UNEQUAL ERROR PROTECTION FOR H.264/SVC BITSTREAM

A Thesis
Presented to the
Faculty of
San Diego State University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Electrical Engineering

by
Shireen Shankar
Summer 2010
SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the
Thesis of Shireen Shankar:

Unequal Error Protection for H.264/SVC Bitstream

Sunil Kumar, Chair
Department of Electrical and Computer Engineering

Santosh Nagraj
Department of Electrical and Computer Engineering

Carl Eckberg
Department of Computer Science

06/07/2010
Approval Date
Copyright © 2010

by

Shireen Shankar

All Rights Reserved
DEDICATION

To my Loving Parents
Dr. Om Shankar Srivastava and Mrs. Neelima Srivastava
Once you start working on something, don't be afraid of failure and don't abandon it. People who work sincerely are the happiest.

- Chanakya
ABSTRACT OF THE THESIS

Unequal Error Protection for H.264/SVC Bitstream
by
Shireen Shankar
Master of Science in Electrical Engineering
San Diego State University, 2010

Today, the wide variety of devices in the digital world ranges from desktops to mobile phones. Within the currently available interactive multimedia applications, there are demands in terms of video quality and coding efficiency, the cost as well as scalability. That is why there is a need for scalable video coding schemes which provide fully progressive bit streams and supports scalability.

Scalable Video Coding targets on seamless delivery of digital content and access to the same, enabling optimal user centered multi-channel and cross-platform media services, providing a straightforward solution for universal video delivery to a broad range of applications. Scalable video coding gives a nice way to perform rate shaping for video streams adapting to the available transmission resource.

The work in this thesis deals with the overview and practical implementation of the H.264 Scalable Video Codec. All the major building blocks of H.264/SVC codec are discussed and implemented. Various kinds of Scalabilities and Error Concealment methods are achieved and comparative studies are performed.

This thesis proposes and discusses Unequal Error Protection scheme in Scalable Video Coding. Unequal Error Protection method is an error resiliency scheme in which we protect the bit stream of the base and the enhancement layers based on priority levels. This priority information is based on the dependency, temporal, and quality scalabilities values.

We have also provided an external protection to the bit stream in the form of Reed Solomon code. A detailed implementation of this scheme is done and results obtained through these simulations and video quality evaluation, are provided, showing the system performance under various network conditions.
TABLE OF CONTENTS

ABSTRACT ............................................................................................................................. vi
LIST OF TABLES ..................................................................................................................... x
LIST OF FIGURES ................................................................................................................. xi
ACKNOWLEDGEMENTS ................................................................................................... xiv
CHAPTER

1 INTRODUCTION .........................................................................................................1
  1.1 H.264/AVC Basic ..............................................................................................2
  1.1.1 Introduction of H.264 AVC ......................................................................3
  1.1.2 Structure of H.264/AVC ...........................................................................3
    1.1.2.1 Network Abstraction Layer ..............................................................3
    1.1.2.2 Video Coding Layer .........................................................................4
  1.1.3 Important Features ....................................................................................5
  1.1.4 Error Resiliency Schemes in H.264/AVC ....................................................7
  1.2 Research Scope .................................................................................................7

2 SCALABLE VIDEO CODING .....................................................................................9
  2.1 Introduction ..........................................................................................................9
  2.2 Basic Concepts for Extending H.264/AVC Towards H.264/SVC ....................10
  2.3 Scalability Dimension .......................................................................................12
    2.3.1 Temporal Scalability ..............................................................................13
    2.3.2 Spatial Stability .......................................................................................15
      2.3.2.1 Basic Building Block of Spatial Scalability .....................................16
      2.3.2.2 Inter-Layer Motion Prediction ..........................................................18
      2.3.2.3 Intern-Layer Residual Prediction ......................................................19
      2.3.2.4 Inter-Layer Inter Prediction ...............................................................19
      2.3.2.5 Complexity Consideration ...............................................................20
      2.3.2.6 Coding Efficiency ..........................................................................21
    2.3.3 SNR /Quality Scalability .........................................................................22
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.3.1 Progressive Refinement Slice</td>
<td>23</td>
</tr>
<tr>
<td>2.3.3.2 Coarse Grain SNR Scalability (CGS)</td>
<td>25</td>
</tr>
<tr>
<td>2.3.3.3 Fine Grain SNR Scalability (FGS)</td>
<td>25</td>
</tr>
<tr>
<td>2.3.3.4 Bit-Stream Extraction</td>
<td>26</td>
</tr>
<tr>
<td>2.3.3.5 Coding Efficiency</td>
<td>27</td>
</tr>
<tr>
<td>2.3.4 Combined Scalability</td>
<td>28</td>
</tr>
<tr>
<td>2.4 NAL Unit Syntax</td>
<td>29</td>
</tr>
<tr>
<td>2.4.1 NAL Unit Header</td>
<td>30</td>
</tr>
<tr>
<td>2.4.2 Payload Structure</td>
<td>31</td>
</tr>
<tr>
<td>2.5 Backward Compatibility for SVC</td>
<td>31</td>
</tr>
<tr>
<td>2.6 Error Concealment in SVC</td>
<td>32</td>
</tr>
<tr>
<td>2.6.1 Frame Copy (FC)</td>
<td>32</td>
</tr>
<tr>
<td>2.6.2 Temporal Direct (TD)</td>
<td>32</td>
</tr>
<tr>
<td>2.6.3 Motion and Residual Upsampling (BLSKIP)</td>
<td>33</td>
</tr>
<tr>
<td>2.6.4 Reconstruction Base Layer Upsampling (RU)</td>
<td>33</td>
</tr>
<tr>
<td>2.7 Extended Spatial Scalability</td>
<td>33</td>
</tr>
<tr>
<td>2.8 SVC Profile</td>
<td>37</td>
</tr>
<tr>
<td>2.9 Applications</td>
<td>37</td>
</tr>
<tr>
<td>3 LITERATURE REVIEW OF DIFFERENT PROPOSED ERROR RESILIENCY SCHEMES</td>
<td>39</td>
</tr>
<tr>
<td>3.1 Error Resilient Unequal Error Protection of Fine Granularity Scalable Video Bitstream</td>
<td>39</td>
</tr>
<tr>
<td>3.2 Robust Transcoding for Scalable Video Coding</td>
<td>41</td>
</tr>
<tr>
<td>3.3 Packet Loss Protection of Scalable Video Bitstreams Using Forward Error Correction and Feedback</td>
<td>44</td>
</tr>
<tr>
<td>4 IMPLEMENTATION AND RESULTS OF SVC</td>
<td>47</td>
</tr>
<tr>
<td>4.1 Spatial Scalability</td>
<td>47</td>
</tr>
<tr>
<td>4.1.1 Discussion of Results</td>
<td>48</td>
</tr>
<tr>
<td>4.1.2 Results for Spatial Scalability</td>
<td>48</td>
</tr>
<tr>
<td>4.2 SNR /Fidelity/Quality Scalability</td>
<td>49</td>
</tr>
<tr>
<td>4.2.1 Discussion of Results</td>
<td>51</td>
</tr>
<tr>
<td>4.2.2 Results for SNR Scalability</td>
<td>51</td>
</tr>
<tr>
<td>4.3 Temporal Scalability</td>
<td>51</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 3.1 SNR in dB of the Average Expected Distortion of the Proposed System and the MD System for a Packet Erasure Channel with an Erasure Probability of Pe=0.1 and Various Transmission Bitrates.................................................................46
Table A.1. Executable Provided by the JSVM Software (9.8).................................................................81
Table A.2. Different Sequence Formats Used for the Executions...............................................................83
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Structure of H.264/AVC encoder</td>
<td>4</td>
</tr>
<tr>
<td>1.2</td>
<td>H.264/AVC NAL unit syntax</td>
<td>4</td>
</tr>
<tr>
<td>1.3</td>
<td>H.264/AVC encoder structure</td>
<td>5</td>
</tr>
<tr>
<td>1.4</td>
<td>H.264/AVC Decoder structure</td>
<td>5</td>
</tr>
<tr>
<td>2.1</td>
<td>Principle of encoding</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>Principle of decoding</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>Block diagram of SVC for combined scalability</td>
<td>11</td>
</tr>
<tr>
<td>2.4</td>
<td>Hierarchical B picture concept in H.264/SVC</td>
<td>13</td>
</tr>
<tr>
<td>2.5</td>
<td>Temporal scalability</td>
<td>15</td>
</tr>
<tr>
<td>2.6</td>
<td>Multi-layer structure with additional inter-layer prediction for enabling</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>spatial scalable coding</td>
<td></td>
</tr>
<tr>
<td>2.7</td>
<td>Block diagram of spatial scalability</td>
<td>17</td>
</tr>
<tr>
<td>2.8</td>
<td>Spatial prediction of data</td>
<td>18</td>
</tr>
<tr>
<td>2.9</td>
<td>Analysis of the efficiency of the inter-layer prediction concepts in SVC for</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>different prediction structures</td>
<td></td>
</tr>
<tr>
<td>2.10</td>
<td>Fine granular scalability</td>
<td>26</td>
</tr>
<tr>
<td>2.11</td>
<td>Performance of SNR scalable coding</td>
<td>27</td>
</tr>
<tr>
<td>2.12</td>
<td>Comparison of concepts with different tradeoffs between enhancement layer</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>coding efficiency</td>
<td></td>
</tr>
<tr>
<td>2.13</td>
<td>H.264/SVC NAL unit structure</td>
<td>30</td>
</tr>
<tr>
<td>2.14</td>
<td>Comparison BLSkip, RU, TD, FC and no error cases with “Foreman” sequence</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>for different base layer packet loss (p) and enhancement layer packet loss</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.15</td>
<td>Subjective quality comparison of concealed pictures for “Foreman”, with a</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>base layer packet loss rate of 3% and an enhancement layer packet loss rate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>of 3%, QP=28. (a) BLSkip; (b) RU; (c) TD; (d) FC.</td>
<td></td>
</tr>
<tr>
<td>2.16</td>
<td>Relations between enhancement layer and base layer</td>
<td>36</td>
</tr>
<tr>
<td>2.17</td>
<td>Cropping window example</td>
<td>37</td>
</tr>
</tbody>
</table>
Figure 3.1. The ER-UEP scheme. ................................................................. 40
Figure 3.2. (a) Normal packetization (b) RD optimal packetization. ......................... 41
Figure 3.3. (a) Comparative evaluation of the proposed scheme at different packet loss ratios (b) Comparative evaluation on channels with prediction errors. The enhancement-layer rate is set to 768 kbps. ................................................................. 42
Figure 3.4. Proposed scalable packet unit. ........................................................................ 43
Figure 3.5. Proposed Error-Resilient protection scheme. .................................................. 44
Figure 3.6. Performance comparisons for 3-part scheme and 4-part scheme in SNR scalability. .................................................................................................................... 44
Figure 3.7. The packet loss protection scheme. Darker colors represent EL sublayers that are more important than the ones represented by lighter colors. ......................... 46
Figure 4.1. Spatial output. ........................................................................................................ 49
Figure 4.2. Command prompt output for spatial scalability. ................................................... 49
Figure 4.3. Output of SNR scalability. .................................................................................... 52
Figure 4.4. Command prompt output for SNR scalability. .................................................. 52
Figure 4.5. Temporal output. ................................................................................................. 54
Figure 4.6. Another view. ................................................................................................. 54
Figure 4.7. Temporal output at command prompt. ................................................................. 55
Figure 4.8. Output result for temporal-spatial combination. ................................................... 56
Figure 4.9. Command prompt output for the temporal-spatial combination. ........................................ 56
Figure 4.10. Command prompt output for the SNR-temporal combination. ........................................ 57
Figure 4.11. Result of error resiliency. .................................................................................... 58
Figure 5.1. Block diagram of the scheme. ............................................................................... 60
Figure 5.2. The UEP comparative output (a) UEP output (b) Original image. ....................... 62
Figure 5.3. UEP with error concealment (a) Output with UEP and error concealment together (b) Output with UEP only. ............................................................................. 62
Figure 5.4. (a) UEP output for spatial scalability (b) UEP output for temporal scalability (c) UEP output for SNR scalability. ................................................................. 63
Figure 5.5. UEP output for combined scalability (a) Output from temporal SNR combined scalability (b) Output from spatial temporal combined scalability. 66
Figure 5.6. UEP comparative output for SNR scalability....................................................... 67
Figure 5.7. Graphical representation of PSNR(Y) with UEP ................................................. 67
Figure 5.8. Graphical representation of PSNR(U) with UEP ................................................. 68
Figure 5.9. Graphical representation of PSNR(V) with UEP ................................................. 68
Figure 5.10. Average PSNR values with 3% PLR using different ER/EC method in case of temporal scalability..................................................................................................68

Figure 5.11. Average PSNR values with 3% PLR using different ER/EC method in case of spatial scalability. ....................................................................................................69

Figure 5.12. Average PSNR values with 3% PLR using different ER/EC method in case of SNR scalability. .......................................................................................................69

Figure 5.13. Average PSNR values with 3% PLR using different ER/EC method in case of temporal-SNR scalability. .......................................................................................69

Figure 5.14. Average PSNR values with 3% PLR using different ER/EC method in case of temporal-spatial scalability. .....................................................................................70

Figure 5.15. Average PSNR values with 5% PLR using different ER/EC method in case of temporal scalability..................................................................................................70

Figure 5.16. Average PSNR values with 5% PLR using different ER/EC method in case of spatial scalability. ....................................................................................................70

Figure 5.17. Average PSNR values with 5% PLR using different ER/EC method in case of SNR scalability. .......................................................................................................71

Figure 5.18. Average PSNR values with 5% PLR using different ER/EC method in case of temporal-SNR scalability. .......................................................................................71

Figure 5.19. Average PSNR values with 5% PLR using different ER/EC method in case of temporal-spatial scalability. .....................................................................................71

Figure 5.20. Average PSNR values with 10% PLR using different ER/EC method in case of temporal scalability.................................................................................................72

Figure 5.21. Average PSNR values with 10% PLR using different ER/EC method in case of spatial scalability. .................................................................................................72

Figure 5.22. Average PSNR values with 10% PLR using different ER/EC method in case of SNR scalability. .................................................................................................72

Figure 5.23. Average PSNR values with 10% PLR using different ER/EC method in case of temporal-SNR scalability. .......................................................................................73

Figure 5.24. Average PSNR values with 10% PLR using different ER/EC method in case of Temporal-Spatial Scalability. .................................................................73

Figure A.1. Libraries provided by JSVM (9.8) software.........................................................82

Figure C.1. Reed solomon codeword.....................................................................................87

Figure C.2. Architecture of reed solomon encoder.................................................................88
ACKNOWLEDGEMENTS

It is a pleasure to thank many people who made this thesis possible. I am grateful to my advisor Dr. Sunil Kumar for his Continuous encouragement and guidance. I especially thank him for his infinite patience. The discussions I had with him were invaluable.

