\hat{I}(n)$, which in combination with the decoded prediction error results in the decoded frame. The approach to transmit both prediction information and the coded prediction error is commonly called hybrid coding.

In backward prediction, the motion estimation is based on (at least) two previously decoded frames $I_{\text{dec}}(n-1)$ and $I_{\text{dec}}(n-2)$. The motion field is not transmitted to the receiver in this case. Therefore the decoder needs its own motion estimation block.

An important aspect of the two types of motion estimation is the temporal allocation of the motion estimates. In forward prediction, the motion between the current and the previous frame is used to predict the current frame. In contrast, backward prediction...
estimates the motion between the previous field and the field before. Therefore a backward coder actually needs to temporally predict the motion of the current frame from the motion estimate which refers to the previous frame.

Most video coding standards, such as MPEG-2 and MPEG-4, use forward prediction. The advantage is the simplicity of the decoder, which does not need to perform motion estimation. Thus the computation power which is required for decoding is by far lower than for coding. On the other hand, this approach only makes sense if the motion information can be transmitted with reasonable bandwidth.

The video coder that is discussed below is based on backward prediction. It therefore does not need to transmit the motion field which would need a lot of bandwidth in this special case. The drawback is that the decoder needs its own motion estimation and thus is almost as slow as the coder.

### 2.3 Peak signal-to-noise ratio

A common quality measure for image coding algorithms is the *peak signal-to-noise ratio*. The PSNR of two images $I(x)$ and $\hat{I}(x)$ is defined as

$$PSNR = 20 \log \left( \frac{b}{|I(x) - \hat{I}(x)|} \right),$$  

where $b$ is the largest possible pixel value, $x = (x, y)^T$ are the spatial coordinates, and $|I(x) - \hat{I}(x)|$ is the root mean square difference of the two images. Typically, $\hat{I}(x)$ would be an image which has been processed by an image compression algorithm, and $I(x)$ would be the original image.
The PSNR is given in decibel units (dB). High values indicate a low difference between the images, i.e. low error due to compression. Typical values are in the range from 20 dB to 40 dB for compressed images and video sequences.

### 2.4 A Novel Video Coder

The following section gives an overview of a video coder which contains new approaches to motion compensation. Those approaches have been presented in a dissertation [And03] and in [AAJ+03]. The resulting video coder has been the basis for the experiments within this thesis.

Almost all video coding standards, such as MPEG-1, MPEG-2, MPEG-4, H.261 and H.263 use block based motion compensation. The frame that is to be predicted is divided into blocks of typically $8 \times 8$ or $16 \times 16$ pixels. For each block, a corresponding block within a given search region of the previous frame is selected, so that the RMS error is minimised. The selected blocks form the motion compensated prediction. The transmitted prediction data describes rather block displacements than real motion.

Block based motion compensation is used in standard coders because it combines simplicity and good performance. The drawback of block based algorithms is that block artifacts are likely to appear in the decoded sequence.

The coder that is described in the following uses a dense motion field for motion compensated prediction, i.e. the velocity is estimated in each pixel to overcome the problem of block artifacts. The central part of the coder is the estimation of this dense motion field. The overhead required to transmit this motion field would be very high, therefore a backward prediction scheme as described in section 2.2 is chosen.

![Figure 2.4: Outline of the motion estimation and motion compensated prediction of the discussed video coder](image-url)
A SPIHT coder is used to compress the residual image before transmission. SPIHT (set partitioning in hierarchical trees) is a wavelet-based image compression algorithm which achieves very high compression ratios. The bit rate can be precisely controlled, which is helpful if a constant bit rate video stream has to be generated. Further information on the algorithm is available from [SP96].

The coder has been designed for interlaced video, as described in section 2.1. The coding procedure is carried out for each field. This means that the decoder has to be aware of the displacement between odd and even fields (as shown in figure 2.1) whenever previous field images are referenced.

Figure 2.4 outlines the steps that are performed by the decoder to calculate the motion compensated prediction. The motion estimation takes place in the Fourier domain, therefore a filter bank is used to calculate a quadrature channels representation of each field image. The estimation of the dense motion field is based on phase differences. An explanation of these steps follows below.

### 2.4.1 Quadrature Filter Bank Representation

The first step of the motion estimation process is to calculate an image representation from a set of quadrature filters in different orientations and scales covering the Fourier domain. Quadrature filters are chosen because the magnitude of the output is invariant to the phase of the input signal. Additionally, a number of physiological experiments indicate that the human visual system has similar characteristics.

A set of nine quadrature filters is used. A filter net, i.e. a combination of simple one dimensional filters with 5-7 real valued coefficients each, calculates the filter responses. This filter implementation requires only 141 real-valued multiplications per pixel, while nine conventionally designed filters with similar performance would need a total of about 3000 filter coefficients. A Fast Fourier Transform approach would be still about ten times slower than the filter net.

Figure 2.5 outlines the structure of the filter net. Each circle represents a simple one dimensional filter. The dots within each circle denote both the number of non-zero coefficients and the spatial position of those coefficients.

The magnitude of the resulting quadrature filters is shown in figure 2.6. The image representation cover four directions of filtering in five different scales.

In order to achieve support for interlaced video, the filter net has been designed using a restricted Fourier domain in the vertical direction, and allowing filter coefficients in appropriate spatial positions only. The resulting filter net handles interlaced images without preprocessing.
In the following the quadrature filter kernels are denoted as \( \tilde{f}_k \), \( k = 1 \ldots 9 \). The complex-valued filter response to a given real-valued image \( I(x) \) is denoted as \( q_k(x) \),

\[
q_k(x) = \tilde{f}_k * I(x) = m_k(x)e^{i\phi_k(x)},
\]

(2.2)

where \( x = (x, y)^T \) are the spatial coordinates, \( * \) is the convolution operator, \( m_k(x) \) denotes the magnitude and \( \phi_k(x) \) the phase of \( q_k(x) \).

**2.4.2 Calculation of Phase Differences**

The relation between the velocity \( \mathbf{v} \) and phase is described by the optical flow constraint equation,

\[
\phi_x v_x + \phi_y v_y + \phi_t = 0,
\]

(2.3)
Figure 2.6: Magnitude of the quadrature filters $\tilde{f}_1$ to $\tilde{f}_9$
where \( v_x \) and \( v_y \) are the velocity components in the spatial directions \( x \) resp. \( y \), \( \phi_x \) and \( \phi_y \) denote the partial derivatives of \( \phi \) in the spatial directions, and \( \phi_t \) is the temporal derivative. It is assumed that the phase is linear within the filter’s operating range.

The velocity field \( \mathbf{v} \) can be determined by solving equation (2.3) if the phase derivatives are known. It is possible to use the phase differences between neighbouring pixels in the temporal or spatial direction as an approximation of the phase derivatives. The phase differences are computed by a conjugate complex multiplication,

\[
qq_{kx}(x) = q_k(x - \left(\begin{array}{c} 1 \\ 0 \end{array}\right), t) \cdot q_k(x, t)^* + q_k(x, t) \cdot q_k(x + \left(\begin{array}{c} 1 \\ 0 \end{array}\right), t)^* \tag{2.4}
\]

\[
qq_{ky}(x) = q_k(x - \left(\begin{array}{c} 0 \\ 1 \end{array}\right), t) \cdot q_k(x, t)^* + q_k(x, t) \cdot q_k(x + \left(\begin{array}{c} 0 \\ 1 \end{array}\right), t)^* \tag{2.5}
\]

\[
qq_{kt}(x) = q_k(x, t - 1) \cdot q_k(x, t)^* + q_k(x, t) \cdot q_k(x, t + 1)^*. \tag{2.6}
\]

The phase differences are given by the argument of those functions.

\[
\phi_{kx}(x) = \arg(qq_{kx}(x)) \tag{2.7}
\]

\[
\phi_{ky}(x) = \arg(qq_{ky}(x)) \tag{2.8}
\]

\[
\phi_{kt}(x) = \arg(qq_{kt}(x)) \tag{2.9}
\]

2.4.3 Calculation of Motion Estimates

The phase differences are weighted in order to avoid points where the phase differences are unreliable,

\[
c_k(x) = c_k \left(\begin{array}{c} \phi_{kx} \\ \phi_{ky} \\ \phi_{kt} \end{array}\right), \tag{2.10}
\]

where the weighting factor \( c_k \) is a combination of several functions which describe the reliability of the phase estimates at a given point. For instance, points with high phase differences or high velocity are avoided. The spatial coordinates \( x \) have been omitted for the sake of readability.