I would also like to sincerely thank my committee members Dr. Santosh Nagraj and Dr. Carl Eckberg for being a part of the culmination of my thesis and for showing confidence in me.

I would also like to extend my acknowledgement towards my friends especially Shishir Srivastava for his immense support and help during this thesis. I would also like to thanks my friends Mohammed Muslehuddin, Kashyap Kambhatla and others for helping me in building my concepts and clearing my doubts.

On the other note, many people have been a part of my graduation and I am highly grateful to all of them.

Finally I would like to take this opportunity to thank my family for there unconditional love and support.

Above all I would like to thank GOD who has always supported me in my endeavors, without whom it would not have been possible to come this far.

Shireen.
CHAPTER 1

INTRODUCTION

The proliferation of multimedia on the World Wide Web and the emergence of broadband wireless networks have brought great interest in video communication. With the materialization of Web as a strong competitor for conventional distribution networks, a main challenge relates to the production of easily adaptable video content capable of optimally fitting into various evolving platforms. Network supported multimedia applications like in-home digital networks, video streaming over IP networks, surveillance systems, mobile video, wireless LAN video, multi-party video telephony/conferencing involve many different transmission capabilities. These applications are used to deliver content to a wide range of terminals and users surrounded by different environments and acting under totally different circumstances. The challenge now is to make information easily retrievable for a variety of systems.

Improvement in compression efficiency between MPEG-2 and MPEG-4 is not significant and new techniques are required to overcome this limitation and to enable a quick and easy access to large multimedia data repositories [1]. To reach this goal, information must be customized in accordance with the various network systems and the features of their devices.

Many uncertain parameters exist in the network such as speed, load and bandwidth. Consequently, requirements for bandwidth availability and quick and easy access to large multimedia databases will be more and more stringent. Therefore, meeting bandwidth requirements and maintaining acceptable image quality, simultaneously is a challenge. Continuous rate scalable applications can prove valuable in scenarios where the channel is unable to provide a constant bandwidth to the application. Rather than terminating the session, a decoder can adjust the data rate to use the limited resources, yet produce video of acceptable quality. Such decoders are particularly attractive because of their flexibility. Scalable Video Coding (SVC) [HHI] opens the door to some new video coding techniques with the following features [2]:

- Reduced bitrate
- Reduced spatial-temporal resolution
- Coding efficiency comparable to non-scalable video systems
- Possibility to employ hierarchical prediction structures for providing temporal scalability with several levels, for improving the coding efficiency, and for increasing the effectiveness of SNR and spatial scalable coding
- Inter-layer prediction of motion and residual for improving the coding efficiency of spatial scalable and coarse-grain SNR scalable coding
- Key picture concept for efficiently controlling the drift of SNR scalable coding with hierarchical prediction structures.

Scalable Video Coding addresses the issue of reliably delivering video to diverse systems over heterogeneous networks using available system resources, particularly in scenarios where the downstream system capabilities, resources, and network conditions are not known in advance. The goal is to provide scalability at bitstream level with good compression efficiency by allowing free combinations of scalable modes such as spatial, temporal and SNR/fidelity scalability. This new work is presently developed by the Joint Video Team (JVT) as an extension of AVC/H.264.

1.1 H.264/AVC Basic

H.264/MPEG-4 Part 10 also known as H.264/AVC (Advanced Video Coding) is the latest single layer coding standard jointly developed by ITU-T Video Coding Expert Group (VCEG) and ISO/IEC Motion Pictures Expert Group (MPEG). The joint group is known as JVT (Joint Video Team). H.264/AVC standard aims at achieving significant enhancement in coding efficiency and error robustness in comparison to previous video coding standards like MPEG-2, H.263 and MPEG-4 Part2 with a range of features supporting better quality and low bitrate for streaming video over fixed and wireless networks and over different transport protocols [3, 4].

Numerous papers and tutorials have been written about video coding theory in general and also about industry standard H.264/AVC. In this chapter we take a quick review of distinguishing features of the H.264/AVC standard. Reader is suggested to refer [5, 6] for comprehensive information about video coding techniques and standards. Detailed information about H.264/AVC can be found in [7, 8].
1.1.1 Introduction of H.264 AVC

Similar to previous standards (MPEG-1, MPEG-2 and MPEG-4) H.264/AVC standard does not explicitly define encoder-decoder pair specifications. Rather it specifies syntax of a valid encoded bitstream along with method to decode it [9]. With this defined the implementation details of encoder are completely left to the developers.

H.264/AVC uses the same basic functional elements as in previous standards [10] i.e. block transform to exploit spatial redundancy, motion compensated prediction to exploit temporal redundancy, quantization to control bitrate, entropy encoding to reduce statistical correlation. However important changes occur in details of each element. It introduces a new intra-picture prediction technique, new 4x4 integer transform, variable block sizes, deblocking filter, multiple reference frames, quarter pixel precision for motion compensation and improved lossless coding. In order to reduce complexity it introduces new multiplier free integer transform.

Multiplier operation for exact transform is combined with quantization scaling. To cope with degradation arising due to channel noise H.264/AVC adds parameter setting, flexible macro block ordering, switched slice, redundant slice methods to data partitioning for error resilience.

1.1.2 Structure of H.264/AVC

The H.264/AVC bitstream has been coded in two layers (shown in Figure 1.1) Network Abstraction Layer (NAL) and Video Coding Layer (VCL) [11, 12]. Video Coding Layer contains the actual coded video information. Purpose of Network Abstraction Layer is to abstract Video Coding Layer data such that it would be convenient to store on storage media or transmit it on variety of communication channels or networks.

1.1.2.1 Network Abstraction Layer

NAL formats the compressed video (VCL) data and provides additional non-VCL information such as parameter setting etc [13] in such a way that it can be conveniently coded as byte-stream or packet-based format. Format of NALU is shown in Figure 1.2.

All data related to video stream is encapsulated in NAL Units (referred as NALU). First byte of each NALU is a header byte and rest all is data. First bit is always zero. Next two bits indicate whether content of NALU is sequence or picture parameter set or a slice of
reference picture. Next five bits specify NALU type corresponding to payload being carried in NALU which may be VCL or non-VCL type. The picture and parameter sets play pivotal role during decoding. They define some parameters of data being encoded which are used for decoding. So during transmission, these two sets are sent frequently. If the bitstream has to be played from a random point, these parameters along with next IDR (Instantaneous Decoder Refresh) picture are used.

1.1.2.2 VIDEO CODING LAYER

The VCL design follows block based hybrid video coding approach. The basic source coding algorithm is, to exploit inter-picture and intra-picture redundancies in temporal, spatial domains and apply transforms and lossless coding to further exploit statistical redundancy.

There is no single functional block in VCL which gives dramatic improvement in coding gain but it is the cumulative effect of modifications done in implementation details of these blocks in H.264/AVC standard with reference to previous standards.
Figure 1.3 and Figure 1.4 shows the H.264/AVC Encoder and Decoder Structure respectively.

**Figure 1.3. H.264/AVC encoder structure.**

**Figure 1.4. H.264/AVC Decoder structure.**

### 1.1.3 Important Features

- The main features of H.264/AVC, which distinguish it from the previous video compression standards, are briefly discussed below [14, 15].

- This standard supports more flexibility in the selection of motion compensation block sizes and shapes than any previous standard, with a minimum luma motion compensation block size as small as 4x4 as compared to 8x8 block size in earlier standards. Block sizes such as 8x4, 4x8 and 4x4, are supported. The minimum luma motion compensation block size can be as small as 4x4.

- The Discrete Cosine Transform (DCT) is replaced by integer transform that is exact match inverse transform, thus avoiding drift during inverse transform.
• Two context adaptive coding schemes, CAVLC (context-adaptive variable-length coding) and CABAC (context-adaptive binary arithmetic coding), improve coding efficiency by adjusting the code tables according to the surrounding information.

• B-frame can be used as reference frame.

• Multiple reference frame motion compensation is allowed, which also improves the prediction accuracy. The restriction that only the immediate previous frame can be used as reference frame is thus removed.

• \( \frac{1}{4} \) pixel motion estimation which improves prediction accuracy.

• In-loop deblocking filtering removes the blocking artifacts caused by transform and quantization.

• Parameter sets are used between the encoder and decoder to achieve synchronization in terms of syntax.

• Flexible macroblock ordering (FMO) partitions a frame into different slice groups.

• Data partitioning groups a slice in up to three packets by their importance.

• SP/SI synchronization/switching frames reduce the penalty of switching between ongoing video bit streams by avoiding transmission of an I-frame.

• Encoder can send redundant representation of some regions of a frame to enhance robustness to data loss.

• Decoupling of referencing order from display order: allowing the encoder to choose the ordering of pictures for referencing and display purposes with a high degree of flexibility constrained only by a total memory capacity bound imposed to ensure decoding ability.

• In prior standards, there was a strict dependency between the ordering of pictures for motion compensation referencing purposes and the ordering of pictures for display purposes.

• The slices/data partitions are conceptually transferred by the network abstraction layer as NAL units, which are the basic transport units. Essential picture header information is assembled in parameters sets, more specifically the sequence parameter sets (SPSs) and picture parameter sets (PPSs).

• Hypothetical Reference Decoder (HRD) is a very important part in H.264, which represents a set of normative requirements on bit stream for the purpose of avoiding buffer (coded picture buffer - CPB) overflow and underflow [16]. HRD is conceptually connected to the output of an encoder and consists of a decoder buffer, a decoder and a display unit. HRD and CPB employ a leaky bucket algorithm.

• There are three Profiles in the standard. The Baseline profile supports I and P slices, and entropy coding with CAVLC. Also, it utilizes redundant slices and arbitrary slice ordering (ASO) for error resilient coding. Potential applications are video telephony, videoconferencing and wireless communications. The Main profile includes support for interlaced video, B slices, inter coding using weighted prediction and entropy
coding with CABAC. Well suited application areas are television broadcasting and video storage. The Extended profile does not support interlaced video or CABAC but includes SP/SI slices to enable efficient switching and data partitioning for improved error resilience. This profile may be particularly useful for streaming media applications.

1.1.4 Error Resiliency Schemes in H.264/AVC

Robustness to channel noise and data errors/loss and operation over wide variety of networks is facilitated by following features. Details can be found in [17].

- Semantics, Syntax, Error detection
- Flexible Macro block Ordering (FMO)
- Arbitrary Slice Ordering (ASO)
- Data Partitioning (DP)
- Parameter set structure
- NAL unit syntax structure
- Flexible slice size
- Redundant pictures
- Data Partitioning
- SP/SI synchronization/switching pictures

1.2 RESEARCH SCOPE

The continuing growth of multimedia applications leads to a greater expansion of video transmission over heterogeneous channels as well as iterative delivery platforms with specific content requirements. Conventional Video Coding Systems encode video content using a fix bit rate tailored to a specific application. As a consequence, conventional video coding does not fulfill the basic requirements of new flexible digital media applications. Contrasting this, Scalable Video Coding emerges as a new technology able to satisfy the underlying requirements. New applications on the Internet require large amount of bandwidth and have high requirements on the latency, jitter and loss experienced by viewers.

We implemented the new H.264/SVC scheme and its features. In the proposed prototype, the encoder separates a single video file into different layers. The decoder can process different kind of layers, a base layer that displays the minimally acceptable quality of the original file and one or more enhanced layers that improve the video quality during the
playback. The layering scheme not only reduces the network traffic but also provides additional flexibility. Decoding all the layers would enable viewing of the original video quality or the highest possible quality.

The H.264/SVC Standard has no error resiliency schemes for this codec. So we tried to implement this scheme Unequal Error Protection Scheme and Forward Error Correction Scheme to the original H.264/SVC codec. We tested the scheme for different scenarios and error conditions to obtain the desired performance charts.
CHAPTER 2

SCALABLE VIDEO CODING

In this chapter, we will discuss about the H.264 Scalable video coding and types of scalabilities.

2.1 INTRODUCTION

Advances in video coding techniques and standardization along with the rapid development and improvements of network infrastructures, storage capacity and computing powers are enabling an increasing number of video applications. Applications area, today, range from MMS, video telephony and video conferencing over mobile TV etc. For these applications, a variety of video transmission and storage systems may be employed.

So with the growing number of devices, there is a need for compressed video sequence to be “scalable” so that data can be selectively removed from the bit stream to yield a sequence with degraded characteristics, such as lower frame rate, spatial resolution or visual quality. This need for scalability has been acknowledged by the standardization bodies of ISO/IEC and ITU-T, which have tasked the JVT (Joint Video Team) with creating a scalable extension to H.264/AVC. This output of JVT is known as Scalable Extension of H.264/AVC or H.264/SVC (Scalable Video Coding). The SVC extension is build on H.264/MPEG 4 AVC and reuses most of its innovative components. This includes the Motion Compensation and Intra Prediction, the transform and entropy coding etc.

A video bit-stream is called scalable when parts of it can be removed in a way that the resulting sub-stream forms another valid bit-stream for a given decoder, which represents the source content with are reduced reconstruction quality compared to the original bit-stream. Bit-streams that do not provide this property are referred to as single-layer streams. SVC enables the transmission and decoding of partial bitstreams to provide video services with lower temporal or spatial resolution or reduced quality while retaining the reconstruction quality that is high relative to the rate of the partial bit stream. Hence, SVC provides functionalities such as graceful transmission environments as well as bit rate, format and
power adaptation. These functionalities provide enhancements to transmission and storage applications. SVC have achieved significant improvements in coding efficiency with an increased degree of supported scalability relative to the scalability profiles of prior video coding standards.

### 2.2 Basic Concepts for Extending H.264/AVC Towards H.264/SVC

The basic SVC principle is One Encoding and Multiple Decoding as shown in Figure 2.1. This implies that our encoder have to encode one single bitstream which contains all the details for Temporal, Spatial and SNR scalability. We generate different combination of bitstreams according to user’s requirements and send it to the user. Thereafter user obtains the desired bitstream and decodes it accordingly.

![Figure 2.1. Principle of encoding.](image)

The principle of decoding can be easily understood by the Figure 2.2.

The encoder structure depends on the scalability dimensions that are to be achieved. Each of the core encoder block basically consist of an AVC encoder which is extended by
Figure 2.2. Principle of decoding.

inter layer prediction and SNR scalability capabilities. The block diagram of the Scalable extension is shown in Figure 2.3.

Figure 2.3. Block diagram of SVC for combined scalability.

As in Advance Video Coding, the encoding of the input video is performed at the Macro block basis. As the codec is based on the layer approach to enable spatial scalability, the encoder provides a down sampling filter stages that generates the lower resolution signal for each spatial layer. Encoder algorithm (not mention here in this thesis) may select between inter and intra coding for block shaped regions of each picture.
The video sequence is temporally decomposed into texture and motion information. Motion information from the lower layer may be used for prediction of the higher layer. The application of this prediction is switchable on a macro block or block basis. In case of intra coding, a prediction from surrounding macro blocks or from co-located macro blocks of other layers is possible. These prediction techniques do not employ motion information and hence, are referred to as intra prediction techniques. Furthermore, residual data from lower layers can be employed for an efficient coding of the current layer [18].

The redundancy between different layers is exploited by additional interlayer prediction concepts that include prediction mechanisms for motion parameters as well as for texture data (intra and residual data). The residual signal resulting from intra or motion compensated inter prediction is transform coded using AVC features. Three kinds of prediction applied here are – Inter layer motion prediction, inter layer residual prediction and Inter layer intra prediction.

An important feature of the SVC design is that scalability is provided at a bit-stream level. Bit-streams for a reduced spatial and/or temporal resolution are simply obtained by discarding NAL units (or network packets) from a global SVC bit-stream that are not required for decoding the target resolution. NAL units of PR slices can additionally be truncated in order to further reduce the bit-rate and the associated reconstruction quality.

Thus, one of the main design goals was that SVC should represent a straightforward extension of H.264/AVC. As much as possible, components of H.264/AVC are re-used, and new tools are only be added for efficiently supporting the required types of scalability. As for any other video coding standard, coding efficiency has always to be seen in connection with complexity in the design process.

### 2.3 Scalability Dimension

The usual modes of scalability are temporal, spatial, and quality scalability. Spatial scalability and temporal scalability describe cases in which subsets of the bit stream represent the source content with a reduced picture size (spatial resolution) or frame rate (temporal resolution), respectively. With quality scalability, the sub-stream provides the same spatio–temporal resolution as the complete bit stream, but with a lower fidelity—where fidelity is often informally referred to as signal-to-noise ratio (SNR). Quality scalability is also
commonly referred to as fidelity or SNR scalability. More rarely required scalability modes are region-of-interest (ROI) and object-based scalability, in which the sub-streams typically represent spatially contiguous regions of the original picture area. The different types of scalability can also be combined, so that a multitude of representations with different spatio-temporal resolutions and bit rates can be supported within a single scalable bit stream [8].

### 2.3.1 Temporal Scalability

In contrast to older video coding standards as MPEG-2/4, the coding and display order of pictures is completely decoupled in H.264/MPEG4-AVC. Any picture can be marked as reference picture and used for motion-compensated prediction of following pictures independent of the corresponding slice coding types. These features allow the coding of picture sequences with arbitrary temporal dependencies. Temporal scalable bit-stream can be generated by using hierarchical prediction structures, as shown in Figure 2.4, without any changes to H.264/MPEG4-AVC. So-called key pictures are coded in regular intervals by using only previous key pictures as references [9].

![Figure 2.4. Hierarchical B picture concept in H.264/SVC.](image)

A bit stream provides temporal scalability when the set of corresponding access units can be partitioned into a temporal base layer and one or more temporal enhancement layers with the following property. Let the temporal layers be identified by a temporal layer
identifier, which starts from 0 for the base layer and is increased by 1 from one temporal layer to the next. Then for each natural number, the bit stream that is obtained by removing all access units of all temporal layers with a temporal layer identifier greater than forms another valid bit stream for the given decoder [9].

Actually all prior standards support temporal scalability to some degree. H.264/AVC provides significant flexibility in temporal scalability with its reference picture memory control. It allows coding of picture sequences with arbitrary temporal dependencies which are only restricted by decoded picture buffer (DPB) size. Therefore, to support temporal scalability no major changes in H.264/AVC are needed except signaling the temporal layers.

Temporal scalability with dyadic enhancement layers can be efficiently implemented with concept of hierarchical B or P pictures as shown in Figure 2.5(a). Enhancement layers are typically coded as B pictures with reference picture lists 0 and 1 corresponding to temporally preceding and succeeding picture respectively, with a temporal identifier less than that of the picture being predicted. Since the backward prediction is not necessarily coupled with B frames, structure in Figure 2.5(a) can also be realized using P frames. Each set of temporal layers \{T0 \ldots Tk\} can be decoded independent of frames corresponding to \(T > k\). Group of Pictures (GOP) corresponds to all the frames between two successive frames of temporal base layer (i.e. T0) including the second T0 frame.

To represent generalized non-dyadic case, hierarchical prediction structures for temporal scalability can be combined with ‘multiple reference picture’ concept in H.264/AVC, meaning that reference picture lists can be constructed by using more than one reference picture and they can also include frames with same temporal level as the one being predicted. Figure 2.5(b) shows the non-dyadic case with two independently decodable sub-sequences at 1/9th and 1/3rd of full frame rate. Figure 2.5(c) shows further case where it is possible to adjust encoder/decoder structural delay by restricting prediction from frames that follow the frame to be predicted in display order. Figure 2.5(a) and Figure 2.5(c) represent same temporal scalability but structural delay of 7 and 0 respectively. However low delay coding structures usually suffer from coding efficiency problems. In hierarchical prediction structures the reference frames should be coded before they can be used for prediction of other frames.
Coding efficiency can be improved by carefully choosing quantization parameters for different temporal layers. Typically the base layer is coded with highest fidelity (or lowest quantization parameter) and quantization parameter is incremented for each subsequent temporal level. Further improvement in selection of quantization parameters can be achieved by computationally expensive rate-distortion analysis [19]. A simpler and sufficiently robust approach has been discussed in [20]. Coding efficiency of B-frames can be improved by using a weighted sum of list 0 and list 1 predictions is used during motion search [21]. It has been verified in [8] that coding efficiency of hierarchical temporal prediction structures can be improved by increasing GOP size and thus the encoding/decoding delay; the maximum coding efficiency is achieved for GOP sizes between 8 and 32.

When higher coding delay can be tolerated, hierarchical temporal prediction structure not only provides temporal scalability but also improves coding efficiency.

2.3.2 Spatial Stability

For supporting spatial scalable coding, SVC follows the conventional approach of multiple-layer coding, which is also used in MPEG-2 Video / H.262, H.263, and MPEG-4 Visual. Each layer corresponds to a supported spatial resolution and is identified by a layer or
dependency identifier $D$. The layer identifier $D$ for the spatial base layer is equal to 0, and it is increased by 1 from one spatial layer to the next. In each layer, motion-compensated prediction and intra coding are employed as for single-layer coding. But in order to improve the coding efficiency in comparison to simulating different spatial resolutions, additional inter-layer prediction mechanisms are incorporated. Although the basic concept for supporting spatial scalable coding is similar to that in prior video standards, SVC contains new tools that simultaneously improve the coding efficiency and reduced the decoder complexity overhead in relation to single-layer coding. In order to limit the memory requirements and decoder complexity, SVC requires that the coding order in base and enhancement layer is identical. All representations with different spatial resolutions for a time instant form an access unit and have to be transmitted successively in increasing order of their layer identifiers $D$. But lower layer pictures do not need to be present in all access units, which make it possible to combine temporal and spatial scalability as illustrated in Figure 2.6.

![Figure 2.6. Multi-layer structure with additional inter-layer prediction for enabling spatial scalable coding.](image)

2.3.2.1 **Basic Building Block of Spatial Scalability**

Similar to MPEG-2 Video / H.262 and MPEG-4 Visual, SVC supports spatial scalable coding with arbitrary resolution ratios. The only restriction is that both the horizontal and vertical resolution must not decrease from a base to an enhancement layer (as shown in Figure 2.7). With the SVC design it is further possible that an enhancement layer picture represents only a selected rectangular area of the corresponding base layer picture, which is coded with a higher or identical spatial resolution. Or in the enhancement layer picture,
additional parts beyond the borders of the base layer picture are added. The base and enhancement layer cropping, which may also be combined, can even be modified on a picture basis. Furthermore, the SVC design also includes tools for spatial scalable coding of interlaced sources. For both extensions, the generalized spatial scalable coding with arbitrary resolution ratios and cropping as well as for the spatial scalable coding of interlaced sources; the three basic inter-layer prediction concepts are maintained. But especially the derivation process for motion parameters as well as the design of appropriate upsampling filters for residual and intra blocks needed to be generalized. For a detailed description of these extensions the reader is referred to [22] and [23].

It should be noted that in an extreme case of spatial scalable coding, both the base and enhancement layer have an identical spatial resolution and no cropping is applied. This case actually represents SNR scalable coding, which is also referred to as coarse-grain SNR scalable (CGS) coding (sec. 2.4). As a specific feature of this configuration, the deblocking of the base layer intra signal for inter-layer intra prediction is omitted, since the transform block boundaries in base and enhancement layer are aligned. It is however still possible to use a 4×4 transform in the base layer and an 8×8 transform in the enhancement layer, or vice versa.
2.3.2.2. Inter-Layer Motion Prediction

For spatial enhancement layers, the SVC design includes a new macroblock mode, which is referred to as BLSkip. In this mode only a residual signal, but no additional side information as intra prediction modes or motion parameters is transmitted. With conventional dyadic spatial scalability a macroblock in an enhancement layer corresponds to an 8x8 sub-macroblock in its base layer. When a macroblock is coded using the BLSkip mode and the corresponding 8x8 base layer block lies inside an intra-coded macroblock, the macroblock is predicted by inter-layer intra prediction as it will be explained in sec. 2.3.2.3. When, however, the base layer macroblock is inter-coded, the enhancement layer macroblock is inter-coded, too, and its macroblock partitioning together with the associates reference indices and motion vectors are derived from the co-located 8x8 block in the base layer. The macroblock segmentation is obtained by upsampling the partitioning of the co-located 8x8 block in the lower resolution layer.

When the base layer 8x8 block is not divided into smaller blocks, the enhancement layer macroblock is not partitioned. Otherwise, each axb submacroblock partition in the base layer block corresponds to a (2a)x(2b) macroblock partition in the enhancement layer macroblock as shown in Figure 2.8. For macroblocks or submacroblocks that are coded in direct mode, the partitioning usually depends on the derived motion vectors. But, identical decoding results are obtained when it is assumed that these blocks are always divided into 4x4 sub-macroblock partitions.

Figure 2.8. Spatial prediction of data.
For the obtained macroblock partitions, the same reference indices as for the corresponding sub-macroblock partitions of the 8x8 base layer block are used; and both components of the associated motion vectors are scaled by a factor of 2. In addition to this new macroblock type, the SVC concept includes the possibility to use a scaled motion vector of the lower resolution as motion vector predictor for the conventional motion compensated macroblock modes. A flag that is transmitted with each motion vector difference indicates whether the motion vector predictor is build by conventional spatial prediction or by the corresponding scaled base layer motion vector.

2.3.2.3 **INTER-LAYER RESIDUAL PREDICTION**

When employing the inter-layer motion prediction by using the BLSkip mode, the motion rate that has been transmitted for the co-located sub-macroblock in the base layer is virtually at least partly re-used in the enhancement layer. However, the bits that have been transmitted for coding the base layer prediction error represent useless side information for the enhancement layer. In order to provide the possibility to benefit from this information for the enhancement layer coding, interlayer residual prediction was added to the SVC design.

The inter-layer residual prediction can be employed for all inter-coded macroblocks regardless whether they are coded in the new BLSkip mode or with any of the conventional macroblock types. A flag is added to the macroblock syntax for spatial enhancement layers, which signals the usage of interlayer residual prediction. When this flag is true, the residual signal of the corresponding 8x8 base layer sub-macroblock is blockwise upsampled using a bi-linear filter and used as prediction for the residual signal of the enhancement layer macroblock, so that only the corresponding difference signal is coded in the enhancement layer. The upsampling of the base layer residual is done on a transform block basis in order to ensure that no filtering is applied across transform block boundaries, by which disturbing signal components could be generated.

2.3.2.4 **INTER-LAYER INTER PREDICTION**

When an enhancement layer macroblock is coded in BISkip mode and the co-located 8x8 sub-macroblock in its base layer is intra-coded, the prediction signal of the enhancement layer macroblock is obtained by inter-layer intra prediction, for which the corresponding
reconstructed intra signal of the base layer is upsampled. For upsampling the luma component, one-dimensional 6-tap FIR filters ([1, -5, 20, 20, -5,1]/32) are applied horizontally and vertically. The chroma components are upsampled by using a simple bi-linear filter. The filtering is always performed across submacroblock boundaries using the samples of neighboring intra blocks. When the neighboring blocks are not intra-coded, the required samples are generated by specific border extension algorithms. It is always avoided to reconstruct inter-coded macroblocks in the base layer in order to allow single-loop decoding, which will be explained in sec. 4.4.3. To prevent the generating of disturbing signal components, the H.264 / MPEG-4 AVC deblocking filter is applied to the reconstructed intra signal of the base layer before it is used for upsampling.