The phase constraint equation is solved by minimising

\[
E_v = \sum_k \sum_x W(x)(c_k^T(x) \bar{\mathbf{v}}(x))^2, \tag{2.11}
\]
where \( k = 1 \ldots 9 \) refers to the quadrature filter channels, \( W(x) \) is a windowing function, and \( \hat{v}(x) = (v_x, v_y, 1)^T \). The velocity \( v(x) \) that minimises equation (2.11) can be calculated as

\[
v(x) = -\left( \frac{\sum_{k,x} W(x) c_k \phi_{kx} \phi_{kx} + \sum_{k,x} W(x) c_k \phi_{ky} \phi_{ky}}{\sum_{k,x} \sum_{k,x} W(x) c_k \phi_{kx} \phi_{ky}} \right)^{-1} \left( \sum_{k,x} W(x) c_k \phi_{kx} \phi_{kx} \phi_{xt} \sum_{k,x} W(x) c_k \phi_{ky} \phi_{ky} \phi_{yt} \right).
\] (2.12)

The resulting velocity \( v(x) \) is post-filtered to further reduce spatial inconsistencies. This post-processing uses the local covariance to measure spatial consistency. See [And03] for further details.

### 2.4.4 Motion compensated prediction

The coder uses previously decoded field images to estimate the motion field \( v'_{t-\Delta t} \), but the current motion \( v'_t \) is needed to generate the motion compensated prediction \( \hat{I}_t \) of the current field image. Thus a temporal prediction of the motion field has to take place (cf. section 2.2 on backward coding).

There are two ways to carry out the temporal prediction of the motion field and the following motion compensated prediction. The first alternative is a prediction in one step from the previously decoded field image \( \hat{I}_{t-\Delta t} \).

\[
\hat{I}_t(x + \Delta t \cdot v'_{t-\Delta t}(x)) = \hat{I}_{t-\Delta t}(x). \tag{2.13}
\]

The second alternative is a prediction in two steps, first predict \( \hat{v}_t \) from \( v'_{t-\Delta t} \), then predict \( \hat{I}_t \) using the predicted motion.

\[
\hat{v}_t(x + \Delta t \cdot v'_{t-\Delta t}(x)) = v'_{t-\Delta t}(x). \tag{2.14}
\]

\[
\hat{I}_t(x) = \hat{I}_{t-\Delta t}(x - \Delta t \cdot \hat{v}_t(x)). \tag{2.15}
\]

The result of equations (2.13) to (2.15) will be irregularly sampled, i.e. the position of the predicted points will not be aligned with the desired regular output sample pattern. Problems will arise where several motion estimates end up in the same point or where no motion estimates end at all. The dissertation [And03] introduces a method to recalculate a regularly sampled image from the irregularly sampled prediction. This concept is called Continuous Normalised Convolution.
2.4.5 Multiple Hierarchical Motion Estimation

A common problem of phase-based motion estimation is to achieve both a wide range of velocities that can be handled and a good spatial resolution. Filters based on a low spatial frequency are able to handle large velocities, but the spatial resolution is bad. This typically causes problem around edges of moving objects. On the other hand, filters using higher spatial frequencies enable a better localisation of object motion, but fail at high velocities.

A common approach to overcome the described problems is hierarchical motion estimation. It starts with motion estimation based on low spatial frequencies to calculate a coarse motion estimate. This coarse motion is then used to pre-compensate another motion estimation unit which is based on higher spatial frequencies. This means that the second motion estimator analyses the motion that remains after the first coarse estimation, so that the motion field can be further refined.

The given coder extends this idea by using several hierarchical motion estimators in parallel. This so-called multiple hierarchical motion estimation (MHME) starts with a motion estimator which compensates global motion, i.e. a global shift of the image caused by camera panning. The remaining motion is analysed by three different hierarchical motion estimators starting at different estimation levels.

Figure 2.7 illustrates the MHME as described above. MC refers to a motion compensated motion estimation unit, while the size of the according symbol indicates the scale of the filters which are used. The top half of figure 2.7 shows the global motion estimation. The lower half contains three hierarchical motion estimators. The estimation of \( \hat{v}_1 \) uses all scales, while \( \hat{v}_3 \) is based on the smallest scale (high spatial frequency) only. \( \hat{v}_4 \) is equal to the global estimate.

The multiple hierarchical motion estimation results in a total of four motion fields that can be used for motion compensation prediction. To combine the four motion fields, a decision is made between the four estimates on a per-pixel level. For each pixel, the motion estimate is selected that gives the least local motion compensated prediction error between previously decoded field images.
Figure 2.7: Multiple hierarchical motion estimation (MHME)
Chapter 3
Motion Compensated Prediction for Error Concealment

This chapter shows how the motion compensated prediction of a given video coder can be used to replace field images that are lost due to transmission errors. The described method is tested with a number of video sequences. The results are evaluated in terms of the peak signal-to-noise ratio and overall visual impression.

3.1 Motivation

The video coder that is used throughout this thesis uses motion compensated prediction to achieve a high coding performance. The coder (resp. decoder) estimates a motion field based on the last two or three received field images. This motion information is used to predict the next field. Only the residual, i.e. the difference between the predicted and the reference field image, is transmitted from the coder to the decoder. Simplified block diagrams of the coder and the decoder are presented in figure 3.1 and figure 3.2.

Figure 3.1: Block diagram of the backward video coder
When the residual information of a field is lost because of transmission errors, it is not possible to calculate the correct field image. In this case, a possible approach is to just use the predicted image, i.e. we assume that the residual for this field is equal to zero. By doing so, we get an incorrectly decoded image that replaces the field image whose information is lost.

This procedure may also be repeated for several field images. In this case the motion estimation within the decoder will be based on incorrectly decoded images, causing the decoder output to diverge more and more from the reference sequence (cf. figure 3.3). Of course it is impossible to reconstruct any information that is lost during transmission. The prediction is based on the motion of the last correctly decoded fields. Any later changes to the motion will not appear in the predicted sequence. In the worst case, a scene change takes place while the transmission is interrupted. The decoder continues the prediction based on the old scene if this happens.

Today’s most commonly used video coders use forward coding, i.e. not only the residual but also motion information is transmitted to the decoder. The discussed procedure for error concealment could be used at such forward coders too, if the residual information of a field is lost. But a forward decoder will need the motion information to accomplish the motion compensated prediction. There are other algorithms for error concealment if motion information is affected by transmission errors. However, error concealment at forward coding is not within the scope of this thesis.

Figure 3.2: Block diagram of the backward video decoder
3.2 Assumptions and Objectives

We assume that the residual information is lost for one complete field or for several complete fields. That means that any error during transmission renders the residual of the corresponding field useless. A reason for this might be an advanced coding of the residual that is disturbed even by one erroneous byte. In our case this assumption is made for the sake of simplicity. For transmission interruptions that last over a couple of fields, at least the residual information of intermediate fields will be completely lost. Short transmission errors are likely to affect only a certain region of the decoded field, yet the local behaviour within those disturbed regions should be comparable to the case that the residual is lost for the complete field image.

We further assume that an error-free state of the decoder is regained after a number of fields, i.e. after a given number of fields the decoder continues as if no error had occurred. A well-known mechanism for this is the transmission of intra-coded images every n-th frame. However the time to the next intra-coded frame will typically be several seconds. This is too long in our case. A more likely scenario is real-time video transmission over a packet-switched network, where the packets must not exceed a certain delay. If the network is heavily loaded it may happen that packets do not arrive in time to be decoded, but those packets often arrive later or can be re-requested by an ARQ scheme. The decoder is able to reconstruct the regularly decoded sequence as soon as the missing information is received.

It should be pointed out that the algorithms which are proposed within this thesis are
Kap. 3 Motion Compensated Prediction for Error Concealment

intended to fill several frames between a transmission error due to delayed or lost packets and the point of time when the lost information is restored. This applies only to real-time applications that allow only a very limited transmission delay, such as video phones. Typical broadcasting systems will allow enough delay so that there will be no need to skip frames in the situations that are considered here.

The objective of this chapter is to show that the motion compensated prediction of a backward coder can be used for error concealment in the suggested cases. We will also try to determine how many fields can be predicted with bearable error.

3.3 Implementation Issues

Most of the experiments within the scope of this thesis are based on the existing video coder [And03]. A description of the main features of this coder can be found in chapter 2. The coder is implemented in MATLAB, for some time-critical routines C is used.

One important task of the thesis work was to build a test environment that consists of a video coder and a decoder and that can be used to investigate the effects of transmission errors as described above. The source code that was taken from [And03] consisted only of routines for video coding, no decoder had been implemented because all necessary measures could be derived within the coder. Therefore the implementation of a decoder was a part of the thesis work.

From the block diagram of the decoder in figure 3.2 it was obvious to reuse most of the coder routines. The most important difference of the coder and the decoder is the handling of the residual image. The coder calculates the residual by subtracting the reference image from the current motion compensated prediction. The decoder has no access to the reference image, instead the residual is transmitted from the coder and then used to calculate the received image.

The coding of intra frames is not shown in the block diagrams. The first two frames of a test sequence are intra-coded. The coder uses wavelet compression for the very first field and a combination of block matching prediction and wavelet coding for the following fields. For the sake of simplicity the decoder of the test environment simply loads the reconstruction of the intra-coded fields from the coder, i.e. the intra fields are transmitted uncoded but including adequate coding distortion.

The coder [And03] originally performs an additional refinement of the backward coded images using a forward prediction to increase coding performance. This so called “backward-forward motion compensation” scheme is not used throughout this thesis.

A mechanism to drop the residual information for one or more selected fields has been included in the decoder in order to simulate transmission errors. The residual image
of the selected fields is set to zero instead of loading the reconstructed residual (cf. section 3.2).

### 3.4 PSNR Performance

The ability of the motion compensated prediction to replace lost frames was tested on a number of video sequences. Figure 3.4 shows a typical example. Most examples presented here will use the “rugby sequence”. This excerpt from a rugby match provides both changes in global motion (camera pan) and rapidly changing local motion. Therefore it is a challenge to any motion estimation algorithm and well suited to demonstrate most effects that will be discussed.

Figure 3.5 presents the difference between the predicted sequence (without residual information) and the regularly decoded sequence in terms of the peak signal-to-noise ratio. The PSNR is defined as

$$\text{PSNR} = 20 \log \left( \frac{b}{\sqrt{\sum (\hat{I} - \tilde{I}_{\text{correct}})^2}} \right),$$

where $b$ is the largest possible pixel value, $\hat{I}$ is the predicted image, $\tilde{I}_{\text{correct}}$ is the correctly decoded image (including residual information for all fields) and $\sqrt{\sum (\hat{I} - \tilde{I}_{\text{correct}})^2}$ is the root mean square difference of those two images.$^1$

Field 1 of figure 3.5 refers to the first field that is predicted without residual information. Any preceding field is correctly decoded, i.e. the difference is zero. Two interlaced fields make up one frame. As we can see, the PSNR decreases quickly. This means that the predicted sequence can be used to replace the correctly decoded sequence for a small number of fields only, otherwise the error will probably not be tolerable.

Figure 3.5 also demonstrates that it depends on the sequence how fast the prediction error increases. The ability of the discussed algorithm to bridge a certain number of lost fields heavily depends on the given video sequence. Chapter 4 will address this problem in more detail.

The rapid decrease in PSNR during the first predicted fields is supposed to have various reasons:

- Changes in global motion (camera panning) that cannot be foreseen. The prediction continues the global motion of the last field before the residual information is lost. If the global velocity changes after that, the prediction will be displaced with regard to the correctly decoded field. A global displacement is likely to cause a steep increase in PSNR.

---

$^1$cf. the definition of the PSNR in section 2.3
Figure 3.4: Example sequence that has been predicted without residual information (left side) and the associated correctly decoded fields (right side). Prediction for 1 field (top), 3 fields, 5 fields and 10 fields (bottom) after residual information is lost.
Figure 3.5: Difference between the predicted fields and the regularly decoded fields of two example sequences
• Changes in local motion (moving objects) that cannot be foreseen. Without residual information the decoder assumes a uniform motion of any local object based on the local velocity of the last known-good field.

• Distortion caused by inaccuracies of the motion estimation. The predicted sequence will look very blurry after several fields because the estimated motion field is lowpass filtered by the decoder. Other problems are incorrect detection of object edges or the handling of covered or uncovered background.

The problem of changing (global or local) motion cannot be solved, because the information that would describe those changes is lost together with the residual. A slight improvement might be achieved if advanced models are used in the motion estimation, such as accelerated motion or a physical modeling. However, this would be a very complex approach.

Changing motion which is not predicted correctly does not have much impact on the visual impression of the first few predicted fields, in spite of the decreased PSNR. After a certain number of fields, the prediction gets more and more deformed, as shown in the lower part of figure 3.4.

The inaccuracy of the estimated motion field heavily depends on the used algorithms. It is likely to cause serious errors, e.g. artifacts at object edges.

### 3.5 Visual Impression

A number of test sequences was generated in order to investigate the ability of the discussed algorithm to conceal field losses. After a presumed transmission error, a certain number of fields was predicted without residual information. After that the decoder output was switched back to the correctly decoded sequence (including residual information for all preceding fields). The result was then judged whether there is noticeable error.

It turns out that the loss of one field can be concealed without noticeable error. The difference between the first predicted field and the corresponding correct image is quite small, figure 3.5 indicates a PSNR of typically more than 24 dB for the first predicted field. Additionally, one erroneous field is probably too short to be recognised unless there is a extremely large difference to the temporally adjacent fields.

In the range from two to five lost fields the proposed algorithm is able to replace the loss with fair visible error. The error increases with increasing number of fields that need to be predicted. Of course the visual impression depends on the source sequence that is used. Generally the algorithm is working quite well in the mentioned range.

A problem that arises is the (global or local) displacement of the predicted sequence with regard to the correctly decoded sequence. When the decoder switches back to
the correctly decoded sequence, there will be a discontinuity, giving the impression of a “jump” or a “flash” at the respective field. Chapters 4 and 5 will discuss methods to attenuate those undesirable effects.

Predicting seven or more fields is only possible with serious error. The predicted fields will get seriously deformed in this case.

3.6 Summary

This chapter introduces a procedure to conceal lost fields of a backward coded video sequence. When the residual information for one or more fields gets lost due to transmission errors, the decoder assumes the residual to be zero. This causes the decoding process to be continued based on the last correctly decoded field.

Throughout this thesis we assume that the residual information of complete fields is lost, i.e. not only a part of a field image. We further assume that the decoder is reset to the error-free state after a certain number of fields, e.g. by decoding information that arrives late over the network.

The performance of the proposed algorithm is evaluated in terms of prediction error and visual impression. The prediction error increases rapidly during the first few fields. This increase is mainly caused by changes to global or local motion in the original sequence that cannot be foreseen. However, the visual quality is very good if only one field is lost and still tolerable for about five to seven dropped fields. The overall impression is decreased by noticeable discontinuity when the decoder switches from the predicted fields back to the error-free state.
Chapter 4

Cross-Fading to Remove Temporal Discontinuity

This chapter investigates how cross-fading can be used to conceal the temporal discontinuity that arises after transmission errors when the decoded sequence switches back from purely predicted fields to the error-free state. The results of cross-fading of several example sequences are presented and analysed. Additionally, an algorithm is proposed which aborts the cross-fading when not appropriate.