2.3.2.5 Complexity Consideration

The design of the SVC inter-layer prediction concepts was not only conducted from a coding efficiency point of view, but also by complexity considerations. The possibility of employing inter-layer intra prediction is restricted to selected enhancement layer macroblocks. The coding efficiency can generally be improved by allowing this prediction mode for all enhancement layer macroblocks as it was done in the initial design [24]. In [25] and [26] it was however shown that the decoder complexity can be significantly reduced by constraining the usage of inter-layer intra prediction. The general idea is to avoid the computationally complex operations of motion-compensated prediction and deblinking for all intercoded base layer macroblocks. This can be realized when the usage of inter-layer intra prediction is only allowed for enhancement layer macroblock, for which the co-located base layer signal is completely intra-coded. It is further required that the base layer and all intermediate layers are coded using constrained intra prediction, so that the intra macroblocks can be constructed without reconstructing any interceded macroblock.

With these restrictions, which are mandatory in SVC, each supported layer can be decoded with a single motion-compensation loop. Note that the complexity reduction is even more important for CGS coding than for e.g. dyadic spatial scalable coding. The decoder complexity overhead in comparison to single-layer coding for SVC is smaller than that for previous video coding standards, which all require multiple motion compensation loops at the decoder side.
The feature of single-loop decoding also reduces the memory requirement, since decoded samples of lower layers do not need to be stored in the decoded picture buffer for inter-layer prediction. This is also a reason why inter-layer prediction is only allowed inside an access unit. Additionally, it should be mentioned that a CGS or spatial enhancement layer NAL unit can be parsed independently of the corresponding base layer NAL units, which provides the possibility to further reduce the complexity of decoder implementations [27].

### 2.3.2.6 Coding Efficiency

The effectiveness of the inter-layer prediction concepts for spatial scalable coding is evaluated in comparison to single-layer coding as well as simulcast. The base layer was coded at a fixed bit-rate, for encoding the spatial enhancement layers, the bit-rate as well as the amount of enabled inter-layer prediction mechanisms was varied. Additional simulations were run with multiple-loop decoding. For these runs, the restriction for the inter-layer intra prediction in the current SVC design was removed. Only the first access unit was intra-coded and CABAC was used as entropy coding method. Simulations have been carried out for a GOP size of 16 pictures as well as for IPPPP coding. All encoders have been rate-distortion optimized following [28].

For each access unit, first the base layer picture is encoded, and given the corresponding coding parameters, the enhancement layer picture is coded [29]. The inter-layer prediction tools are considered as additional coding options for the enhancement layer pictures in the operational encoder control. The lower resolution sequences have been generated following the method in [29]. The simulation results for the sequence “City” are depicted in Figure 2.9. All inter-layer prediction (ILP) tools, intra (I), motion (M), and residual (R) prediction, improve the coding efficiency in comparison to simulcast.

However, the effectiveness of a tool strongly depends on the sequence characteristic. Multiple-loop decoding can further improve the coding efficiency. But the gain is often minor and comes along with a significant increase in decoder complexity. Particularly, the coding efficiency for multi-loop decoding with only inter-layer intra prediction enabled, which is comparable to the concepts of MPEG-2 Video / H.264, H.263, and MPEG-4 Visual, is usually worse than the coding efficiency for the SVC design.
Figure 2.9. Analysis of the efficiency of the inter-layer prediction concepts in SVC for different prediction structures. The rate-distortion point for base layer is plotted inside the diagrams, but it should be noted that it corresponds to a different spatial resolution.

Furthermore, it can be noted that the usage of hierarchical prediction structures does not only improve the overall coding efficiency, but also the effectiveness of the inter-layer prediction mechanisms.

2.3.3 SNR /Quality Scalability

SNR scalability can be considered as a special case of spatial scalability for which the picture sizes of base and enhancement layer are identical. As already mentioned in sec. 2.3.2, this case is supported by the general concept for spatial scalable coding and it is also referred
to as coarse-grain SNR scalable (CGS) coding. The same inter-layer prediction mechanisms as for spatial scalable coding are employed, but without the corresponding upsampling operations. However, with this multi-layer concept for SNR scalable coding only a few selected bit-rates can be supported in a scalable bit-stream. In general, the number of rate points is identical to the number of layers. A switching between different layers can only be done at defined points in the bit-streams. Furthermore, as it will be demonstrated in sec. 2.3.2 the multi-layer concept for SNR scalable coding becomes inefficient, when the relative rate difference between successive layers is small. Although CGS coding is simple and characterized by a low decoder complexity overhead in relation to single-layer coding, it does not provide enough flexibility for all applications.

Especially for increasing the flexibility for bit-stream adaptations and the error robustness, but also for improving the coding efficiency for bit-streams that have to provide a variety of bit-rates, an additional approach for fine-granular SNR scalable (FGS) coding is included in the SVC design. FGS coding is based on so-called progressive refinement (PR) slices. The SVC syntax allows that up to 3 layers of PR slices are coded on top of a base layer picture. These SNR refinement layers are identified by a quality layer identifier Q, which is equal to 0 for the base layer pictures and increases by 1 for each SNR refinement layer. The NAL units containing PR slices have the unique property that they can be truncated at any byte-aligned position or arbitrarily discarded from an SVC bit-stream without influencing its decodability. When PR slices are employed in connection with a suitable encoder configuration, any truncation or discarding of the corresponding NAL units reduces the reconstruction quality of the bit-stream in a fine-granular way. FGS coding is thus especially suitable for streaming applications in which the video bit-rate has to be frequently adapted to the channel conditions, or in combination with unequal error protection for environments that are characterized by significant packet loss rate.

2.3.3.1 PROGRESSIVE REFINEMENT SLICE

Progressive refinement slices have been designed for efficiently representing SNR refinements and allowing the truncation of the corresponding NAL units. Each PR slice basically corresponds to a bisection of the quantization step size or an increase by 6 of the quantization parameter QP. The quantization parameter QP for a macroblock can only be
freely chosen, when no non-zero transform coefficient was transmitted in the co-located macroblock of any subordinate layer. Otherwise, the quantization parameter is derived from the QP of the corresponding lower layer macroblock. At the decoder side, the transform coefficient levels of the base and all enhancement layers are scaled by the scaling factor which is determined by the quantization parameters, these scaled values are added up, and a single inverse transform is applied to obtain the reconstructed residual signal.

Although the quantization step size is usually halved from one layer to the next, the SVC concept for SNR scalable coding substantially differs from bit-plane coding as it is applied in MPEG-4 Visual or most of the 3-d wavelet codecs. The SVC design generally allows a greater freedom in encoder decision for determining transform coefficient levels. It was mainly influenced by the observation that the possibility to arbitrarily adjust the transform coefficient levels of each SNR refinement layer has a much greater impact on the coding efficiency than an efficient bit-plane coding for given transform coefficients. As an example, with bit-plane coding, a transform coefficient level unequal to 0 has to be transmitted for any transform coefficient that lies outside the quantization interval around zero; otherwise, a very large reconstruction error in following refinement layers would be obtained. With the SVC concept of re-quantization, however, a transform coefficient level equal to 0 could be transmitted in the base layer, because this would minimized the rate-distortion cost in the base layer and thus maximizing its coding efficiency. Then, in the next enhancement layer, a transform coefficient level greater than 1 could be transmitted, since this would maximize the coding efficiency of this enhancement layer. The only restriction in the current SVC draft is that for transform coefficients, for which a non-zero level was transmitted in one of the subordinate layers, only the level values -1, 0, and 1 are allowed.

With PR slices, the SNR refinement signal is represented in a coarse-to-fine representation. The transform coefficient levels are usually not transmitted macroblock by macroblock. Instead they are processed in several scans, and in each scan only a few transform coefficient levels for each transform block are coded. The coefficient scanning can be influenced by syntax elements of the slice header, and thus it is possible to adjust the trade-off between decoder complexity, which increases with the number of scans, and the quality of the coarse to fine representation, which determines the coding efficiency for
truncated FGS layers. For more details on this so called cyclic block scanning in SVC the reader is referred to [30].

In SVC, it is also possible to include a refinement of motion parameters in PR slices. This is especially useful, when SNR scalability has to be provided for a large bit-rate interval. With a single motion vector field the trade-off between motion and texture rate can only be optimized for a single bit-rate or a small interval. Similarly to CGS coding, the possibility to refine the motion vector field of the base layer for the enhancement layer coding allows to increase the coding efficiency for larger rate intervals. In order to limit the decoder complexity, the motion refinement in PR slices is not allowed for key pictures, here only the concept of leaky prediction can be employed.

More details about the refinement of motion information in PR slices are given in [31]. It should be further noted that it is not only possible to truncate NAL units that contain PR slices, but also to distribute the data of a PR slice to several NAL units. Thus, for example only the first part of a PR slice is employed as a reference for inter-layer prediction of the next spatial layer. But the entire PR slice can be used for decoding the lower layer resolution. The PR NAL units that are not employed for interlayer prediction are also called discardable sub-streams.

### 2.3.3.2 Coarse Grain SNR Scalability (CGS)

Coarse-grain SNR scalable coding is achieved using the concepts for spatial scalability. The only difference is that for CGS the upsampling operations of the inter-layer prediction mechanisms are omitted. Note that the restricted interlayer prediction that enables single-loop decoding is even more important for CGS than for spatial scalable coding.

### 2.3.3.3 Fine Grain SNR Scalability (FGS)

In order to support fine-granular SNR scalability, so-called progressive refinement (PR) slices have been introduced. Each PR slice represents a refinement of the residual signal that corresponds to a bisection of the quantization step size (QP increase of 6). These signals are represented in a way that only a single inverse transform has to be performed for each transform block at the decoder side. The ordering of transform coefficient levels in PR slices
allows the corresponding PR NAL units to be truncated at any arbitrary byte-aligned point, so that the quality of the SNR base layer can be refined in a fine-granular way. Figure 2.10 shows general concepts of Fine Granular Scalability in terms of layers.

![Figure 2.10. Fine granular scalability.](image)

The main reason for the low performance of the FGS in MPEG-4 is that the motion compensated prediction (MCP) is always done in the SNR base layer. In the SVC design, the highest quality reference available is employed for the MCP of non-key pictures as depicted in Figure 2.10. Note that this difference significantly improves the coding efficiency without increasing the complexity when hierarchical prediction structures are used. The MCP for key pictures is done by only using the base layer representation of the reference pictures. Thus, the key pictures serve as resynchronization points, and the drift between encoder and decoder reconstruction is efficiently limited. In order to improve the FGS coding efficiency, especially for low-delay IPPP coding, leaky prediction concepts for the motion-compensated prediction of key pictures have been additionally incorporated in the SVC design. In a method for further improving the FGS coding efficiency by allowing the coding of motion parameter refinements as part of the PR slices has been proposed. Performance comparison of different SNR scalable coding is shown in Figure 2.11.

### 2.3.3.4 Bit-Stream Extraction

For extracting a sub-stream with a specific average bit-rate from a given SNR scalable bit-stream usually a huge number of possibilities exist. The same average bit-rate can be adjusted by truncating or discarding different SNR refinement packets. However, the coding efficiency that corresponds to the bit-rate is dependent on the extraction method. With a very simple method that is widely used in experiments, all refinement packets are truncated by the same percentage. In a more sophisticated method, a priority identifier is assigned to each packet by an encoder. During the bitstream extraction, first all packets with the lowest
Figure 2.11. Performance of SNR scalable coding.

Priority value are truncated or discarded, and when the target bit-rate is not reached the packets of the next priority values or truncated or discarded, etc. The priority identifiers can either be fixed by the encoder based on the employed coder structure or determined by a rate-distortion analysis. The SVC syntax provides different means to include such priority information in the bitstream. For more detailed information about the concept of optimized bit-stream extraction, which is also referred to as quality layers, the reader is referred to [32].

2.3.3.5 Coding Efficiency

In a first experiment the different concepts for controlling the drift are evaluated for both hierarchical B pictures with a GOP size of 16 pictures and IPPPP coding. With exception of the 2-loop control, all configurations could be realized with an SVC compliant coder. The results for the sequence “City” and “Crew” are summarized in Figure. When the motion compensation loop is only closed in the base layer (BL-only control) as in MPEG-4 FGS, no drift occurs, but the enhancement layer coding efficiency is very low, especially for sequences like “City” for which the motion-compensated prediction works very well. By closing the loop only at the enhancement layer (EL-only control) as it is done in the SNR scalable mode of MPEG-2 Video / H.262, a high enhancement layer coding efficiency can
be achieved. But any modification to the enhancement layer sub-stream results in a serious drift, and the reconstructed video quickly becomes unusable, especially for IPPPP coding structures.

A similar behavior can also be observed for the 2-loop control, but here the reconstruction quality stabilizes for low rates at the base layer level. For the sequence “Crew” these impacts are less obvious, since a substantial part of the macroblock is intra-coded and the differences only apply for inter coding. With the SVC key picture concept (adapt. BL/EL control), in which the pictures of the coarsest temporal level are coded as key pictures, a reasonable coding efficiency for the entire supported rate interval can be achieved in connection with hierarchical prediction structures. For IPPPP coding, the best coding efficiency over the entire rate range is obtained with the leaky prediction concept. For hierarchical B pictures, however, the additional gain relative to the simply key picture concept is minor (shown in Figure 2.12).