4.1 Motivation

In chapter 3 it was concluded that prediction without residual information can be used at the decoder to conceal transmission losses for a certain number of fields. A drawback of this approach is the temporal discontinuity that occurs when the decoder switches back to the correctly decoded stream as soon as the lost information has been recovered (see section 3.2 on how this should happen). Incorrectly predicted motion leads to a displacement of certain areas or the whole predicted field compared to the correctly decoded field. This displacement causes a jolty or flashing impression in the sequence.

An approach to suppress the discontinuity is to cross-fade from the predicted sequence to the error-free sequence over some fields. The cross-fade increases the number of fields that are affected by a transmission error, but hopefully the overall impression of the decoded sequence can be improved because the flashing effect is concealed.

4.2 Experimental Results

Experiments were carried out on a number of test sequences in order to check whether the temporal discontinuity can be reduced by cross-fading. After a presumed transmission error a number $k$ of fields was predicted without residual information. Thereafter the prediction was continued for another $l$ fields while a linear cross-fade to the correctly decoded sequence was performed according to (4.1),

$$I_{out}(n) = \left(1 - \frac{n-n_0+1}{l+1}\right) \hat{I}(n) + \frac{n-n_0+1}{l+1} \hat{I}_{correct}(n), \quad n_0 \leq n < n_0 + l \quad (4.1)$$
where \( \hat{I}(n) \) is the predicted field image, \( \hat{I}_{\text{correct}}(n) \) is the correctly decoded field image, and \( n_0 \) means the number of the first cross-faded field.

\( l = 0 \) means no cross-fading, i.e. the decoder switches immediately to the error-free state as discussed in the previous chapter.

Figure 4.1 gives an example for cross-fading. Three consecutive fields are shown. After an assumed transmission error \( k = 3 \) fields are predicted without residual information. The top part of figure 4.1 shows the third predicted field. Now a cross-fade is performed during \( l = 1 \) field (middle). At the bottom image the next decoded field can be seen.

A problem when evaluating the experiments was that a quality measure that would describe the effect of short errors within a video sequence was not on hand. Therefore the subjective impression of different test sequences was judged. Table 4.1 summarises the results for various combinations of \( k \) and \( l \) in a typical experiment. Grades in the range from 1 to 5 were assigned to the overall impression of each generated sequence, where 1 means no noticeable error, 3 means tolerable error and 5 stands for a very disturbed sequence. Additionally, the observed discontinuity is indicated by the symbols +, o or -, where + means that there is no noticeable discontinuity, o is assigned if there is some discontinuity and - means that there is a serious “flashing” effect.

<table>
<thead>
<tr>
<th>cross-fade duration ( l )</th>
<th>predicted frames ( k ) before cross-fade</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (none)</td>
<td>1/+ 2/o 3/o 4/- 4/- 5/-</td>
</tr>
<tr>
<td>1</td>
<td>2/+ 2/+ 3/o 4/o 4/-</td>
</tr>
<tr>
<td>3</td>
<td>2/+ 3/+ 3/+ 4/+ 4/o</td>
</tr>
<tr>
<td>5</td>
<td>3/+ 3/+ 4/+ 4/+ 4/o</td>
</tr>
</tbody>
</table>

**Table 4.1:** Evaluation of the perceived quality of a test sequence depending on the number of predicted fields and the cross-fade duration

Of course it is very difficult to state an absolute error for the given test sequences. The results that are presented in table 4.1 and later in this chapter should rather be seen as a relative evaluation to describe the impact of field losses and the proposed cross-fading on a certain test sequence.

We conclude from table 4.1 that the prediction is very good if only one field is lost. This is consistent with the results of chapter 3. No blending is required in this case.

After 2 to 5 predicted fields cross-fading is well suited to reduce the temporal discontinuity without affecting the overall quality too much. In most of the test sequences a + could be reached by applying the tested cross-fading algorithm.

At 7 or more predicted fields the overall quality decreases quickly. Cross-fading is less effective in this case as the difference between the prediction and the error-free state...
Figure 4.1: Example for cross-fading from the predicted sequence to the correctly reconstructed sequence
Kap. 4 Cross-Fading to Remove Temporal Discontinuity

gets very large.

An important question is how the cross-fade duration \( l \) should be chosen. If the decoder cross-fades too quickly, the discontinuity will not be suppressed well. If the cross-fading is expanded over too many fields the overall impression will be deteriorated. A probable blending scheme would be to perform no cross-fading when only one field is lost, to cross-fade over \( l = 1 \) field after two lost fields, and to use \( l = 3 \) fields cross-fading for losses of three or more fields. This blending has been tested in a number of experiments and is proposed as a default that should work with many sequences.

Unfortunately there are also situations when cross-fading should not be used. Figure 4.2 gives an example where the overall error will be seriously increased when the predicted and the error-free sequence are cross-faded. Again, three consecutive fields of the decoder output are shown. The first field is a prediction after three fields loss of residual. In the second field the decoder cross-fades back to the error-free state with \( l = 1 \).

The problem that arises in the presented example is a change in global motion that takes place after the transmission is interrupted and therefore cannot be predicted. This will result in a global displacement of the predicted sequence compared to the original sequence. The overall quality will be impaired when both sequences are cross-faded. It is likely that a ghost will be visible, as seen in figure 4.2.

Section 4.4 will present an algorithm which tries to detect whether the decoder should perform cross-fading on a given sequence.

4.3 Using Wrong Residual Information for Prediction

In the preceding sections we always assumed that no residual information is available between a transmission error and the point of time when the decoder is able to switch back to an error-free state. Actually it could also happen that the residual of one field is lost, e.g. due to the loss of one packet on the network, but succeeding fields arrive regularly. We assume that the lost packet can be reconstructed a certain number of fields later, e.g. by retransmission, and then the decoder will be able to switch to the error-free state. The question is whether the residual information that arrives after a field loss can be used in the prediction of the succeeding fields.

The decoder will not be in the same state as the coder after predicting a field without residual information. The residual information that arrives for the succeeding fields will be incorrect for the decoder, because it refers to the state of the coder. But as the prediction error is quite low, it should be possible to use the “wrong” residual information nevertheless.

Figure 4.3 presents an example of a correctly decoded sequence (right) and its prediction if the residual information of the first field is dropped. The prediction of the first two
Figure 4.2: Example for a situation where cross-fading does not work
Figure 4.3: Example sequence that has been predicted with residual information dropped for one field (left side) and the associated correctly decoded fields (right side). Prediction for 1 field (top), 3 fields, 5 fields and 10 fields (bottom) after residual information is dropped.
4.3 Using Wrong Residual Information for Prediction

fields is very good in this case. The quality of the third and the fifth predicted field is increased compared to the experiments that did not use residual information after the transmission error (figure 3.4). The prediction without residual information in figure 3.4 looks very blurred after several fields. The use of residual information in figure 4.3 obviously adds new detail to the sequence. New edges appear within the predicted images.

It is even possible to predict ten or more fields when “wrong” residual information is used. The quality of such a long prediction will be strongly increased compared to the case where no residual information is used, but there still will be notable error and the difference to the correct sequence will still increase over time.

In the following experiment one field of a test sequence was predicted without residual. The succeeding \( k - 1 \) fields were predicted by using “wrong” residual information. After that cross-fading to the error-free sequence was performed. Table 4.2 summarises the results. The evaluation was done as in section 4.2.

<table>
<thead>
<tr>
<th>cross-fade duration ( l )</th>
<th>predicted frames ( k ) before cross-fade</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (none)</td>
<td>(1/+(^1) 2/+ 2/(\circ) 3/(\circ) 4/(\circ) 4/(\circ))</td>
</tr>
<tr>
<td></td>
<td>1/+ 2/+ 2/+ 3/+ 4/+</td>
</tr>
<tr>
<td>1</td>
<td>1/+ 2/+ 2/+ 3/+ 4/+</td>
</tr>
<tr>
<td>3</td>
<td>2/+ 2/+ 3/+ 4/+ 4/+</td>
</tr>
<tr>
<td>5</td>
<td>2/+ 2/+ 3/+ 4/+ 4/+</td>
</tr>
</tbody>
</table>

\(^1\) same as section 4.2

Table 4.2: Perceived quality of a test sequence depending on the number of predicted fields \( k \) and the cross-fade duration \( l \) where residual information is dropped only for the first predicted field

The results are generally better than those in section 4.2 where no residual information was used, especially for \( k = 3 \) and \( k = 5 \) predicted fields. The “flashing” effect caused by the temporal discontinuity is not a problem in this experiment. It turns out that the residual information adds edges at correct positions to the predicted image, although the prediction is based on incorrectly decoded fields.

The proposed blending scheme (cross-fade three fields if more than two fields needed to be predicted) works fine in this experiment. The cross-fading algorithm is able to conceal the temporal discontinuity even after ten predicted fields.

Other experiments were performed to see if wrong residual information is still helpful after three or five purely predicted fields, i.e. longer transmission interruptions. It turned out that in this case the overall quality could not be improved except for very long predictions (more than ten fields). If the wrong residual information is included...
after several purely predicted fields, suddenly new edges are added to the sequence at unexpected positions. This deteriorates the predicted sequence more than being helpful and increases the “flashing” impression because of new temporal discontinuity.

There might be problems if wrong residual information is used immediately after a scene change. In this case, the state of the decoder could be considerably different from the coder. It has not been investigated whether it is helpful to use residual information in this case.

### 4.4 Automatic Abort of Cross-Fading

In section 4.2 it turned out that the proposed cross-fading algorithm should not be used in certain situations. Problems occurred especially when there was a global or local displacement of the predicted sequence to the error-free sequence. In those cases cross-fading is not able to reduce the discontinuity, but distinctly decreases the perceived quality. Therefore an algorithm that detects whether the decoder should cross-fade a certain sequence would be helpful.

A displacement of the predicted sequence is likely to cause a high per-pixel difference to the correctly decoded sequences. Therefore it should be possible to use the RMS difference between those two sequences as a measure to decide whether to cross-fade.

Figure 4.4 shows the RMS difference of the predicted fields to the correctly decoded sequence in some experiments. In addition to the RMS difference, the mean absolute difference as seen in figure 4.5 is also investigated. Sequence A refers to a typical example where the proposed cross-fading algorithm produces good results, as described in section 4.2. Sequence B means the sequence of figure 4.2 where cross-fading deteriorated the overall quality. The third graph results from an experiment where wrong residual information was used after one purely predicted field according to section 4.3. Field 1 refers to the first predicted field after residual information is lost.

It is obvious that the difference increases very steeply at sequence B which performed bad with cross-fading. On the other hand, the sequence which uses wrong residual information rises slowly compared to the other examples. There was almost no temporal discontinuity in this case. This verifies the assumption that the presented difference measures can be used to select cross-fading depending on the sequence.

It is interesting to see that the RMS difference and mean absolute difference are very similar in shape. Both could be used within an algorithm. An advantage of the mean absolute difference is that it needs much less computational effort.

A proposed algorithm would be as follows: After a transmission error the decoder predicts several fields without residual information. As soon as the error-free state can be reconstructed, the decoder calculates the difference between the currently predicted
Figure 4.4: Root mean square difference of the predicted field images after a transmission error to the correctly decoded fields of some example sequences.
Figure 4.5: Mean absolute difference of the predicted field images after a transmission error to the correctly decoded fields of some example sequences.
4.4 Automatic Abort of Cross-Fading

Field and the corresponding correctly decoded field. If this difference exceeds a given threshold, the decoder switches immediately to the correctly decoded stream. Otherwise, the current field is cross-faded according to (4.1). This procedure is repeated for each field until the cross-fade is complete or the threshold is exceeded. In other words: the decoder aborts the cross-fading immediately as soon as the (RMS or mean average) difference of the predicted and the original sequence gets too large.

This algorithm was tested on various example sequences. Empirical threshold values were 0.15 in case of the RMS difference and 0.1 in case of the mean absolute difference. These values refer to pixel values in the range from 0 to 1. A RMS difference of 0.15 corresponds to a PSNR of 16.5 dB (see section 2.3 for a definition of the PSNR).

Let us have a look at how the algorithm operates at the examples of figure 4.4 and 4.5. Sequence B reaches the threshold of 0.15 resp. 0.1 after three predicted fields. That means that no cross-fading will take place after three or more predicted fields. This is exactly what we intended for sequence B, where cross-fading does not work. According to the scheme that was proposed earlier in this chapter, we will cross-fade over one field after two predicted fields. Actually this is the sole combination where the decoder will cross-fade this sequence.

In case of sequence A, the decoder stops to cross-fade after seven predicted fields (resp. six predicted fields if using mean average difference). Sequence A represents a typical example where cross-fading should be used to reduce “flashing” effect. The proposed algorithm will allow cross-fading over three fields after at most four predicted fields. After seven or more predicted fields, no cross-fading will be performed at all. However the overall perceived quality decreases quickly after more than five predicted fields, as seen in table 4.1. The more important range up to five predicted fields is handled well by the proposed algorithm.

Temporal discontinuity turned out not to be a problem if residual information is dropped for only one field and continued for the remaining fields of the prediction, as described in section 4.3. In this case considerably more than ten fields can be predicted until the threshold is reached. The proposed algorithm allows blending in all useful cases.

The presented algorithm was tested with several other test sequences too, with reasonable results. It can be concluded that the algorithm is a useful extension to the proposed blending scheme of section 4.2. It will prevent cross-fading when not appropriate in many cases.

The presented algorithm also handles the problem of scene detection. If the scene of the original sequence changes while the transmission is interrupted, the decoder will keep on predicting based on the last received frames, i.e. on the old scene. The decoder switches to the new scene when it reconstructs the error-free state. Cross-fading to conceal discontinuity is not needed in this scenario, it would instead lead to an unwanted cross-
fade between the scenes, adding additional error to the new scene. But as the RMS
difference (resp. mean average difference) between fields of two different scenes is very
large, the cross-fade will be suppressed if the decoder uses the detection mechanism that
is proposed in this section.

4.5 Summary

To overcome the problem of the temporal discontinuity that occurs after a transmission
error when the decoder switches from the predicted stream to the reconstructed error-
free stream, cross-fading can be used. A cross-fade algorithm is discussed. Experiments
show that the proposed algorithm is able to reduce the “flashing” or jolty impression
caused by the temporal discontinuity in many situations.

A blending scheme is suggested that uses no cross-fading if there has been one field
predicted, performs a cross-fade over one field if two fields have been predicted, and uses
a cross-fade over three fields after three or more predicted fields.

Some experiments using “wrong” residual information are discussed, i.e. residual infor-
mation is only dropped for one field. Succeeding fields use residual information from the
coder, even though this residual information is based on the state of the coder, not on
the erroneous state of the decoder.

There are also situations when cross-fading should not be used. If the local or global
displacement caused by the prediction during the lost field gets too large, a cross-fade
is likely to deteriorate the perceived quality. Therefore an algorithm is proposed which
immediately aborts the cross-fade if the difference of the predicted fields to the error-
free state exceeds a given threshold. This algorithm also prevents a cross-fade when the
scene changes while the transmission is interrupted.
Chapter 5
Block Matching Prediction to Remove Temporal Discontinuity

This chapter discusses a different approach to remove the temporal discontinuity which occurs when the decoder switches back to the error-free sequence. This method is based on block matching.

5.1 Idea

We still assume that several fields are lost due to transmission problems and that the decoder replaces those fields by their motion compensated prediction. After a given number of fields, the decoder is able to reconstruct the error-free state, e.g. by evaluating packets that arrive late, and switches back to the correctly decoded sequence.

Chapter 3 concluded that the perceived quality of the decoded sequence suffers from a discontinuity when the decoder switches back. This temporal discontinuity causes a “flashing” or jolty impression. In chapter 4, cross-fading was used to conceal those effects.

In those chapters it turned out that displacements within the predicted field image compared to the error-free image are the main reason for the temporal discontinuity. These may be local displacements if object motion is incorrectly predicted, or a global displacement of the whole image. The displacement is obvious in figure 4.2.