### 2.3.4 Combined Scalability

The general concept for combining spatial, SNR, and temporal scalability is illustrated in Figure 2.3 (p.11) which shows an example encoder structure with two spatial layers. The SVC coding structure is organized in layers. A layer usually represents a specific spatial resolution. In an extreme case it is also possible that the spatial resolution for two layers is identical (CGS). Layers are identified by a layer identifier D. The spatial resolution must not decrease from one layer to the next. For each layer, the basic concepts of motion-compensated prediction and intra prediction are employed as in single-layer coding, the redundancy between layers is exploited by additional interlayer prediction concepts. The concepts for fine-grain SNR scalability as described in sec. 2.3.3 are inserted within a layer. That means that progressive refinement slices can be coded in each layer, usually in order to refine the reconstruction quality of the specific spatial resolution. The SNR refinement levels inside each layer are identified by a quality level identifier Q. When however the spatial base layer contains different SNR representation, it needs to be signaled which of these is employed for inter-layer prediction. There is a tradeoff between enhancement layer and coding efficiency as shown in Figure 2.12.
The 1 byte header in H.264/AVC is extended by additional 3 bytes for SVC NAL unit types (shown in Figure 2.13). The extended header includes identifiers D (for spatial), T (for temporal) and Q (for quality) as well as additional information to assist easy bitstream manipulations. One such additional syntax is priority identifier P signaling importance of a NAL unit. Also, in order to attach SVC relayed information to non-SVC NAL unit, prefix-NAL units are introduced. SVC also specifies additional Supplementary Enhancement Information (SEI) messages, which contain information like spatial resolution or bitrate of layers included in coded scalable bitstream that can assist in bitstream adaptations.

The first 1 byte represents the NAL Unit header of the H.264/AVC bitstream which is used when only base layer is decoded.
Figure 2.13. H.264/SVC NAL unit structure.

### 2.4.1 NAL Unit Header

The 3 byte NAL unit extension for SVC is divided into different segments [33]:

- **r (reserved_one_bit)** - This value should be specified as 0. The value of reserved_one_bit may be specified by future extension of this Recommendation | International Standard. Decoders shall ignore the value of reserved_one_bit.
- **i (idr_flag)** - Its value specifies that the current access unit is an IDR access unit.
- **p (priority_id)** specifies a priority identifier for the NAL unit.
- **n (no_inter_layer_pred_flag)** specifies, whether inter layer prediction may be used for the decoding of the coded slices. When its value is 1 that implies no inter layer prediction is there.
- **d (dependency_id)** specifies a dependency identifier for the NAL unit. dependency_id shall be equal to 0 in VCL prefix NAL units.
- **q (quality_id)** specifies a quality identifier for the NAL unit. quality_id shall be equal to 0 in VCL prefix NAL units.
- **t (temporal_id)** specifies a temporal identifier for the NAL unit. The value of temporal_id shall be the same for all prefix and coded slice in scalable extension NAL units of an access unit.
- **u (use_ref_base_pic_flag)** equal to 1 specifies that reference base pictures are used as reference pictures for the inter prediction process. use_ref_base_pic_flag equal to 0 specifies that decoded pictures are used as reference pictures during the inter prediction process. The values of use_ref_base_pic_flag shall be the same for all NAL units of a dependency representation.
- **x (discardable_flag)** equal to 1 specifies that the current NAL unit is not used for decoding NAL units of the current access unit and all subsequent access units that have a greater value of dependency_id than the current NAL unit. discardable_flag equal to 0 specifies that the current NAL unit may be used for decoding NAL units of the current access unit and all subsequent access units that have a greater value of dependency_id than the current NAL unit.
- **o(output_flag)** affects the decoded picture output and removal processes
• **r2** (reserved three 2bits) shall be equal to 3. The values of reserved_three_2bits may be specified by future extension of this Recommendation | International Standard. Decoders shall ignore the value of reserved_three_2bits.

### 2.4.2 Payload Structure

In H.264/AVC and SVC, all coded bits for representing a video signal are encapsulated in NAL Units.

H.264/AVC specifies three basic payload structures:

1. Single NAL Units
2. Aggregation Packet Units
3. Fragment Units
4. In addition to these three, SVC defines three new NAL types specifically for H.264/SVC[35]. Those are-
5. Coded Slice in Scalable extension NAL units
6. Prefix NAL units
7. Subset Sequence parameter set NAL units

### 2.5 Backward Compatibility for SVC

It is desirable in SVC scheme that a so called base layer be compatible with non Scalable video coding standards like AVC. It is also desired that additional scalable layers should additional scalable layers should be carried out in such a way that non-scalable video decoders, which have no knowledge of scalability, will ignore all scalable layers and only decode the base layer [34].

For these coded data that follow h.264/AVC and to ensure compatibility with existing H.264/AVC decoder, another new type of NAL (type 20) is used. This NAL carry the header information [35].

The base layer by design is compatible to H.264/AVC. During transmission, the associated prefix NAL units, which are introduced by SVC and when present are ignored by H.264/AVC decoders, may be encapsulated within the same RTP packet as the H.264/AVC VCL NAL units, or in a different RTP packet stream (when Multi session transmission mode is used) [36].

When using Multi session transmission mode-When a H.264/AVC compatible subset of the SVC base layer is transmitted in its own session in multi session transmission mode,
the packetization of RFC3984 must be used, such that RFC 3984 of receivers can be part of multi transmission mode and receive only this session [37].

When using Single session transmission mode-When an H.264/AVC compatible subset of SVC base layer is transmitted using single session transmission, the packetization of RFC 3984 must be used, thus ensuring compatibility with RFC 3984 receivers [35].

2.6 Error Concealment in SVC

There are four error concealment algorithms (2 intra layer methods and 2 inter layer methods) for cases when the base layer has no loss and when base layer has loss [38]. The error concealment algorithms are:

1. Frame copy (FC)
2. Temporal direct motion vector generation (TD)
3. Motion and residual up sampling (BLSkip)
4. Reconstruction base layer up sampling (RU).

2.6.1 Frame Copy (FC)

This is an intra layer error concealment method. In the “frame copy” (FC) algorithm, each pixel value of the concealed frame is copied from the corresponding pixel of the first frame in Reference Picture List. This algorithm can be invoked for both base layer and enhancement layer. The base layer FC will only be done when it needs to be rendered. For the two layer spatial scalability case, a lost enhancement layer frame will be concealed using the first frame in the reference.

2.6.2 Temporal Direct (TD)

This is an intra layer error concealment method. In the “temporal direct motion vector generation” (TD) algorithm, the MVs and reference indices of each sub blocks of the missing frame are calculated as if they were coded using the “temporal direct mode”. This algorithm can be invoked for both base layer and spatial enhancement layer. For enhancement layer, if TD is used, each MB in the lost high resolution frame will use TD mode to get and scale the MVs from the collocated MB at the enhancement layer. So the error concealment process does not need base layer information.
2.6.3 Motion and Residual Upsampling (BLSKIP)

This is an inter layer error concealment method. In the “motion and residual upsampling” (BLSkip) algorithm, SVC tools are used and the BLSkip mode is set in the enhancement layer. Residual upsampling is also used to upsample the residual of the base layer for enhancement layer. However, the motion compensation is done at the enhancement layer using the upsampled motion fields. This algorithm can directly be used for the enhancement layer if there is no packet loss in the base layer. If base layer is also lost, it needs to generate the MVs using TD method before BLSkip can use motion field upsampling [39].

2.6.4 Reconstruction Base Layer Upsampling (RU)

This is an inter layer error concealment method. In the “reconstruction base layer upsampling” (RU) algorithm, the base layer picture is reconstructed and upsampled using the H.264/AVC 6-tap filter for the lost enhancement layer picture. If base layer packet is also lost, the FC method is used for the enhancement layer, rather than using an upsampled concealed base layer picture. For packet loss of key frames of the base layer, frame copy error concealment is used. Memory Management Control Operation (MMCO) and Ref Reference Picture List Reordering (RPLR) commands are created during the error concealment process, to emulate the reference picture and non-reference picture patterns at the encoder [40]. MMCO commands will be generated to delete all the non-key pictures of the previous GOP and the key picture before the previous key picture (if it exists). RPLR commands will be used to guarantee the lost key picture refers to the previous key picture. This algorithm only works properly if regular GOP coding patterns were used, e.g., adaptive GOP size (AGS) cannot be detected at the decoder and will not be concealed properly [41].

Comparison of the results of the above described error concealment methods are shown in Figure 2.14 and Figure 2.15.

2.7 EXTENDED SPATIAL SCALABILITY

Extended Spatial Scalability enables a generalized relation between successive spatial layers. A picture of a lower spatial layer may represent a cropped area of the higher resolution picture and the relation between successive spatial layers does not need to be
Figure 2.14. Comparison BLSkip, RU, TD, FC and no error cases with “Foreman” sequence for different base layer packet loss (p) and enhancement layer packet loss.
Figure 2.15. Subjective quality comparison of concealed pictures for “Foreman”, with a base layer packet loss rate of 3% and an enhancement layer packet loss rate of 3%, QP=28. (a) BLSkip; (b) RU; (c) TD; (d) FC.
dyadic. Geometrical parameters defining the cropping window and the down-sampling ratio can either be defined at the sequence level, or evolve at the picture level.

Extended Spatial Scalability includes two tools:

- Cropping
- Generic up-sampling (any horizontal and vertical inter-layer size ratios)

Let’s consider two successive spatial layers, a low layer (base layer) and a high layer (enhancement layer), linked by the following geometrical relations (see Figure 2.16). Width and height of enhancement layer pictures are defined respectively as $w_{\text{enh}}$ and $h_{\text{enh}}$. Base layer pictures dimensions are defined as $w_{\text{base}}$ and $h_{\text{base}}$. Base layer pictures are sub-sampled versions of pictures of dimensions $w_{\text{extract}}$ and $h_{\text{extract}}$, partially or totally inside the enhancement layer pictures, positioned at coordinates $(x_{\text{orig}}, y_{\text{orig}})$ in the enhancement layer pictures coordinates system. In Figure 2.16, the enhancement and base layer pictures are divided in macroblocks. $w_{\text{extract}} / w_{\text{base}}$ and $h_{\text{extract}} / h_{\text{base}}$ correspond actually to the upsampling factors between base layer pictures and extraction pictures in enhancement layer. The parameters $(x_{\text{orig}}, y_{\text{orig}}, w_{\text{extract}}, h_{\text{extract}})$ completely define the geometrical relations between high and low layer pictures. In a standard dyadic scalable configuration (extended_spatial_scalability = 0), these parameters are equal to $(0, 0, 2*w_{\text{base}}, 2*h_{\text{base}})$.

![Figure 2.16. Relations between enhancement layer and base layer.](image)

Figure 2.16 shows result for cropping window. Figure 2.17 (a) shows the original Forman qcif sequence while Figure 2.17 (b) shows cropping window output of the sequence with cropping height of 144 and cropping width of 176.
2.8 SVC PROFILE

SVC amendment supports following profiles for scalable video coding [42]:

- Scalable Baseline Profile: Mainly targeted for mobile broadcast, conversational and surveillance applications that require low decoding complexity.
- Scalable High Profile: Designed for broadcast, storage and streaming applications.
- Scalable High Intra Profile: Mainly intended for professional applications.

2.9 APPLICATIONS

There is multitude of application scenarios where such scalable video stream can be used [8]: For instance, a video server serving variety of end user devices with different display capabilities with the same source content over variety of network connections with different bandwidths. With properly configured encoder, the bitstream is encoded only once with highest desired resolution and bitrate and then extracted and substreams are formed as per need to serve various clients.

Another interesting scenario is video transmission over a channel with unpredictable throughput variations and/or relatively high packet loss rates. Since the scalable video stream usually contains different parts with different importance in terms of quality, they can be coded with unequal error protection schemes such that stronger protection is provided to more important information i.e. base layer information and relatively weaker protection to
subsequent enhancement layers. Such scheme can help in graceful degradation up to certain degree of channel error/loss rates. They can be assisted with the Media Aware Network Elements (MANE) by removing unwanted parts from the bitstream before forwarding it to terminals as per their feedback [43].

One different kind of application is for video archiving in video recorders, home networking or for video surveillance applications. In such scenario, the high quality parts of the video stream can be deleted after some expiration time to save storage space assuming that as time passes the probability that they will be viewed again and again lowers down.

For web browsing of video library, scalable video coding can generate a low resolution preview without decoding a full resolution picture. In general, Scalable Video Coding addresses the issue of reliable delivery of video to diverse systems over diversified network connections using available system resources, especially in scenarios where the end system capabilities, resources and network conditions are not known beforehand.
CHAPTER 3

LITERATURE REVIEW OF DIFFERENT PROPOSED ERROR RESILIENCY SCHEMES

Practically there is no Error Resiliency scheme in H.264/SVC. Currently, only the error concealment tools are supported by JSVM (for JSVM software, see Appendix A). However, the reconstruction quality is poor. A lot of research is taking place in this area these days. Most of the error resiliency methods which are considering the FGS (Fine Granular Scalability) bitstream as the reference for their experiments. Here we will discuss some of the important error resiliency schemes which are proposed and of our area of interest.

3.1 ERROR RESILIENT UNEQUAL ERROR PROTECTION OF FINE GRANULARITY SCALABLE VIDEO BITSTREAM

Hua Cai, Bing Zeng, Guobin Shen, Zixiang Xiong and Shipeng Li of the Hong Kong University of Science and Technology in this paper deals with the optimal packet loss protection issue for streaming the fine granularity scalable (FGS) video bitstreams over IP networks. Unlike many other existing protection schemes, they developed an error-resilient unequal protection (ER-UEP) method that adds redundant information optimally for loss protection and, at the same time, cancels completely the dependency among bitstream after loss recovery. In their ER-UEP method, the FGS enhancement-layer bitstream is first packetized into a group of independent data packets, while each packet can be truncated to represent the original video signal at any fidelity (i.e., scalability). Parity packets are then created with intrinsic UEP capabilities that can easily adapt to the current channel conditions.