In order to reduce the discontinuity, an algorithm could try to identify the displacement of the predicted image as soon as the correctly decoded image is reconstructed at the decoder. For the next few fields the decoder could shift parts of the image towards the right position, so that a smooth transition is achieved.

Figure 5.1 illustrates this approach. We consider a sequence that contains only one very simple object. Position A is the object position after some predicted fields. B would be the correct position, i.e. where the object should be when the decoder switches to the error-free state.

The suggested algorithm consists of two steps. At first, a displacement vector \( d = (\Delta x, \Delta y) \) is estimated which describes the object motion between the images A and B.
In the second step, the displacement \( d \) is multiplied by a scalar \( \alpha \), where \( 0 < \alpha < 1 \), and used to generate an image where the object has been shifted partially from A towards B. The resulting position in our example would be P1.

The two steps of the algorithm are repeated for a given number of fields. The factor \( \alpha \) is increased at each iteration. The object position at the next field of our example sequence would be P2 if we assume that the predicted image A and the correct image B are unchanged. The object moves towards its correct position, and the discontinuity is suppressed.

Figure 5.2 gives an overview of the processes within the decoder after a transmission error. It shows the field images of an example sequence in chronological order. The images which are output by the decoder are marked with a thick border. The field images on the left side are regularly decoded.

In this example, the two field images B4 and B5 are lost due to a transmission error. The decoder replaces the lost fields by a prediction A4 and A5 without residual information, as described in the previous chapters of this thesis. The correctly decoded fields B4 and B5 are reconstructed as soon as the missing information arrives. Immediately switching back to B6 would cause a discontinuity at the output sequence. Therefore, two fields P6 and P7 are generated to smooth the transition, using the algorithm which is discussed in this chapter. In chapter 4, this smoothing was achieved by cross-fading.
Figure 5.2: Procedure at a transmission error
5.2 Identifying Object Displacements by Block Matching

The description of the algorithm leaves open how the decoder should determine the local displacement between two images A and B. This thesis suggests a method which is based on block matching. The main reason to choose block matching has been the ease of implementation with the given test environment, which already contained some block matching routines.

A typical block matching algorithm selects blocks of a given image as an approximation of the blocks of a reference image, so that the RMS error is minimised. This means that there are two possible approaches in our case. One could select blocks of image B that fit image A, or vice versa. A more detailed discussion of those two approaches follows below.

5.2.1 Forward Estimation

At first we consider the approach in figure 5.3, which will be called “forward estimation” in the following. Image B is divided into blocks of equal size, where a typical block size would be 8 or 16 pixels. For each block of B, a block of image A is chosen that minimises the RMS error to the block of B. The location of the selected block is expressed in terms of the block displacement $d_i$, where $i$ refers to the block number. The selected block of image A is then used to predict image B. Figure 5.3 illustrates the procedure for one block of image B.

![Figure 5.3: Forward estimation: Use blocks of A to predict image B](image)

The block search is limited to a certain search region, which is usually defined as a maximum displacement. There are also block matching algorithms that use interpolation to determine block displacements with sub-pixel accuracy.

Forward estimation uses blocks of image A to generate a prediction of image B. In case of the algorithm in section 5.1, which increases the block displacements from zero to the
estimated value, this means that the sequence starts with image A, but image B is never reached, only its prediction.

The top row of figure 5.5 shows two example images A and B of a test sequence after a transmission interruption for a few fields. Image A has been predicted without residual information. Image B is the image that has been reconstructed. The prediction of B using blocks of A, i.e. the forward estimation approach, is shown on the right side of the bottom row. The reverse estimation, which is described below, is presented, too.

### 5.2.2 Reverse Estimation

The reverse estimation approach just exchanges the images A and B. Now image A is divided into blocks. For each block of A, a block of image B with a displacement $d_i$ is chosen which minimises the RMS error, as depicted in figure 5.4. The selected blocks of B are used to predict image A.

![Figure 5.4: Reverse estimation: Use blocks of B to predict image A](image)

Image A is never reached in case of reverse estimation, only its prediction. The algorithm in section 5.1 now starts at the estimated displacement, i.e. at the prediction of image A. The displacements are decreased over several fields. The sequence ends with image B when a displacement of zero is reached.

The bottom left image of figure 5.5 shows the reverse estimation of A using blocks from image B.

### 5.2.3 Combining Forward and Reverse Approach

We have seen that block matching uses only one of the two images as source for gray values, and therefore the other image cannot be reached. In case of the algorithm of section 5.1, it would be better if the generated sequence started with image A and
Figure 5.5: Two example images and their block matching estimations. Top left: Image A. Top right: Image B. Bottom left: Estimation of image A using blocks from B (reverse estimation). Bottom right: Estimation of image B using blocks from A (forward estimation).
ended exactly in image B. This can be achieved by combining the forward and the reverse estimation approach.

A possible solution would be to start with forward estimation and switch to reverse estimation after a few fields. This would however introduce new discontinuity to the sequence.

In the following we will use a combination of forward and reverse estimation and cross-fading. According to the beginning of this chapter, a scaling factor $\alpha$ is used to scale the block displacements. This $\alpha$ is increased from 0 to 1 over a given number of fields to smooth the transition between the sequences A and B. For each field both a forward and a reverse estimation is generated. A cross-fade similar to the previous chapters is carried out between the two estimations.

$$I_{\text{out}}(n) = (1 - \alpha) \hat{I}_{\text{fwd}}^{(\alpha)}(n) + \alpha \hat{I}_{\text{rev}}^{(1-\alpha)}(n),$$  \hspace{1cm} (5.1)

where $n$ is the field number, $\hat{I}_{\text{fwd}}^{(\alpha)}(n)$ is the forward estimation between the predicted and the correct field image, which is generated with block displacements multiplied by $\alpha$. $\hat{I}_{\text{rev}}^{(1-\alpha)}(n)$ means the reverse estimation, which is generated with block displacements multiplied by $(1 - \alpha)$.

The cross-fading process starts at $\alpha = 0$, which means the field image of sequence A without displaced blocks, and end at $\alpha = 1$, which refers to the unchanged field image of sequence B. Intermediate fields are blended between the two types of estimation.

A drawback of the supposed cross-fading is that two block matching procedure have to be performed for each field during the cross-fade, i.e. the computational effort is doubled. As we will see, computation time is critical with the proposed algorithm.

### 5.2.4 Selection of Block Size and Search Region

Both the block size and the search region need to be selected with the block matching approach (see section 5.2.1).

The search region, i.e. the area of the image which is searched for a given block, should be large in order to identify even large object displacements. If several fields of a sequence have been predicted without residual information, the displacement of fast objects is likely to reach a value of a few tens of pixels. Those objects will only be identified if the search region is large enough.

On the other hand a large search region increases the possibility that “wrong” blocks are selected. This means that the algorithm finds blocks within the search region that minimise the RMS error, but do not belong to the object that we are looking for. In this case the identified block displacement does not describe object motion and therefore cannot be used by the suggested smoothing algorithm.
The maximum search region is limited by the available processing power, too. The computational effort of the block search increases quadratically with the maximum block displacement, if an exhaustive search is used. There are advanced search methods that are much faster [Lun01], but such methods were not tested within this thesis.

Most problems with block matching algorithms are caused by block artifacts. In order to reduce block artifacts, the block size should be kept small. But in our case, a small blocks size will again cause more “wrong” block selections. The extreme example would be a block size of only one pixel, where the algorithm would pick any other pixel within the search region that has a matching gray value. The resulting block displacement will not be related to object motion at all.

Within the following experiments, a block size of 16 by 16 pixels was chosen, with a search region of 16 pixels (maximum block displacement in each direction). With these values, the computation time of a block search for one field image was about half the time that was necessary to decode the image, i.e. the algorithm which is suggested in section 5.2.3 used about the same computational effort as the complete video decoder.