Unlike conventional UEP schemes that suffer from bitstream contamination due to the dependency among packets, their method guarantees the successful decoding of all received bits, thus leading to a better error resilience as well as higher robustness (under varying and/or unclean channel conditions). See Appendix B for loss simulators.

The complete ER-UEP framework consists of four STEPS, namely Data packets generation, Parity packets generation, Data and parity rate calculation, and Rate shaping.
In this paper, they assumed that the base layer of an FGS video bitstream is always received correctly and focus on the error protection for the enhancement layer. Figure 3.1 show the principle diagram of our ER-UEP method.

![Figure 3.1. The ER-UEP scheme.](image)

It is important to notice that bits in each data packet are processed on the symbol-by-symbol basis. For example, the kth data packet is composed of symbols sk(0), sk(1), ..., where sk(i) denotes the ith data symbol of the kth data packet. Next, all K data symbols at the same position within their individual data packets are grouped to form a codeword. In this way, all bits in K data packets are organized into a list of codewords, and the ith codeword consists of data symbols s1(i), s2(i), ..., sK(i). Channel coding is finally applied to generate T parity symbols for each codeword via the Reed-Solomon code RS(K + T,K). The generated T parity symbols for the same codeword are then filled into T parity packets, one symbol for each packet. In the end, K data packets and T parity packets together form an FEC block.

The data here is then packetized. Normally, the enhancement-layer bits of each video frame are sequentially ordered on a bit-plane by bit-plane and MB by MB basis, i.e., from the most significant bit-plane of all MBs to the least significant bit-plane of all MBs. For such normal ordered bitstream, the bits after truncation are put into packets continuously with the constraint of maximum packet length, as depicted by the normal packetization in Figure 3.2-(a).

This packetization strategy has two drawbacks: (1) it is difficult to choose the most efficient bits for transmission (2) it introduces strong dependency among packets as bits from
different bitplanes of the same MB, which are causally dependent, are often packetized into different packets.

To overcome the drawbacks of normal packetization, an RD optimal packetization strategy for FGS enhancement-layer bits was developed in [44] and used here. It first performs a R-D optimal bit allocation across bit-planes and MBs. It then packetizes those selected bits into packets by grouping bits from the same MBs into one packet, and thus packet dependency is completely eliminated. Figure 3.2(b) shows one scenario of performing this optimal packetization. Clearly, any lost packet will not contaminate the decoding of other received packets. Note that each packet is still fine scalable due to the MB by MB and bit-plane by bit-plane sequential scan ordering, as depicted by the packetizing order in Figure 3.2(b).

3.2 Robust Transcoding for Scalable Video Coding

J. C. Chiang, and C. F. Lu of National Chung Chen University China- Taiwan In this paper, we propose an unequal error protection (UEP) technique to provide the scalable video bitstream generated by JSVM platform with error resilience ability. The proposed method guarantees successful decoding of all received bits, thus leads to strong error resilience at various channel bandwidth and high robustness for streaming scalable video bitstream over a noisy link (results are shown in Figure 3.3).

The situation considered here is that the base layer and enhancement layer will be transmitted through the same channel due to some specified applications. Thus, the base
layer maybe suffered from channel noise. The goal of this work is to provide a robust decoding for any received bit stream under various channel conditions.

The proposed method guarantees successful decoding of all received bits, thus leads to strong error resilience at various channel bandwidth and high robustness for streaming scalable video bit stream over a noisy link.

This paper is an extension towards Error Resilient Unequal Error Protection for FGS bit stream as discussed in Section 3.1 and it added some new protection schemes (shown in Figure 3.4):

Figure 3.3. (a) Comparative evaluation of the proposed scheme at different packet loss ratios (b) Comparative evaluation on channels with prediction errors. The enhancement-layer rate is set to 768 kbps.
Figure 3.4. Proposed scalable packet unit.

1. Consider that the Base Layer is no error free
2. Not only the Quality Scalability but also Combined Scalability (Spatial and Quality) are introduced.

Figure 3.5 shows the proposed error resilient protection scheme. The left part of Figure 3.5 represents the scalable bitstream generated by JSVM. After packetization, each packet contains one frame and scalability is ensured within this packet. The scalability index D, T, and Q denotes the index of spatial layer, temporal layer and quality layer, respectively. For example, the base layer will be specified as (0, 0, 0). To reduce the computation complexity during the minimization process for the overall expected distortion, a discard and protect [45] scheme is adopted here. The original data bits inside a packet will be divided into three groups, namely protected part, unprotected part and discarded part, according to channel bandwidth and packet loss rate, as shown in Figure 3.6, where different colors denote different layers. The idea here is to sacrifice the less important data and reserve room for the protection of the most important data.

Figure 3.6 shows the performance comparisons for 3-part scheme and 4-part scheme in SNR scalability.
Youssef Charfi and Raouf Hamzaoui of University of Konstanz proposed a system for packet loss protection of quality scalable video coders. The compressed bitstream is composed of a non-scalable base layer and an embedded enhancement layer as in MPEG4-Fine Granularity Scalability (FGS) and the emerging H26L-FGS. But the system can also be used with embedded video bitstreams like the 3DSPIHT bitstream. Our system allocates a
given transmission bitrate between the base layer bitstream and the enhancement layer bitstream in an efficient way using feedback. The base layer is protected with equal packet loss protection. For the enhancement layer, unequal packet loss protection is used to provide graceful degradation of video quality in the presence of packet erasure. Preliminary results for a hypothetical operational distortion-rate function and the 3D-SPIHT video coder show promising performance of the proposed system.

In this paper, they proposed a new system that provides packet loss protection to both the base layer and the enhancement layer. The base layer is non-scalable and therefore any loss in the base layer makes the enhancement layer useless and results in a very poor reconstructed video quality. In the proposed system, the base layer bitstream is protected using a technique similar to the one given in [46]. The base layer (BL) bitstream is protected using Reed Solomon (RS) codes (see Appendix C) and sent followed by the protection packets until the receiver acknowledges that the BL is correctly recovered. The remaining bandwidth is spent for the protection of the enhancement layer bitstream using ULP to provide graceful degradation in video quality in the presence of packet loss.

As shown in Figure 3.7 in this packet loss protection system, the transmitter starts by sending the k BL packets to the receiver, followed by the RS protection packets until an acknowledgment (ACK) from the receiver arrives meaning that the k BL packets were correctly received, or the time deadline for the transmission of the GOF is reached (see Figure 3.6, p. 44). Due to the real-time requirements of video, we limit the transmission deadline to packets. We assume that k channel packets were sufficient for the correct reconstruction of the BL bitstream (see Figure 3.7). Once the ACK is received, the transmitter starts the transmission of the embedded enhancement layer (EL) using the remaining bandwidth for the considered GOF.

Table 3.1 shows SNR values of the average expected distortion of the proposed scheme and the MD scheme for a packet erasure channel with an erasure probability of 0.1 and at various transmission bitrates.
Figure 3.7. The packet loss protection scheme. Darker colors represent EL sublayers that are more important than the ones represented by lighter colors.

Table 3.1 SNR in dB of the Average Expected Distortion of the Proposed System and the MD System for a Packet Erasure Channel with an Erasure Probability of $P_e=0.1$ and Various Transmission Bitrates

<table>
<thead>
<tr>
<th>Rata (kbps)</th>
<th>48</th>
<th>64</th>
<th>96</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD</td>
<td>11.56</td>
<td>16.11</td>
<td>24.62</td>
<td>33.09</td>
</tr>
<tr>
<td>Proposed</td>
<td>13.49</td>
<td>17.56</td>
<td>25.93</td>
<td>34.42</td>
</tr>
</tbody>
</table>
CHAPTER 4

IMPLEMENTATION AND RESULTS OF SVC

In continuation to Chapter 2, this chapter presents a practical implementation of all the scalabilities, error concealment and other interesting features of SVC and their practical output. We are using JSVM (Joint Scalable video model) reference software for all the simulation and results. More details about the JSVM software can be found in Appendix A of this thesis. Microsoft Visual Studio 2003 or newer versions is also required to build the batch files and make changes in the configuration files and C++ programs as we are simulating it in the window environment. When we build the JSVM software then we will generate different configuration files and workstations. We can enable flags and values of different parameters according to our requirements and experimental conditions in configuration files as well as in the C++ programs. See Appendix D for codes and results.

4.1 Spatial Scalability

To achieve Spatial Scalability firstly we have to make some changes in the configuration files of the encoder and different layers we are taking into consideration. Firstly we will change maximum frame rate, number of frames to be encoded, GOP (Group of Picture) size, Intra period, and search mode (0 for block search and 4 for fast search). We will also change number of enhancement layers we want to enable and take into consideration. They can be enable by removing pound (#) sign in front of them, change other parameters in encoder.cfg except CgsSnrRefinement, MGSControl, MGSKKeyPicMotRef, parameters, as they are for SNR Scalability; we can change any other parameter according to our experimental conditions.

Then we will make some changes in the base layer configuration file named as Layer0.cfg in the software. Note that since we are taking spatial scalability into consideration so we have to enter different resolution of video sequences in the input file. Then we will enter width and height of the input sequence, according to our hierarchy and placing lowest resolution at the base layer. Frame rate in and frame rate out so also be mentioned.
Similar changes have to be made in the other enhancement layer configuration files named layer1.cfg etc.

After doing all the desired changes we have to invoke command prompt which can be called as: Go to “Start” (Windows XP, SP2) then “Run” and then type cmd and hit enter.

Then change the directory to bin folder which is located in the JSVM folder. Once you are in bin directory type all the JSVM commands you need to run. Firstly we will call encoder by command 
\[ H264AVCEncoderLibTestStatic.exe –pf encoder.cfg \] will generate the output in the form of .264 file. Then we will extract the bit stream by calling the Bit Streams Extractor command As 
\[ BitStreamExtractor.exe input.264 \] (i.e. input bit stream file name). The bit stream extractor also allows other option of extracting bitstream. Other options can also be specified when using the bitstream extractor to obtain the bitstream of our interest (More results on this is specified later).

Then we have to run the decoder to produce the output. Now we have to execute 
\[ H264AVCDecoderLibTestStatic.exe bitstream_file.264 output.yuv \] command on the command prompt. The video obtained here in in YUV format and can be played easily in any YUV player. We then calculate the PSNR values for the computational and observational purposes by calling PSNR Calculator.

**4.1.1 Discussion of Results**

Now in this section we discuss results achieved.

**4.1.2 Results for Spatial Scalability**

Some of the outputs of the Spatial Scalability are shown in Figure 4.1. Here we can see that in Figure 4.1 the comparative output of the snapshot of the videos is shown which gives the output of different layers and shows how different layers produces different resolutions. Figure 4.2 is the snapshot of the command prompt output which shows that the when we enable the Spatial Scalability mode only the Spatial Scalability identifier i.e. D is active while other two identifiers T for Temporal Scalability and Q for Quality Scalability shows no changes.
4.2 SNR/FIDELITY/QUALITY SCALABILITY

To achieve SNR Scalability first of all we have to make some changes in the configuration files of the encoder and different layers we are taking into consideration.

We will make some changes in maximum frame rate, number of frames to be encoded, GOP (Group of Picture) size, intra period, search mode (0 for block search and 4 for fast search), We will also change number of enhancement layers we want to enable and take into consideration. They can be enable by removing pound (#) sign in front of them.
Enable CgsSnrRefinement, EncoderKeyPicture, MGSControl, MSGKeyPicMotRef
parameters, as they are for SNR Scalability;

**CgsSnrRefinement** value of 1 implies that SNR enhancements are coded using the
EI,EP or ER slices while its value of 0 implies that the coding is done using the PR slices.

**EncoderKeyPicture** specifies whether the picture temporal level is coded as key
picture. Its value 0 implies that no key picture are coded as key picture. Its value of 1 says
that picture with MGS refinement is coded as key picture and value of 2 says that all pictures
at the temporal level are coded as key picture.

**MGSControl** specifies that what picture is used as reference for motion estimation
and motion compensation (determination of the residual signal to be coded) for MGS coding.

**MSGKeyPicMotRef** specifies whether motion refinement is allowed in the layer
refinement layers of MGS key picture.

Then we will make some changes in the base layer and enhancement layer
configuration files. At the layers the resolutions will be kept same irrespective of the number
of layers we are enabling. Then we will enter width and height of the input sequence. Frame
rate in and frame rate out should also be mentioned. Also enable MGS vector values

After doing all the desired changes we have to invoke command prompt which can be
called as- Go to “Start” (Windows XP, SP2), then “Run” and then type “cmd” and hit
“enter”.

Then change the directory to bin folder which is located in the JSVM folder. Once
you are in bin directory type all the JSVM commands you need to run. Firstly we will call
encoder by command **H264AVCEncoderLibTestStatic.exe –pf encoder.cfg** will generate the
output in the form of .264 file. Then we will extract the bit stream by calling the Bit Streams
Extractor command As **BitStreamExtractor.exe input.264** (i.e. input bit stream file name).The
bit stream extractor also allows other option of extracting bitstream. Other options can also
be specified when using the bitstream extractor to obtain the bitstream of our interest (more
results on this is specified later).

Then we have to run the decoder to produce the output. Now we have to execute
**H264AVCDecoderLibTestStatic.exe bitstream_file.264 output.yuv** command on the
command prompt. The video obtained here in YUV format and can be played easily in any
YUV player. We then calculate the PSNR values for the computational and observational purposes by calling PSNR Calculator.

4.2.1 Discussion of Results

Now in this section we discuss results achieved.

4.2.2 Results for SNR Scalability

Some of the outputs of the SNR Scalability are shown below. Here we can see that in Figure 4.3 the comparative output of the snapshot of the videos is shown which gives the output of different layers and shows how different layers produces different quality. Figure 4.4 is the snapshot of the command prompt output which shows that the when we enable the SNR Scalability mode only the SNR Scalability identifier i.e. Q is active while other two identifiers T for Temporal Scalability and D for Spatial Scalability shows no changes.