5.3 Experimental results

The block based algorithm has been tested with several video sequences. The assumptions are the same as in the previous chapters. The residual information is dropped for a given number $k$ of fields; those fields are predicted by the decoder using motion compensated prediction. Thereafter the decoder applies the block based algorithm which has been discussed in sections 5.1 and 5.2.3 for a number $l$ of fields, in order to switch back to the error-free sequence.

A typical test sequence is presented in figure 5.6. Four consecutive field images are shown. In this example, $k = 3$ fields have been predicted without residual information. We consider the part of the sequence where the block-based algorithm changes back to the error-free state over $l = 3$ fields.

The top left image of figure 5.6 refers to the first image that is generated based on block matching. An $\alpha$ of 0.25 is used here, i.e. the blocks are only little displaced from their position in the erroneous sequence (cf. equation (5.1)). The resulting image still looks very similar to the erroneous image.

The succeeding field (top right) is halfway to the error-free image, it is generated with $\alpha = 0.5$. Now it is obvious that the image is composed of blocks. The blocks are visible especially in the middle of the image, where most of the local motion is located.

The third generated field uses an $\alpha$ of 0.75 and is already quite close to the corresponding error-free image. There are still visible block artifacts, e.g. in the centre of the image or around the legs of the man who is running in the front. At the bottom right of figure 5.6,
Figure 5.6: Example sequence: changing back from the erroneous to the error-
free sequence using the suggested block matching approach. Top left: First
field that is generated by the algorithm ($\alpha = 0.25$). Top right: Second field
($\alpha = 0.5$). Bottom left: Third field ($\alpha = 0.75$). Bottom right: Error-free
sequence is reached.
the transition to the error-free sequence is complete. The decoder continues with regular decoded field images.

Although the still images may look quite promising, the impression is quite different if the discussed example sequence is played at correct frame rate. The perceived quality is seriously decreased compared to the cross-fade algorithm of chapter 4. The problems are caused by block artifacts which are very obvious in the context of an image sequence. The block edges are literally flashing up.

The annoying block edges can be reduced by a suitable post-processing of the generated image. A number of experiments has been carried out with a simple unidirectional Gaussian lowpass filter. The filtering is applied to complete fields as long as the block-based algorithm is active. No filtering is done in the error-free state. It turned out that a standard deviation of 0.7 pixel is sufficient to conceal visible block edges. Most experiments use a standard deviation of 1 pixel, which further reduces the perceived error caused by block artifacts. The filtered images are noticeably blurred in this case, but this does not affect the overall impression very much.

More sophisticated approaches to conceal block edges are available, too. An example would be filtering which is aware of the local orientation of block edges. However, none of those advanced algorithms have been investigated in this thesis.

The block based approach, which is discussed in this chapter, remains inferior to a simple cross-fade as described in chapter 4 even if the block edges are suppressed. The overall quality is affected by incorrect blocks, i.e. block displacements which do not reflect object motion as described in section 4.1. If the difference between the two images that should be matched is too large, it is likely that no appropriate block is available within the search region. On the one hand the search region could be too small to cover fast motion. On the other hand the changes within a block could be so serious that this block is not recognised at the reference image.

In both cases, a block displacement within the given search region is chosen which minimises the RMS error to the reference image, but does not describe any object motion. Multiplying the displacement by a scaling factor $\alpha$ does not make sense here and will generate blocks that are moving in a completely wrong direction. Such an erroneous block can be seen at the top right image of figure 5.6 at the left should of the person in the foreground. This block causes a very obvious error.

Another important disadvantage of the investigated algorithm is the large computational effort which already has been mentioned in section 5.2.4. The cross-fade algorithm of chapter 4 needs much less computation power.

The question remains whether the idea to reduce temporal discontinuity by identifying and shifting objects, which has been outlined in section 5.1, would perform better if it was not based on block matching but on another algorithm. The block matching approach
fails because the difference between the predicted and the error-free sequence is too large after several fields have been predicted without residual information. The predicted sequence is likely to get distorted after such a long prediction, and the local displacement may reach 50 or more pixels. It is doubtful whether an alternative algorithm will be able to identify object displacements under these circumstances.

5.4 Summary

This chapter presents an alternative algorithm to conceal the temporal discontinuity that is perceived when the decoder switches back to the error-free state after several purely predicted frames. The algorithm is based on the fact that this discontinuity is caused mainly by a displacement of regions of the predicted sequence compared to the error-free sequence. The idea is to determine the displacement and shift the identified regions towards their correct position over a few fields. This should smooth the transition.

A block matching approach is used to identify displaced regions between two images (namely the predicted and the correct image of the current field). This information is used to generate an intermediate image. Common block matching algorithms use only pixel values of one image to generate their output, therefore a method is suggested which combines two block matching procedures in order to include gray values of both reference images into the generated image.

The impact of block size and search region on the proposed algorithm is discussed. A relatively large search region has to be chosen, but this considerably slows down the algorithm.

The suggested algorithm is investigated in experiments with a number of test sequences. It turns out that it basically works, but the perceived quality is seriously decreased by block edges which appear in the generated sequence. Lowpass filtering is used to reduce those block artifacts.

The block based algorithm remains inferior to the cross-fade algorithm of chapter 4 in terms of perceived quality and computational effort, even if block edges are suppressed. This is due to object motion which is not identified correctly at certain blocks. These blocks introduce serious error to the generated sequence.
Chapter 6
Increased Stability Coding

In this chapter, the given video coder is modified so that additional energy of the reference image is included in the residual. This will limit the effects of field losses to the decoder to several fields. This approach is different to other mechanisms described in this thesis. It is meant as a reference to evaluate the performance of those mechanisms.

6.1 Idea

All experiments throughout this thesis are based on the backward video coder which is described in chapter 2. The sole information that the coder transmits is the residual image, which is calculated by subtracting the reference image of each field from its motion compensated prediction. We now modify this coder so that the predicted image is multiplied by a constant $0 < \gamma \leq 1$ before calculating the residual. This multiplication is done, too, when the reconstructed image is calculated from the reconstructed residual and the predicted image. Figure 6.1 exhibits the modified coder. The decoder is changed in the same way.

A factor $\gamma = 1$ would describe the normal mode of operation of the coder, as it is used in the other chapters of this thesis. A multiplication by $\gamma < 1$ artificially increases the prediction error. In this case, the residual image contains additional information of the reference image. A $\gamma = 0$ would mean to transmit the reference image only without any motion compensated prediction.

After transmission errors, a $\gamma < 1$ should cause the decoder to converge towards the error-free state, i.e. the error should decrease over a certain number of fields instead of propagating forever. On the other hand, the quantisation error will be increased because energy is added to the residual image. A $\gamma < 1$ will decrease the PSNR\(^1\) even when the transmission is undisturbed.

As opposed to other algorithms that are discussed within this thesis, the modified decoder will not need to reconstruct information that has been lost in a transmission error. In other chapters we assume that the decoder recalculates the error-free state by using

\(^1\) see section 2.3
packets that arrive late or are retransmitted. No such mechanism is needed here. Actually, there is no need for any dedicated error concealment mechanism in this case. The decoder should start to regain an undisturbed state as soon as residual information is received again.

6.2 Experiments

A number of test sequences has been generated by using the modified coder and decoder as described in section 6.1. It is assumed that a given number $k$ of fields were lost during transmission. The residual images of those fields are set to zero at the decoder. The decoded sequence is evaluated for different combinations of the number of dropped fields $k$ and values of $\gamma$.

Figure 6.2 presents some decoded fields of a typical test sequence. In this case, $k = 7$ fields have been predicted without residual information during a transmission interruption. The upper left of figure 6.2 shows the seventh predicted field. Now the residual information is switched on again. We see how the decoder changes back to the error-free state over several fields. The visible error fades away. After about seven fields (not shown here), there is almost no visible difference to the sequence that would have been decoded in case of no transmission interruption. $\gamma = 0.7$ has been used in this example.

The motion compensated prediction is a highly nonlinear algorithm. It was difficult to anticipate what value would be sufficient for convergence of the decoder. A low $\gamma$ should cause a quick error recovery. On the other hand a low $\gamma$ impairs the coding performance. Therefore the investigation focussed on the question how $\gamma$ should be chosen.
Figure 6.2: Example sequence which shows how the decoder recovers from a transmission interruption. Top left: Last field that has been predicted without residual information. Top right: First field after residual is continued. Top left: Third field. Bottom right: Fifth field.
Within all experiments, $\gamma = 0.7$ provided enough stability to recover from field losses quickly. The results were compared to the experiments in section 4.2, which perform cross-fading after a transmission error. The overall perceived quality is found to be similar to the case where three field cross-fading is used.

There are some noticeable differences to the cross-faded sequences. The decoder with $\gamma = 0.7$ adds new detail to the sequence very soon after the residual information is continued. Figure 6.2 demonstrates that there is a distinct change in the very first field which includes new residual information. It is likely that this sudden change is perceived as a discontinuity, although it takes several more fields until the error has faded away.

Other experiments have been carried out in order to evaluate whether higher values of $\gamma$ can be used. It turned out that $\gamma$ should not be chosen greater than 0.7 to guarantee that the error does not propagate for more than a few fields. A value of $\gamma = 0.85$ was critical with many test sequences. In this case the error was likely to propagate over many fields, the test sequences showed noticeable errors even more than 40 fields after the transmission interruption. Problems showed up especially around the edges of moving objects. Typically, flattering or wavy distortion appeared within the sequence.

### 6.3 Summary

This chapter investigates a modification of the given video coder. The motion compensated prediction is multiplied by a constant $0 < \gamma \leq 1$ before calculating the residual. Under certain conditions this will cause the decoder to converge to the error-free state after transmission errors.

The selection of $\gamma$ is crucial for the ability to recover from errors. The impact of $\gamma$ is investigated on various test sequences. It is concluded that $\gamma \leq 0.7$ should be chosen. In this case, the ability to conceal transmission errors is comparable to the cross-fading algorithm that is discussed in chapter 4. Higher values of $\gamma$ cause an error propagation over many fields.

The suggested modification is a very simple means to add error resilience to the given video coder. No dedicated algorithm to for error concealment is needed, the decoder starts to recover as soon as residual information is received again.

The main drawback of the modified coding is the increased quantisation error, especially if $\gamma$ is chosen considerably smaller than one. The PSNR is increased even when there are no transmission errors. It is also not yet clear whether $\gamma = 0.7$ is sufficient in any situation or even lower values should be necessary.

The discussed approach seems not to be well suited for error concealment with the given coder due to its disadvantages. The algorithm which is discussed in chapter 4 will perform similar without the described problems.
Chapter 7

Summary

This thesis investigates a procedure to conceal lost fields of a coded video sequence. The considered coder uses backward motion compensated prediction, i.e. the prediction is based on previously decoded images. When the residual information of one or more (interlaced) fields is not available due to transmission errors, the decoder assumes the residual to be zero. By doing so, the motion compensated prediction of the decoder replaces the fields which cannot be correctly decoded.

The scenario of the experiments within the scope of this thesis is as follows: The residual information of a certain number of fields gets lost during transmission. Those fields are replaced by their motion compensated prediction at the decoder. After that, the decoder is reset to the error-free state, i.e. as if all previous fields had been correctly decoded. A possible way to achieve this would be to decode information that arrives late or is retransmitted. A typical application would be real-time video transmission over a packet-switched network, where the packets must not exceed a given delay in order to be regularly decoded.

The suggested algorithm is able to replace one lost field without noticeable error, and gives results with tolerable error up to seven dropped fields, depending on the video sequence. The perceived error rapidly increases if more fields need to be predicted. The overall impression of the resulting sequences is decreased by noticeable discontinuity when the decoder switches back from the prediction to the error-free state.

An algorithm is presented which uses cross-fading to reduce the temporal discontinuity. The procedure is the same as above, but now the decoder does not switch immediately from the prediction to the error-free sequence. A cross-fade between the two sequences is performed instead. The duration of the cross-fade is chosen depending on how many fields have been predicted. No cross-fading is used if only one field is lost. A cross-fade over one field is performed after two predicted fields. Cross-fading over three fields is selected after three or more predicted fields. Experiments show that this procedure reduces the “flashing” effects, which are caused by the temporal discontinuity, in many test sequences.

The proposed cross-fade algorithm should not be used in situations where the global
or local displacement of the prediction to the error-free field gets too large. In this case cross-fading deteriorates the perceived quality. Thus an extension to the algorithm is suggested which immediately aborts the cross-fade if the difference of the predicted image to the error-free field exceeds a given threshold.

An alternative approach to conceal the temporal discontinuity is based on block matching. The idea is to determine the displacement of objects in the predicted image to their position in the error-free image. This displacement is used to generate intermediate images in which the objects are shifted towards their correct position. Thus the transition should be smoothed.

It turns out that test sequences which are generated by the block based algorithm suffer from block artifacts. Those artifacts are obvious in the context of an image sequence. A simple lowpass filtering is used to remove block edges, but certain blocks still introduce very obvious error to the generated sequence. These errors occur if the estimated block displacement does not correspond to object motion. Another disadvantage of the block based approach is the high computational effort which is needed for the block search.

Finally, a modification to the coding algorithm is discussed that causes the decoder to converge towards the error-free state after a transmission error. This is achieved by multiplying the motion compensated prediction by a constant lower than 1 before the residual image is calculated. No dedicated error concealment algorithm is needed in this case. This approach is investigated for reference purposes, but it is not recommended for regular coding, as the coding error is increased even if the transmission is undisturbed.

It can be concluded that the motion compensated prediction of a backward video decoder can be used to conceal transmission interruptions. The ability to predict up to seven fields without information from the coder is more than was expected. Two approaches to reducing the discontinuity in the resulting sequences are presented. The first approach, which uses cross-fading, is very simple and gives good results. The second, block-based approach is much more complex, but the results are not satisfying yet. On the other hand, the block-based algorithm may be improved in many ways. Some ideas are outlined in the following chapter.
Chapter 8
Suggestions for Further Research

The results of the block-based algorithm of chapter 5 are not satisfying yet, but there should be a number of approaches to improve the algorithm:

- The block matching could be improved in several ways. The experiments use a very simple post-processing to remove visible block edges. A more sophisticated post-processing should be able to further reduce block artifacts.
  
  The generated test sequences are disturbed by blocks which are moving in “wrong” directions. It should be possible to detect such blocks by analysing the spatial consistency of the block displacements. Block displacements that differ very much from neighbouring blocks could be treated as unreliable and replaced.
  
  The computation time of the block search can be decreased if alternative search algorithms are used, such as a hierarchical search or reduced search strategies [Lun01].

- An optical flow algorithm could be used to estimate the displacement between the sequences. An optical flow algorithm would avoid the problems with artifacts and inherently provide spatial consistency. [BFBB94] reviews various optical flow techniques.

- An alternative approach would be to track displacements between consecutive fields, starting at the last correctly decoded field. The idea is to accumulate the displacements for each field of the purely predicted sequence. The same is done for the reconstructed sequence. The difference of those accumulated displacements should give the displacement between the predicted and the error-free image. The required information is taken from the motion estimation of the decoder.
  
  Unfortunately, there are a number of pitfalls in this approach. The resulting displacement will be irregularly sampled. Displacement vectors may point to the same point, while other regions are not covered by the resulting displacement (similar to the problem of covered background). It will also happen that the displacement exceeds the image boundaries.
  
  When the decoder begins to shift objects towards their correct position, new “incorrect” motion will be introduced to the output sequence. This artificial motion is likely to cause noticeable errors, no matter how good the displacements are es-
estimated. It might also be that objects cross each other while they are shifted to their correct position.

- Maybe the algorithm will give better results if the estimated displacements are considerably smoothed in order to ensure spatial consistency. This would limit the idea of shifting image regions to a coarse level. The algorithm could be even reduced to estimate the global displacement and shift the predicted image globally for a few fields.
Bibliography


På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extra-ordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnt som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är krävande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet – or its possible replacement – for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Claudius Volz