4.3 Temporal Scalability

To achieve Temporal Scalability first of all we have to make some changes in the configuration files of the encoder and different layers we are taking into consideration. We will make some changes in maximum frame rate, number of frames to be encoded, GOP (Group of Picture) size, intra period, search mode (0 for block search and 4 for fast search), We will also change number of enhancement layers we want to enable and take into consideration. They can be enable by removing pound (#) sign in front of them. Do not enable CgsSnrRefinement, EncoderKeyPicture, MGSControl, MGSKeyPicMotRef, parameters, as they are for SNR Scalability.

Again we will make some changes in the base layer and enhancement layer configuration files. At the layers the resolutions will be kept same irrespective of the number of layers we are enabling. Then we will enter width and height of the input sequence. Frame rate in and frame rate out should also be mentioned. Also enable MGS vector values. After doing all the desired changes we have to invoke command prompt which can be called as- Go to “Start” (Windows XP, SP2) then “Run” and then type “cmd” and hit “enter.”

Then change the directory to bin folder which is located in the JSVM folder. Once you are in bin directory type all the JSVM commands you need to run. Firstly we will call
encoder by command `H264AVCEncoderLibTestStatic.exe -pf encoder.cfg` will generate the output in the form of .264 file. Then we will extract the bit stream by calling the Bit Streams Extractor command as `BitStreamExtractor.exe input.264` (i.e. input bit stream file name). The bit stream extractor also allows other option of extracting bitstream. Other options can also
be specified when using the bitstream extractor to obtain the bitstream of our interest (More results on this is specified later).

Then we have to run the decoder to produce the output. Now we have to execute $H264AVCDecoderLibTestStatic.exe$ bitstream_file.264 output.yuv command on the command prompt. The video obtained here in YUV format and can be played easily in any YUV player. We then calculate the PSNR values for the computational and observational purposes by calling PSNR Calculator.

### 4.3.1 Discussion of Results

Now in this section we discuss results achieved.

### 4.3.2 Results for Spatial Scalability

Some of the outputs of the Temporal Scalability are shown below. Here we can see that in Figure 4.5 the comparative output of the snapshot of the videos is shown which gives the output of different layers and shows how at the same frame number we can obtain the different frame rate. Figure 4.6 shows that how the same output can be obtained at different frame rate at different frame numbers. Figure 4.7 is the snapshot of the command prompt output which shows that the when we enable the Temporal Scalability mode only the Temporal Scalability identifier i.e. T is active while other two identifiers D for Spatial Scalability and Q for Quality Scalability shows no changes.

### 4.4 COMBINED SCALABILITY

Combined Scalability takes into consideration all three Temporal, Spatial and SNR, but due to removal of FGS this combination is not possible. But it is possible to achieve other combination i.e. Temporal-Spatial Combination and Temporal-Quality Combination.

To achieve Combined Scalability we have to make some changes in the configuration files of the encoder and different layers we are taking into consideration. For both the cases the changes as mentioned above is done and hence we will be able to get the combination of the two scalability parameters mentions in the configuration files. These changes will be made in the encoder as well as the layers configuration files.

After doing all the desired changes we have to invoke command prompt which can be called as- Go to “Start” (Windows XP, SP2) then “Run” and then type “cmd” and hit “enter”.
Figure 4.5. Temporal output.

Figure 4.6. Another view.
Then change the directory to bin folder which is located in the JSVM folder. Once you are in bin directory type all the JSVM commands you need to run. Firstly we will call encoder by command `H264AVCEncoderLibTestStatic.exe –pf encoder.cfg` will generate the output in the form of .264 file. Then we will extract the bit stream by calling the Bit Streams Extractor command as `BitStreamExtractor.exe input.264` (i.e. input bit stream file name). The bit stream extractor also allows other option of extracting bitstream. Other options can also be specified when using the bitstream extractor to obtain the bitstream of our interest (More results on this is specified later).

Then we have to run the decoder to produce the output. Now we have to execute `H264AVCDecoderLibTestStatic.exe bitstream_file.264 output.yuv` command on the command prompt. The video obtained here in in YUV format and can be played easily in any YUV player. We then calculate the PSNR values for the computational and observational purposes by calling PSNR Calculator.

### 4.4.1 Discussion of Results

Now in this section we discuss results achieved.

### 4.4.2 Results for Spatial Scalability

Some of the outputs of the Combined Scalability are shown below. Here we can see that in Figure 4.8 the comparative output of the snapshot of the videos is shown which gives the output of Temporal-Spatial Combination. Here we can see that different resolution is obtained at different frame rate. Figure 4.9 is the snapshot f the command prompt output which shows that the when we enable the Temporal and Spatial Scalability mode is there the

<table>
<thead>
<tr>
<th>Layer</th>
<th>Resolution</th>
<th>FrameRate</th>
<th>Bitrate</th>
<th>MinBitrate</th>
<th>DIQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>176x144</td>
<td>0.7375</td>
<td>14.00</td>
<td>14.00</td>
<td>(0,0)</td>
</tr>
<tr>
<td>1</td>
<td>176x144</td>
<td>1.8750</td>
<td>20.00</td>
<td>20.00</td>
<td>(0,1)</td>
</tr>
<tr>
<td>2</td>
<td>176x144</td>
<td>3.7500</td>
<td>29.00</td>
<td>29.00</td>
<td>(0,2)</td>
</tr>
<tr>
<td>3</td>
<td>176x144</td>
<td>7.5000</td>
<td>40.73</td>
<td>40.73</td>
<td>(0,3)</td>
</tr>
<tr>
<td>4</td>
<td>176x144</td>
<td>15.0000</td>
<td>53.93</td>
<td>53.93</td>
<td>(0,4)</td>
</tr>
</tbody>
</table>

Figure 4.7. Temporal output at command prompt.
Figure 4.8. Output result for temporal-spatial combination.

Figure 4.9. Command prompt output for the temporal-spatial combination.
Temporal Scalability identifier i.e. T and Spatial identifier Q is active while other Q for Quality Scalability shows no changes. Similarly in Figure 4.10 when SNR-Temporal case is done the output shows that only D identifier is inactive or shows a value of zero.

![Command prompt output for the SNR-temporal combination.](C:\Documents and Settings\shishir\Desktop\jsxv98\bin\BitStreamExtractor\Static.exe testforeman_new.264)

**Figure 4.10. Command prompt output for the SNR-temporal combination.**

### 4.5 Error Concealment

For achieving and executing error concealment we have to introduce error in the sequences by introducing some packet loss using simulator.exe. More about simulator.exe can be found at the Appendix B.

Error Concealment can be applied to any kind of scalability. To achieve Temporal/Spatial/SNR/Combined Scalability follow the steps mentioned above at the encoder side.

Then pass the bitstream obtained from the bitstream extractor to the simulator.exe and obtain the erroneous bitstream. Use of the command simulator.exe bitstream_file_name.264 name_of_loss_file.264 3.dat (we can use 3, 5, 10 or 20 in place of 3 in 3.dat for different losses).

Switch, “-ec”, specifies the error concealment method. The following values are supported: Then we have to run the decoder to produce the output. Now we have to execute
command H264AVCDecoderLibTestStatic.exe name_of_loss_file.264 output.yuv –ec 1 (1, 2 or 3) -ec is for error concealment feature and it can take any value from 1 to 3. The parameter

1. All macroblock are assumed to be coded using BLSkip
2. Frame copy
3. All macroblocks are assumed to be coded in Direct mode

The default value for ec is 0. If the value is equal to 0, we do not perform any error concealment and no packet loss detection. For the bit-streams which are not supported by the current packet loss detection, uiErrorConceal equals to 0 means that it will perform as JSVM without error concealment on the command prompt. The video obtained here in in YUV format and can be played easily in any YUV player. We then calculate the PSNR values for the computational and observational purposes by calling PSNR Calculator.

4.5.1 Discussion of Results

Now in this section we discuss results achieved.

4.5.2 Results for Spatial Scalability

The result of error concealment in Figure 4.11 shows the comparative output of the three error concealment schemes.

![Figure 4.11. Result of error resiliency.](image)
CHAPTER 5

UNEQUAL ERROR PROTECTION SCHEME

With the popularity and availability of the internet, video streaming across the packet erasure networks has received much attention over the last few years. In the current deployment of the internet, routers do not differentiate between the importance of the packets, and may arbitrarily discard packets for congestion avoidance [46]. Such a simple network design methodology poses great challenges to scalable video streams produced by some state-of-the-art video encoding methods [41, 47], such as multirate three-dimensional (3-D) subband video coding, 3-D set partitioning in hierarchical trees (3D-SPIHT) and MPEG-4 fine granular scalability (FGS) just to name a few. This is because, in scalable video coding, the embedded bitstream for a coding unit [e.g., a group of pictures (GOP) encoded with 3D-SPIHT] is only able to be decoded progressively, and a lost packet in the earlier part of the bitstream renders all the following packets (of the same coding unit) useless. Therefore, in transmitting scalable encoded video, it is essential to make the bitstream insensitive to the position of packet loss.

In order to achieve this goal a widely known manner of protection of data from transmission errors is the used i.e. forward-error correcting (FEC). One way of applying FEC is to add parity symbols to the original data. In this thesis the use of Unequal Error Protection (UEP) instead of Equal Error Protection (EEP) for the error-resilient transmission of H.264/SVC encoded data is investigated. By introducing UEP it is possible to individually determine a priority level for every part of the video stream that has to be transmitted. Using H.264/SVC, all parts of the encoded video sequence are stored in Network Abstraction Layer (NAL) units and we will provide priority to the data of higher importance. A higher priority level will result in better protection of the encoded video data, by encoding it with a stronger error correction code. In the design proposed in this paper, this is done by the RS encoder, which will add more parity symbols to data with a higher priority.

The priority or importance of the NAL units is based on the information contained in two parameters, the temporal level and the quality level, which reside in the header of every scalable NAL unit.
A simplified block diagram of the scheme of the proposed setup is as shown in Figure 5.1.

![Block Diagram of the Scheme](image)

**Figure 5.1. Block diagram of the scheme.**

### 5.1 Encoding

Changes are made at the NAL unit and other programs at the encoder side and provide priority to the base layer and enhancement layer according to requirement. These values can be dependent on temporal and quality identifiers. Then regular execution of the SVC encoder is made to generate the encoder bitstream.

This bitstream is feed to the Reed Solomon (RS) encoder. The Reed Solomon (RS) encoder forms code words from the stream of NAL units it receives. These code words consist of data bytes and parity bytes. The RS coder will add a certain number of parity symbols, based on the priority level, to the data to provide protection against erasures. The formed code words are forwarded to the UDP/IP.

The data packets produced by the Reed Solomon decoder are subsequently wrapped in an RTP packet and sent to the decoder over UDP/IP. Since UDP is used, packet losses can occur. Lost packets can be detected by observing the RTP sequence number. If numbers are missing from the linear sequence, it means that the corresponding RTP packets are lost.

### 5.2 Decoding

At the decoder side, the RTP depacketizer first reorders the RTP packets that arrived out of order. Thereafter it pushes the payload of the received packets to the Reed Solomon decoder.
The RS decoder, located at the decoding side, gets the code words from the RTP packetizer. These code words will contain (a part of) the original data of the video sequence, as well as the parity symbols, that were added by the RS coder. The purpose of the RS decoder is to correct erasures in the packets it receives, by applying a complex mathematical operation. The decoding will only succeed if the number of losses/erasures in each codeword does not exceed the number of parity symbols. If the decoding of all the code words containing the information of a NAL unit succeeds, the NAL unit is passed to the decoder. Otherwise, the NAL unit is discarded.

5.3 Priority Assignment

In our approach we determine a priority level for every individual NAL unit to enable unequal error protection. Each NAL unit is first analyzed in order to determine its importance in the bit stream. The different priority levels can be determined because hierarchical prediction in frames is used. In methods, applying hierarchical prediction for frames, the first picture of a video sequence is intra-coded as IDR(instantaneous decoder refresh) picture. These pictures are referred to as key pictures and are usually coded in regular intervals. Each priority index $pr$ is assigned to each NAL unit type, such that a lower priority index is allocated to a more important NAL unit. For the first scalable enhancement information (SEI) NAL unit, for the sequence parameter set (SPS) and picture parameter set (PPS) NAL units a priority index $pr = 0$ is given. The base layer has given more priority than the enhancement layer because all other layers are builds by upsampling the residuals of the base layer.

5.4 Discussion of Results

The following figures show the various outputs at different scenarios. Figure 5.2 shows the comparative output between the UEP scheme and the original sequence. The quality of the reconstructed video sequence(after 5% error introduction) is lower than the original sequence. Figure 5.3 shows the result when the Unequal Error protection scheme is combined with the Error Concealment scheme and the output and PSNR value are calculated. We can see the quality is slightly improved but the difference is not worth the complexes it causes. Figure 5.4 shows the output when we apply the error to the spatial/temporal/SNR scalable bitstream and then protect it. All the results show that the scheme is working very efficiently with the scalabilities and preserving their basic natures as well.
Figure 5.2. The UEP comparative output (a) UEP output (b) Original image.

Figure 5.3. UEP with error concealment (a) Output with UEP and error concealment together (b) Output with UEP only.
Figure 5.4. (a) UEP output for spatial scalability (b) UEP output for temporal scalability (c) UEP output for SNR scalability.
Figure 5.5 represents the output when the combined scalabilities are taken into consideration and the erroneous bitstream is protected and reconstructed. Figure 5.6 shows another comparative output. Figure 5.7, Figure 5.8, Figure 5.9 shows the PSNR values graphical representation and comparisons. Figure 5.10 to Figure 5.24 shows PSNR value graphs with different error ratios and with ER, EC and ER and/or EC options.

Figure 5.5. UEP output for combined scalability (a) Output from temporal SNR combined scalability (b) Output from spatial temporal combined scalability.
Figure 5.6. UEP comparative output for SNR scalability.

Figure 5.7. Graphical representation of PSNR(Y) with UEP.
Figure 5.8. Graphical representation of PSNR(U) with UEP.

Figure 5.9. Graphical representation of PSNR(V) with UEP.

Figure 5.10. Average PSNR values with 3% PLR using different ER/EC method in case of temporal scalability.
Figure 5.11. Average PSNR values with 3% PLR using different ER/EC method in case of spatial scalability.

Figure 5.12. Average PSNR values with 3% PLR using different ER/EC method in case of SNR scalability.

Figure 5.13. Average PSNR values with 3% PLR using different ER/EC method in case of temporal-SNR scalability.
Figure 5.14. Average PSNR values with 3% PLR using different ER/EC method in case of temporal-spatial scalability.

Figure 5.15. Average PSNR values with 5% PLR using different ER/EC method in case of temporal scalability.

Figure 5.16. Average PSNR values with 5% PLR using different ER/EC method in case of spatial scalability.
Figure 5.17. Average PSNR values with 5% PLR using different ER/EC method in case of SNR scalability.

Figure 5.18. Average PSNR values with 5% PLR using different ER/EC method in case of temporal-SNR scalability.

Figure 5.19. Average PSNR values with 5% PLR using different ER/EC method in case of temporal-spatial scalability.
Figure 5.20. Average PSNR values with 10% PLR using different ER/EC method in case of temporal scalability.

Figure 5.21. Average PSNR values with 10% PLR using different ER/EC method in case of spatial scalability.

Figure 5.22. Average PSNR values with 10% PLR using different ER/EC method in case of SNR scalability.
Figure 5.23. Average PSNR values with 10% PLR using different ER/EC method in case of temporal-SNR scalability.

Figure 5.24. Average PSNR values with 10% PLR using different ER/EC method in case of Temporal-Spatial Scalability.
CHAPTER 6

CONCLUSION

H.264/Scalable video coding provides improved adaptation capability to heterogeneous network compared to the earlier SVC standards. Error resilient coding and error concealment are highly desired for the robustness and flexibility of SVC-based applications. In this thesis we studied the structure of Scalable H.264 and improvement for the quality of the bitstream by providing unequal error protection.

Simulation results showed that unequal error protection for SVC, the proposed error resiliency methods, and their combination with Error Concealment methods improve the average picture quality under erroneous channel conditions when compared to the design applying no error-resilient tools at the encoder and only picture copy error-concealment method at the decoder. We were able to execute all the important building blocks of H.264 and were able to achieve the better quality of bitstream using Unequal Error Protection Scheme. When combined with error concealment it is able to provide even better results. We have implemented the error resiliency scheme in temporal, spatial and combined scalability individually and have observed a lot improved and refined bitstream compared to the original bitstream decoded after receiving channel errors or packet losses.

The Unequal Error Protection also helps in obtaining better quality results when using different scalable and temporal layer. But error concealment and error resiliency technique does not work for more than two layers. Any valid bitrate, framerate, resolution can only be achieved, if that is in the range of original bitstream extractor and can be extracted successfully.

Overall it is a good protection scheme and is capable of producing comparative results.
REFERENCES


APPENDIX A

REFERENCE SOFTWARE FOR H.264/SVC - JSVM
The Joint Scalable Video Model (JSVM) software is the reference software for the Scalable Video Coding project of the Joint Video Team (JVT) of the ISO/IEC MPEG and the ITUT Video Coding Experts Group (VCEG). Since the SVC project is still under development, the JSVM Software is also under development. The JSVM software is written in C++ and is provided as source code.

A.1 How To Download JSVM Software
1. Download TortoiseCVS from http://www.tortoisecvs.org/
2. Install TortoiseCVS
3. Right click on desktop and select CVS Checkout
4. Copy following line in CVSROOT
5. :pserver:jvtuser@garcon.ient.rwth-aachen.de:/cvs/jvt
6. Type "jsvm" in Module
7. Click Fetch list.
8. When ask for password, type "jvt.Amd.2"
   And you will get the latest JSVM version on your desktop.

A.2 Building the JSVM Software
1. Extract the Software obtained and follow path:
2. JSVM_9_8 then JSVM then H.264Extension then Build and then Windows
3. In Windows there is H264AVCVideoEncDec.sln. Open this workspace.
4. In H264AVCVideoEncDec.sln workspace go to Build option on the toolbar then to Batch Build option This will open in a new window. There select the Select All option and hit Rebuild.
5. Build then Batch Build then Select All and then Rebuild.
6. This execution will produce a number of executable and library files and other cfg files.
### A.3 Executables Provided by the JSVM Software

<table>
<thead>
<tr>
<th>Executable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DownConvertStatic</td>
<td>Resampler</td>
</tr>
<tr>
<td>H264EncoderLibTestStatic</td>
<td>AVC/SVC encoder</td>
</tr>
<tr>
<td>H264DecoderLibTestStatic</td>
<td>SVC decoder</td>
</tr>
<tr>
<td>BitStreamExtractorStatic</td>
<td>Bitstream extractor</td>
</tr>
<tr>
<td>QualityLevelAssignerStatic</td>
<td>Quality Level Assigner</td>
</tr>
<tr>
<td>MCTFPreProcessor</td>
<td>MCTF pre-processor</td>
</tr>
<tr>
<td>PSNRStatic</td>
<td>PSNR tool</td>
</tr>
<tr>
<td>FixedQPEncoderStatic</td>
<td>fixed Quantization Parameter Encoder</td>
</tr>
<tr>
<td>AVC Rewriter</td>
<td>SVC to AVC Rewriter</td>
</tr>
<tr>
<td>SIP Analyser</td>
<td>SIP Analyzer tool used to make the selective inter-layer prediction decision</td>
</tr>
</tbody>
</table>
### Libraries provided by the JSVM software

<table>
<thead>
<tr>
<th>Library</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H264AVCCommonLibStatic</td>
<td><em>common lib</em></td>
</tr>
<tr>
<td></td>
<td>This library provides classes that are used by both the encoder and decoder.</td>
</tr>
<tr>
<td>H264AVCEncoderLibStatic</td>
<td><em>encoder lib</em></td>
</tr>
<tr>
<td></td>
<td>This library provides classes that are only used by the encoder.</td>
</tr>
<tr>
<td>H264AVCDecoderLibStatic</td>
<td><em>decoder lib</em></td>
</tr>
<tr>
<td></td>
<td>This library provides classes that are only used by the decoder.</td>
</tr>
<tr>
<td>AvcRewriterLibStatic</td>
<td><em>SVC to AVC rewriter lib</em></td>
</tr>
<tr>
<td></td>
<td>This library shares the same source files as H264AVCDecoderLibStatic, but compiled with compiler define SHARP_AVC_REWRITE_OUTPUT. It provides classes that are only used by the AvcRewriter.</td>
</tr>
<tr>
<td>H264AVCVideoIoLibStatic</td>
<td><em>io lib</em></td>
</tr>
<tr>
<td></td>
<td>This library provides classes for reading and writing NAL units in the byte-stream format as well as classes for reading and writing raw video data.</td>
</tr>
</tbody>
</table>

Figure A.1. Libraries provided by JSVM (9.8) software.
A.5 Different Format Used for the Executions

Table A.2. Different Sequence Formats Used for the Executions

<table>
<thead>
<tr>
<th>Format</th>
<th>Luminance Resolution (Horizontal x Vertical)</th>
<th>Chrominance Resolution (Horizontal x Vertical)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQCIF</td>
<td>128 x 96</td>
<td>64 x 48</td>
</tr>
<tr>
<td>QCIF</td>
<td>176 x 144</td>
<td>128 x 96</td>
</tr>
<tr>
<td>CIF</td>
<td>352 x 288</td>
<td>176 x 144</td>
</tr>
<tr>
<td>4CIF</td>
<td>704 x 576</td>
<td>352 x 288</td>
</tr>
</tbody>
</table>

A.6 Test Sequences Used for the Executions

- Example of the Test Clip ‘Foreman’ which we have taken as a sequence and renamed it.
  
  FOREMAN_176x144_7.5_orig_01_yuv  
  FOREMAN_176x144_15_orig_01_yuv  
  FOREMAN_352x288_15_orig_01_yuv  
  FOREMAN_352x288_30_orig_01_yuv  

- We can obtain the test sequence from the website-  
  ftp://ftp.tnt.uni-hannover.de/pub/svc/testsequences/  

- We need to have these files in our folders in order to get the output.
  We have used a number of different sequences to obtain outputs.
APPENDIX B

LOSS SIMULATOR AND CONDITIONS
The Loss Simulator we are using here is provided and written by Yi Guo specially for H.264/SVC and H.264/AVC Video Codec and donated to JVT. This loss simulator discards NAL units from the input bitstream according to the error patterns and outputs the possibly lossy bitstream. The input bitstream must be AVC or SVC bitstream in the byte stream format, i.e. each NAL unit is prefixed with a start code. The output bitstream is also in the byte stream format.

As Proposed in the Busan meet, only packet losses are considered and bit errors are not considered, as UDP would discard packets with bit errors. The 4 packet loss patterns with average packet loss rates of 3%, 5%, 10% and 20% included in ITU-T VCEG Q15-I-16r1 are employed. Details about the generation, file format, and usage of the error patterns are available in Q15-I-16r1.
APPENDIX C

REED SOLOMON CODEC
Reed Solomon (RS) codes are the subclass of BCH codes (Bose, Chaudhuri, and Hocquenghem). Its name is given in honor of their discoverers Irving S. Reed and Gustave Solomon. Reed Solomon codes have widely used for error control in-

- Storage Devices (including tape, compact disk, DVD etc)
- Wireless or mobile communications (including cellular telephones, microwave links)
- Satellite Communication, Digital TV etc

A Reed Solomon code is specified as RS(n,k) with s-bit symbols. This means that the encoder takes ‘k’ data symbols of ‘s’ bits each and add parity symbols to make an ‘n’ symbol codeword.

![Figure C.1. Reed solomon codeword.](image)

Some important parameters for ‘t’ error correcting codes are (Shown as Figure C.1)-

- Block Length: \( n = q - 1 \)
- No. of Parity check symbol: \( n - k = 2t \)
- Dimension: \( k = n - 2t = (q-1) - 2t \)
- Minimum Distance \( d_{\text{min}} = 2t + 1 \)

Where \( q = 2s \) and its number of elements

Data Length = Code length – Fec. Length

This helps us conclude that-

- The length of the code is one less than the size of the code alphabet.
- The minimum distance is 1 greater than the number of parity check symbols.

The RS encoder and decoder are two sections of the system.

**REED SOLOMON ENCODER:**

As discussed before Reed-Solomon (R-S) codes in terms of the parameters \( n, k, t, \) and any positive integer \( m > 2 \).

\[
(n, k) = (2m - 1, 2m - 1 - 2t)
\]
where \( n - k = 2t \) is the number of parity symbols, and \( t \) is the symbol-error correcting capability of the code. The generating polynomial for an R-S code takes the following form:

\[
g(X) = g0 + g1 X + g2 X 2 + \ldots + g2t - 1 X 2t - 1 + X 2t
\]

We describe the generator polynomial in terms of its \( 2t = n - k = 4 \) roots, as follows:

\[
g(X) = (X - \alpha) (X - \alpha^2) (X - \alpha^3) (X - \alpha^4)
\]

\[
= X^2 - (\alpha + \alpha^3) X + \alpha^2
\]

\[
= (X^2 - \alpha^3 X + \alpha^0) (X^2 - \alpha^6 X + \alpha^0)
\]

\[
= X^4 - (\alpha^4 + \alpha^6) X^3 + (\alpha^3 + \alpha^{10} + \alpha^9) X^2 - (\alpha^4 + \alpha^9) X + \alpha^3
\]

\[
= X^4 - \alpha^3 X^3 + \alpha^6 X^2 - \alpha^4 X + \alpha^3
\]

Following the low order to high order format, and changing negative signs to positive, since in the binary field \( +1 = -1 \), \( g(X) \) can be expressed as follows:

\[
g(X) = \alpha^3 + \alpha^1 X + \alpha^0 X^2 + \alpha^3 X^3 + X^4
\]

The output codeword, \( U(X) \), written in polynomial form:

\[
U(X) = \alpha^0 + \alpha^2 X + \alpha^4 X^2 + \alpha^6 X^3 + \alpha^1 X^4 + \alpha^3 X^5 + \alpha^5 X^6
\]

The roots of a generator polynomial, \( g(X) \), must also be the roots of the codeword generated by \( g(X) \), because a valid codeword is of the following form:

\[
U(X) = m(X) g(X)
\]

Therefore, an arbitrary codeword, when evaluated at any root of \( g(X) \), must yield zero. It is of interest to verify that the codeword polynomial in does indeed yield zero when evaluated at the four roots of \( g(X) \). In other words, this means checking that

\[
U(\alpha) = U(\alpha^2) = U(\alpha^3) = U(\alpha^4) = 0
\]

The architecture of the RS encoder is shown in Figure C.2 –

![LFSR encoder for a (7, 3) R-S code.](image)

**Figure C.2. Architecture of reed solomon encoder.**
**REED SOLOMON DECODER:**

The received corrupted-codeword polynomial, \( r(X) \), is then represented by the sum of the transmitted-codeword polynomial and the error-pattern polynomial as follows:

\[
r(X) = U(X) + e(X)
\]

We add \( U(X) \) from above to \( e(X) \) and so \( r(X) \) yields, as follows:

\[
\begin{align*}
   r(X) &= (100) + (001)X + (011)X^2 + (100)X^3 + (101)X^4 + (110)X^5 + (111)X^6 \\
   &\quad = \alpha^0 + \alpha^2X + \alpha^4X^2 + \alpha^0X^3 + \alpha^6X^4 + \alpha^5X^5 + \alpha^5X^6
\end{align*}
\]

The more detailed information about Reed Solomon Codes can be obtained from:

APPENDIX D

CODES AND RESULTS
Due to the extensive size of the project, some of simulations, results not attached. If results, scripts or video codec is required, please contact the following individuals:

Dr. Sunil Kumar
Email: skumar@mail.sdsu.edu

Shireen Shankar
Email: shireenshankar@gmail.com